English-to-IPA Transcription

Riley Roberts
Utah State University

Follow this and additional works at: https://digitalcommons.usu.edu/honors
Part of the Computer Engineering Commons

Recommended Citation

This Thesis is brought to you for free and open access by the Honors Program at DigitalCommons@USU. It has been accepted for inclusion in Undergraduate Honors Capstone Projects by an authorized administrator of DigitalCommons@USU. For more information, please contact digitalcommons@usu.edu.
English-to-IPA Transcription

by

Riley Roberts

Capstone submitted in partial fulfillment of the requirements for graduation with

University Honors

with a major in

Computer Engineering

in the Department of Electrical and Computer Engineering

Approved:

Capstone Mentor
Dr. Don Cripps

Departmental Honors Advisor
Dr. Todd Moon

University Honors Program Executive Director
Dr. Kristine Miller

UTAH STATE UNIVERSITY
Logan, UT

Fall 2021
Abstract

The purpose of this project was to create a tool that could automate English-to-IPA (International Phonetic Alphabet) transcription. Research was done to determine what would be required to perform such a transcription. After researching and experimenting with existing tools, it was determined that developing the signal processing and Artificial Intelligence model portion of the application would be too intensive to successfully complete within the timeframe of this project.

The choice was made to develop an iOS application, with the Python library Allosaurus used to do the speech processing and as the Artificial Intelligence model. This model was then deployed in the cloud using AWS, and the two components, iOS and cloud, were then integrated to work together. The deployed cloud model was placed behind a secure API layer, requiring authentication for access, to ensure security and consistency of requests. The final application functions as intended and is capable of performing the desired transcriptions. The application follows conventional iOS user interface styling so-as to be conducive to easy pickup by the intended users, Speech Therapists.

The application is not yet ready for public release, due to the inability to meet some accuracy and security requirements that were originally specified. However, the researcher has plans to continue the work beyond the scope of this project and presents ideas for how to solve the current problems the application is facing. Additionally, the researcher discusses ethical considerations of the project, and supplies a User Guide with non-technical instructions for operating the application.

Acknowledgements

I’d like to thank Dr. Don Cripps and for guiding me through the process of selecting a project and helping me to understand the steps involved in designing a system. Dr. Todd Moon also helped me at the beginning of the project to point me in the proper direction of existing tools, in order to know where to start on such a large project. I’d also like to thank Jolynne Berrett for her help in the preparation of the documentation for this project. Additionally, I would like to thank my wife, Kaitlyn Roberts, for helping me to choose a project and for giving me guidance along the way from the perspective of a Speech-Language Pathologist.
# Table of Contents

1. **INTRODUCTION** ..................................................................................................................... 1  
   Objectives ................................................................................................................................................................... 1  

2. **METHODS** ............................................................................................................................. 1  
   Research ..................................................................................................................................................................... 1  
   Model ......................................................................................................................................................................... 1  
   Cloud .......................................................................................................................................................................... 2  
   User Interface ............................................................................................................................................................. 3  
   Transcription List ................................................................................................................................................... 4  
   Transcription Creation ............................................................................................................................................ 4  
   Account Management ............................................................................................................................................ 4  
   Ethical Considerations ................................................................................................................................................ 5  

3. **RESULTS** ............................................................................................................................... 5  
   Interface Requirements .............................................................................................................................................. 5  
   Functional Requirements ............................................................................................................................................ 7  
   Support Requirements ................................................................................................................................................ 8  

4. **DISCUSSION AND CONCLUSION** .............................................................................................. 8  

5. **HONORS CAPSTONE REFLECTION** ............................................................................................ 10  

6. **WORKS CITED** ............................................................................................................................... 12  

7. **APPENDIX** ............................................................................................................................................. 13  

8. **AUTHOR BIO** ............................................................................................................................................. 18
1. INTRODUCTION

The International Phonetic Alphabet (IPA) is a set of symbols that represent the basic sound components that make up speech (International Phonetic Association). Speech therapists and other professionals often transcribe speech into IPA in order to better analyze patterns in the speech. This process can be highly time consuming and difficult, so automation of the process would lighten the load of therapists significantly, allowing them to provide better therapy. The purpose of this project was to develop a tool that provides this automation.

Specifically, an iOS-based application was developed. Recently, technology has become more prevalent in the therapy setting, and using a smart-phone or tablet to assist in the sessions has become commonplace. As such, making the tool an iOS application allows it to be easily introduced into therapy without requiring any additional training to be completed by the therapist, and without requiring the purchase of any additional technology.

Objectives

1. Provide an interface for users to import audio or record new audio files
2. Convert English speech from audio files to IPA symbols with 75% accuracy
3. Organize and present recordings and transcriptions together to the user
4. Visibly mark transcribed symbols which the model predicts with less than 75% confidence

2. METHODS

Research

The field of Speech Recognition, the ability of a computer to identify and respond to the sounds produced in human speech, has seen tremendous progress in recent years, mostly due to the introduction of AI assistants such as Siri, Alexa, etc. However, most existing tools focus on transcribing the speech simply into written English, whether to provide voice-to-text services or with the intent to understand and respond to the meaning itself. However, some advanced tools attempt to convert speech into phonemes (the phonetic units that IPA represents) as a step in this process.

The first step in the speech recognition task is to perform signal processing on the waveform described by the audio file (Huang and Deng). This process converts the audio into a set of features to pass to the downstream models. These features vary depending on the system, but typically include the frequency-domain transformation of the signal. After this feature extraction, an acoustic model is then used to decode the features into acoustics, such as the phonemes used by this project. In a typical speech recognition system, the acoustic model would then be followed by a language model. A language model is trained to know the structure of a particular language, such as which words are likely to follow other words. The accuracy of many modern-day systems relies on the combined strength of both the acoustic and language models in tandem.

In this project, a language model is not helpful, as the application intends to directly present the decoded phonetic information. As such, the accuracy cannot be expected to compare to that of modern speech-to-text systems. In fact, due to the intended use case, transcribing disabled speech, the acoustic model used in this project should not utilize information regarding which phonemes are likely to follow others in the English language, as this may guide the model to transcribe phonemes as they would have been said if a word were said correctly, rather than how the word was actually said.

Model

It was the original intent of the project to utilize outside tools for the audio processing portions of the code but to develop the AI models that make the actual symbol predictions as part of the project. Unfortunately, the technology behind this process is extremely complicated and developing them turned out to be beyond the scope of a one-year senior project. However, there are existing tools that were able to be leveraged for this project. Specifically, the
Python library Allosaurus was identified. Allosaurus provides the capability to use pre-trained artificial intelligence models to predict sequences of IPA symbols from sound files in the WAV format, helping to achieve Objective 2 (with the accuracy to be determined in the Results section). Additionally, Allosaurus provides the model confidence metric for each predicted symbol, which is needed by the application to complete Objective 4.

Cloud

Python code cannot be run on an iOS device, and even if it could, the Allosaurus models require more compute power and RAM than most older iOS devices can provide. As such, in order to leverage these models, they were deployed in the cloud using Amazon Web Services (AWS). AWS was chosen over other cloud platforms for a variety of reasons, as shown in the decision matrix in Figure 1. In the end, although Google Cloud Platform and AWS scored relatively closely, AWS is more prevalent in industry, and learning to use it is currently a more valuable skill, so it was selected.

<table>
<thead>
<tr>
<th>Criteria (Weight 0 - 1)</th>
<th>Cost (1.0)</th>
<th>Ease of Development (0.5)</th>
<th>Built-In Login Service (0.5)</th>
<th>Compatibility with iOS (0.7)</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Web Services (AWS)</td>
<td>(5) 5.0</td>
<td>(4) 2.0</td>
<td>(5) 2.5</td>
<td>(4) 2.8</td>
<td>12.3</td>
</tr>
<tr>
<td>Google Cloud Platform</td>
<td>(3) 3.0</td>
<td>(5) 2.5</td>
<td>(4) 2.0</td>
<td>(4) 2.8</td>
<td>10.3</td>
</tr>
<tr>
<td>Microsoft Azure</td>
<td>(2) 2.0</td>
<td>(3) 1.5</td>
<td>(4) 2.0</td>
<td>(4) 2.8</td>
<td>8.3</td>
</tr>
</tbody>
</table>

Figure 1. Cloud Platform Decision Matrix

Specifically, the models were setup using AWS Lambda, a tool that allows for python code to be run without the need for a server to be setup. Instead, resources are allocated to the function only when it needs to run. Using Lambda comes with many advantages: it is much more cost-efficient to allocate resources only as needed, Lambda can run Docker images containing all of the setup and libraries needed for a particular script, and Lambda eliminates the need to manage a server full-time.

Because Allosaurus is a library that is not included with the standard Python installation, Docker was used to package all the dependencies together with the script to be run on Lambda. Docker is a tool that bundles code and its dependencies together in what is called a “container”. Each container is a virtual machine that can run on any machine with docker installed, regardless of that machines Operating System or available libraries. This allows code to be quickly and easily run on new machines with little to no additional setup.

Originally, the design called for the audio files to be sent directly to the Lambda function, with the transcriptions returned in one clean transaction. Unfortunately, audio files cannot be sent directly to a Lambda function, so they are instead uploaded to AWS’s storage system S3, and then a reference to their location is given to the Lambda function when it is invoked. Additionally, in order to provide uniformity, the design was altered so that the Lambda function
can only be accessed through AWS AppSync, a GraphQL Application Programming Interface (API). This helps with ensuring security, as it requires authentication to access, and also guarantees uniformity of requests to the Lambda function and the formatting of the responses. AWS was also used to manage the accounts of users of the application. Figure 2 shows how the iOS application interacts with the cloud components.

<table>
<thead>
<tr>
<th>Internet</th>
<th>Cloud Platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login Request</td>
<td>Login/Account Server</td>
</tr>
<tr>
<td>Accept/Deny Request, Send Account Info</td>
<td></td>
</tr>
<tr>
<td>Upload Recording</td>
<td></td>
</tr>
<tr>
<td>Request Transcription – Send File Location in Storage</td>
<td></td>
</tr>
<tr>
<td>Send Back Transcription</td>
<td></td>
</tr>
<tr>
<td>Application/User Interface</td>
<td>Storage Service</td>
</tr>
<tr>
<td>Completed Transcriptions</td>
<td>Transcription Model</td>
</tr>
<tr>
<td>User Interaction/Audio Recordings</td>
<td>File Name</td>
</tr>
<tr>
<td></td>
<td>Transcription</td>
</tr>
</tbody>
</table>

**Figure 2.** System diagram illustrating the interactions between the application and the cloud components

Request timeouts and memory limits turned out to be an additional challenge associated with integrating the components into the cloud. The GraphQL API has a hard-coded request timeout of 30 seconds that cannot be extended by developers. However, some longer transcriptions required more time than this to transcribe, and then the transcriptions were unable to be obtained from the cloud. Additionally, longer audio files can quickly overrun the current maximum RAM allotments to the Lambda function. More RAM can be configured, but for a higher cost per second of use. A solution to both these problems would be to split the audio file at the word or sentence level and send each word/sentence individually to be transcribed, thus eliminating the need to transcribe long audio files. This would also help with accuracy checking, as it is difficult to automatically discern the accuracy of transcriptions of long audio files, as an extraneous symbol added early on can make the entire transcription not match the transcription in the test set being used to assess accuracy.

Due to the cloud interface, requires an internet connection to perform a transcription. However, the application does not currently programmatically verify that an internet connection is available prior to requesting a transcription. This is because the application also uses the cloud resources to authenticate the user at start-up. Without this authentication, users cannot access the rest of the application and could not request a transcription. However, if a user signs-in to the application and then loses an internet connection, they may be allowed to request a transcription, which will fail, resulting in a recording that does not have a transcription. Future iterations of this work will programmatically check for an internet connection prior to allowing a transcription request. If a connection is not available, the user will still be allowed to save the audio recording for later transcription. As part of this, a functionality to request a transcription at a later time, i.e. not at audio creation or upload time, will be required.

**User Interface**

The graphical user interface (GUI) of the application is an iOS app written in Swift. The GUI provides three major navigation areas: the transcription list, the transcription creation area, and an account management area. Each area is accessible by a tabbing feature located at the bottom of the GUI. From within each tab, the user can then access
elements related to that functional area. Figure 3 illustrates the main screen for each of these tabs. See the Appendix for the User Guide, a user-friendly explanation of how to use the application and its features.

![Figure 3. The transcription list (left), new transcription page (middle), and account page (right)](image)

**Transcription List**

This major area is meant to achieve Objective 3. The transcription list screen provides a list of transcription names, along with the dates they were created, as shown above in Figure 3. The user may scroll through this list to locate a particular transcription. The list is sorted with the oldest transcription at the top. Tapping on a particular transcription activates navigation to another screen and fetches the details of that particular screen to be displayed.

Once a transcription is selected, the full IPA transcription is displayed on the page. The transcription itself is stored on the iOS device as a raw string, as obtained from the Allosaurus cloud model. It is then parsed into a more presentable format prior to being displayed to the user. At the parsing stage, the code alters the attributes of IPA symbols that Allosaurus predicted with low confidence so that they appear a different color, achieving Objective 4. This page also provides a button for the user to listen to the audio file the transcription is based on.

**Transcription Creation**

This major area is designed to meet Objective 1. It provides two primary means for creating a transcription: recording a new audio file to be transcribed or transcribing a file that already exists on the device. For the latter, the app utilizes Apple’s built-in functions for accessing the on-device file structure, and then uploads that audio to the cloud and requests a transcription from the API. In order to create a new recording, the GUI guides the user through a series of steps.

The user is first instructed to give the transcription a name. This name is not only used for distinguishing the transcriptions to the user, but also by the application for distinguishing between the different audio files it stores. Following that, the user is presented with a screen where they may interact with the GUI to start and stop recording until they are ready to save the recording, at which point it is sent to the cloud and a transcription is requested.

**Account Management**
The account management view is a simple screen that displays the username of the current user, as obtained from the AWS authentication system, along with the subscription level of the user. All users currently have a standard subscription level, giving them access to all of the application’s functionalities. However, the idea of a subscription was introduced such that the app could be marketed in the future at various different price points allowing for different use levels.

**Ethical Considerations**

The main ethical considerations in relation to this project are: 1. The use of external code, and 2. The storage of audio files. This project heavily utilizes the Allosaurus library, which is licensed under an open-source license allowing for reuse and redistribution within a new product, but only if the new product is released under the same open-source license. As such, if this application is ever released to the public, or used at all outside of my own personal use, all source code will need to be made available to the public, and the other requirements of the license met. This does not mean that the product could not be distributed commercially, only that somewhere the source code would have to be made available. At this stage, the application is not being distributed or used by anyone other than myself to avoid any possible legal issues. Following the conclusion of this project, the application may or may not continue to use Allosaurus. If it does, the licensing requirements will be followed.

The recordings on the device, which are intended to include recordings of individuals with disabled speech, must be sent to the cloud for transcription. These recordings are only momentarily stored in the cloud; they are deleted as soon as the transcription is completed. All recordings require authentication to access; however, if a malicious user were to have their own account, the knowledge of the proper transcription name, and request a transcription at the exact right time, they may be able to get a transcription of another user’s recording prior to it being deleted. While this would require excellent timing and foreknowledge, the application in its current state could not be distributed to the public ethically. Much more rigorous security requirements would need to be implemented in the cloud, along with extensive testing to ensure the security requirements could not be bypassed.

### 3. RESULTS

The application was successfully created and installed on the student’s iPhone. It performs the basic functions of recording and transcribing successfully. The speech processing and AI model was successfully deployed in the cloud and is only accessible by authorized users. The application has been tested on the student’s device and has not been observed to crash.

Due to the timeout issue discussed in Section 2, occasionally a request for transcription fails. This is particularly likely to occur on the first transcription following the opening of the application. During the timeline of this project, the student was unable to eliminate this issue. As mentioned in Section 2, dividing the audio into separate files for each word and then requesting transcriptions for each would greatly alleviate the issue. Once this is completed, the program could then be instructed to retry obtaining a transcription, up to a maximum amount of retries, to ensure that a valid transcription is eventually obtained, even if an error occurs during one of the attempts.

Prior to beginning work on the application, a set of requirements were designed to try and describe in detail what a completed project would look like. Some of these requirements did not end up being implemented at all, and others changed during the design process, but they served as a roadmap for the duration of the project. The following subsections show the results of testing for the requirements set forth for this project, along with any applicable notes. The pass/fail criteria for each requirement were determined based on either visual verification or a test, as noted in the notes.

**Interface Requirements**

Table 1 shows the results of testing the requirements related to the user interface of the application. The only failed requirement had to do with payment methods, which was not implemented at this stage because it has not yet been determined whether the product will be carried forward to a public release. Several requirements were marked as
partial, as some aspect of the requirement was met, but the project was changed in some way such that the requirement was not met in the way it originally stated. For example, accuracy could not be assessed at this point in time (see Requirement 3.2.2.6), resulting in no critical value to use for testing Requirement 3.1.2.9.

**Table 1.** Pass/Fail results for the user interface requirements.

<table>
<thead>
<tr>
<th>Requirement Number</th>
<th>Requirement</th>
<th>Pass/Fail/Partial</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.1.1</td>
<td>The application shall be compatible with either iOS or at least one web browser. The application may be compatible with both iOS and at least one web browser.</td>
<td>Pass</td>
<td>Visually verified the application runs on an iOS device.</td>
</tr>
<tr>
<td>3.1.1.2</td>
<td>The iOS version of the application shall follow the Apple Human Interface Guidelines for iOS apps, as referenced in Section 2 of the specification.</td>
<td>Partial</td>
<td>Most guidelines are met. Some guidelines regarding security, iPad screen sizes, and Dark Mode features still need work.</td>
</tr>
<tr>
<td>3.1.1.3</td>
<td>The web browser version of the application shall follow the Google Web Fundamentals guidelines as reference in Section 2 of the specification.</td>
<td>N/A</td>
<td>Only the iOS version was produced. Specification only required one.</td>
</tr>
<tr>
<td>3.1.2.1</td>
<td>The application shall open to a login/sign-up screen where the user can create an account or sign in.</td>
<td>Pass</td>
<td>Visually verified.</td>
</tr>
<tr>
<td>3.1.2.2</td>
<td>The application shall require a user login prior to allowing access to saved transcriptions or allowing the production of new transcriptions.</td>
<td>Pass</td>
<td>Manually verified that the user must login prior to access.</td>
</tr>
<tr>
<td>3.1.2.3</td>
<td>During sign-up, the application shall provide a method to collect user payment.</td>
<td>Fail</td>
<td>Payment methods were not implemented.</td>
</tr>
<tr>
<td>3.1.2.4</td>
<td>The application shall provide a method to upload audio files from the user device into the application’s memory.</td>
<td>Pass</td>
<td>Manually verified that a file could be uploaded and transcribed.</td>
</tr>
<tr>
<td>3.1.2.5</td>
<td>The application shall be capable of displaying IPA symbols to the user, matching the ASCII encoded IPA symbols defined by TIMIT (see Section 2 of the specification) produced by the application.</td>
<td>Partial</td>
<td>Displays IPA symbols, uses Allosaurus set rather than TIMIT.</td>
</tr>
<tr>
<td>3.1.2.6</td>
<td>The main page (home page) of the application shall contain a one-click pathway to the recording tool.</td>
<td>Pass</td>
<td>Manually verified.</td>
</tr>
<tr>
<td>3.1.2.7</td>
<td>Every page of the application shall provide a one-click pathway to the main/home page.</td>
<td>Pass</td>
<td>Manually verified.</td>
</tr>
<tr>
<td>3.1.2.8</td>
<td>The application shall provide a view/screen where the IPA transcription for a given recording is visible to the user.</td>
<td>Pass</td>
<td>Visually verified.</td>
</tr>
<tr>
<td>3.1.2.9</td>
<td>The application shall provide a visible indicator to the user of IPA symbols for which the prediction confidence was less than the critical value as defined by requirement 4.2.2.5.</td>
<td>Partial</td>
<td>Visual indication of low confidence is present (less than 0.7), critical value based on accuracy was not computed.</td>
</tr>
<tr>
<td>3.1.2.10</td>
<td>The application shall not allow storage of Personally Identifiable Information (PII), as defined by the Handbook for Safeguarding Sensitive PII, for anyone other than the application user.</td>
<td>Partial</td>
<td>The app does not have anywhere to save PII, however it does not prevent the user from using it as a title.</td>
</tr>
<tr>
<td>3.1.2.11</td>
<td>The application shall provide a method that does not</td>
<td>Pass</td>
<td>Title plus transcription</td>
</tr>
</tbody>
</table>
Functional Requirements

Table 2 contains the results of testing the requirements related to the function of the application. Three requirements were not met due to time constraints: MP3-WAV conversion capabilities, time constraints on transcription, and accuracy requirements, which couldn’t be met at this time due to difficulty of assessing accuracy on sentence-length (or longer) transcriptions. Future iterations will implement the former, as well as test methods for achieving the latter two. These failures limit the quality of the application, but do not prevent the application from achieving its overall objective of providing automated transcriptions. Additionally, some of the requirements mention TIMIT (Garofolo), which is a dataset that was originally planned to be used, but was not used in the final implementation.

Table 2. Pass/Fail results for the functional requirements of the application.

<table>
<thead>
<tr>
<th>Requirement Number</th>
<th>Requirement</th>
<th>Pass/Fail/Partial</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2.1.1</td>
<td>All software used for this project shall follow the Google style guide for its respective programming language, as referenced in Section 2 of the specification.</td>
<td>Partial</td>
<td>Mostly following, did not have time to go through and ensure strict compliance.</td>
</tr>
<tr>
<td>3.2.2.1</td>
<td>The application shall be capable of recording and storing audio files.</td>
<td>Pass</td>
<td>Manually verified.</td>
</tr>
<tr>
<td>3.2.2.2</td>
<td>The application shall be capable of converting MP3 files into WAV files representing the same audio content. The application may be able to convert other audio file formats into WAV files.</td>
<td>Fail</td>
<td>This functionality was not implemented, all recording are WAV files, and uploaded files must be WAV files.</td>
</tr>
<tr>
<td>3.2.2.3</td>
<td>The application shall be able to convert WAV audio files containing English speech into the matching ASCII encoded IPA symbols, as defined by the TIMIT reference in Section 2 of the specification.</td>
<td>Partial</td>
<td>Capable of conversion, uses Allosaurus IPA set instead of TIMIT.</td>
</tr>
<tr>
<td>3.2.2.4</td>
<td>The application shall take no longer than one one-hundredth of the duration of the audio file to transcribe the audio file.</td>
<td>Fail</td>
<td>A 10 second audio file took 10 seconds to transcribe. Splitting the audio on words and transcribing all words simultaneously would alleviate this issue.</td>
</tr>
<tr>
<td>3.2.2.5</td>
<td>The application shall record the prediction certainty (predicted probability) of the ASCII encoded IPA symbols it assigns being correct.</td>
<td>Pass</td>
<td>Manually verified prediction probability is stored in the code.</td>
</tr>
<tr>
<td>3.2.2.6</td>
<td>The application shall define a critical value of prediction certainty such that ASCII encoded IPA symbol predictions with certainty greater than the critical value are 90% accurate.</td>
<td>Fail</td>
<td>Accuracy was not determined on the TIMIT set due to the difficulty of determining accuracy when the model may predict a different number of symbols entirely. Accuracy can be assessed if</td>
</tr>
</tbody>
</table>
Support Requirements

Table 3 shows the results of testing the requirements related to the support aspects of the application. These requirements refer to how the application behaves within the larger system ecosystem. The application passed each of these requirements.

Table 3. Pass/Fail results for support related requirements.

<table>
<thead>
<tr>
<th>Requirement Number</th>
<th>Requirement</th>
<th>Pass/Fail/Partial</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3.1.1</td>
<td>The mobile version of the application shall be no larger than 250MB.</td>
<td>Pass</td>
<td>Manually verified app size is 23.9 MB</td>
</tr>
<tr>
<td>4.3.1.2</td>
<td>The application shall consume no more than 500MB of RAM at one time.</td>
<td>Pass</td>
<td>During simulation, the app never exceeded 25 MB of RAM usage.</td>
</tr>
</tbody>
</table>

4. DISCUSSION AND CONCLUSION

Overall, the project met the goal of producing a functioning iOS application capable of performing IPA transcriptions of audio files. The application meets most functional requirements but was unable to achieve the standards of security and accuracy prescribed by the specification document. Additionally, the project utilized a Python library for the AI portion rather than providing a custom-built model as was originally intended. However, the architecture of the application was set up so that the model is independent from the rest of the system, allowing for future development of improved AI models without any need for altering the rest of the application.

This project taught me a lot about what goes into producing a fully functioning system, not just a program. The code containing the model that performed the transcriptions already existed, but it still took a lot of effort to package that into a user-friendly application, particularly considering the complexities of deploying the model in the cloud. I learned how all of these different pieces interact to create a seamless experience for the user. This project also served as my Engineering capstone; however, I believe the extra effort required to learn two entirely new skillsets, mobile app development and cloud computing, qualify it for the honors capstone as well.

On the cloud side, I learned how to set up a serverless function that can be run at any time and uses only the resources allocated to it. I learned how to store files in the cloud programmatically, and how to access them from within the serverless function. Finally, I learned how to restrict access to the serverless function through a secure API layer, ensuring only authorized users can access it, and guaranteeing uniformity of data input to the function. The combination of these skill sets showed me how to set up runnable code in a manner that can easily scale in compute and memory as more users attempt to access it simultaneously. I think this sets me up to be able to deploy future applications in the cloud with significantly less time on learning and setup than was needed for this project.

On the application side, I had never before created an iPhone application, or even a GUI at all. Up to this point in my education, all programs I had created used only the command line or external files for input/output. It was really eye opening to see how things change when data needs to be presented to the user in a more visual fashion. I found that some steps were surprisingly simple, as the iOS toolset provided easy to use libraries to accomplish complex visual tasks, while other items that seemed straightforward required much more effort than I would have thought. I learned how to setup layouts that can adapt to different screen sizes and still look correct, how to interface with the...
iOS operating system to access hardware elements such as the microphone and speakers, and how interface the application with the external cloud portion.

Finally, I learned about the complexity of speech recognition. I originally wanted to play a more active role in the speech processing and IPA symbol prediction components of the code, but quickly learned that not only is the code incredibly complex, but there is also a lot of domain knowledge (signal processing, acoustics, phonetics, etc.) that is used in creating the algorithms and models. While I didn’t get to directly code this as I had hoped, I still learned a lot about the process through my research, and I have set myself up to continue working on this application after graduation, with the intention to eventually take a more hands-on approach to the transcription process.
5. **Honors Capstone Reflection**

My capstone project really pushed the boundaries of what I was capable of, along with rounding out my education here at Utah State University. I studied Computer Engineering and focused the majority of my coursework on the programming-heavy courses. However, throughout my courses, I have only ever developed programs that can perform a specific function, never a full product that a non-engineer would be capable of using.

For my capstone project, I developed an iOS application that can transcribe English audio recordings into the International Phonetic Alphabet (IPA), a set of symbols commonly used to describe the fundamental sounds that make up speech. This project bridged the gap between what I had done in coursework and a true final product, requiring a user-interface that anyone familiar with mobile apps would be capable of using, and requiring the integration of several different programs into a full system.

The project also served as my Engineering capstone, though I feel it meets the requirements for an Honors capstone because in addition to using the skills I had learned in my degree, I had to self-teach myself two entirely new skillsets: mobile application development and cloud development. In order for this system to work, the speech processing had to be done in the cloud, due to iOS system resources being limited. Getting this processing to run properly in the cloud and to interact with the user interface portion of the code running on the local iOS device was extremely challenging, as I had no prior experience with cloud technologies.

This project fits into the field of speech recognition, which prepares me well for my future career, as I have recently accepted a position with Apple working on the Siri team following graduation. I believe the work I did throughout this capstone will not only prepare me to succeed on that team, but also to be innovative and improve the field of speech recognition as a whole. I cannot be sure but discussing my work on this project during the interview stages may have contributed to my hiring.

I have always been interested in Artificial Intelligence (AI) and chose this project in order to further my exposure to that field. Unfortunately, due to the complexity of this particular problem and the time constraints of the project, I didn’t get to work directly with the AI portion of the system as much as I would have liked. However, I think the exposure I got to integrating the AI components into a whole system is going to help me immensely as I continue to work in AI, as I will have a greater understanding of how my models may be used and how to prepare them best to succeed in a system rather than only as an individual component.

I originally got the idea for this project from my wife, who is completing her master’s degree in Speech-Language Pathology. I discussed with her many of the challenges associated with that field in order to help come up with ideas for a project. This discussion with her, which continued throughout the project, helped me to think outside of my particular discipline and see how I could create great systems to benefit many different fields if only I spent the time to learn their individual challenges. It was challenging at first to listen to the advice I received about what kind of a product a speech-pathologist would want, what features they would most like, and how they would prefer it to look, as I already had in mind how I would accomplish the goal. This forced me to remember that applications are hardly ever developed for the sole use of software engineers, they are developed for some customer, and they need to be optimized for that customer’s use case.

While my work on the project is complete for the semester, I plan to revisit it after graduation, with the hope that eventually I can release it to the general public. While it may take a lot more effort to get it ready for such a release, the opportunity this project has given me as a steppingstone to a first-time entrepreneurial experience is something I really appreciate. I have always wanted to pursue entrepreneurship, whether as a side hobby or as a full-time job if things went well, but I have always found it hard to find the time to really get started. This project has, in a way, forced me to take those first steps, and now I’ve seen that I’m capable of creating a functioning product, and all that remains is to try and improve it. I will always be grateful for that opportunity, even if this particular avenue doesn’t pan out, as I think it will give me the confidence to explore other future projects as well if the chance arises.

For me, one of the best things about this particular project is the impact it could have on some individuals. While speech therapists and others who frequently transcribe IPA would be saved effort through the product, I think the clients could benefit the most. From what I understand, the transcription process is integral to diagnosing individuals...
with a particular speech disorder, and in knowing what to target during therapy in order to try and improve their speaking ability. However, human IPA transcription can be unreliable and inconsistent, changing greatly from therapist to therapist, as it leaves a lot up to the interpretation of the therapist. While the application I developed isn’t ready to replace human transcription, it is a first steppingstone, and as speech processing and AI technology continues to advance, it will one day greatly increase the quality of speech disorder diagnosis and therapy.

Overall, the project was a lot of work, and was the cause of a lot of stress over the previous two semesters, but I am really glad that I was able to do it. I have grown a lot in my technical skills, but also in my ability to think about problems from a larger viewpoint, seeing how all of the different moving pieces will and should interact.

Word Count – 1020
6. WORKS CITED


7. APPENDIX

MyScribe
English-to-IPA Transcription

User Guide
Introduction

MyScribe is an iPhone application designed to automate the process of transcribing English speech into the International Phonetic Alphabet (IPA). This guide will provide you with a step-by-step guide for obtaining the application, as well as instructions on how to perform each of the app’s basic functions:

- Creating an Account / Logging In
- Making a new recording for transcription
- Uploading an external file for transcription
- Viewing and managing past transcriptions
- Managing your account

Required Materials

In order to utilize MyScribe, you will need to own an Apple mobile device running iOS 13 or later, an AppleID, and an internet connection.

Instructions

Obtaining the Application

1. On your iOS device, open up the Apple App Store.
   a. You will need to login with your AppleID or create one if you don’t already have one.
2. Using the App Store search bar, search for “MyScribe”.
3. Tap the install button located to the right of the “MyScribe” application.

Creating an Account / Logging In

1. Tap the MyScribe application on your iOS device to open it.
   a. You may need to wait a few seconds as the application loads.
2. Tap the “Sign-In” button.
3. If a prompt requesting permission for an Amazon Cognito webpage appears, tap continue.
4. Sign-Up or Sign-In
   a. Sign-Up
      i. If you do not already have an account, tap the words “Sign up”.
      ii. Enter your desired username, your email, and your desired password.
      iii. Tap “Sign-Up”.
      iv. Enter the verification code sent to your email address.
      v. Tap “Confirm Account”.
   b. Sign-In
      i. If you have already created an account, enter your username and password.
      ii. Tap “Sign In”.

Creating a New Transcription

1. Tap the “New Transcription” tab at the bottom-middle of the screen to navigate to the new transcription page.
2. Recording or Upload
   a. New Transcription from Recording
      i. Tap “New Recording”.
      ii. Enter a name for the new transcription. *Note the application will not allow you to enter a particular name if that name already exists as a file for the application.
      iii. Tap “Save Name”.
      iv. If you are prompted to allow the application microphone access, tap “OK”.
      v. Tap the “Tap to Record” button to begin recording.
      vi. Tap the “Stop Recording” button to stop the recording.
      vii. If you need to try the recording again, repeat the previous two steps.
      viii. When you are satisfied with the recording, tap “Save Recording”.

   b. New Transcription from File Upload
      i. Tap “Upload Audio File”.
      ii. Select the desired audio file from the iPhone file browser.

Viewing and Managing Past Transcriptions

1. Tap the “Transcriptions” tab in the bottom-left of the screen to open the transcription list screen.
2. Scroll through the list to locate the desired transcription using the displayed name and transcription date.
3. Tap on the desired transcription to open the transcription page.
4. On the transcription page, tap the play button in the top right to listen to the original recording.

5. After tapping the play button, tap the stop button in the same location to stop the playback.
6. Scroll through the transcription in the center of the screen to read it all.
   a. Symbols colored red indicate the application is not confident in its transcription, and they may need to be double-checked.

Managing your Account

1. Tap the “Account” button in the bottom right of the screen to view the account screen
   a. On this screen, you can view your username and subscription level.
Conclusion

You are now ready to use MyScribe an automated transcription assistant. Feel free to practice creating recordings and transcriptions of yourself to become familiar with the tool and remember to return to this guide if you have questions about MyScribe’s capabilities.

8. Author Bio

Riley Roberts is a Computer Engineering undergraduate student at Utah State University and will graduate at the end of the Fall 2021 semester. Riley has three minors: Computer Science, Mathematics, and Japanese. He has a 4.0 GPA and has received honors such as the A-Pin award for his academic performance. Riley has published research papers in two professional conferences, FMICS 2021 and VMCAI 2022 (to be published January 2022), as the first author and was also a co-author on a journal paper (ACS Synthetic Biology). He was selected as the Undergraduate Researcher of the Year for the Department of Electrical and Computer Engineering and is in consideration for the award of the same name from the College of Engineering. Following graduation, Riley has accepted a position with Apple as part of the AI/ML team. He also plans to obtain a master’s degree part-time while working at this position.