Singing Data Labeling Tool
Milestone 3 Progress Evaluation Report

Group members:
Nandith Narayan nnarayan2018@my.fit.edu
Avinash Persaud apersaud2018@my.fit.edu
Carlos Cepeda ccepeda2018@my.fit.edu

Advisor:
Dr. William Shoaff wds@fit.edu

Client:
Caleb Matthew Long, Appalachian State University

Task	Completion %	Nandith Narayan	Avinash Persaud	Carlos Cepeda	To Do
1. Add Spectrogram View to the main window of the tool's GUI.	100%	80%	20%		
2. Split classification of phonemes based on type and train models for classification of phonemes.	100%	90%	10%		
3. Create and train models for phoneme boundary detection.	80%		80%		Implement a method to convert the output to the boundaries between phonemes
4. Custom theme and color map support	100%	50%	50%		

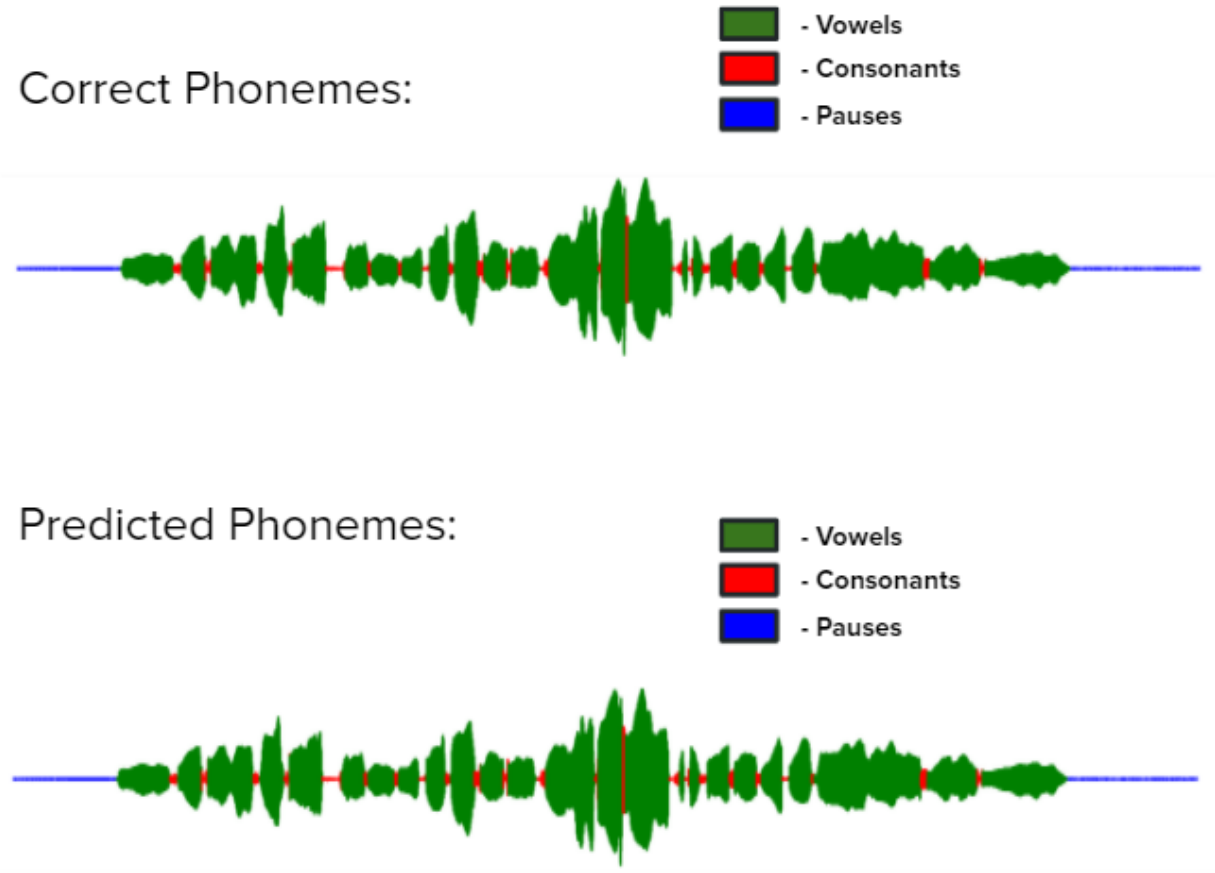
Task Discussion:

Task 1:

Adding the spectrogram to the GUI proved to be a difficult task as it posed multiple challenges we had to overcome. To compute the spectrogram, we wrote an implementation of the Cooley-tukey Fast Fourier Transform algorithm. We then sliced the audio input into batches or windows. Each window overlapped with the previous and next windows a little bit. Each window was passed as input into the Fast Fourier Transform. The output was then used to create the spectrogram. Here we ran into an issue; the outputs of the Fourier transform have huge ranges, some values can be as large as a few billion while others are in the millions. To solve this, we took the log of the spectrogram. This made our ranges way easier to handle. Then, we wrote a python script to convert an image into a color map. We used this color map to tell the UI what color each value should be displayed as. Our next issue was that computing the spectrogram was computationally expensive. Our initial builds had the UI become unresponsive due to the computation of the spectrogram taking too long. We fixed this by multi-threading the computation of the spectrogram. This allows us to have the UI and the computation on two separate threads. We cache the spectrogram for every audio file that's imported in order to prevent having to recompute the spectrogram. This lets us compute the spectrogram only once. To get the spectrogram to scroll, we calculated the percentage of the audio that the current zoomed screen covered and scaled the spectrogram accordingly.

Task 2:

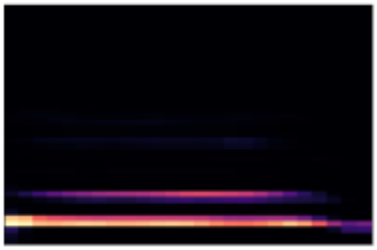
We were successfully able to classify phonemes by splitting the task into smaller subproblems. First we needed a dataset. The old dataset we were using was for speech. However, singing is very different from speech. So, we tried a dataset for children's songs. However, that dataset grouped phonemes into syllables. This meant that we had to separate the phonemes. Since this proved to be too difficult, we simply went with another dataset which was made by Nagoya Institute of Technology. First, we converted all the phonemes from raw audio to a 2D image representation of the frequencies present in the audio using the MEL spectrogram. Then, we passed these images (20 by 40 size) to a simple CNN. This CNN classified whether the given phoneme was a pause(break between words), consonant, or vowel. We got this model to achieve a 99% accuracy on our test set. Below is a graphical representation of a prediction made by the model.



As the diagram above shows, this model was near 100% accurate. The classification report for this model is given below.

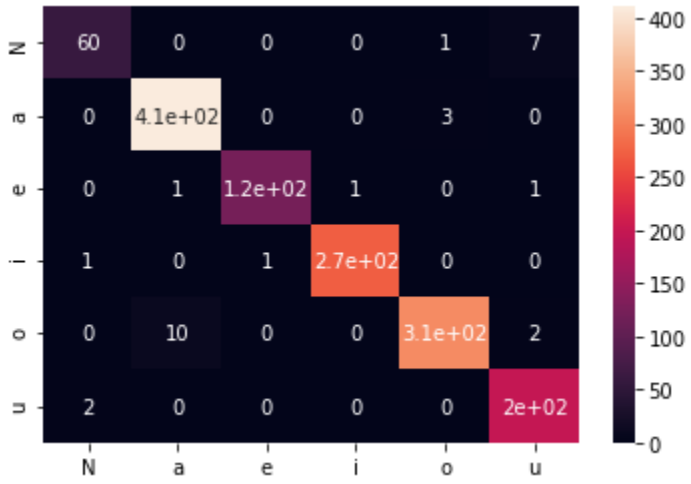
	precision	recall	f1-score	support
Consonant	0.99	0.98	0.98	1097
Pause	0.95	0.97	0.96	117
Vowel	0.99	0.99	0.99	1402
accuracy			0.99	2616
macro avg	0.98	0.98	0.98	2616
weighted avg	0.99	0.99	0.99	2616

MEL Spectrogram example:



Then, we trained a model to classify the vowel phonemes. Note that these are not letters that are vowels, rather they are phonemes which sound like vowels that are represented by letters. We attained an accuracy of 98% on vowels. The classification report and heatmap are below for the vowel model.

	precision	recall	f1-score	support
N	0.95	0.88	0.92	68
a	0.97	0.99	0.98	413
e	0.99	0.98	0.98	126
i	1.00	0.99	0.99	274
o	0.99	0.96	0.97	323
u	0.95	0.99	0.97	199
accuracy			0.98	1403
macro avg	0.98	0.97	0.97	1403
weighted avg	0.98	0.98	0.98	1403



Then, we trained a model to classify the consonant phonemes. This was a harder task for the network to learn, hence we only got an accuracy of 81%. We would need to further subdivide consonants into sub groups for better classification. The classification report is to the right.

	precision	recall	f1-score	support
b	0.53	0.62	0.57	32
ch	1.00	0.97	0.98	30
d	0.59	0.42	0.49	45
f	0.71	0.42	0.53	12
g	0.46	0.62	0.53	39
h	0.83	0.52	0.64	29
j	1.00	0.62	0.77	8
k	0.92	0.96	0.94	152
ky	1.00	0.33	0.50	3
m	0.81	0.78	0.79	106
n	0.76	0.81	0.78	125
p	1.00	0.14	0.25	7
py	1.00	0.40	0.57	5
r	0.85	0.85	0.85	157
ry	0.62	0.70	0.65	23
s	0.90	1.00	0.95	66
sh	0.97	1.00	0.99	39
t	0.82	0.95	0.88	93
ts	1.00	0.71	0.83	24
ty	0.95	1.00	0.97	19
w	0.94	0.71	0.81	21
y	0.87	0.82	0.84	55
z	0.67	0.74	0.70	19
accuracy			0.81	1109
macro avg	0.83	0.70	0.73	1109
weighted avg	0.82	0.81	0.81	1109

Task 3:

More research has to be done for proper language processing tools to be implemented in the application. This will also be taken care of over the break and be ready for M4.

Task 4:

The color map is responsible for mapping the spectrogram values to a corresponding color. To create this color map, we wrote a simple python script that took an image of a color map and created a one-to-one mapping from horizontal pixel value to color. Then, we mapped the spectrogram's values to the horizontal pixel value to get the color. We also implemented support for custom themes. Upon startup, the tool checks a directory for the presence of any theme files and uses them if present. Currently, we only have 1 theme apart from the default.

Contribution Discussion:

Milestone 4 task matrix:

Task	Nandith Narayan	Avinash Persaud	Carlos Cepeda
1. Speed up spectrogram computation	50%	50%	
2. Speed up rendering of waveforms	50%	50%	
3. Create GUI elements and windows for common tasks like project creation and customization	10%	10%	80%
4. Integrate phoneme boundary detection and classification	40%	60%	
5. Allow the User to manually add phonemes	40%	30%	30%
6. Output the created labels	33%	34%	33%

Milestone 4 Task Discussion:

Task 1:
Currently, the spectrogram takes a significant amount of time to compute. While this is fine for small audio files, this doesn't work well for larger audio files. We plan to use the FFTW library to speed up our FFT calculation which in turn will speed up our spectrogram.

Task 2:
Currently, the waveform takes longer to render when in full screen due to the increased number of lines needing to be drawn to the screen. We plan to increase the speed of this by caching the wave form as an image and simply cropping and scaling said image for zoom.

Task 3:
We plan to let the user customize things like the theme and color maps as well as other miscellaneous settings. To do this, we need to create multiple windows for accessing settings.

Task 4:
The next step is to integrate both the phoneme boundary detection and the phoneme classification to create an automated tool for detecting phonemes.

Task 5:
We want to let the user manually add phonemes to the project. This requires the creation of a new GUI element that contains the phonemes and lets the user drag and drop phonemes.

Task 6:
The most important part of this tool is being able to export the labeled data as a label file. We are targeting the HTS singing label format for our output format.

Client Feedback

Got feedback from Client on **11/18/21** via Discord.

Date of meeting with Faculty Advisor: **11/29/21**

Faculty Advisor feedback on milestone tasks:

- Evaluation by Faculty Advisor
- Faculty Advisor: detach and return this page to Dr. Chan (HC 214) or email the scores to pkc@cs.fit.edu
- Score (0-10) for each member: circle a score (or circle two adjacent scores for .25 or write down a real number between 0 and 10)

Carlos Cepeda	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Avinash Persaud	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Nandith Narayan	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

- Faculty Advisor Signature: _____ Date: _____