
VCU College of Engineering

# 346 AI/Linguistics
# Preliminary Design Report

Prepared for

Clinton Farrell

DoD

By

Nathan Devore

Nate Eldering

Connor Kohout

Allen Lee

Under the supervision of

Tamer Nadeem

5 December 2024

## Executive Summary

Military operations occur in noisy environments and involve specific terminology that current speech-to-text models cannot handle effectively. Air communications, in particular, are hindered by wind, radio static, and engine noise, degrading transcription quality. The solution to ensuring clear communications is a speech to text software that utilizes artificial intelligence to display in text what is being spoken in real time.

Of available options, our team had decided that the cheapest and more reliable solution is a fine-tuned version of OpenAI's Whisper model. OpenAI's Whisper model has been installed on VCU's HPRC nodes for its high performance, remote accessibility, and collaborative working environment. It will be trained in a dataset that we will create to optimize the model so it can better handle the background noise encountered in Army environments.

The model has been tested on real world air traffic control data from ATCLive.net to establish a baseline accuracy. Testing with real air traffic communications shows the Whisper model performs inconsistently in these environments.. By the start of the second semester we will have training data to improve Whisper's accuracy. We will acquire this data by having a pilot read from a script while flying. Any further training will be done with synthetically created noisy audio.

We hope to have the fine-tuned model accurate to a minimum of 82% accuracy for noise levels up to 100 db based on word error rate. Our project will adhere to necessary codes and standards.

# Table of Contents

## Section A. Problem Statement

Automatic caption generation has become increasingly essential due to today's current landscape of remote work and online communications, the technology to support this has also made great advancements in the recent years since COVID-19. However, existing solutions still face significant shortcomings, especially in challenging environments characterized by disruptive noise such as ones seen in military environments. Clear and precise communication is essential in high risk environments and this can be aided with the help of speech to text technology.

The technologies that exist today use a variety of algorithms to enhance audio clarity and minimize background noise, some work in the time domain, treating the audio as a waveform and performing operations to optimize the quality of the desired sound by isolating it from other background sounds. Others function in the frequency domain, identifying which frequencies are present in the audio and focusing on the primary sources while attempting to enhance their quality by reducing unwanted frequencies. Although these algorithms have made significant strides they still fall short in providing optimal performance in noisy environments. Furthermore, the effectiveness of models using these algorithms heavily relies on the quality of their training datasets. Acquiring well-labeled, large-scale datasets that accurately represent various types of noisy audio can be a substantial challenge. This scarcity of high-quality training data complicates the training process, hindering the model's ability to generalize and perform reliably in real-world applications.

The client for this project created it with the idea of it being used in military aviation settings, for example a helicopter in which communication is through radio, it is important for that radio transmission to be clear and the help of automatically generated captions can aid this process. Additionally, in the setting of military air travel, the vocabulary and dialect differs from that of everyday conversation enough to where it is necessary to take into account when designing a software to turn this text into speech. For example the NATO phonetic alphabet is commonly used, these words when used out of context could create confusion for a model that is expecting something else but with the correct architecture and training it is possible to learn what these words mean and when they should be used. Additionally, accents are very prevalent in the army population and this also should be accounted for. A study done by Qxf2 (2023) to test the accuracy of OpenAI’s Whisper model on detecting words with various English accents found that for many different accents like Arabic, French, and German there was a low word error rate. For accents like Polish, Turkish, and Russian the word error rate was high. While there was no correlation drawn between which accents performed better than others these results are indicative that with the correct training the model has the ability to understand the English language through heavy accents.

Collecting valuable labeled data for testing is the main cost of this project, ensuring that the audio is distorted in an accurate way, whether that be collected in real-world scenarios or modified after collection to appear more distorted. OpenAI’s Whisper model, which was trained

on a diverse range of languages and types of audio interference, has proven to be more robust without the need for dataset-specific fine-tuning to achieve high-quality results (Radford et al., 2023). This is relevant to our project, as we plan to use dataset-specific fine-tuning with the goal of our system being specialized for military environments and the paper supports the idea that this approach will be effective. The whisper model will be the main one used in this project as it is open source and provides a range of sizes, the smaller models perform very quickly while taking in a smaller number of parameters and producing output with less accuracy than the larger models which can take more parameters and perform slower. This range will be helpful as we gauge which features are more important in the context of this project, whether that be a faster or more accurate transcription.

Services like Microsoft and Google offer paid options that support fast live transcription in a large variety of languages. Google Cloud Speech-to-Text is customizable and allows users to adjust the domain of their speech recognition to specific modes like phone calls or video calls. Additionally this model offers the option to adjust the expected vocabulary so in specific settings similar words will be given higher priority. Google's model is good at handling pre-processed audio as well as live transcription in a variety of languages. This model has been used extensively across industries for applications such as transcription, voice commands, and customer service automation. However, it has limitations in extremely noisy environments, such as those found in military or industrial settings, where background noise can significantly degrade performance. Although the service includes features like noise-canceling algorithms and speaker diarization, achieving robust performance in environments with intense noise, such as helicopters or heavy machinery, remains a challenge. Microsoft's service offers similar customization options while also suffering from similar shortcomings such as their dependence on an internet connection and their service fee. Due to these limitations we decided they were not the best direction to move forward with for this project.

According to Keller (2010), the DARPA Robust Automatic Transcription of Speech (RATS) development program was launched with the goal of determining speech activity, identifying speakers, and spotting keywords in highly degraded audio environments. The system created through the RATS program took communication channels and put them into a system of four sub systems differing in their voice activation detection (VAD) algorithm (Thomas, Saon, Van Segbroeck, & Narayanan, 2015). The combination of these VADs and their respective focuses resulted in a robust model that was very good at handling communication channels filled with both stationary and dynamic noise. In the general field of automatic speech recognition, the RATS program is significant for its emphasis on robustness in extreme noise conditions, an area where traditional ASR systems struggle. The program's innovations, including the use of diverse acoustic features and deep neural network models, have laid a foundation for further advancements in speech recognition under adverse conditions.

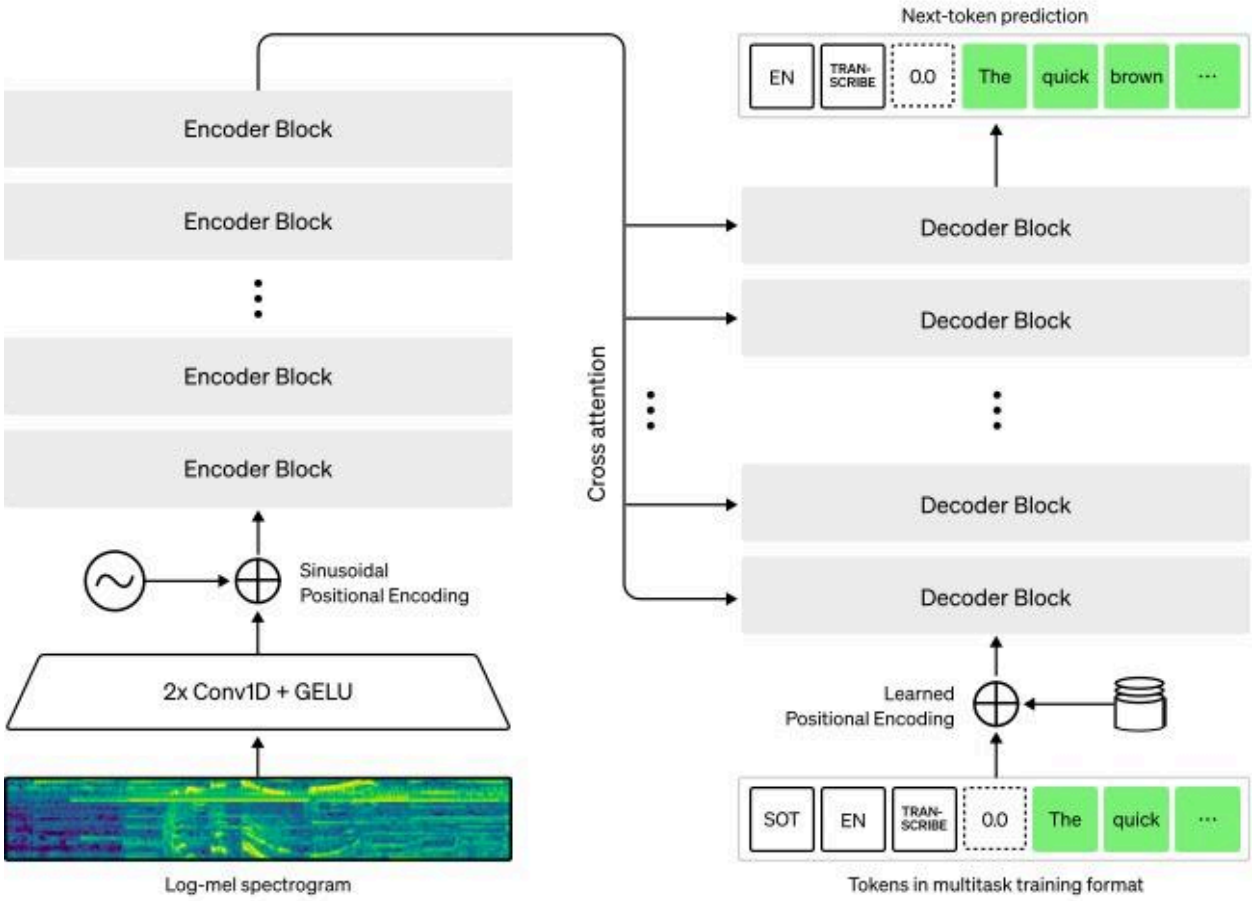



**Figure 1. Open AI's Whisper model architecture**

## Section B. Engineering Design Requirements

### B.1 Project Goals (i.e. Client Needs)

Create an automated solution to accurately caption live audio in military scenarios. Current systems for speech recognition do not effectively address the unique challenges presented by the military environment, such as background noise, diverse accents, and specialized vocabulary. Therefore, the main goal is to create a proof of concept that shows it is possible to tailor existing speech recognition models for this application.

- Develop a system to automatically caption live audio in military settings.
- Ensure the system is robust against various types of background noise (e.g., helicopter sounds).
- Address challenges such as regional accents and English as a second language for a large portion of personnel.

- Incorporate military-specific vocabulary and protocols (e.g., phonetic alphabet and repeated communications).
- Achieve accuracy comparable to current systems used in civilian applications, tailored for military use.

**B.2 Design Objectives**

The design will focus on specific measurable objectives that align with the needs of the military environment. These objectives ensure the model developed can function under different conditions and accommodate the unique needs of the military.

- The design will transcribe audio with at least 82% accuracy in noisy environments.
- The design will correctly interpret and transcribe military-specific vocabulary.
- The design will process speech with varying accents and dialects common among military personnel
- The design will allow for real-time transcription with latency
- The design will support transcription accuracy tests with audio from various noise levels and conditions.

**B.3 Design Specifications and Constraints**

The system will be required to meet certain technical constraints relating to noise levels, processing power, and speech recognition accuracy. These constraints are essential for ensuring that the system can work under real-world military conditions and handle the varied demands of this environment.

The design will focus on testing and optimizing the model under the following constraints:

- **Noise environment constraints**: The system must function in environments with noise levels up to 100 dB (helicopter noise) and still maintain an accuracy threshold of 82%.
- **Accent and dialect constraints**: The model must be able to transcribe speech from personnel with southern U.S. accents, as well as non-native English speakers, with at least 82% accuracy.
- **Vocabulary constraints**: The system must recognize and transcribe military language, phonetic alphabets, and repeated communications without exceeding a word error rate of 18%.
- **Latency constraint**: Transcription of live audio must have a latency of no more than 2-3 seconds.

## B.4 Codes and Standards

There are no specific codes that need to be followed, however there are standards.

- ISO/IEC 27001 - Captions should only be accessible to authorized persons and systems.
- ITU-T P.800 - The accuracy of the system should be tested under various conditions.

## Section C. Scope of Work

### Project Scope:

The primary objective of this project is to develop an audio transcription system capable of accurately transcribing audio files, even in challenging environments with significant background noise. The system will leverage the Whisper model, running on a cluster, to process and analyze audio files efficiently. The project aims to develop solutions that enhance the model's performance under conditions such as loud sounds, white noise, or other disruptive environments.

### Key Objectives:

1. Implement the Whisper model on the school's cluster.
2. Process audio files and transcribe them with a focus on noisy data.
3. Train the model to handle diverse audio conditions and voice distinctions.
4. Ensure the system can recognize and differentiate between multiple speakers in an audio clip.
5. Develop a user-friendly interface or tool for testing and validating transcriptions.

### Timeline & Milestones:

- **Phase 1:** Initial setup of the Whisper model on the cluster. Completion of setup, including all dependencies such as ffmpeg and Rust (End of Month 1).
- **Phase 2:** Gathering and preparing audio data for model training, potentially synthesizing additional data (Month 2).
- **Phase 3:** Testing transcription accuracy under varied conditions, including environments with high noise levels (Month 3).
- **Phase 4:** Model optimization and refinement based on test results. Implementation of voice distinction capabilities (Month 4).
- **Phase 5:** Final testing and deployment of the transcription tool for user interaction (Month 5).

### Responsibility of the Team:

- Set up and maintain the Whisper model on the cluster, ensuring smooth operation.
- Gather and preprocess the required audio data.
- Train, test, and refine the model.
- Collaborate with the project sponsor and faculty advisor to ensure timely progress and meet deliverables.
- Maintain regular communication and provide updates on the project's status.
- Develop user documentation and a final report summarizing project outcomes.

### Exclusions:

- This project does not involve developing audio recording hardware or other tools outside the transcription scope.
- The system is not responsible for handling real-time audio processing or live transcription beyond testing datasets.

### C.1 Deliverables

In order to mitigate risks associated with the completion and delivery of the project deliverables, provide an outline of the most potentially disruptive, foreseeable obstacles. Some important issues to discuss with the design team, sponsor, and faculty advisor include the following:

- What deliverables require access to campus? Which/how many students regularly access campus and are physically available to complete tasks?
- What work can be done remotely? What resources might be needed in order to ensure that remote work can be completed effectively (e.g. software licenses, shared drives/folders, etc.)?
- What deliverables require ordering from third-party vendors? Will any components potentially require extended lead times? What can the team do in order to mitigate potential supply chain disruptions?

The following is a list of all agreed-upon project deliverables for the audio transcription capstone project, applying the Whisper model on the cluster:

- **Whisper Model Setup:** Successful setup of the Whisper model on the school's cluster, including all dependencies (e.g., ffmpeg, Rust).
- **Data Collection & Preprocessing:** A dataset of audio files, including both clean and noisy environments, to train and test the transcription model. This may involve synthesizing additional data if necessary.
- **Trained Whisper Model:** A fully trained Whisper model, optimized to handle noisy environments in an audio file for radio voices.
- **Transcription Tool:** A functioning transcription tool with an interface for users to upload audio files and receive
- **Project Documentation:** User documentation, detailing how to use the transcription tool, as well as a technical report summarizing the model setup, data preprocessing, training process, and final performance metrics.
- **Academic Deliverables:**
    - Team Contract
    - Project Proposal
    - Preliminary Design Report
    - Fall Poster and Presentation
    - Final Design Report
    - Capstone EXPO Poster and Presentation

## Risk Mitigation and Obstacles

- **Access to Campus:** The model is being run on the school's cluster, which requires access to campus resources. Regular campus access will be necessary to manage the model's training and maintenance, as the cluster resources are hosted by the school.
    - **Mitigation:** Team members who need physical access to campus resources have been identified and will coordinate their access. Remote access to the cluster is available to mitigate the need for constant campus presence.
- **Remote Work:** Most work, including data preprocessing, model testing, and report writing, can be done remotely. Shared drives and collaboration tools like GitHub will be used to manage code and documentation across the team.
    - **Mitigation:** Ensuring that all necessary software (e.g., Python, Whisper model dependencies) is installed and configured for remote access. Regular virtual check-ins will be held to maintain progress.
- **Third-Party Dependencies:** There are no components requiring third-party ordering. However, the project does rely on open-source software (Whisper, ffmpeg, Rust), which may present compatibility challenges or delays if updates or changes occur during the project.
    - **Mitigation:** The project team will monitor updates to dependencies and ensure compatibility through testing.

By identifying these potential risks and obstacles, the team is prepared to manage and mitigate disruptions to the project timeline.

**C.2 Milestones**

| Milestone | Description | Estimated Time | Completion Date |
|---|---|---|---|
| Whisper Model Setup | Install and configure Whisper model on the school's cluster, including ffmpeg and Rust dependencies. | 2 weeks | October 31, 2024 |
| Data Collection & Preprocessing | Collect and preprocess audio data, including noisy environments and synthesized data. | 3 weeks | November 21, 2024 |
| Initial Model Training | Train the Whisper model using the collected audio data to handle noisy environments. | 3 weeks | December 12, 2024 |
| Testing & Evaluation of Transcriptions | Evaluate model performance under various conditions, including multiple speakers and | 3 weeks | January 2, 2025 |

| | | | |
|---|---|---|---|
| | noisy environments. | | |
| Speaker Differentiation Implementation | Add functionality for recognizing and differentiating multiple speakers in an audio file. | 3 weeks | January 23, 2025 |
| Transcription Tool Interface | Develop user interface for uploading audio files and displaying transcriptions. | 3 weeks | February 13, 2025 |
| Final Model Optimization | Optimize the model for final testing and evaluation, ensuring all requirements are met. | 2 weeks | March 5, 2025 |
| Project Documentation & Reports | Complete all necessary documentation, user manuals, and final project reports. | 3 weeks | March 26, 2025 |
| Capstone EXPO Preparation | Prepare poster, presentation materials, and final demonstration for Capstone EXPO. | 2 weeks | April 16, 2025 |

This table provides a breakdown of key milestones to guide the project. Each milestone is designed to ensure steady progress while allowing for iterative improvements, especially following the Agile approach.

**C.3 Resources**

The Whisper model training project primarily relies on access to the school's server cluster and the open-source Whisper software. However, certain software, hardware resources, and possible cloud services were considered in case additional computing power was required. This section outlines all project expenditures and, if relevant, any experimental setups or prototypes developed.

Since we are utilizing open-source software and libraries, the direct cost for software tools is minimal. However, potential costs for commercial or cloud-based services are also considered.

| Software | Vendor | Cost |
|---|---|---|

| | | |
|---|---|---|
| Whisper | OpenAI | $0 |
| Python | Python.org | $0 |
| FFmpeg | FFmpeg.org | $0 |
| VSCode | Microsoft | $0 |
| AWS Cloud Computing (Optional) | Amazon | $70 |

The project utilizes the school's server cluster for training, which incurs no direct hardware costs. However, for future scalability or more significant computation needs, we considered cloud computing resources.

We expect to receive some datasets from our advisor, which incur no direct cost. Synthesizing data may require additional storage and processing but has no direct monetary expense in the current setup.

| Dataset | Vendor | Cost |
|---|---|---|
| LibriSpeechASR corpus | LibriSpeech | $0 |
| Advisor Provided Data | DoD | $0 |
| Synthesised Data | Self made | $0 |

## Section D. Concept Generation

Design concept A was a multi model machine learning system that would optimize for different situations, the idea was to identify specific aspects of noisy audio that could be targeted and extracted to train a set of speech to text models to be used together. When audio is encountered it would first be passed to a decision making model that could analyze the noise and decide which speech to text model would best translate, then it would pass the audio to that model and text would be output. The main advantage of this concept was the possibility of the same audio having different types of interruptions, in this case the design process would break up the audio into segments and pass each to a translator, combining them together after transcription. A disadvantage to this concept is the reliance on a way to identify and categorize different types of noise that commonly appear. This concept would require multiple clean and prepared datasets each with different characteristics which could also have potentially made the execution very challenging. Overall this concept was not chosen however aspects of it, such as training a speech to text model on noisy data, were helpful in generating the idea for the concept we did choose to move forward with.

Design concept B was to implement a noise reduction feature to OpenAI's Whisper model. This idea was aimed at transforming the audio in a way that allowed the base whisper model to better translate it. This solution addresses the issue of translation of noisy audio that's being delivered over radio waves by attempting to isolate the voice of the speaker. This would require some form of audio manipulation which was a concern when discussed as that could present a challenge. The benefit to this idea was that it would not require any retraining as the Whisper model would be used as it comes, however after analyzing all ideas it was decided the retraining was not cumbersome enough of a task to make this idea more effective than the one we decided to move forward with.

Design concept C was to create a dataset of labeled audio and use that to retrain OpenAI's Whisper model so it can perform better under the noisy conditions of air travel. This concept addresses the problem by using the current state of the art technology and narrowing the scope of conditions with which it can handle. Since our project focuses only on noisy audio that is obtained from aircrafts and radio transmissions this narrow scope will not only still be effective but allow for accurate results given conditions that would otherwise make it unable to obtain the same outcome. Potential downsides to this concept that were discussed are the requirement for a clean representative dataset, the ability to retrain the Whisper model, and the need for the proper computational power. After discussing as a team, with our faculty mentor, and our sponsor from the DoD it was clear that resourcing the compute power and retraining the model would not be an issue for the scale we are working in. Additionally multiple solutions for obtaining a dataset were put forward, there are archives of radio communications between airplanes and air towers that are free to download on the internet, our sponsor has been working to obtain audio from the pilots he works with which would be the best solution as it would likely be the most representative, we have also begun exploring ways to synthetically make clean audio noisy in a way that is similar the noisy from audio taken inside aircrafts. This concept proved to have the most potential since it relies on a very fast and powerful machine learning model that has already

been made and distributed in a variety of sizes and speeds which allows for quick and easy scaling.

## Section E. Concept Evaluation and Selection

In order to properly evaluate each of our design options, we came up with the following criteria. The selection criteria we chose were cost, reliability, performance, time intensity, and computational resources required.

Cost refers to how expensive each design choice would be, including, but not limited to, the cost of data collection and manipulation, any cost related to the acquisition of a machine learning model, and the potential cost of running the model on a paid server. Reliability, in our case, is not only based on how reliably we can run each model

While it is very important to us that our project be as accurate and reliable as possible, it is also important to consider what is reasonable to create and what resources we would need to run the model on. In table 1, we labeled each of the criteria as criteria 1 through criteria 5: cost is criteria 1, reliability is criteria 2, performance is criteria 3, time intensity is criteria 4, and computational resources required is criteria 5.

We assigned value to each criterion on a scale of 1 to 10, 1 being the lowest and least favorable and 10 being the highest, representing a more favorable rating. All concepts scored relatively high in the first criteria cost, the most cost intensive aspect of any of the concepts is the ability to store and run the model being used for transcription on a remote server however after identifying our ability to use VCU's HPRC for free it was clear that would not present as a major issue for any concept. The methods used in concepts B and C are relatively similar, since concept A focused on retraining a model on specific aspects of noisy data, as opposed to concept C which is training on a set that contains a variety of data B was given a lower score than C in criteria 2.

Concept B required an efficient way to soften the noise in audio and isolate speaking voices which resulted in its lower score in criteria 2. Criteria 3 assessed performance again concepts A and C were given higher scores since they were using a very robust model that has lots of documentation supporting its ability to transcribe audio. Concept B was given a lower score due to the research that was done assessing the ability of current noise reduction technology which showed to be a challenge given the specific conditions of this project. For criteria 4 concept A was given a low score because it would require multiple sets of differently unique noisy audio, this was seen as a potential issue and ensuring the data is clean and formatted correctly would be a very time consuming task. Concept B was given a mid range score again due to the research required to identify and produce a method to silence audio. Concept C was given a higher score because it only required one dataset to be made that represented a range of conditions. Criteria 5 assessed computational requirements, concept A was given the lowest score because it would require storing multiple fully trained and functional models, while this would not be an issue during development since we are using VCU's HPRC which is free to use, it could become an issue when trying to distribute or when finding a solution to implementing the software in the cockpit of the aircraft. Concept B was given a mid range score as processing audio can be both computationally and time consuming. Concept C was given the highest score because it only

required one retrained model to be stored and the process of implementing it on a smaller scale would be less of a challenge.

**Table 1. Decision Matrix.**

| | Design Concept A | Design Concept B | Design Concept C |
|---|---|---|---|
| Criteria 1 | 7 | 8 | 9 |
| Criteria 2 | 5 | 3 | 7 |
| Criteria 3 | 5 | 3 | 7 |
| Criteria 4 | 1 | 5 | 8 |
| Criteria 5 | 3 | 5 | 9 |
| | | | |
| Total Score | 21 | 24 | 33 |

## Section F. Design Methodology

### F.1 Experimental Methods

The primary experimental method we will be using in our project is in regards to data collection. In order to train the model to be as accurate as possible on military communication we need data that reflects the environment it will be used in. Because our main priority is to transcribe data coming from military aircraft, we plan to collect data using a real aircraft. We plan to write a script and have the pilot or passenger of the aircraft read the script when the aircraft is taking off, in the air, and landing. We are unsure how long we will be able to collect data while in the air, so we may end up with too little usable data to train the model on.

### F.2 Computational Methods

Due to how difficult it is to collect relevant data to train the model on, we are working to create synthetic data as a backup. We do this by using FFmpeg to alter many audio files at a time, and by choosing audio files that already have transcripts. Choosing files that we have transcripts for saves us the enormous amount of time it takes to manually label audio. We alter the audio files in ways that mimic aircraft radio communication like overlaying the sound of a plane's engine or radio static to the entire audio file. If we do not collect enough real world data to train the model on, we will mix the real world data we do have with the synthetic data and train the model on the mixed data. Since the synthetic data can be tailored to what we need, we will create the synthetic data based on what we are missing from our real world data, so that the model is well rounded and less likely to memorize.

Another computational method we use is removing the silence from audio files. We use FFmpeg for this as well. During our testing, we found the Whisper model hallucinating during long periods of silence. We decided that the most effective way to combat this is to remove the silence in our audio files. In a live setting where transcriptions are happening in the moment, the solution would be to not send any audio segments to Whisper that don't have speakers. In our case, however, in order to test and train the model we have to use pre-recorded clips, meaning the silence must be removed.

### F.3 Architecture/High-level Design

The model itself is hosted on VCU's server. Through this server, we can run and train the model without needing to pay for computational resources. This also allows every member of the group to use and change files as necessary. On the server, we have the model itself, the necessary dependencies required to run the model, and a Jupyter Notebook file containing the required code to train the model. This Jupyter Notebook file requires many imported libraries which will all be stored on the server as well. As well as everything relating to the model, we also have all of our data stored on the server, so Whisper can access it. We may switch to storing our training data on huggingface.co if we find that it is easier to train the model that way. However, all of our test data will be on VCU's server. Once the model is trained, it will be tested using the validation procedure outlined below.

**F.4 Validation Procedure**

We plan to have three different validation tests. If we are able to collect enough data to fine tune the model without needing synthetic data, we will save 10% of it to use solely for testing purposes. If we collect only enough data to partially train the model, we will mix the collected data with synthetic data and use K-Fold Cross-Validation to assess the quality of the model. In K-Fold Cross-Validation, a section of the data, in our case 10% of the data, is used to test the model, while the other 90% is used to train it. The model is then assessed, and the training is repeated with the next 10% of data, with the rest being used to train it. For example, if we have 100 audio files of equal length, audio files 1 through 10 would be used to test the model, while audio files 11 through 100 would be used to train the model. Then, audio files 11 through 20, would be used to test the model, and this would be repeated until all sections of the data have been used to test the model. We will also get an updated score for the four airports used for initial testing to visualize the improvements made by the model. Lastly, additional validation will be done with synthetic audio, since our real world data will be limited. Results and progress will be shared with the sponsor at regular intervals. We plan to have the model fully trained and tested by February 28, 2024.

## Section G. Results and Design Details

### G.1 Experimental Results

To establish a baseline for transcription accuracy, we manually transcribed ATC audio and compared the results to outputs from the Whisper model. Because manually labeling data is so time consuming, we will not be using data from LiveATC.net for training the model. The dataset comprises four audio files of varying clarity, sourced from LiveATC.net archives. These recordings represent diverse real-world scenarios from multiple pilots, aircraft, and radio equipment. The audio was collected from the airports KCHO (Charlottesville Albemarle Airport, Virginia), KHMT (Hemet-Ryan Airport, California), K0V4 (Brookneal-Campbell County Airport, Virginia), and KTOC (Toccoa Airport, Georgia). This provides a robust foundation for evaluating the Whisper model. When we tested the model on these four audio files, we got the results shown in figure 2. While some of these tests went rather well, it was incredibly inconsistent.

In its current state, Whisper struggles with identifying proper nouns and the phonetic alphabet, significantly hurting its accuracy. For example, it was unable to recognize "Hemet" (the name of the airport), incorrectly labeling every instance as "have it." When we fine-tune the model, we will target proper noun recognition and the phonetic alphabet; this includes airport names, company names, aircraft manufacturers, and types.

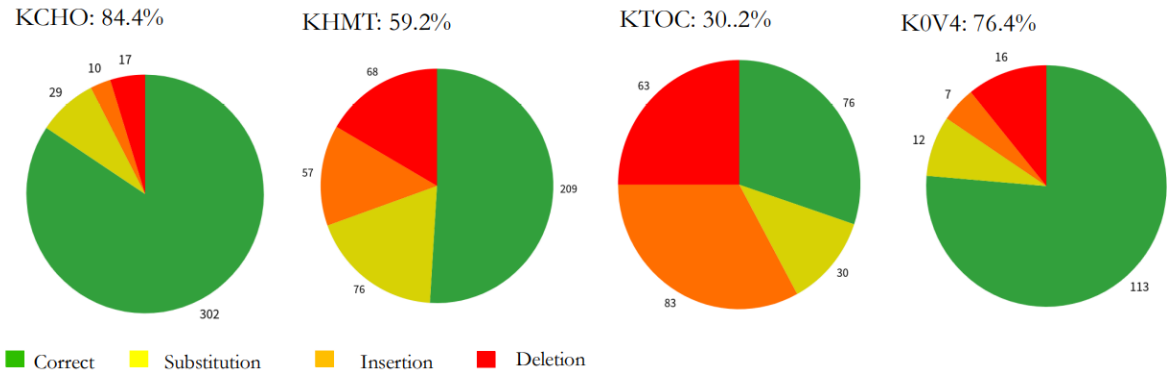



**Figure 2: Test Results of Whisper on LiveATC.net Data**

## G.2 Final Data Collection Methods

The data collection methods that we are using in our final design can be divided into two categories. The first category focuses on real world data collection. We have decided to collect data from an aircraft by paying a pilot to read a script while in the air. The script will be created beforehand and used as the labels for the audio data acquired from the flight. Because of the unknowns surrounding this tactic, we are not yet sure how much data we will realistically be able to collect with this method. Due to this, we may need to supplement the real world data with synthetic data. This leads us to the second category of our data collection process.

The second category of our data collection methods is through the generation of synthetic data. This data will be created by collecting and altering free audio files that already have transcripts, or labels, associated with them. The audio files will be edited using FFmpeg, which allows us to edit large audio files all at once. The edits being made range from adding radio static and engine noise to altering the files' bitrates. All of these edits simulate a different aspect of aircraft communication, and when mixed with the real world audio will lead to a more robust version of the model.

## G.3. Final Design Details/Specifications

Whisper's baseline performance can be enhanced through targeted fine-tuning using aviation-specific data. By having pilots read scripted communications during flights and collecting audio through LiveATC archives, we ensure the training data matches real-world radio conditions. This precisely labeled dataset will help the model filter out cockpit and radio interference while improving speech detection in noisy environments. We also plan to use data to train the model such that the fine-tuning process will also enhance recognition of proper nouns and the phonetic alphabet.

If the amount of real world data we collect is not enough to fully train the model on, we will mix the real world data with the synthesized data as described above. The mixed, labeled audio will then be used to train the model with. We also plan to formulate a way for our model to be implemented in a real-time environment. We would like to see the model, once trained, reach an 85% accuracy. This accuracy will be an average accuracy across all of our test data. We plan on either using K-Fold Cross-Validation, or splitting our data into two sections, 90% training and 10% testing, depending on how much synthetic data we use. Either way, the accuracy will be measured using two different metrics. The first metric will be the standard word error rate, while the second metric will be focusing on how many keywords the model incorrectly transcribes. These two metrics will tell us both the model's general accuracy and how well we trained the model to detect keywords like nouns.

## Section H. Societal Impacts of Design

### H.1 Public Health, Safety, and Welfare

This project may have adverse safety implications if the model is providing inaccurate information to pilots. Though since we are very far from implementing a system that will interact with pilots in real time, and likely will not with this project, this safety concern is not as realistic.

### H.2 Societal Impacts

In terms of public safety, we had to consider the accuracy involved in the output of our model. If we were to inaccurately report locations or information to pilots due to a design flaw or an inaccuracy in our model, then we could potentially cause real world harm as aviation relies heavily on accurate real-time reporting.

### H.3 Political/Regulatory Impacts

Due to the safety concerns associated with inaccurate reporting, we need to consider the potential need for standards or regulation regarding this technology. Privacy considerations need to be taken into account as the information about locations of planes could be potentially regulated.

### H.4. Economic Impacts

This technology could be put to commercial use with airlines or different industries taking advantage of this. These changes would impact design considerations and could potentially impact on safety or welfare.

## Section I. Cost Analysis

The cost of development is very low, since all the software we are using is open source and free to use. Hosting the speech to text model in the cloud is also free for us as VCU students. Additionally, our vision for the direction of the project after the semester completes is to potentially work out a way for the software to run in the cockpit with the pilot, in this case there would be no additional cost to run the software as it would be stored and processed locally in the aircrafts computer system.

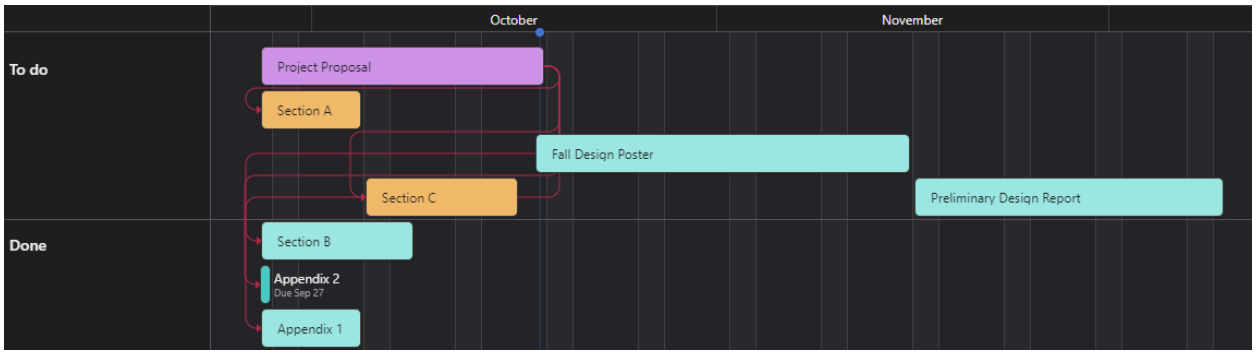
## Section J. Conclusions and Recommendations

The design process for this project began with a clear definition of the problem: existing speech to text technologies struggle to maintain accuracy in environments with high noise level and specialized vocabularies. The first steps we took were doing initial research on the topic, generating multiple concepts for solutions as well as the criteria to measure their effectiveness, and scoring those concepts based on our research. Our team ultimately selected the fine-tuning of OpenAI's Whisper model on aviation specific data as the most feasible and effective solution. Due to the limitations in noise reduction techniques and multi model approaches to speech recognition the single fine tuned Whisper model proved to be the best given the scope and timeframe of this project. Our process from idea generation to criteria scoring was very iterative, our team would zone in on one component and try to improve upon it and assess the potential performance to identify weak spots and then repeat that process, this approach helped to flesh out the issues that could potentially arise when implementing the proposed solution.

We designed a clear set of metrics and results that we aim to achieve with our fine tuned model at the end of the spring semester. The model is to recognize proper nouns, the phonetic alphabet, and diverse accents, while maintaining a word error rate below 18% in noise levels up to 100dB. Based on our preliminary testing of the base Whisper model and results from research done on similar projects we believe this to be attainable and a good bar to set to ensure accurate communication through the model.

There are many possible directions we envision our project moving in after our time working on it is over. Enhanced data collection and continuous testing would be a great way to further ensure the robustness of our retrained model and ensure any changes in speech and communication are reflected. Additionally, it would be possible to implement noise reduction techniques, similar to what was discussed in our concept generation phase, that could be used on top of the transcription. Our team also has identified an important component to our solution is a reliable way for the transcription to be displayed. Including a visual aspect to our project is an important addition that could be explored if this project continued.

# Appendix 1: Project Timeline



## Appendix 2: Team Contract (i.e. Team Organization)

### Step 1: Get to Know One Another. Gather Basic Information.

**Task:** This initial time together is important to form a strong team dynamic and get to know each other more as people outside of class time. Consider ways to develop positive working relationships with others, while remaining open and personal. Learn each other's strengths and discuss good/bad team experiences. This is also a good opportunity to start to better understand each other's communication and working styles.

| ***Team Member Name*** | ***Strengths each member brings to the group*** | ***Other Info*** | ***Contact Info*** |
|---|---|---|---|
| Nathan DeVore | NLP, Python, Java, C, Engineering, problem solving | | devoreni@vcu.edu<br><br>(434) 282-8258 |
| Allen Lee | Experience with Machine Learning, Java, C(++,#), Python in that order. | Currently in a few classes that may also help, like Artificial Intelligence and Databases. Have had some fairly unresponsive groups in the past. | leea18@vcu.edu<br><br>(804) 683-8526 |
| Nate Eldering | Communication, technical skills, problem solving, Python, Java, C | currently taking machine learning, artificial intelligence and databases. | elderingn@vcu.edu<br><br>703-935-6689 |
| Connor Kohout | C, Java, Python, sql | currently in ML | kohoutck@vcu.edu<br><br>703-508-6386 |

| ***Other Stakeholders*** | ***Notes*** | ***Contact Info*** |
|---|---|---|
| Tamer Nadeem | Faculty advisor. | tnadeem@vcu.edu |
| Nibir Dhar | Contact from project sponsor. | dharnk@vcu.edu |

**Step 2: Team Culture. Clarify the Group's Purpose and Culture Goals.**

**Task:** Discuss how each team member wants to be treated to encourage them to make valuable contributions to the group and how each team member would like to feel recognized for their efforts. Discuss how the team will foster an environment where each team member feels they are accountable for their actions and the way they contribute to the project. These are your Culture Goals (left column). How do the students demonstrate these culture goals? These are your Actions (middle column). Finally, how do students deviate from the team's culture goals? What are ways that other team members can notice when that culture goal is no longer being honored in team dynamics? These are your Warning Signs (right column).

**Resources:** More information and an example Team Culture can be found in the Biodesign Student Guide "Intentional Teamwork" page (webpage | PDF)

| ***Culture Goals*** | ***Actions*** | ***Warning Signs*** |
|---|---|---|
| Getting to each meeting on time. | - Create meetings with everyone's schedules in mind<br>- Communicate in the Discord, posting reminders before each meeting | - If a student misses a meeting, they receive a warning<br>- If a student continues to miss meetings, the issue will be brought to the faculty advisor |
| Making everyone aware of any delays in the schedule | - Keep each other informed about each person's portion of the project<br>- Create achievable benchmarks, and communicate when they cannot be met | - Student has not contributed what they communicated they would in the weekly meeting |
| Keep work well documented | - maintain clean well documented code<br>- keep work consistent with others and up to date on github | - pushes to github with no comments or explanation |
| Communication and understanding | - let others know if responsibilities are too much or not reasonable<br>- adjust responsibilities if needed | - Student seems to never complete anything |
| consistent communication with sponsor | - meeting on zoom or live some other way | - skipping meetings or multitasking during meetings |

### Step 3: Time Commitments, Meeting Structure, and Communication

**Task:** Discuss the anticipated time commitments for the group project. Consider the following questions (don't answer these questions in the box below):

- What are reasonable time commitments for everyone to invest in this project?
- What other activities and commitments do group members have in their lives?
- How will we communicate with each other?
- When will we meet as a team? Where will we meet? How Often?
- Who will run the meetings? Will there be an assigned team leader or scribe? Does that position rotate or will the same person take on that role for the duration of the project?

**Required:** How often you will meet with your faculty advisor advisor, where you will meet, and how the meetings will be conducted. Who arranges these meetings?
See examples below.

| ***Meeting Participants*** | ***Frequency<br>Dates and Times / Locations*** | ***Meeting Goals<br>Responsible Party*** |
|---|---|---|
| Students Only | As Needed, On Discord Voice Channel | Update group on day-to-day challenges and accomplishments |
| Students Only | Weekly, 6pm Thursday, | Actively work on project |
| Students + Faculty advisor | Once a week via Zoom, time to be determined | Update faculty advisor and get answers to our questions |
| Project Sponsor | Twice a month via Zoom, time to be determined | Update project sponsor and make sure we are on the right track |

### Step 4: Determine Individual Roles and Responsibilities

**Task:** As part of the Capstone Team experience, each member will take on a leadership role, ***in addition to*** contributing to the overall weekly action items for the project. Some common leadership roles for Capstone projects are listed below. Other roles may be assigned with approval of your faculty advisor as

deemed fit for the project. For the entirety of the project, you should communicate progress to your advisor specifically with regard to your role.

- **Before meeting with your team**, take some time to ask yourself: what is my "natural" role in this group (strengths)? How can I use this experience to help me grow and develop more?
- **As a group,** discuss the various tasks needed for the project and role preferences. Then assign roles in the table on the next page. Try to create a team dynamic that is fair and equitable, while promoting the strengths of each member.

## Communication Leaders

**Suggested:** Assign a team member to be the primary contact for the client/sponsor. This person will schedule meetings, send updates, and ensure deliverables are met.

**Suggested:** Assign a team member to be the primary contact for faculty advisor. This person will schedule meetings, send updates, and ensure deliverables are met.

## Common Leadership Roles for Capstone

1. **Project Manager:** Manages all tasks; develops overall schedule for project; writes agendas and runs meetings; reviews and monitors individual action items; creates an environment where team members are respected, take risks and feel safe expressing their ideas.
   **Required:** On Edusourced, under the Team tab, make sure that this student is assigned the Project Manager role. This is required so that Capstone program staff can easily identify a single contact person, especially for items like Purchasing and Receiving project supplies.
2. **Logistics Manager:** coordinates all internal and external interactions; lead in establishing contact within and outside of organization, following up on communication of commitments, obtaining information for the team; documents meeting minutes; manages facility and resource usage.
3. **Financial Manager:** researches/benchmarks technical purchases and acquisitions; conducts pricing analysis and budget justifications on proposed purchases; carries out team purchase requests; monitors team budget.
4. **Systems Engineer:** analyzes Client initial design specification and leads establishment of product specifications; monitors, coordinates and manages integration of sub-systems in the prototype; develops and recommends system architecture and manages product interfaces.
5. **Test Engineer:** oversees experimental design, test plan, procedures and data analysis; acquires data acquisition equipment and any necessary software; establishes test protocols and schedules; oversees statistical analysis of results; leads presentation of experimental finding and resulting recommendations.
6. **Manufacturing Engineer:** coordinates all fabrication required to meet final prototype requirements; oversees that all engineering drawings meet the requirements of machine shop or vendor; reviews designs to ensure design for manufacturing; determines realistic timing for fabrication and quality; develops schedule for all manufacturing.

| *Team Member* | *Role(s)* | *Responsibilities* |
|---|---|---|

| | | |
|---|---|---|
| Nathan DeVore | System Engineer<br>Test Engineer | - Keep track of all project specifications and make sure that all aspects of the prototype meet requirements<br>- Test prototype to ensure quality |
| Nate Eldering | Project Manager | - Keep everyone working on a timely consistent schedule<br>- Ensure everyone feels safe and open to share ideas |
| Allen Lee | Logistics Manager | - Communicate with members of the project to coordinate meeting times<br>- Acquiring information the team requires to continue the project |
| Connor Kohout | Manufacturing engineer | - Ensure that the product is made to task |

**Step 5: Agree to the above team contract**

*Team Member:* *Signature:* *Nathan DeVore*

*Team Member:* *Signature:* *Allen Lee*

*Team Member:* *Signature:* *Nate Eldering*

*Team Member:* *Signature:* *Connor Kohout*

## Appendix 3: [Insert Appendix Title]

Note that additional appendices may be added as needed. Appendices are used for supplementary material considered or used in the design process but not necessary for understanding the fundamental design or results. Lengthy mathematical derivations, ancillary results (e.g. data sets, plots), and detailed mechanical drawings are examples of items that might be placed in an appendix. Multiple appendices may be used to delineate topics and can be labeled using letters or numbers. Each appendix should start on a new page. Reference each appendix and the information it contains in the main text of the report where appropriate.

***Note:*** Delete this page if no additional appendices are included.

## References

Provide a numbered list of all references in order of appearance using APA citation format. The reference page should begin on a new page as shown here.


[1] VCU Writing Center. (2021, September 8). *APA Citation: A guide to formatting in APA style.* Retrieved September 2, 2024. https://writing.vcu.edu/student-resources/apa-citations/

[2] Teach Engineering. *Engineering Design Process*. TeachEngineering.org. Retrieved September 2, 2024. https://www.teachengineering.org/populartopics/designprocess

[3] Keller, J. (2010). DARPA launches RATS program for advanced speech-recognition algorithms in noisy conditions. *Military & Aerospace Electronics, 21*(4), 10. Retrieved from http://proxy.library.vcu.edu/login?url=https://www.proquest.com/trade-journals/darpa-launches-rats-program-advanced-speech/docview/338452109/se-2

[4] Dudam, R. (2023, May 26). Testing OpenAI Whisper with different accents - Qxf2 BLOG. Retrieved October 18, 2024, from https://qxf2.com/blog/testing-openai-whisper-with-different-accents/

[5] Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., & Sutskever, I. (2022). *Robust speech recognition via large-scale weak supervision*. arXiv. https://arxiv.org/abs/2212.04356

[6] Thomas, S., Saon, G., Van Segbroeck, M., & Narayanan, S. S. (2015). Improvements to the IBM speech activity detection system for the DARPA RATS program. *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4500–4504. https://doi.org/10.1109/ICASSP.2015.7178822

[7] OpenAI. (2022, September 21). *ASR summary of model architecture* [Image]. Retrieved October 18, 2024, from https://images.ctfassets.net/kftzwdyauwt9/d9c13138-366f-49d3-a1a563abddc1/8acfb590df46923b021026207ff1a438/asr-summary-of-model-architecture-desktop.svg?w=3840&q=90