|  |
| --- |
| **Capstone Project**  **-I Can See My Voice(나의 목소리가 보여)-**  Final Report |

|  |  |  |
| --- | --- | --- |
| **Major** | Computer Science & Engineering | |
| **ddfdf** |  |  |
| **Class** | Capstone Project(1) | |
|  |  |  |
| **Members** | JaeWoong Song, JooHwan Shin, JiHo Lee | |
|  |  |  |
| **Team Name** | Team MJJ | |
|  |  |  |
| **Submission Date** | 2019.12.09 | |
|  |  |  |
| **Professor** | Young Bin Kwon | |
|  |  |  |



**Context**

|  |  |
| --- | --- |
| **Idea Proposal** --------------------------------------------------- | **3** |
| **Development Backgrounds** ------------------------------------ | **3** |
| **Project Aim** ------------------------------------------------------ | **4** |
| **Project History** -------------------------------------------------- | **4** |
| **Process** ----------------------------------------------------------- | **5** |
| **Implementation and Theoretical Base** ------------------------ | **6** |
| **Problem Solving** ------------------------------------------------- | **14** |
| **Limitations** ------------------------------------------------------- | **16** |
| **References** ------------------------------------------------------- | **17** |
| **Development Environment** ------------------------------------- | **18** |
| **Comparison** ------------------------------------------------------ | **19** |
| **Conclusion** ------------------------------------------------------- | **19** |
| **Schedule** --------------------------------------------------------- | **20** |

**1. Idea Proposal**

Hearing impaired people are vulnerable to speech-based communication due to their poor or no hearing ability. As of 2018, the number of deaf people registered in ‘National Indicator System’ is 363,000, and their lack of communication is an important social issue. It takes a very long time of patience for them to learn speaking and is almost impossible without the help of people around them

We will focus on these issues to create a handheld device that helps hearing impaired people to speak in correct pronunciation

**2. Development Background**

**2.1 Background**

**2.1.1 Hardware**

**- Portability**

## Because practice should be done repeatedly, we think that portability is a big factor in order to correct pronunciation anytime, anywhere without much preparation. We decided to make a portable machine-based program, so that users can practice pronunciation anywhere in a quiet place.

**- Sensors**

Our plan is to use voice sensor with high sensitivity so that we can check user’s pronunciation.

For these reasons, we decided to make an application on Raspberry Pi and it’s sensors.

**2.1.2 Conventional Methods for Hearing-impaired Learning Speech**

**- Help from people around**

Hearing-impaired people cannot get feedback on pronunciation by themselves, so they can't practice pronunciation repeatedly and infinitely. They always need someone with them who can help them correcting their pronunciation and give them feedback.

**- Personal statistics**

It is difficult for the hearing-impaired to assess and pronounce their pronunciations, so it is difficult to reconfirm information about what they are doing wrong. They tend to pronunciate badly in some pronunciation repeatedly.

**2.2 Definition of Pronunciation Correction**

Hearing impaired people who have difficulty hearing because of hearing loss can also practice speech using their sense of sight, touch and muscle movement. But they can't hear and evaluate it. Therefore, we will help them by scoring their pronunciation, giving them visual feedback.

Pronunciation Correction : This means that there is no incompatibility, and it is repeated and continuously performed so that the correct pronunciation of the word can be followed. The goal is to make users sound the pronunciation of words perfectly with the correct pronunciation

**3. Project Aim**

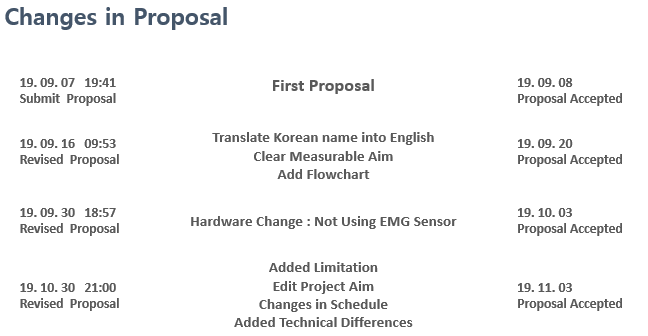
- Make phoneme-level Korean speech dataset.

- Provide service using training model in Raspberry Pi.

- Can score user’s voice data properly and give visual feedback in phoneme-level.

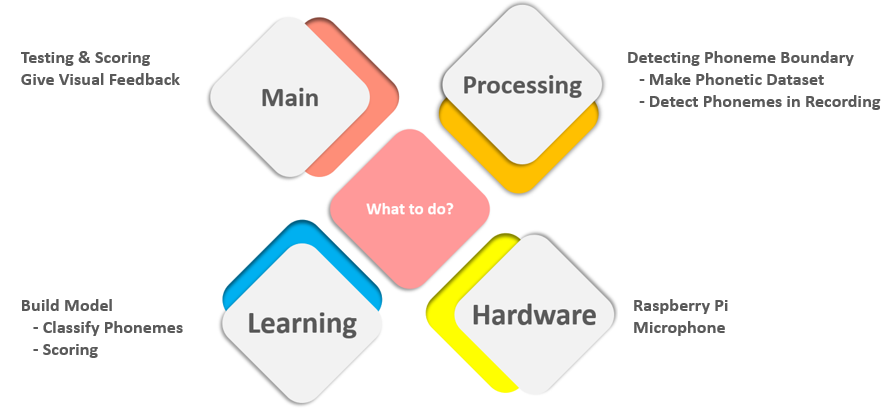
- Use training model which has 75% of accuracy or more.

**4. Project History**

****

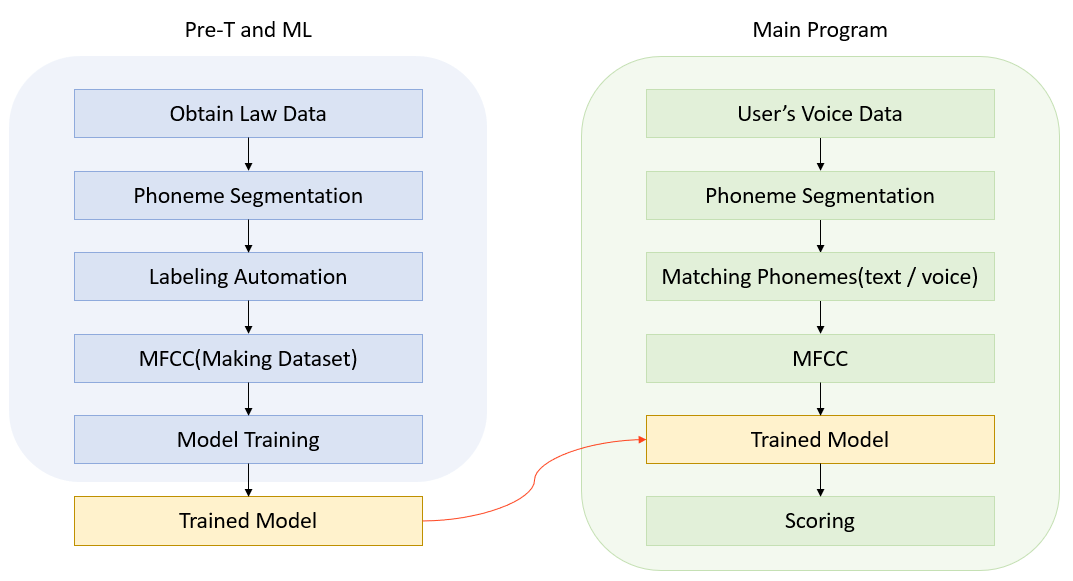
Since the first proposal had been accepted, we added measurable aims and flowcharts. We decieded not to use EMG Sensor because we don't have enough time for testing on September 30th. Finally, Some limitations, measurable aim, and schedule changes was accepted on November 3rd.

**5. Process**



To make our program, we must carry out four major steps. In order to provide feedback on phonemic pronunciation, we have to divide the user's recordings and the voices in our dataset into phonemes. We also transform our text dataset with 'standard' pronunciations. After pre-processing, we will train the dataset with CNN model.

So our process is as follow



**6. Implementation and Theoretical Base**

**(1) Obtaining Dataset**

As we give visual feedback in phoneme-level, we had to make a machine learning model trained with phonetic speech dataset. And there is no public phonetic speech dataset, so we have to develop a processing model to make it, learning model to train it, main program to score user’s pronunciation and give feedback, and hardware which our program works on.

Using dataset downloaded from Aihub.

* Korean free speech voice data for acoustic modeling for improved interactive speech recognition
* 1,000 hours of Korean conversational voice with 2,000 speakers.
* Recorded voices of two people talking freely on a variety of subjects and record their utterances according to ERTI transcription rules.

**(2) Pre-Processing Dataset**

The dataset obtained above is a sentence-based speech. We went through the following procedures to separate them into phonemes.

* Change subtitles with standard pronunciation rule

We made a standard phoneme converter to label phonemes detected in pcm files with subtitles.

* Separate expecting Unvoiced and Voiced sound.

Korean is phonetically divided into voiced and unvoiced sound according to the vibration of the vocal cords. Since these two have very different characteristics, we analyze them separately.

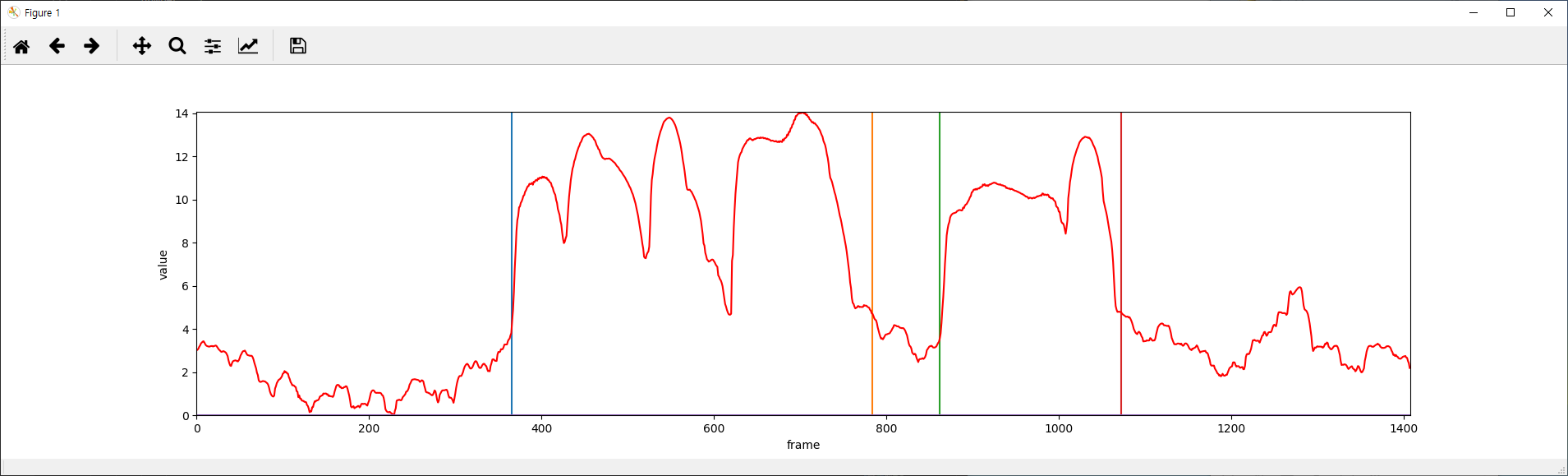
We separate voiced, and unvoiced sections because we are searching phonetic sections in different ways to suit their characteristic. In addition, when the first phoneme is a voiced sound, the beginning of the pronunciation has the same characteristics as an unvoiced sound, so a dummy unvoiced sound ‘S’ is inserted in front of it. (The same goes for the last)

* Read pcm
* Framing

To get Log-Energy, Spectrogram and so on, we need to frame pcm data we’ve got. In order to prevent loss of data, we put interval sections when framing. The size of each frame is 512 samples, and the size of each interval is 480.

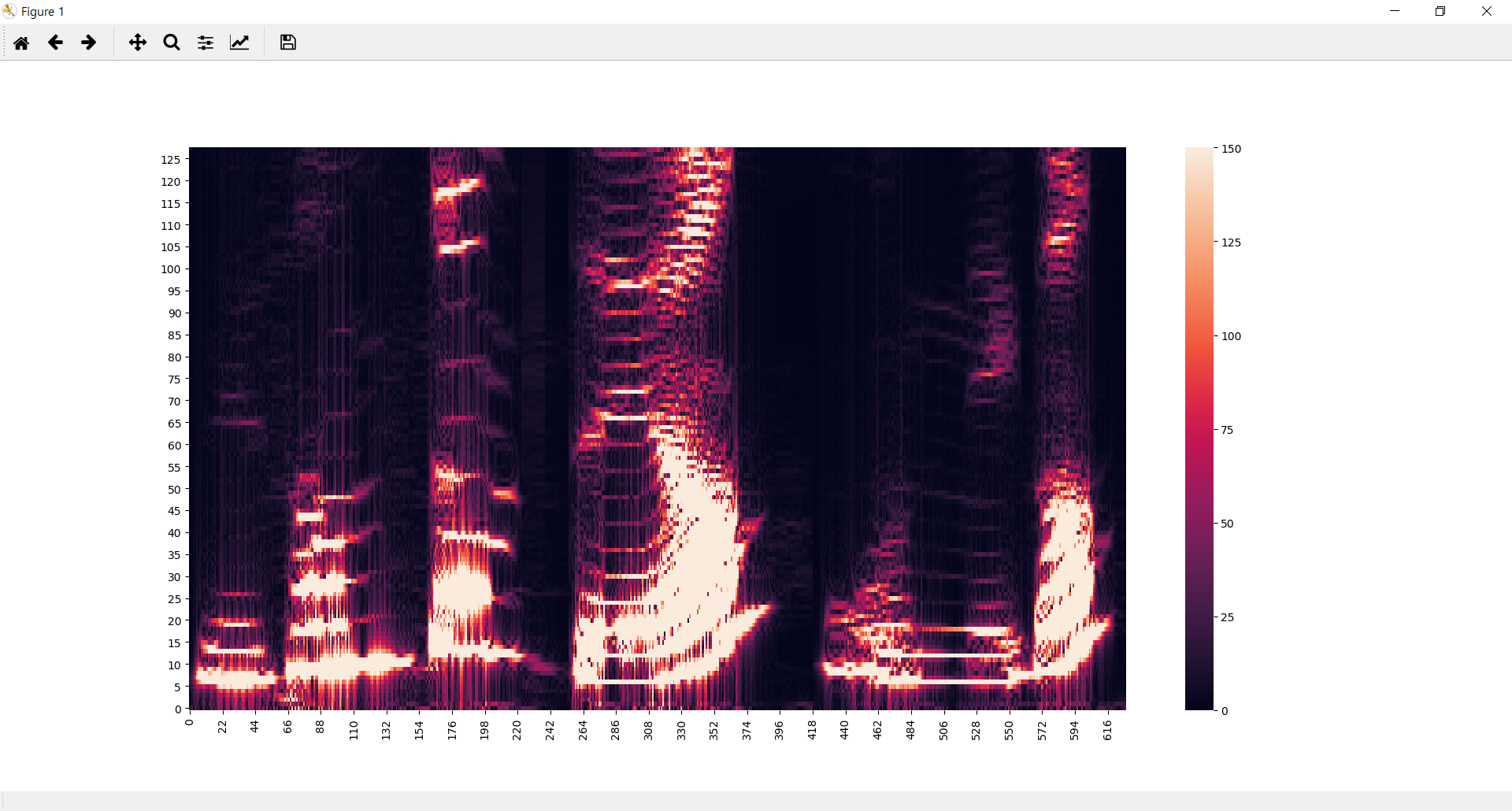


* Get log-energy for each frame



Logarithm Energy is a very important feature that distinguishes between voiced and unvoiced sounds. It is also used in phoneme segmentation. However, it is impossible to get all the information about phoneme separation from this graph itself.

* Get spectrogram (Fast Fourier Transform)



Spectrogram is a tool for visualizing and grasping sound or waves. It combines the characteristics of a waveform and spectrum. We can see the change of the amplitude axis with the change of the amplitude axis with the change of the frequency axis.

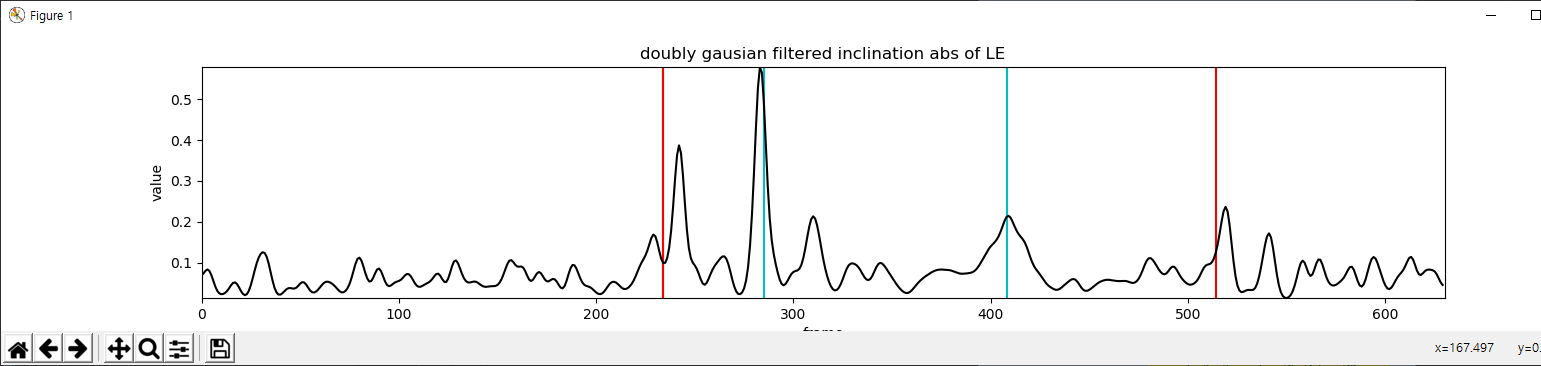
As a result of analysis, the spectrogram shake is large in the voiced section, so we decided to use it with blurring.

* Silence section and noise detection

The first step in analyzing pcm data is to detect silence section and section with voice. The process is as follows.

* In the spectrogram, silence is defined when the maximum value of the frame is less then 1/1000 of the optimal maximum value.
* If the log energy of the frame is less than 20% of optimal max energy, it is defined as silent.
* Separate Unvoiced and Voiced sound in section with voice
* If the y-coordinate of the spectrogram center of mass of the frame is 1/10 or more of the maximum of those of all frames, it is defined as voiced sound.
* If the value of the one whose value is 180 or more is 2.5% or more, it is defined as voiced sound.
* If there is more than 1000 spectrogram values of the frame in the section, it is defined as voiced sound.
* Withis the interval defined as voiced sound, that segmentation is defined as unvoiced sound if it does not last more than 0.05 seconds. This is because the shortest time required to pronounce the voiced sound is 0.05 seconds.
* The section in which the sound contains no voiced sound is defined as noise section. This is because there is no Korean sound does not contain any vowels.
* Matching UV section in standard pronunciation and pcm file

If the UV sections are not matched, no further work is done and the data is not used. It is to ensure the integrity of our dataset.



* Detecting phonemes

For the matched pcm file, We detected phoneme by checking convex points in the graph of doubly Gaussian filtered inclination of absolute log energy. In unvoiced sections, however, al least one phoneme is taken even if no convex exits. It is because usually voiced sound has much more value than unvoiced sound, so there can be no convex point if the pronunciation changed quickly.

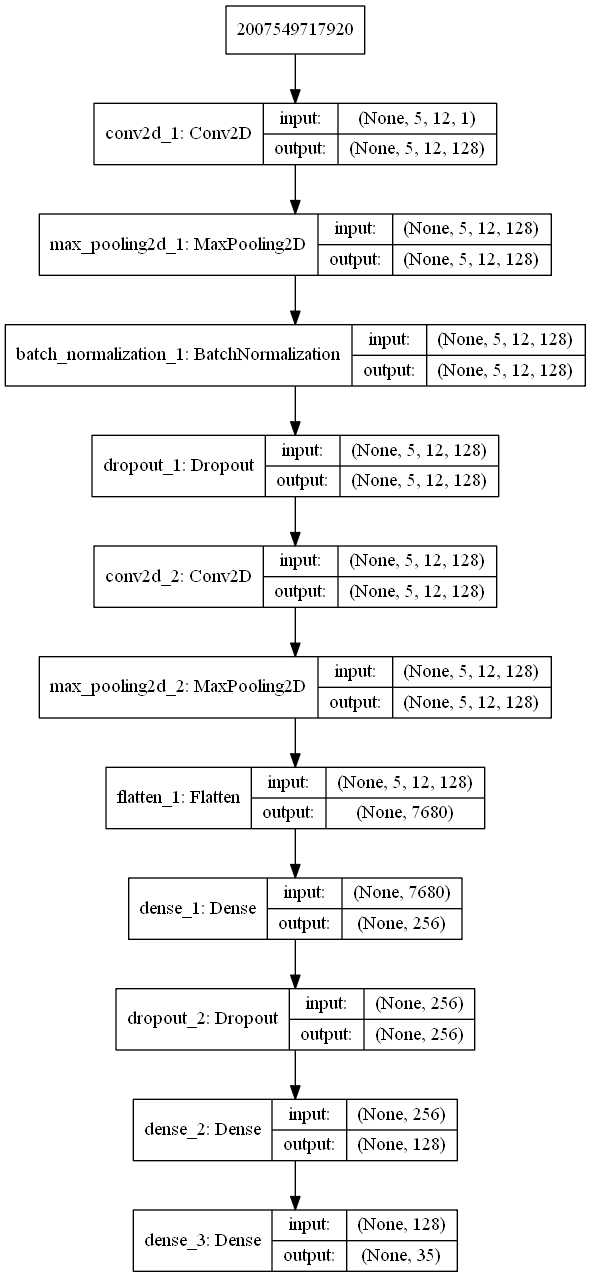
* MFCC

We divided 0.25 seconds to the left, 0.25 seconds to the right, 0.5 seconds in total from the coordinates of the found phonemes in 0.1 second intervals. By applying the MFCC in frames divided by 0.1 second intervals, the top 12 features are brought.

As a result, 5 x 12 sized MFCC data were combined with labels from standard pronunciations to create a dataset.

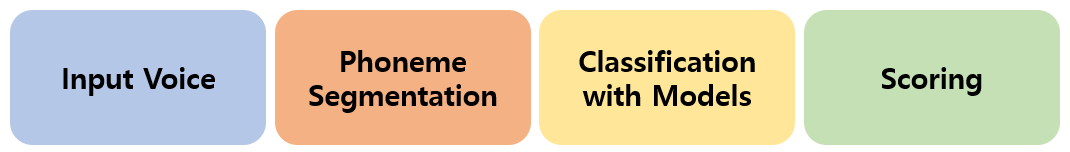
If you want to check more about making dataset, please refer to the link :

<https://github.com/jwoonge/MJJ>

**(3) Machine Learning**

This is the structure of our learning model. We didn't use a pre-trained model because each of our datasets is a small dataset with 12 columns of 5 rows.  
In addition, the existing voice recognition learning model used RNN model mostly, but because our purpose is to create a model taught in phonemic units, we decided not to use RNN but to use an array with five MFCC values per phoneme as a data set, thereby using the CNN model.  
After being advised that our datasets may be lacking, we increased the amount of learning by reducing our CNN kernel size to a smaller size. After selecting the features of the dataset using two CNN layers, two DNN layers for classify feature and one output DNN layer for extract output. And the output DNN layer has softmax activation function for multi class classfication  
Also, the number of outputs of the output layer in the above figure is 35 but we have implemented the multi-class model according to the Korean alphabetical table, such as the model that separates 파열, 파찰, 마찰 sound, etc. by adjusting the type of data set that we want to classify (ex, 파열 sound, 파찰 sound, and 마찰 sound).

**(4) Pre-Processing Voice Data**



- Phoneme Segmentation

Phoneme segmentation in main program and phoneme segmentation in making dataset are different. Though we have expecting proposed word but we cannot sure whether the user says ‘the word’ or not. So we had to pay more attention since we cannot label phonemes with the text.

First, we find the UV section, but set the silent section considering that our headword consists of only one word. In addition, since our headword consists of only three letters or less, the maximum number of appearances of the voiced sound section is utilized.

The PhonemeProc module splits the phonemes and returns the MFCC data and its voiced / unvoiced classification, and enters the scoring module.

The scoring module then provides an STT function in addition to the scoring function. All detected phonemes must be scored by matching the phonemes of the expected ‘head word’.

In other words, in the scoring, we need to choose the location of the phoneme we find is reasonable to match. This is because we should match the result even if the user pronounces the wrong pronunciation.

Since the only information we know about the input phoneme is the voiced / unvoiced category, voiced sounds were compared between voiced sounds and so do the unvoiced sounds. (If they do not match, the score will be 0 and you should record it again.)

If the phoneme belongs to the voiced sound category, a model (20 classes) that classifies the whole voiced sound is determined, and when confidence is judged, the score is given according to the rank of the corrected phoneme.

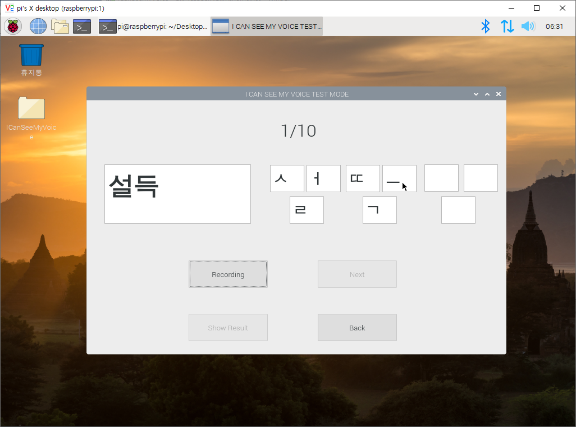
**(5) Scoring**

If the phoneme belongs to the unvoiced sound category, three multi-class models checks it according to the Hangul consonant classification system to see if it belongs to the correct classification.

For example, ‘ㄱ’ belongs to the ‘예삿소리’ (label0), ‘파열음’ (label0) and ‘경구개음’(label3). If a ‘ㄱ’ classified in (0, 0, 2), It takes 25% of deduction.

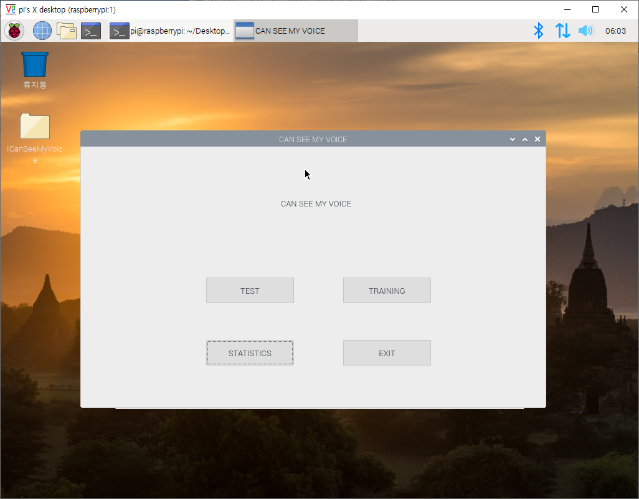
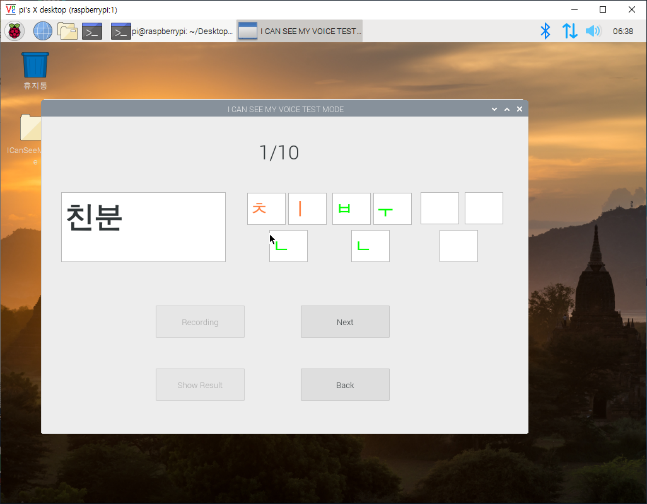
**(6) Hardware & UI**

We borrowed a Raspberry Pi 3 and touch screen from school and bought a microphone.

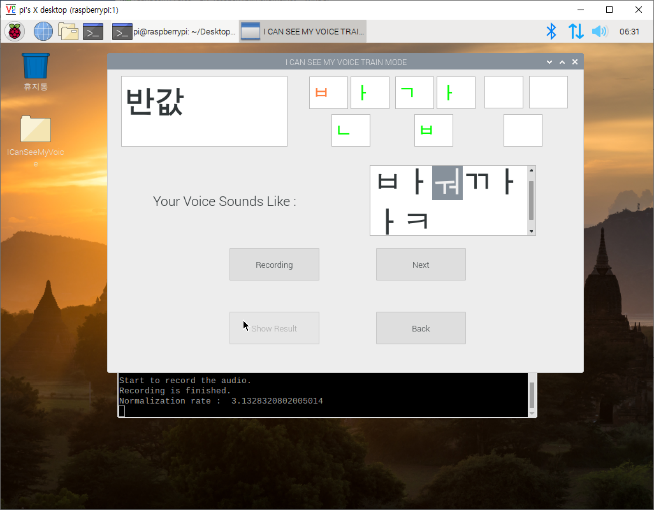
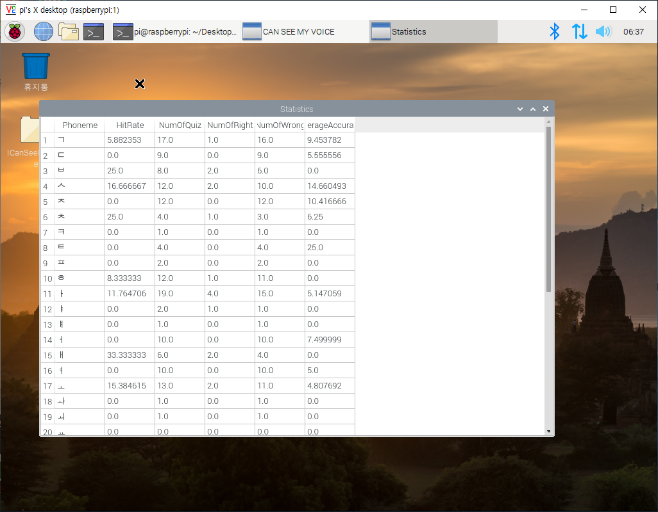


We installed Pajbian OS after formatting Raspberry Pi, and built a development environment using ssh with Putty and WinSCP. As a result, we successfully ported our software and tested its performance.

These following pictures are the result of testing:

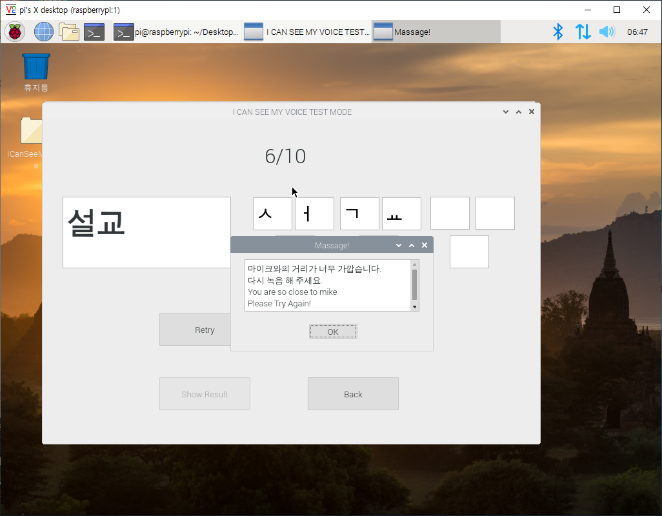
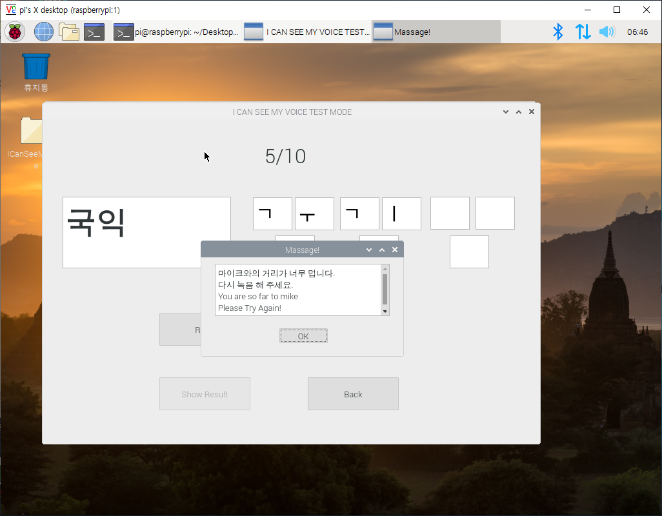
 

<Main Window> <Result of Testing Window>

<Result of Training Window> <Statistics Window>

Testing has shown that the accuracy of our program can vary depending on the sensitivity of the microphone and the volume of the input. So we did an exception hadling on UI for the case user didn’t give the right level of input.

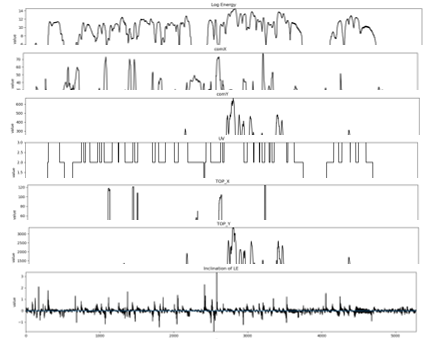
   
<When the speaker is too close> <When the speaker is too far>

**(7) STT Algorithm (Added after Final Demo)**

In ‘Training Mode’, we separated every 0.1sec in pcm file to provide results. In this case, we used a unified model of 35-class replication. It works just like STT, but it shows you the result in the phoneme unit.

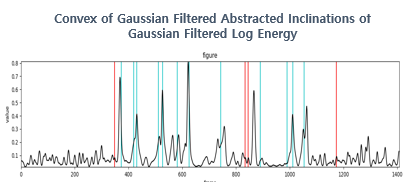
**7. Problem Solving**

**- Too many features to integrate**



As the phoneme changes, there will be a striking changes in frequency, 0crossing rate, and log energy. We can get the phoneme boundaries by analyzing numerable graphs.

The problem was, there were too many features we had to integrate. We were able to specify the area and number of phonemes by dividing voiced / unvoiced sounds, but didn’t know which features to use or not.



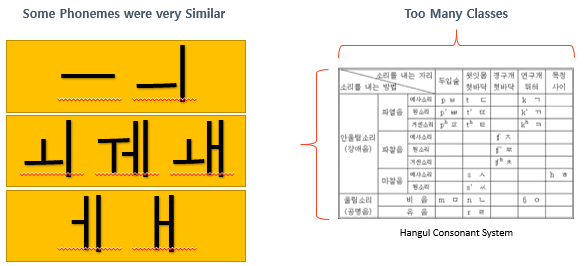
We solved this problem by using gaussian filtered log energy. With this, we could specify features to use, and able to get about 1,200,000 samples.

**- Low accuracy in training model.**

It was very hard to get a model with high accuracy because

* Our samples were not enough to classify 40 phonemes
* Our samples were too biased in some phonemes

So we could get only about 40% of accuracy. It was not possible to make program that completely depends on this model.



To solve this problem, we used multiclassing. We divided our model into smaller models which can classify phonemes based on Korean Pronunciation System. Also we integrated some vowels which has very similar pronunciation. Finally, we applied data augmentation by giving current samples little bit of changes.

**8. Limitations**

**- Limit of Noise Reduction**

We have successfully searched for the noise with our algorithms, but have not done any work to separate it from the voice data we need. Parameters such as log-energy are easily affected by noise. Therefore, **it is recommended to use our program in a quiet environment.**

**- Cannot support sentence level correction**

The longer the length of the recorded data to analyze, the more silent sections between the voices, and the longer the overlap between voiced and unvoiced sounds, the less accurate the phoneme separation is. To ensure the integrity of our program, we decided not to provide sentence level correction.

**- STT function is not accurate**

The STT function was not included in our proposal. Our dataset has the specific amount and shape to compare the expected pronunciation with the user’s pronunciation. Therefore, our dataset is not suitable for implementing STT without using expected pronunciation. It is not recommended you to trust STT function too much.

**- Roughness of the dataset**

The dataset we used is the voice of 2,000 ordinary people talking in their daily lives. Pronunciation becomes inaccurate depending on their emotional state, surroundings, and speed of speech. If we used the voice of an expert(like an announcer) with the correct pronunciation, we would get better results.

**9. References**

We referenced many academic papers to establish and validate our theoretical background. The following documents are referenced:

- Phoneme Segmentation based on Volatility and Bulk Indicators in Korean Speech Recognition

Jae Won Lee, 2015

- Research about auto-segmentation via SVM

Ho Min Kwon, 2003, Dong-A univ.

- MLP - based Phoneme Segmenter for Applying to Spontaneous Speech Recognizer

Nam Gyu Kang, 1998, ETRI Spoken Language Processing Section

- Useful Feature Vectors for a CD-DNN-HMM based LVCSR System

Seong Joo Lee, 2014, 이성주, ETRI Speech Processing Lab

- Speech Recognition of Phoneme Size Using Neural Network

Kee Hee Lee, 1997, The Korean Society of Computer and Information

- Korean Phoneme Recognition Using Neural Networks

Gyun Cha Jung, 1992, POSTECH. E.E.

- A Blind Source Separation Method Based on Independent Vector Analysis for Separation of Speech Signal and Noise Signal

Jae Seung Choi, 2018, Korean Institute of Information Tech

- Development of a Phoneme and Tone Labeling Program

Yun Kyung Lee, 2007, Chungbuk Nat.Univ.

- Phoneme Segmentation Using Phoneme Combination and Formant Scaling in Korean

Chong Hwan Lee, 2003, INHA univ.

- Branch Algorithm for Phoneme Segmentation in Korean Recognition System

Young Wan Seo, 2000, Inha univ.

**10. Development Environment**

**10.1 Language**

- Python 3.6

**10.2 Libraries(requirements)**

- h5py==2.10.0

- Keras==2.3.1

- microdotphat==0.2.1

- numpy==1.16.2

- pandas==0.25.1

- PyAudio==0.2.11

- python-speech-features==0.6

- scipy==1.3.3

- seaborn==0.9.0

- tensorboard==1.13.1

- tensorflow==1.13.1

**10.3 Hardware**

- Raspberry Pi3

- 7-inch touch screen for Raspberry Pi

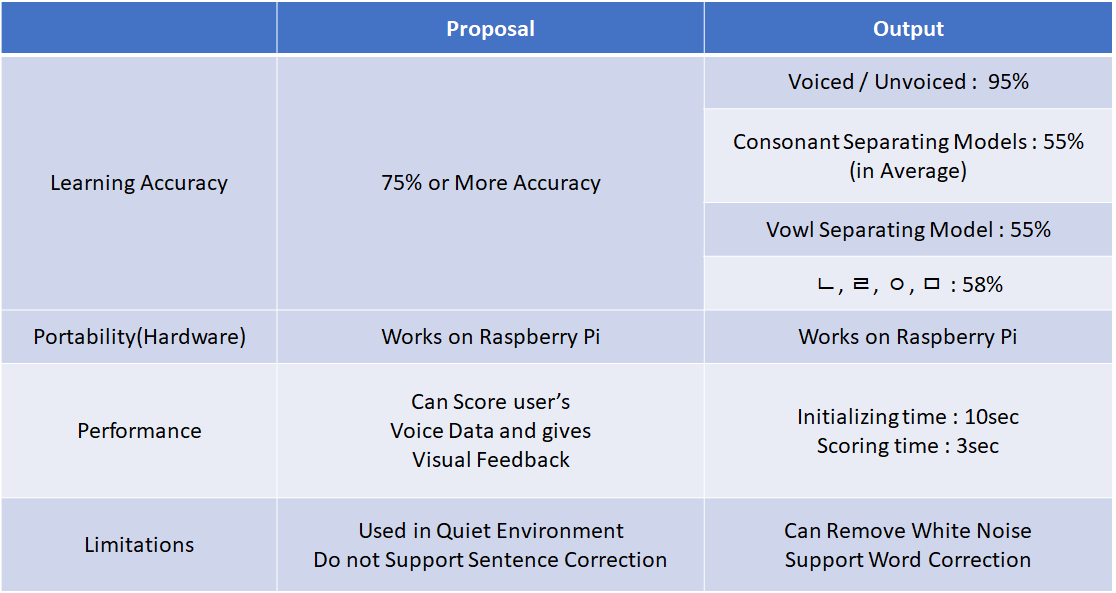
- ‘코시 한방향 USB 마이크’ MK1343UB

**10.4 Cooperation**

- github : <https://github.com/jwoonge/MJJ>

We did agile programming using GitKraken and Github. Codes that used in making datasets and not submitted as our source code are also available on our GitHub.

**11. Comparison**



This is the comparison table between the proposal and the output.

in learning accuracy, we could not get 75% accuracy for classifying 40 phonemes, but we made our learning model works the same or more by multiclassing. We made the program works on Raspberry Pi3 as we wrote in the proposal. We can both score user's voice data and give visual feedback to them but it may takes some time. It takes about 10 sec for loading models. We said that our hardware must be used in quiet environment but, we can remove noises to some extent. And we do not support sentence level correction like we said in our proposal.

**12. Conclusion**

12.1 Project Outputs

- We developed a **phoneme-lever automatic labeling system** by developing our own **algorithmic phoneme separation model**.

- We’ve created **enough dataset** with our system and a **phoneme recognition model**.

- We can now provide **pronunciation correction service** with hardware for the hearing-impaired using this **phoneme recognition model**.

12.2 Impressions

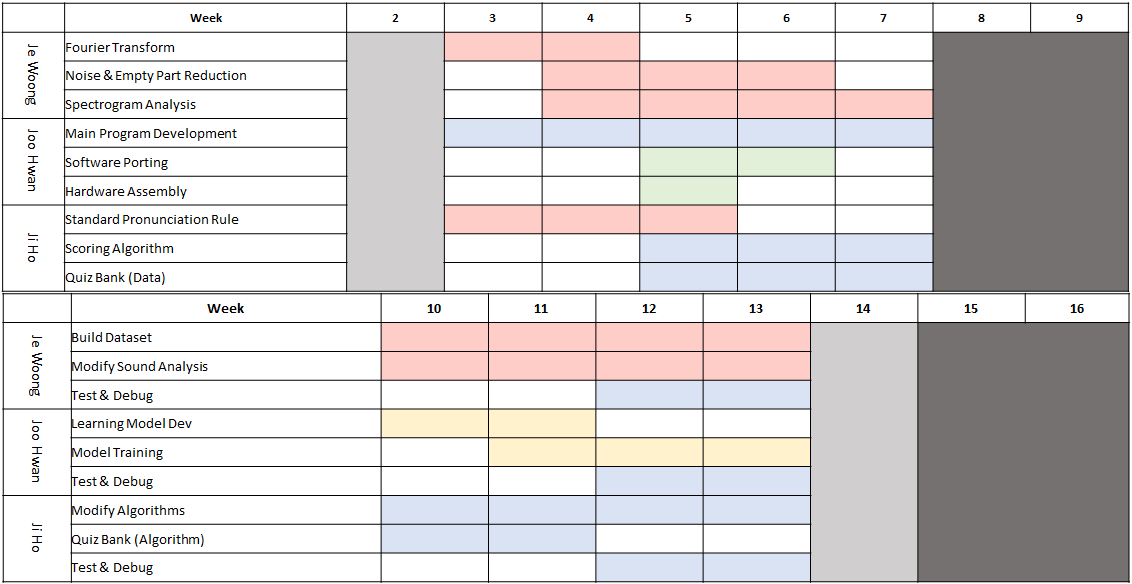
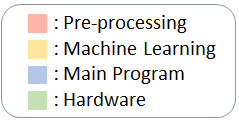
It was very hard to go the road that no one walked ahead before. We think that this project has greatly increased the ability to solve problems by creating algorithms for situations.

12.3 Feedback and Future Works

At Final Demonstration, we received harsh feedback on aspects of delivering solutions. We didn’t give users an intuitive idea of how their pronunciation was, so we had a limit in providing solutions. We realized that our program wasn’t good enough for hearing impaired people because of the problems we’ve written in ‘Limitations’.

In order to provide solutions as far as possible in our current situation, we’ve implemented STT within our limits. The main subject of our program is not implementing STT, so we couldn't produce high accuracy and intuitive output, but we will focus on the STT aspect in the future to create a program that provides a complete solution.

**13. Schedule**

****

We spent the most time pre-processing datasets and user voice data. We had to build new datasets and training model whenever the algorithms changed. By week 14, all algorithms and datasets were established. Since then, we ported and integrated to make it work on Raspberry Pi.

Je Woong Song : Mainly responsible for the pre-treatment process

Joo Hwan Shin : In charge of the training and functioning of the main program

Ji Ho Lee : Dedicated to algorithms and data processing

**This is the end of our final report. Thank you.**