5-2023

ACAT 2.0: An AI Transformer-Based Approach to Predictive Speech Generation

Kairan Quazi

SANTA CLARA UNIVERSITY

Department of Computer Science and Engineering

I HEREBY RECOMMEND THAT THE THESIS PREPARED
UNDER MY SUPERVISION BY

Kairan Quazi

ENTITLED

ACAT 2.0: AN AI TRANSFORMER-BASED APPROACH TO PREDICTIVE SPEECH GENERATION

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

BACHELOR OF SCIENCE
IN
COMPUTER SCIENCE AND ENGINEERING

N. Ling

Thesis Advisor

Thesis Advisor

N. Ling

Department Chair

ACAT 2.0: AN AI TRANSFORMER-BASED APPROACH TO PREDICTIVE SPEECH GENERATION

By

Kairan Quazi

SENIOR DESIGN PROJECT REPORT
(Project Sponsor: Intel Labs)

Submitted to
the Department of Computer Science and Engineering

of

SANTA CLARA UNIVERSITY

in Partial Fulfillment of the Requirements
for the degree of
Bachelor of Science in Computer Science and Engineering

Santa Clara, California

2023

ACAT 2.0: An AI Transformer-Based Approach to Predictive Speech Generation

Department of Computer Science and Engineering
Santa Clara University
2023

ABSTRACT

While constituting a rare family of diseases that afflicts 268,000 people worldwide, motor neuron diseases carry a high fatality rate with one-third of people dying within a year of diagnosis and 50% of people dying within two years (MND Association, 2022). MNDs rapidly and progressively impair muscle movement, making everyday activities like walking, chewing, and speaking almost impossible. In collaboration with famed physicist Dr. Stephen Hawking, Intel Labs developed an assistive communications platform known as ACAT to simulate speech and facilitate electronic tasks. However, the original ACAT can be slow to use, leading to awkward pauses in conversations. This paper presents a solution through a machine learning pipeline that listens in on conversations and generates full sentence responses that more accurately simulates human speech in real time. Our pipeline consists of three main phases: (1) voice activity detection; (ii) diarization; and (iii) response generation. A significant benefit of this technique is that it allows users a flexible substitution of components. Our results show that this speech generation method can significantly improve conversational flow, partly by adapting to user feedback to create more accurate results. To increase efficacy further, we plan to implement additional steps that incorporate fine-tuning the voice activity detector and diarizer models, enhancing and integrating our GUI into ACAT, and upgrading the response generator.

Keywords: ACAT, voice activity detector, language modelling, machine learning, transformer models, MND, response generation, diarization, NLP

TABLE OF CONTENTS

Revision History

Name	**Date**	**Reason for Changes**
Initial (V1)	2/21/23	**First Version Submitted to Advisors**
V2	3/3/23	**Added external requirements to Section 2.2 Solution Architecture**
V3	6/5	**Advisor feedback**

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

The latest figures from a 2019 Global Burden of Disease, Injuries and Risk Factor study place the global population of people suffering from motor neuron disease (MND) at 268,000 (Park, 2022). MND refers to a rare group of neurodegenerative diseases that impair muscle movement. The symptoms of MND begin as cramps and muscle twitches but rapidly progress to respiratory changes, cognitive changes, slurred speech, difficulty swallowing food, and a muscle wasting that debilitates the use of upper and lower limbs (Better Health Channel, n.d.). While rare, MND carries a high fatality rate with one-third of people dying within a year of diagnosis and 50% of people dying within two years. Only 10% of those afflicted will live for more than 10 years (MND Association, 2022). Although there is no cure for MND yet and the causes of the disease remain unknown, big technology companies like Intel Labs are spearheading efforts to help sufferers of MND simulate speech so they can have more agency over their daily tasks.

The most common and well known MND is amyotrophic lateral sclerosis (ALS), a disease that entered the public consciousness through its most famous face, Dr. Stephen Hawking. In partnership with Dr. Hawking in the early 2000s, Intel Labs developed a revolutionary communications technology known as the Assistive Context-Aware Toolkit (ACAT), a program that used a menu layout to speak and perform different computer tasks, such as writing emails. The most iconic images and interviews of Dr. Hawking portray him using the original ACAT.

Our team at Intel Labs has been working on the next generation development of ACAT where sophisticated machine learning and natural language processing tools can more naturally and spontaneously predict conversational responses. This senior design capstone project represents the first phase of a two-phased sub-project within the larger ongoing project. The capstone project involves the design of a pipeline that will predict and generate full-sentence responses. The project pipeline is composed of three phases: (1) detect and record human speech; (2) diarize and transcribe speech; and (3) generate predictive sentences. The second phase, which includes the tuning and optimization of the transformer-based response generator, is expected to be complete in early-to-mid summer 2023 and will not be included in the senior design capstone presentation.

The progression of this project could be understood through three primary components: (1) conception; (2) elaboration; and (3) construction.

The conception phase is represented by our problem statement above, as well as our user scenarios and solution architecture. In this phase, we first identified a specific issue within a problem domain that poses a significant scientific and moral challenge. We then ascertained the use cases and scenarios for our project. Finally, we used these scenarios to design a high-level architecture that would provide the technical vision and framework for implementation.

The elaboration phase is represented by our system prototype, as well as functional and non-functional requirements. In this phase, we explicitly defined the requirements that our implementation should meet to better visualize our final solution. We then built a prototype to understand the technical challenges we would need to solve when implementing our final design.

Lastly, the construction phase is represented by our implementation and results. In this phase, we applied the completed tasks from the conception and elaboration phases to determine the specific technologies that would be appropriate for use. We then implemented our final response generation pipeline. Following the completion of the pipeline, we benchmarked both the individual components and the combined system to determine quantitative metrics such as accuracy as well as qualitive outcomes such as similarity to human speech.

Chapter 2 Conception: Solution Architecture

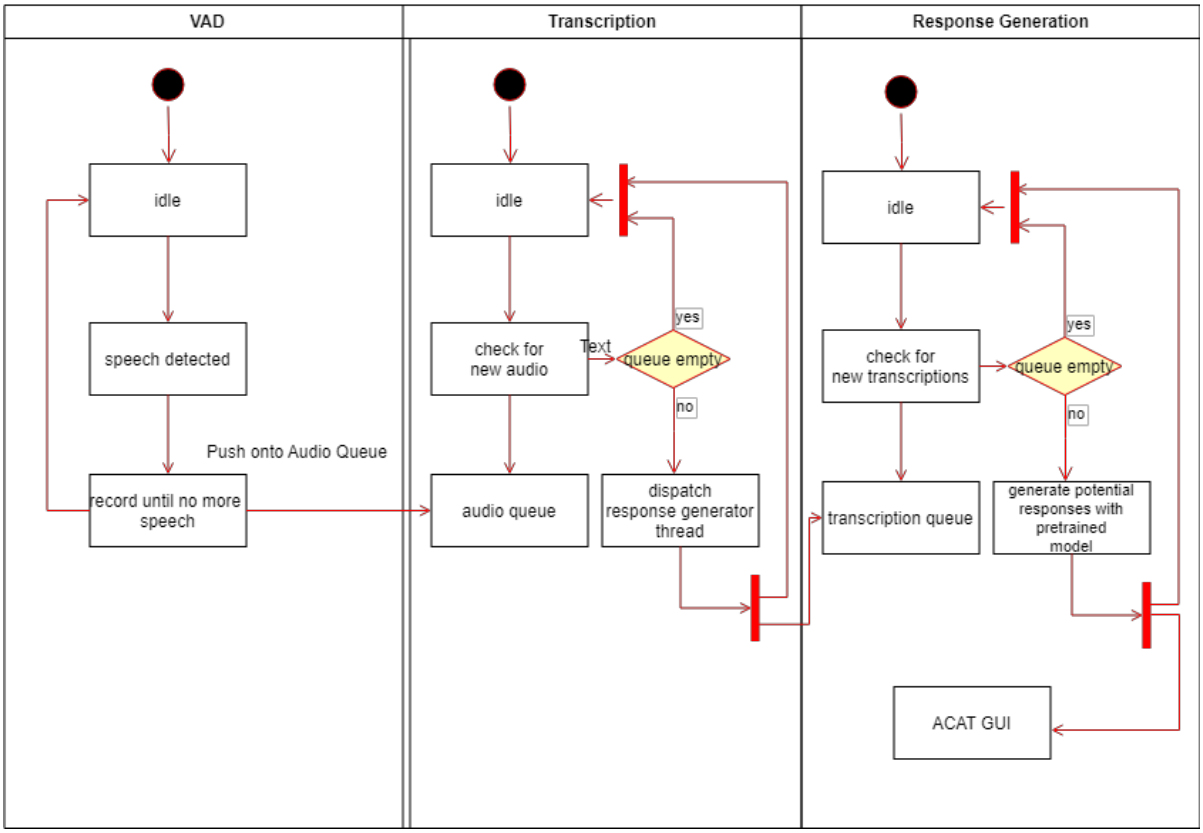
After identifying the problem domain as described in the Chapter 1 Introduction, the next logical step in the project's conception phase was to develop a robust solution architecture that would provide an overarching technical vision for the project, including a framework for implementation.

2.1 Scenarios

To design our solution architecture, we first imagined the operational scenarios for the system. Operational scenarios represent a visualization of our system's requirements. We devised scenarios in which our program, once activated, would run idly until it detected speech. Once speech was detected, the program would record the entire conversation until the conversation ends. Following the recording of the conversation, the program would diarize the audio and feed the transcript into the response generator. The responses would then be transmitted to the ACAT GUI to permit the user to edit and send the message outputs.

Figure 1

UML Diagram



2.2 Solution Architecture

After constructing the use cases for our system, we devised a solution that utilizes a three-stage pipeline to generate responses. Since our project is part of the larger ACAT ecosystem, we needed to satisfy certain external constraints related to our system structure. To satisfy these requirements, we chose to split the solution into a three-stage pipeline where each stage could be solved with independent machine learning models. This allowed us to hot-swap individual components without affecting the larger system.

The first stage of our pipeline commences with the Voice Activity Detector (VAD) that records fixed-sized blocks of audio to determine if there is human speech. Upon speech detection, the VAD saves these audio blocks and continues to do so until human speech is no longer detected for a fixed period. The second stage involves the activation of a system known as a diarizer, which performs the dual roles of transcription and speaker labeling. The diarizer

system allows our pipeline to handle multiple discrete speakers in a single conversation, as opposed to a single transcription engine that can only process a single speaker. The diarizer, therefore, facilitates a more accurate processing of dialogue and natural conversations. The third stage culminates in the generation of full sentence responses. After feeding the transcription and user data into the response generator, the system delivers customized responses that integrates both training data and previous user data. This data is then sent to the frontend through a TCP socket connection using a custom protocol.

With the conception phase complete with an imagined architecture, our project progressed to the elaboration phase where we continued to plan for project risks and constraints.

Chapter 3 Elaboration: Design Constraints

After developing an imagined architecture, our project proceeded to the elaboration phase where we defined the functional and non-functional requirements in preparation for final implementation. We also developed target metrics and a testing plan for the individual components and the full pipeline.

3.1 Functional Requirements and Testing Plan

Functional requirements of a system represent quantitative targets that the final implementation must meet. Since this project requires specific accuracy metrics, we have merged the testing plan within functional requirements to avoid redundancy. We present below five primary functional requirements for our final system.

1. For the Voice Activity Detector, we are aiming for our F1 score and accuracy metrics to be at least 90% over the TIMIT and ESC50 datasets.
2. For the speech-to-text engine, we are aiming for the average cosine distance between the SentenceBert embeddings of the true sentence and the generated sentence (using the TIMIT dataset) to be over 90%.
3. For the combined pipeline of the VAD and S2T, our accuracy goal is 90%. We will measure accuracy with the following process on both TIMIT and ESC50:
    a. If the VAD determines that there is no speech and the audio does not include speech, the accuracy will be set to one.
    b. If the VAD determines that there is no speech, but the audio does include speech, the accuracy will be set to zero.
    c. If the VAD determines that there is speech, but the audio does not include speech, the accuracy will also be set to zero.
    d. If this is not true, the accuracy will be set to the SentenceBert embeddings of the true sentence and the generated sentence.
4. For the response generator, we will use the Hits@1 and F1 metrics for evaluation on the Persona Chat dataset. For both metrics, our goal is 90%. Internally, the model uses two metrics on the Persona Chat dataset:

    a. Language Modelling: We will apply a cross-entropy loss between the ground response and the generated response.
    b. Next-Sentence Prediction: We pass the last token through a linear layer to get a score and apply a cross-entropy loss to correctly classify the true response from random distractors
5. Our VAD should be able to check the sound in under 0.1 seconds to minimize the gap between audio clips.

3.2 Non-Functional Requirements

Non-functional requirements of a system represent qualitative attributes that the final implementation should incorporate. We present below four secondary non-functional requirements for our final system.

1. Our application should be secure:
    a. It should not leak the user's data to untrusted services
2. Our application should feel smooth:
    a. There should not be a significant delay between the conversion ending and the potential responses being served to the user
3. Our application should run on cost-effective hardware:
    a. To be widely accessible, our application should not require a dedicated GPU to perform inference on the different pretrained models.
4. Our application should be accurate:
    a. In addition to the quantitative requirements listed previously, our application's results should feel natural and "human-like"

After elaborating the explicit requirements of our system, including a detailed testing plan, our project entered the construction phase where we commenced work on the final implementation.

Chapter 4 Construction: Implementation and Technologies & Standards Used

The construction phase of our project was focused on determining the technologies that would be deployed and implementing our solution.

4.1 Implementation

The implementation of our proposed architecture required that we test several techniques and model architectures for each of the three stages.

For the first stage, the VAD, we considered three options. The first option we considered was a simple sound threshold. However, this option required frequent tuning and yielded inaccurate results even under ideal conditions. The second option was a relative threshold system that superseded the first option. This iteration would measure the power in the human speech band and compare it to the total auditory power. If the ratio between the two power values is greater than a pre-set threshold, the system would mark speech as having been detected. However, this system also proved ineffective under conditions with environmental noise. Due to the flaws presented in the first two options, we decided to deploy a third option, a machine learning-based approach that could handle more diverse conditions than traditional detectors. In addition, a machine learning approach would be easier to fine-tune since we would have the option of tuning the pretrained model on a new dataset.

For the second stage, the diarizer, we considered different machine learning-based approaches. However, due to the complexity and cost of training a local model, we decided to use a remote service that handles hyperparameter optimization and the myriad steps needed for an accurate diarization engine that can handle different environments. After testing multiple cloud providers, Google Cloud's diarization engine reported the best results over a variety of datasets. Google Cloud's API is also able to produce the top-n results, which is useful for tuning the next stage and providing more accurate services.

For the final stage, the response generator, we used an in-house model based on the architecture of GPT-2. We performed multiple steps to transform the architecture into our

desired model. Specifically, we overlayed persona embeddings over the context string and used transfer learning to optimize our pretrained model. We will continue to work on finetuning this model beyond the Senior Design scope.

Following the completion of the three core stages, we developed a custom protocol to communicate with the front-end application. This protocol communicates the predicted responses as well as telemetry information about the selected or modified responses. We created two packets to transmit this information as described in Table [__] and Table [__] below.

Table 1

Packet: Responses Generated (Backend to Frontend)

Name	Size
Packet ID	4 bytes
Number of Responses	4 bytes
Response 1 Length	4 bytes
Response 1 (ASCII encoded)	Response 1 Length bytes
Response 2 Length	4 bytes
Response 2	Response 2 Length bytes
…	

Table 2

Packet: Responses Selected (Frontend to Backend)

Name	Size
Packet ID (same as the Packet ID it is responding to)	4 bytes
Selected Response ID (from zero)	4 bytes
Original Response Length	4 bytes
Original Response (ASCII encoded)	Original Response Length bytes
Sent Response Length	4 bytes
Sent Response (ASCII encoded)	Sent Response Length bytes
Unmodified Response	1 byte (0x00 or 0x01)

To test our custom protocol, we developed a proof-of-concept Terminal User Interface (TUI) as a front-end for our system. The following figures show the different aspects of the GUI that the user would face:

Figure 2

Main Page

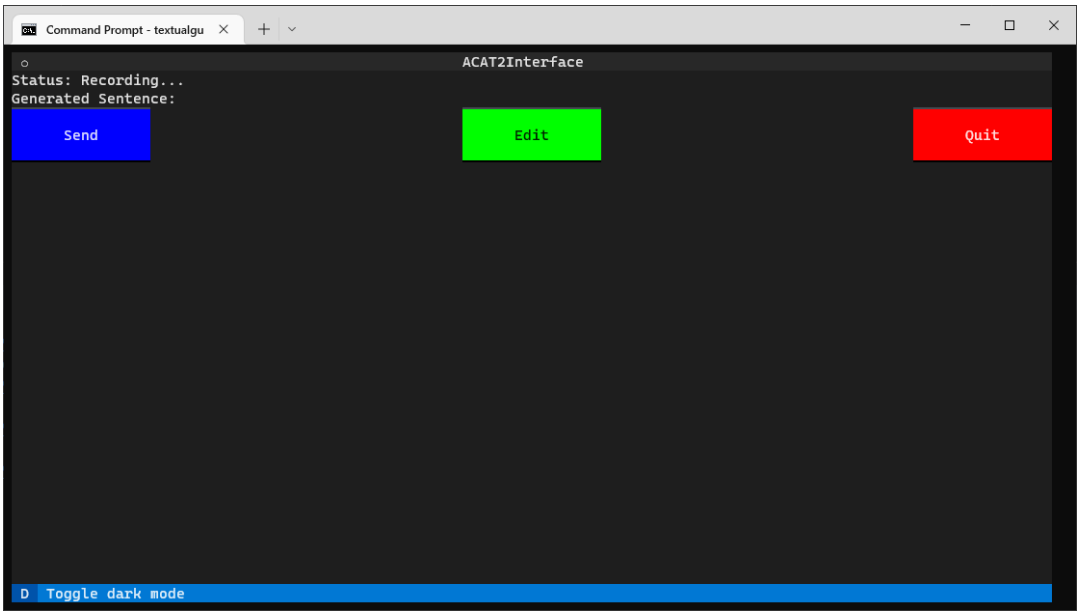


Figure 3

Between Stage 1 and Stage 2

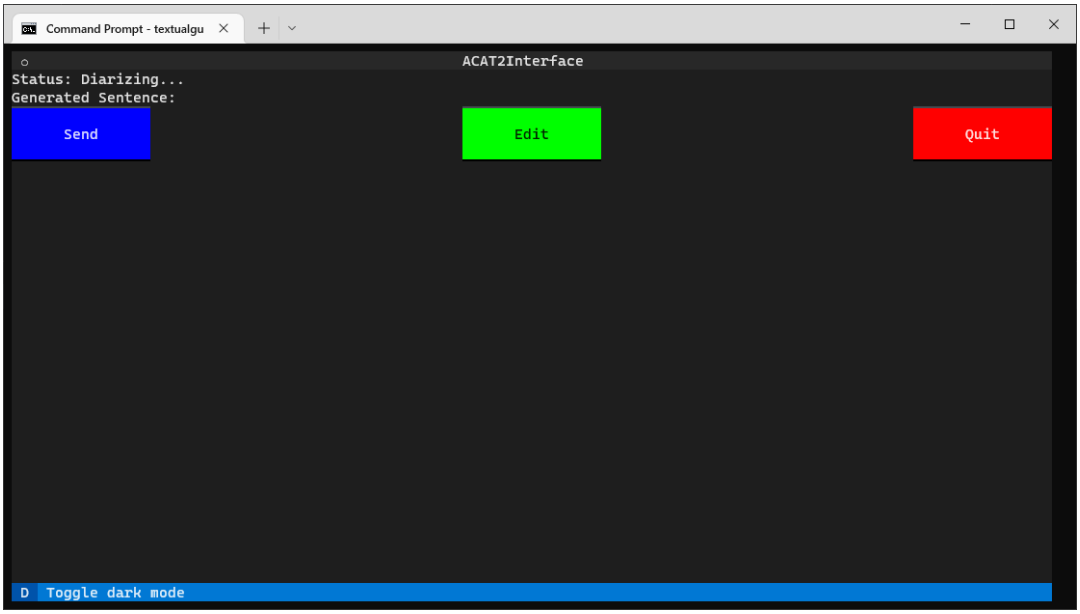


Figure 4

Between Stage 2 and Stage 3

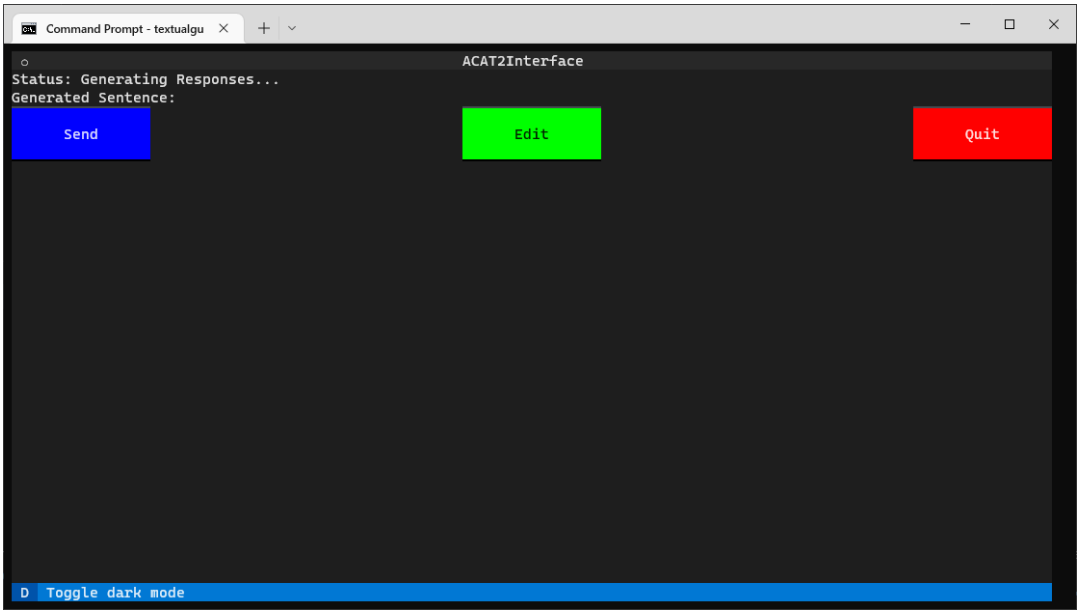


Figure 5

Showcasing Generated Sentence

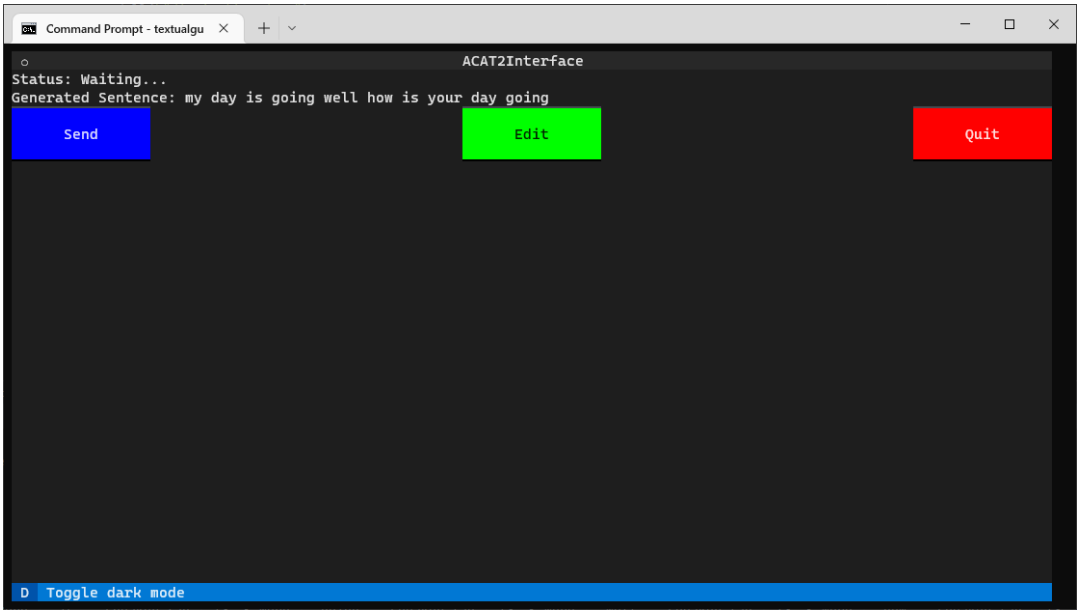
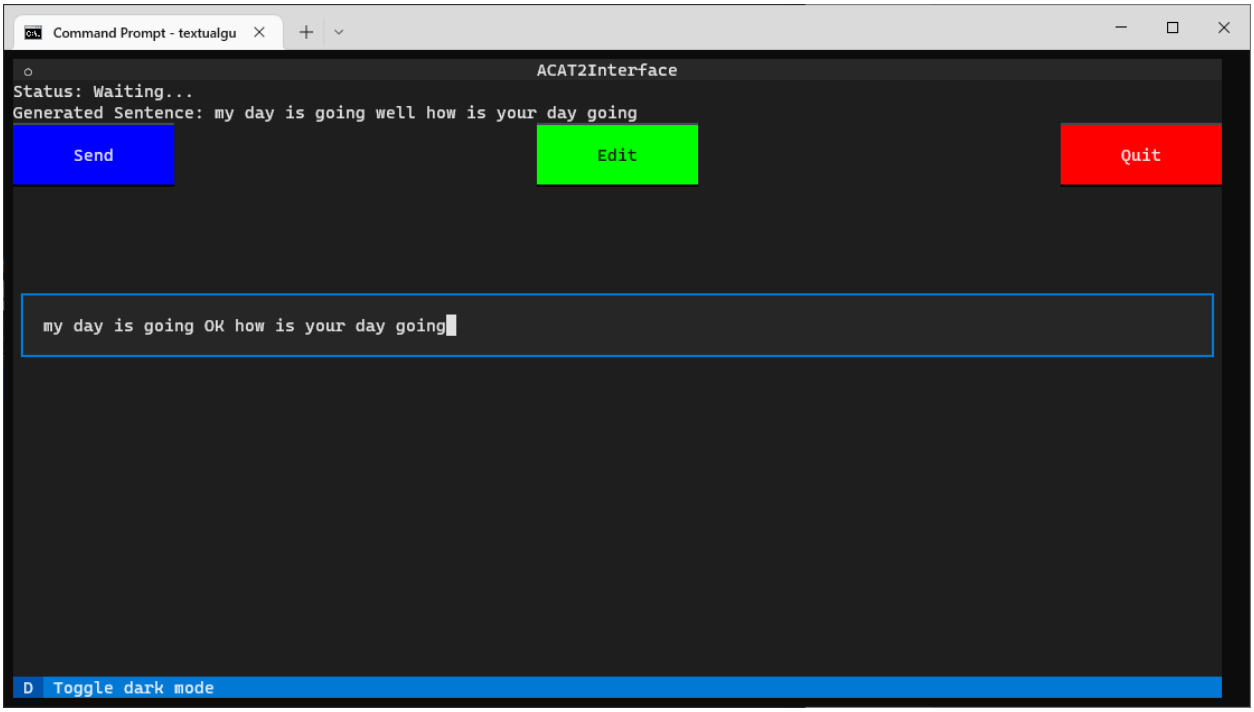


Figure 6

Editing Sentence



4.2 Technologies Used

There are several tools and technologies that this project utilized for development. The project implementation heavily relies on third-party libraries for complex tasks, including machine learning training and evaluation.

Table 3

Technologies Used

Name	Usage
Python	The programming language that was used for the pipeline
Python STL	Python's standard library, used for running shell commands, managing the file system, and creating TCP sockets.
SpeechBrain	A PyTorch-based speech toolkit, used for the pretrained Voice Activity Detector
PyTorch + Ignite	A machine learning framework, used for storing tensors on the GPU and training the response generator.
Google Cloud SDK	The Python interface for interacting with Google Cloud's APIs, used for interacting with the Google diarization tool
Hugging Face Transformers	A library for using pretrained models, used for retrieving GPT and GPT-2 as a base for the response generator

4.3 Results

The results below, corresponding to the earlier discussion in Chapter 3, demonstrate the effectiveness of both the individual components, as well as the overall pipeline. While the VAD time results are not ideal, we were able to work around the time limitation with a multiprocessing queue system.

Table 4

Results

Test Name	Test Result	Minimum Threshold
VAD F1	0.9893	≥0.9
VAD Accuracy	0.9830	≥0.9
Diarization Cosine Distance	0.9291	≥0.9
Combined Pipeline Accuracy	0.9332	≥0.9
Response Generation Hits@1	0.91	≥0.9
Response Generation F1	0.90	≥0.9
VAD time (seconds)	0.44	≤0.1

Chapter 5 Value Compass

When determining project funding, a corporation like Intel uses its value compass to weigh near-term business needs against longer-term investments and broader societal contributions. With a rare disease like MND that afflicts a relatively small population, Intel has had to balance its numerous transactional, instrumental, and intrinsic values in undertaking a project of this magnitude. These values can be observed through a decision matrix that considers an ethical approach, co-creation of value, manufacturability, usability, sustainability, and user data privacy.

5.1 Ethical Approach

Utilitarian ethics offers one potential lens through which we can determine which projects to fund. Utilitarianism's primary philosophy is to provide "the greatest good to the greatest number". In many cases, utilitarianism provides a strong foundation for the ethical allocation of resources. But in the case of rare diseases that afflict a small percentage of people, and therefore do not meet the criteria for the "greatest number", utilitarian ethics will resign to death many people, and in the case of MNDs, that number would be a quarter of a million people. On a human level, such an approach can seem to lack compassion. But on a progress level, such an approach might even be short-sighted from the standpoint of what can be learned from studying rare diseases.

In these cases, companies like Intel Labs are better served by embracing virtue ethics in the vein of Aristotle. Aristotle's virtue ethics dictates the pursuit of the common good through sustained, long-term collective action. A justice-oriented ethics would dictate that the lives of those suffering from MNDs should be given equal consideration in medical and technological resources.

5.2 Co-Creation of Value

In funding this project, Intel has had to connect its internal virtue ethics to rational business decisions and long-term profitability. One way Intel has executed on its value

compass is by releasing ACAT as open-source software. This allows the co-creation of value by inviting diverse perspectives through collaboration of experts across disciplines. By choosing an open-source approach, Intel recognized the transformative solutions that can be achieved through collective knowledge and action. But Intel also made a long-term bet that the research and insights gained from this project can be scaled to domains beyond the original scope and be more directly applicable to Intel's profit motive. For example, the research and development that has gone into ACAT can be applied to the development and refinement of Large Language Models and Conversational AI. As underscored by the media coverage of OpenAI's ChatGPT that was released in December 2022, the tech industry expects these technologies to revolutionize the way information is processed and understood.

5.3 Manufacturability

Because ACAT is a fully software-based toolkit, it is infinitely reproducible, thus making it easy to manufacture. In addition, due to Intel making the software open source, any user can download ACAT and add extensions at no charge. Intel's cost comprises of maintaining a team of engineers and scientists to research and develop the latest methodologies and applications. Separately, a user's cost comprises of paying for Google Cloud's diarization engine in the response generation pipeline. However, we are working to develop a local model that would replace this component and become free to the user.

5.4 Usability

ACAT's model is designed around ease of use for people with MNDs. It relies on a cursor moving between different options and a single trigger to select the current object. This framework is highly extensible as it allows new sensors to be added with ease. ACAT was originally designed for Dr. Hawking's cheek senor but could also operate with a simple push button. It is this framework that allows ACAT to be applied to a class of diseases that express themselves differently in each patient.

5.5 Sustainability

Training large machine learning models has a notoriously large carbon footprint. However, our project minimizes emissions in two ways. First, the response generator is made with transformer blocks, which can be trained in parallel. This minimizes the time spent training and the power used in contrast with many other machine learning advancements. Second, we often opt to fine-tuning a pretrained model rather than training a completely new model and architecture. This minimizes the epochs spent training, increasing the turnaround time, and decreasing emissions. These factors allow us to minimize our emissions while not sacrificing accuracy.

5.6 User Data Privacy

When designing our system, we soon realized that ensuring user privacy needed to be a major priority. Due to the always listening model, we realized that our pipeline could listen to sensitive conversations. In considering potential solutions, we imagined that any data we moved off the local machine would be publicly visible. Because of this, we used a local model to detect speech and record audio. Our next concern was handling informed consent. Initially, we chose to only activate the diarization phase after we had prompted the user for explicit consent. However, after discussing this feature with users, we realized that it was cumbersome, especially since the express goal of our project was reducing the delay in sending responses. After this realization, we instead opted for a toggle system where the default behavior was to not activate our pipeline. This allowed us to protect the user's privacy while still providing ease of use.

Chapter 6 Lessons Learned

While working on this project, we learned many important lessons. One of these lessons was to ensure that we avoid overcomplicating projects with residual code. During the design of the initial VAD, we noticed that the system was too slow to allow for synchronous operation. This forced us to use multiprocessing and asynchronous functions to achieve smoother audio. However, we later upgraded the VAD stage to increase speed, allowing our system to run synchronously. However, we never removed that original multiprocessing code. This led to a compounded code complexity that further slowed down development by forcing us to work around the multiprocessing code. It was only after a major refractor that we realized that our asynchronous code was unnecessary, after which we removed the inefficient code. In future steps, we became more rigorous about checking for unnecessary code especially after major changes in components or techniques.

We also learned a lesson related to the impact of data quality. When testing different datasets for the response generator, we found that the domain, size, and turn length of the dataset has an oversized impact on the results. Due to the complexities of training a large transformer-based model, small changes often led to inconsistent or incoherent outputs from the model.

The final lesson we learned related to stakeholder interaction. Initially, we designed our pipeline to require affirmative consent before processing the data as a solution to privacy concerns. However, we failed to discuss this feature with users until the end of the Fall quarter. After acquiring feedback from users, we implemented significant changes to the privacy settings and architecture regarding our custom protocol for communication between the front- and back-end.

Chapter 7 Next Steps

As ACAT 2.0 is a complex initiative with implications beyond Intel's immediate scope, the project will continue to implement changes and improvements that make the system more versatile for a broader population of users. We envision three key next steps for this project:

1. VAD and Diarizer

We should consider either finetuning an existing model or developing an in-house model to implement the VAD and diarization phases. If we were to move these models in house, we could consider building a single model that directly implements all three stages, potentially leading to increased accuracy and system cohesion. However, this would require a completely new type of dataset that may not be feasible to collect and process.

2. GUI and UX Enhancements

Currently, our GUI is separate from the ACAT infrastructure. We should consider building this functionality as an extension of ACAT for a better user experience. In addition, we should consider refining the UI to use the standard ACAT theme.

3. Response Generation

We will continue to tune and improve the response generation model. We are considering several methods to improve the accuracy and experience of the generator:

- Training on diarized text to account for transcription errors
- Using the top-n diarized results and their associated probabilities to lessen the effect of transcription errors
- Investigating different searching methods including greedy decoding, beam search, and top-k/top-p sampling.

References

Motor neurone disease (MND). (n.d.). Better Health Channel. Retrieved September 24, 2022 from https://www.betterhealth.vic.gov.au/health/conditionsandtreatments/motor-neurone-disease#symptoms-of-motor-neurone-disease-mnd

Park, J., Kim, J., & Song, T. (2022, April 21). *The Global Burden of Motor Neuron Disease: An Analysis of the 2019 Global Burden of Disease Study.* Frontiers in Neurology. https://www.frontiersin.org/articles/10.3389/fneur.2022.864339/full

What is MND? (Updated 2022, June 29). MND Association. https://www.mndassociation.org/about-mnd/what-is-mnd/

Appendix A: Abbreviations and Field-Specific Terminology

Phrase/Shorthand	Meaning
Cross Entropy	A common loss function for ML models
ML	Abbreviation for “Machine Learning”
S2T/Diarization	Speech-to-Text
SentenceBert/BERT/embeddings	A method of converting sentences and words into a list of vectors which represent an index in a large table, where elements close to each other represent synonyms.
TIMIT/ESC50	Datasets for training different audio platforms
VAD	Voice Activity Detector, a system to detect human speech in audio