Faculty of Informatics and Computer Science

Information System/Computer Networks/Software Engineering/ Computer Science

Speech to ASL (American Sign Language)

By: Moaz Ayman Mohamed Zaki

Supervised By

Dr. Khaled Nagati

And

Dr. Ann Nosseir

June 2019

Abstract

In 1975 law no.39 [8] was constructed to prevent deaf students in Egypt from joining high degrees (Universities) because of the difficulty of perceiving all the information from the tutor during lectures, and not all deaf students can read the lip-sync of the lecturer. On the other hand, every tutor must have any knowledge on how to speak sign language to be ready for any question a student might ask. In this project, we are aiming to develop an **ASR** (Automatic Speech Recognition) system that can record and recognize every spoken word by the tutor during his class and to animate it to its signed language animation. This will lead to a step further in making a better learning environment both for students and tutors to communicate with each other. Also, by using supervised neural network model in speech recognition to process every sentence the tutor declares, an avatar will demonstrate it by animating each word in **ASL** (American Sign Language) using Unity. This will give a privilege for deaf students to have a higher opportunity for joining higher education regardless of the disabilities they have.

Attestation &Turnitin Report

I understand the nature of plagiarism, and I am aware of the University's policy on this.

I certify that this dissertation reports original work by me during my University project except for the following (*adjust according to the circumstances*):

- The technology review in Section 2.5 was largely taken from [17].
- The code discussed in Section 3.1 was created by Acme Corporation (*www.acme-corp.com/JavaExpert*) and was used in accordance with the licence supplied.
- The code discussed in Section 3.5 was written by my supervisor.
- The code discussed in Section 4.2 was developed by me during a vacation placement with the collaborating company. In addition, this used ideas I had already developed in my own time.

Signature *Moaz Ayman Mohamed Zaki* **Date** *14/6/2019*

Acknowledgements

Dr Khaled Nagati proposed this idea in helping handicapped students and to enrich them with knowledge. To give them also an opportunity to join high degrees like: “Bachelor Degree or Master Degree”.

Table of Contents

Abstract ii
Attestation &Turnitin Report iii
Acknowledgements iv
Table of Contents v
List of Figures vii
List of Tables viii
1 Introduction 1
1.1 Overview 2
1.2 Problem Statement 3
1.3 Scope and Objectives 4
1.4 Work Methodology 4
2 Related Work (State-of-The-Art) 5
2.1 Literature Survey 5
2.1.1 Speech Recognition using HMM (Hidden Markov Model) 5
2.1.2 Word Recognition Engine using Neural Network to recognize Jawi language 7
2.1.3 Implementation of Neural Network for recognition of Bangla numbers 10
2.2 Analysis of the Related Work 12
3 Proposed solution 14
3.1 Solution Methodology 14
4 Implementation 16
4.1 Set up how to record audio with the Microphone 16
4.2 Extract voice features by MFCC 19
4.2.1 Applying Pre-emphasis 20
4.2.2 Frame Blocking 21
4.2.3 Hamming and Windowing 22
4.2.4 Fast Fourier Transformation (FFT) 23
4.2.5 Mel Filter Bank 25
4.2.6 LOG Energy 26
4.2.7 Inverse Fast Fourier Transformation 27
4.3 Dataset for training the acoustic model 27
4.4 Training an acoustic model by CNN (Convolutional Neural Network) 30
4.4.1 Input Layer in the CNN 32
4.4.2 Convolutional 2D Layer 32
4.4.3 Max Pooling Layer 34

4.4.4 Fully connected layer 35
4.4.5 Activation by Relu and SoftMax 35
4.5 Animating ASL (American Sign Language) by the Avatar by Unity 3D 36
4.5.1 Download an Avatar from Unity Store 36
4.5.2 C# script to listen for any message from the server to receive 36
4.5.3 C# script to animate the avatar 37
4.6 Send Broadcast message by TCP Sockets to clients 38
4.7 Run the Speech To ASL (American Sign Language) system 39
5 Testing and evaluation 40
5.1 Testing 40
5.2 Evaluation 42
6 Conclusions and Future Work 43
6.1 Summary 43
6.2 Future Work 43
References 45

List of Figures

Figure 1 Jawi Architecture....7
Figure 2 Flow of Training the Neural model....14
Figure 3 MFCC Implementation....15
Figure 4 Speech To ASL Flow....15
Figure 5 Frequency Level....16
Figure 6 Sound Level....16
Figure 7 Import Library Pyaudio....17
Figure 8 Start Recording audio....18
Figure 9 Saving and Trimming audio....19
Figure 10 MFCC Steps of Extraction....19
Figure 11 Pre-emphasis flow....21
Figure 12 Framing....21
Figure 13 Hamming and Windowing....23
Figure 14 FFT equation....24
Figure 15 example sound of Whistle A....24
Figure 16 example sound of Whistle B....24
Figure 17 Mel-Filter Bank Flow....25
Figure 18 Log Equation to apply....26
Figure 19 Logarithm applying....26
Figure 20 FFT equation....27
Figure 21 Save MFCC Features....29
Figure 22 CNN Architecture....31
Figure 23 example of applying convolutional filter....33
Figure 24 Rule for the pooling layer....34
Figure 25 example of Max Pooling....34
Figure 26 code of listening client....37
Figure 27 code for animation....37
Figure 28 Send TCP Message....38
Figure 29 screen shot of program running....39

List of Tables

Table 1 Accuracy of Jawi Characters....9
Table 2 Bangla accuracy....11
Table 3 Difference Between HMM and Neural Network....12
Table 4 All words recorded for training the model....28
Table 5 architecture of the CNN....31
Table 6 Testing accuracy for Known Speaker....40
Table 7 Accuracy comparison between acoustic models....42

1 Introduction

Speech recognition is how to make computers hear like the human and understand what is been said. To make a computer listen we want to understand that human ears only recognize frequencies between 20 and 20,000 Hz and frequency is the number of cycles or complete vibration experienced at each certain time. The sound wave contains the value of each frequency at each period to describe us how the tone and pitch of each can vary. This type of difference is called phoneme is one of the units of sound that distinguish one word from another in a language like the sound "hello" can have a different sequence of a high and low pitch as the time varies which will be distinguished by human ear because it has a phenomenon.

The first ever speech recognizer was called Audrey by Bell Labs in 1952[7], which it could only recognize spoken numbers between 0 and 9, it was a big invention back then. Despite her small vocabulary, Audrey was a marvel of science as it was the first generation of Speech Recognition to recognize an analogue wave by a human voice into a digital sound, so the computer can recognize. The first step is recording the voice of the speaker but making him pause every 350ms between each word, so the computer can differentiate between each word as their silence in the recording. The second step using it some electrical operations to sort the speech sound into electrical classes, so it can refer to which word it belongs to. Finally, Audrey flashes the number it recognizes.

The drawback is that Audrey is that it is 6 feet tall; one can only guess how much space the analogue circuit of the system consumed with its amplifiers, integrators, and filters which impossible to be connected to your mobile as nowadays inventions (Siri,

Google assistant). Which made the renowned scientist at Bell Labs John Pierce banned speech recognition because the results weren't promising enough.

In 1971 DARPA funds five years [12] of speech recognition research with finding the machine capable of recognizing the set of words with min. of 1,000, which this program led off the creation of Harpy which it uses 15,000 interconnected nodes and each represented all the possible utterances within the domain. They used BFS (Brute Force Search) algorithm to the map the speech to the right nodes to get the right text. The results were better than Audrey giving less complexity with a bigger domain of words.

Afterwards, HMM (Hidden Markov Model) technique [12] was in introduced in the early 1980s and started to be used in speech recognition. Rather than simply using templates for words and looking for sound patterns, HMM considered the probability of unknown sounds being words and transferred from one state to another according to the probability of this sound.

Finally, in 2007 Geoffrey Hinton and his students made the first deep feedforward (non-recurrent) networks for acoustic modelling. This reportedly experienced a dramatic performance jump of 49%with Google's speech recognition in 2015, which until now found in most smartphone users.

1.1 Overview

Developing a tool to help educational institutions like universities and schools in teaching deaf students. Also, to increase the learning curve of how to talk in sign language without a professional instructor for translation. This can be done through creating a 3D avatar to animate each word dynamically as the person speaks through

the microphone to be clarified better. Unlike the older version of communicating with handicapped people(deaf) in using cut videos for each word does.

1.2 Problem Statement

Firstly, in 1975 law no.39 [8] was constructed for preventing deaf students in Egypt joining high degrees (Universities) because of the difficulty in communicating with the handicapped students. Which caused a result of un employment for 80% of the deaf people. Furthermore, every tutor must have knowledge in speaking sign language for any urgent situations that may occur.

Many deaf patients complain about the use of video interpreting services in emergency rooms [5] unprivileged(deaf) patients claim there is a blurry video and they must set up the video interpreting service themselves because nurses don't know how to operate the equipment. Or they are unable to focus on a small screen in a crowded room and it's hard to convert complex sentences into sign language, making it hard to explain for the patients what the nurses want the patients to do.

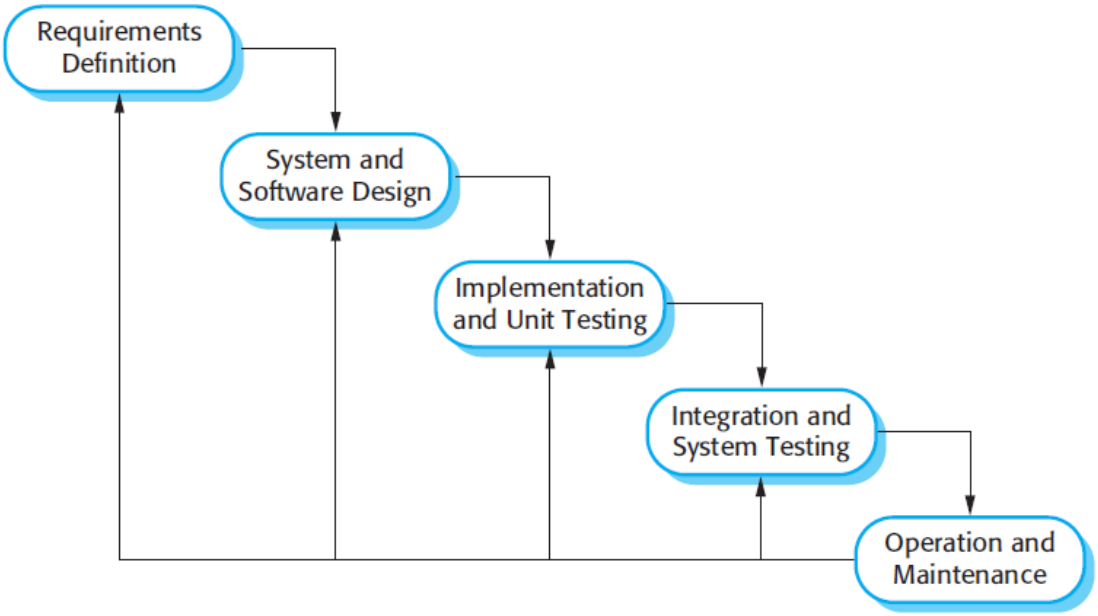
A lot of people see that learning to communicate with deaf friends or family is too costly and time-consuming learning from a professional tutor or playing cut videos for each word. This is also hard for the deaf person to understand it fully.

1.3 Scope and Objectives

The scope of this project is to increase communication and interaction between humans regarding their disability by decreasing the complexity in learning sign language.

On the other hand, making the handicapped people more recognizable to avatar animations. This can be conducted by animating a sentence as a total rather than using playback videos for each word without any relation between the words. The English words can differ in meaning according to segmentation of the sentence. For example: - "Play in the playground" | "play the video" the word play has a different meaning in both sentences, so different animations will be conducted according to the placement of the word play in each sentence.

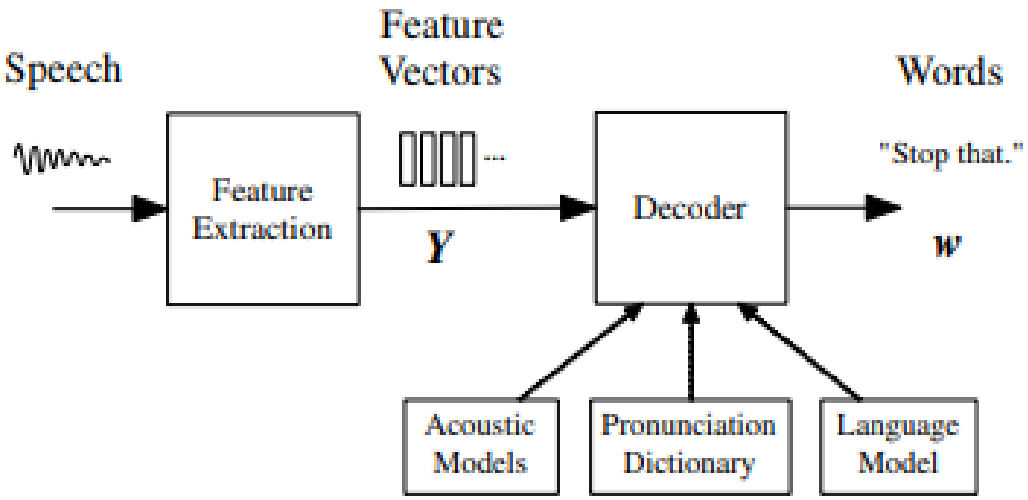
1.4 Work Methodology



2 Related Work (State-of-The-Art)

2.1 Literature Survey

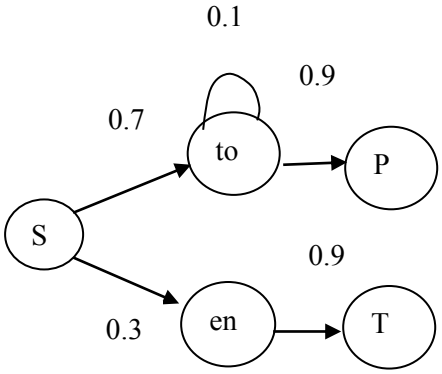
2.1.1 Speech Recognition using HMM (Hidden Markov Model)



As shown in Figure 2 the first step is feature extraction of the Speech wave, which is a very important stage [3] which seeks to provide a representation of the speech waveform, which can increase the model accuracy and can eliminate any distortion or noise in the background. Two methods were used for feature extraction of the phoneme in this project: -

Firstly, was implemented using MFCC (Mel Frequency Cepstral Coefficient)[2] which is one of the most popular algorithms to extract acoustic features and remove any noise or distortion in the background. Secondly, method was using PLP (Perceptual Linear Predictive) model that has been developed by Hermansky [2]. PLP model extract acoustic features according to the concept of psychophysics of hear-

ing. Which improves the speech recognition process by removing and discarding all the irrelevant information of the speech.



Furthermore, to input the features into the HMM model from the input audio that has been recorded. First, we need to transform the speech waveform into digital waveform, then converted into a sequence of fixed vector features from **F** 1**: T=** f1…. f: **T.** This process is called **Feature Extraction** as shown in **Figure 2,** then the decoder will try to break-down the features sequentially of the words into chunks, so it can be used by the Acoustic Models for that specific word. As we can see in **Figure 3** an example of an HMM states of the word **Stop** and **Sent** which is composed of s / to / p and s / en / T. Each composer is identified as a hidden state and then will move from on composer (state) to another according to the phoneme that has been extracted. The next phase is using the pronunciation dictionary to form any sentence "ex: - Stop the car" as it predefined as a correct sentence structure. This can be done by concatenating the phone model of each word recognized and save it as temp of all the possible solutions. The correct sentence is composed by using the decoder in penetrating through all the saved word sequences and eliminated all the hypothetically unlikely to be the right structure of the sentence for example "stop the car or sent the car" the decoder will penetrate that the higher more structured sentence is "stop the car". Also, the decoder keeps track of all the

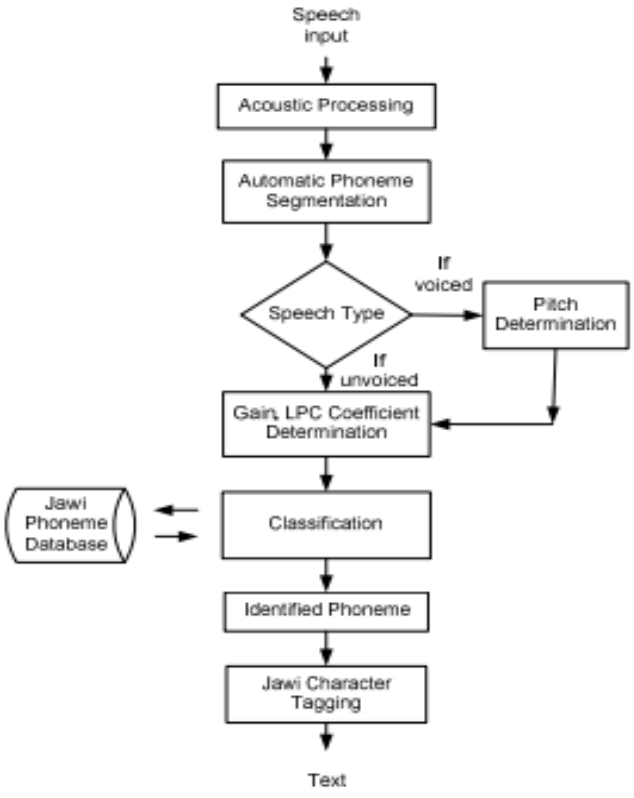
possible solutions for search trackability. Finally, when the end of the utterance is reached and accepted by the decoder, the most likely word will be presented as an output from the acoustic model.

2.1.2 Word Recognition Engine using Neural Network to recognize Jawi language

Jawi is an old version Malay language[6] that uses special characters like Arabic Characters. However, its pronunciations are in the Malay Language. The major objective of this tool to learn kids how to speak Jawi by speaking through the microphone as an input and transformed to text in English, so it enhances into the upcoming generations their Jawi's language.

This uniqueness makes STT (Speech to Text) development a challenging task as it needs to identify the difference of pronunciation between each word, which in this paper it uses **LPC (Linear Predictive Coding)** to extract unique features from the voice signal, highly efficient to classify words easily. Using LPC because it

Figure 1 Jawi Architecture



gained a lot of popularity nowadays because of it's amazing formant estimation techniques, which made it gain a lot of popularity as one of the most powerful speech analysis techniques like MFCC[2].

As shown in the Figure 1 the Speech Type (frequency) increases than the certain threshold given, the speech will be trimmed and saved in a segment from the start of the increased frequency until it is less than the threshold for a certain period. At last, gain LPC coefficient from the saved segment and classify it by using Jawi Phoneme Database.

Afterwards, the acoustic model is made by Neural Network architecture to match the right pattern of classification by a fixed vector of features for all speech signals by gaining LPC Coefficient Determination. In this project Neural Network model is used as a classifier after training and testing. The advancement in researcher in the technology of the computer architecture that been made by researchers to explore more in Neural Network. Due to larger GPU and ROM memory of computers made a major advantage in training and testing Neural models as they need large datasets and a lot of computational power. Also, the most important reason in researching more about Neural Network is that how the brain Neurons of the human brain is close of how the Neural Networks functions, which can advance to how our biological brain functions and implement it with same structure. The Neural Network work based on the Neurons that holds weights and features of the input signal and modify it every iteration of learning[6]. The implementation of the Neural Network uses a MPN known as a Multilayer Perceptron Neural Network, that give a structured layer that each Neuron completely. Also, a fully connected to all the next structured layers of neurons. Which those values will be adjustable to classify the output process of the learning curve in every iteration. Children are functioned the same way, as by learning to rec-

ognize objects and memorize them by asking what is that objects and by time they can identify and classify objects.

To adjust the weights and features of the Neurons is by using the sigmoid function that can map each neuron in every structured multi-layer. Consider the following example that all the neurons have calculated the input signal by the activation function and the output pattern has been executed. The output will be compared with actual result. If the output is equal to the result the weights will be saved for the next iteration, but if a different output has been given the error is calculated and the weights of the neurons will be re-calculated. The recalculation is made by the Backpropagation technique to adjust the weights of the output layer back again into the input layer for the next iteration. To reduce the amount of error weights assigned is by back propagating of the output layer to the previous layer.

Finally, 15 Jawi characters were selected for the learning process of the acoustic model by NN (Nueral Network) in this project while fifteen of male speakers and ten of females to randomize in gender selection. Those speakers are selected for recognition testing. Each speaker recorded for each Jawi character to train and test of 225 samples on the acoustic model on various enviroments. Until they reached an accuracy as shown below at Table 1.

Table 1 Accuracy of Jawi Characters

JAWI CHARACTER	MODERN MALAY WRITING	MEANING	TARGET OUPUT	ACCURACY (%)
كاكق	Kakak	Sister	0001	97.77%
فينم	Pinjam	Borrow	0010	88.88%
بوكو	Buku	Book	0011	97.77%
داري	Dari	From	0100	99.11%

كاون	Kawan	Friend	0101	99.11%
كامي	Kemi	We	0110	80.00%
اكن	Akan	Will	0111	82.22%
داتج	Datang	Come	1000	88.88%
برسام	Bersama	Together	1001	89.77%
ايه	Ayah	Father	1010	98.66%
ساي	Saya	I	1011	99.11%
سوك	Saka	Love	1100	98.22%
بلاجر	Belajar	Learn	1101	71.11%
توليسن	Tulsan	Writing	1110	80.88%
جاوي	Jawi	Jawi	1111	98.22%

2.1.3 Implementation of Neural Network for recognition of Bangla numbers

In this paper, the training set was ten Bangla digits from 0 to 9 were [4] each speaker is recorded and have been recognized from the speech signal the speaker record through the microphone and been translated into number by a trained acoustic model using Neural Network for only word recognition.

The speech is extracted by applying MFCC algorithm that extracts all the features of these speech digits for recognition. The MFCC features were used to be extracted from five speakers, so it can train the neural network by using the back-propagation algorithm and the rest 5 speakers were used to test the accuracy of the supervised neural network. This tested one time with Known Speaker with an accuracy of 96.32%, but for the Unknown speaker with an accuracy of 92%.

The Implementation of developing a Back-propagation Neural Network to recognize Bangla Speech into digit numbers was made in this project. In this project the Neural model recognize a single word (Ten Bangla digit numbers) where Bangla digits recognized one at a time. The training and testing of the Neural model were conducted from 10 speakers from different ages. 5 for training and 5 for testing the accuracy model. The age approximates of the speakers where ranging from 20 to 30 years old to have a clear identification of the model in its performance and accuracy. However, it needed to make a constant setup for the instrument of recording for the amplitude and frequency of the speech wave form. It was examined that different style or habit of the speaker can reproduce different results because of the affection on the system performance, so the speakers need to be trained on how to speak with the microphone. On the other hand, the amplitude and the speed of the speech causes a lot of errors to the extraction features process. Furthermore, the background noises of the environment around the speaker affect into the recognition system.

Table 2 Bangla accuracy

Bangla Digit	No. of samples for testing	Speaker Dependent (Known speakers' speech were used for testing)		Speaker Independent (Unknown speakers' speech were used for testing)	
		No. of Properly Recognized Digit	Recognition Rate (%)	No. of Properly Recognized Digit	Recognition Rate (%)
0	15	14	93.33%	14	93.33%
1	15	15	100%	14	93.33%
2	15	14	93.33%	13	86,.67%
3	15	15	100%	14	93.33%
4	15	15	100%	14	93.33%

5	15	14	93.33%	14	93.33%
6	15	15	100%	14	93.33%
7	15	13	86.67%	13	86.67%
8	15	14	93.33%	14	93.33%
9	15	14	93.33%	14	93.33%
Total	**150**	**143**	**96.32%**	**138**	**92%**

In conclusion, the Back-propagation Neural Network that has been developed and tested has achieved a recognizable result for recognizing single Bangla digit number at a time as word recognition implemented. The system was made to identify and classify ten isolated Bangla digits at a time and has only achieved 92% recognition accuracy for multiple speakers male and females.

2.2 Analysis of the Related Work

Table 3 Difference Between HMM and Neural Network

MODEL	FEATURE EXTRACTION	NO. OF SPEAKERS	ACCURACY
HMM (Hidden Markov Model	PLP or MFCC	15 speakers	92.2%
Neural Network to Jawi characters	LPC	10 (male) and 5(females)	92% and 88% for unknown
Neural Network to Bangla Digit	MFCC	5 speakers	95.32% and 91% for the unknown

According to Table 3 above the comparison in extraction of the voice features with Neural Network. Applied once by the LPC (Linear Predictive Coding) and another time with MFCC (Mel Frequency Cepstral Coefficient) which two of the most powerful speech analysis techniques. Which can provide an extremely accurate estimates of speech parameters. However, we can see in **row 2 (LPC)** the maximum accuracy reached for using LPC and known speakers is 95%, but in **row 3 (MFCC)** the maximum accuracy for using MFCC and known speakers is 96.32%; indicating that extraction is better by 1.32% which the difference can be increased if the number of samples increased to 225 sample rather than only using 150 sample. So, we can clearly identify MFCC is a better speech analysis technique to be implemented.

Furthermore, in Table 3 the comparison between model classification at the ASR (Automatic Speech Recognition) between HMM (Hidden Markov Model) and NN (Neural Network) with applying the same algorithm (MFCC) to extract the voice features of the speaker and analyse it. In applying the HMM it reached **85.8%** for known speaker and 3300 sample of audio for training the model. On the other hand, the NN reached **96.32%** for known speakers with only 150 sample. Which making a difference of **10.52%** with additional of using lesser quantity of samples for training the model.

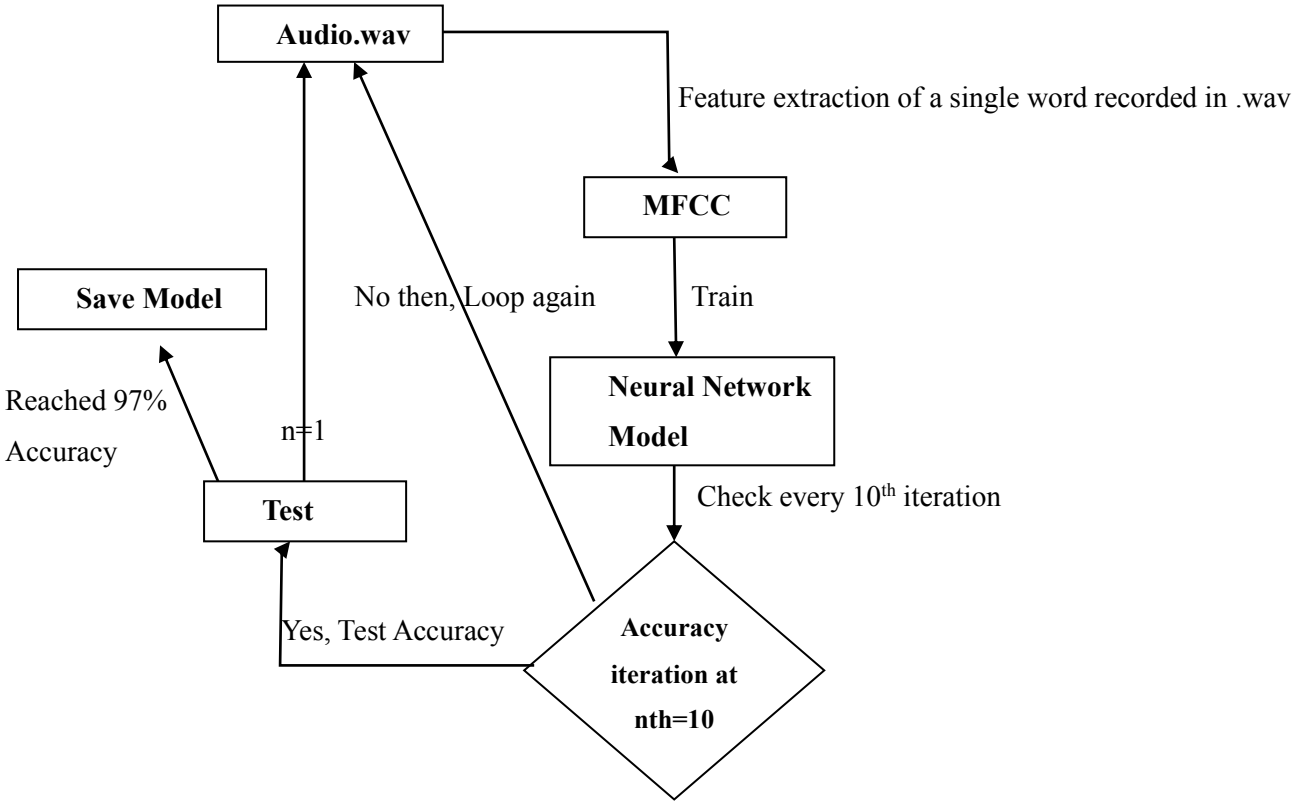
In conclusion, to reach the best architecture of ASR (Automatic Speech Recognition) to be applied is obviously using the NN (Neural Network) model as a classified as it can reach better values comparing with HMM (Hidden Markov Model). And the MFCC (Mel Frequency Cepstral Coefficient) to extract all the voice features of the speaker and fed it for the NN model to train and test.

3 Proposed solution

3.1 Solution Methodology

After analysing the related work, we can now identify that the best technique to make ASR (Automatic Speech Recognizer) is to extract the features of an audio signal by MFCC and fed it to the Neural Network to train and test on a known speaker for 10 spoken words, to reach the best accuracy it can reach as the previous projects.

Figure 2 Flow of Training the Neural model



As shown in Figure 2 to train the supervised model for every word in the dataset until reach the max accuracy without overfitting the neural model.

In Figure 3 below is the flow of how to extract the voice features by MFCC from a signal wave, for classification and recognition of the acoustic sound.

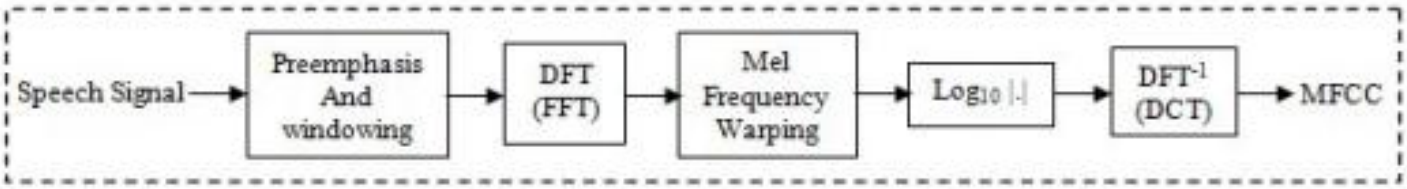


Figure 3 MFCC Implementation

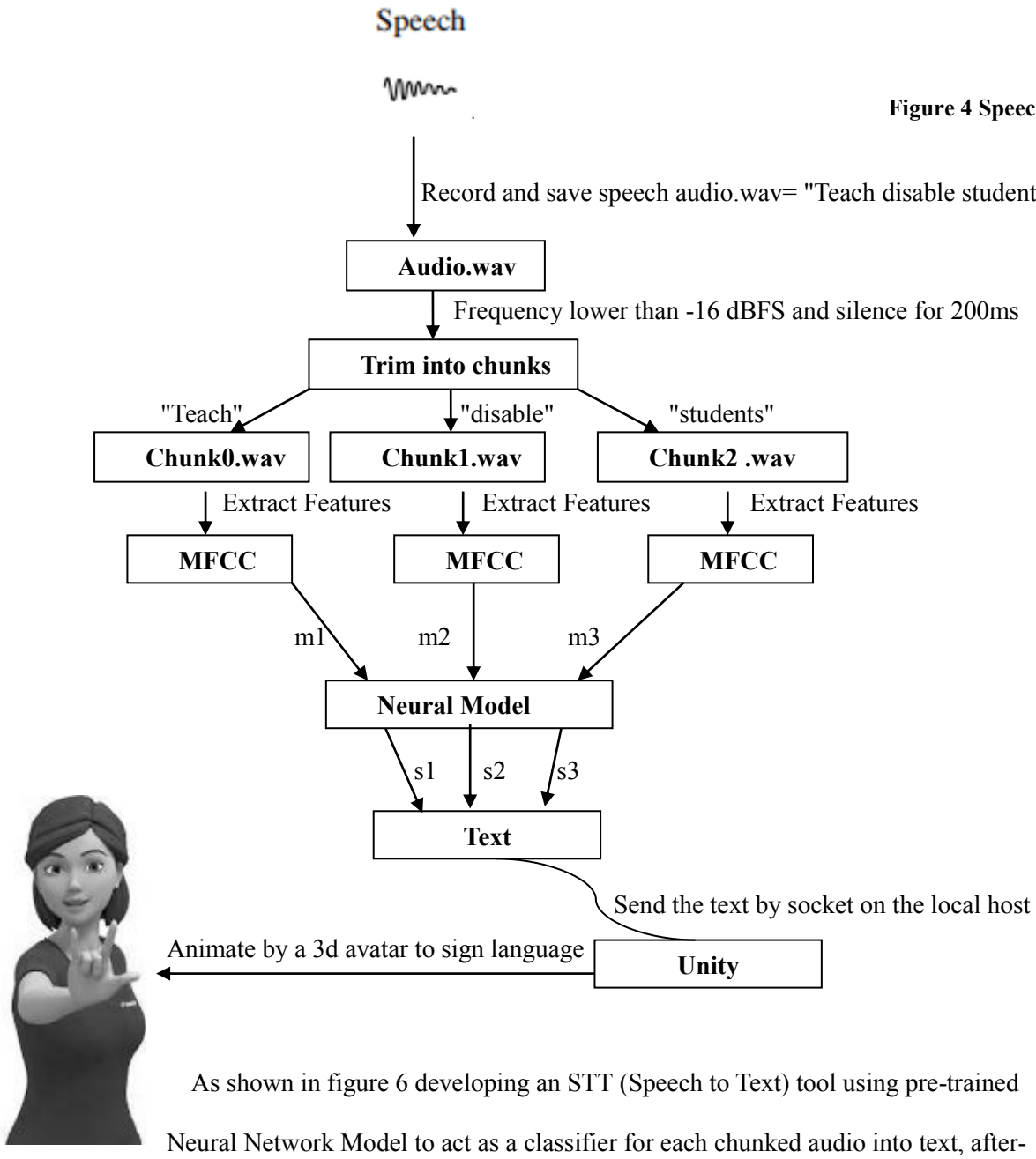


Figure 4 Speech To ASL Flow

As shown in figure 6 developing an STT (Speech to Text) tool using pre-trained Neural Network Model to act as a classifier for each chunked audio into text, afterwards animate onto the 3d avatar.

4 Implementation

4.1 Set up how to record audio with the Microphone

The Microphone is an instrument that converts acoustical energy (soundwaves) to electrical energy (the audio signal) [9]. At first, we need to set up the microphone to transform the soundwaves at an accurate range of frequency, so it can act like a human ear. Which the human ear of a regular person can detect from 20 Hertz to 20,000 Hertz [1]. To clarify more, the number of vibrations that are produced per second is called frequency as shown in the Figure 6. Secondly, we need to make provision of the amplitude (sound level) that ranges from 0 to 120 dB as shown in **Figure 5 Sound Level**.

Figure 5 Frequency Level

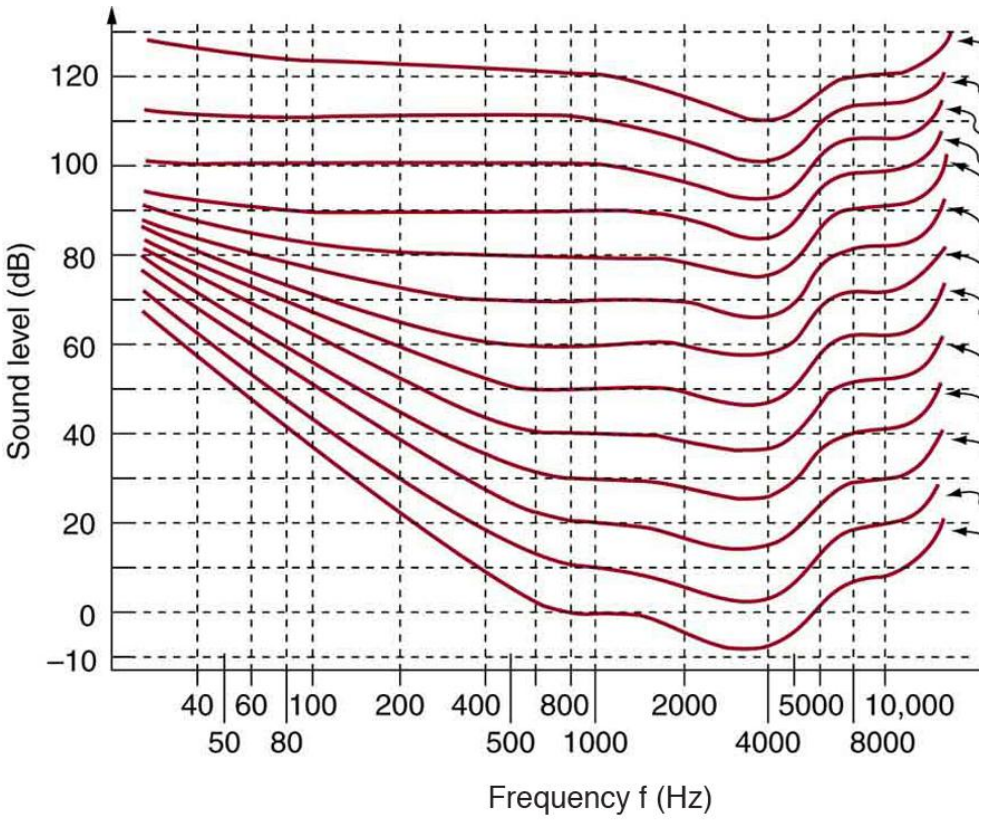
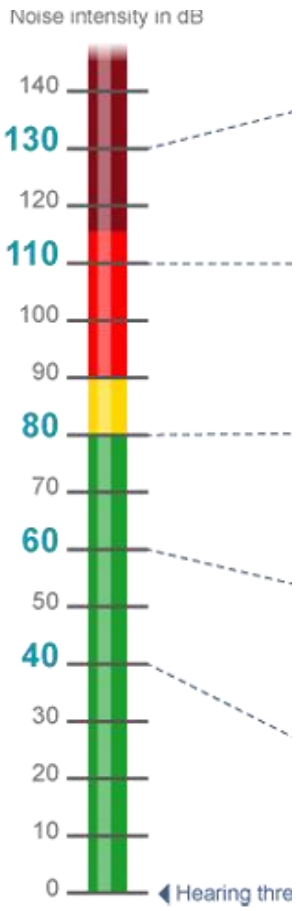


Figure 6 Sound Level



We need to set up the environment using Anaconda Navigator which is an open source distribution tool based on python and R programming, to set-up for you the environment for any library you want to install, initially used for Data Science and Machine Learning. Then made an environment to install all the required libraries. As we can see in Figure 7 the most important library is pyaudio that setup the Format, Channel, Rate, Chunk, and time of recording of the Microphone. The rate = 24,100 which means is the maximum to collect information is frequency 24,100 Hz; the channel means the microphone in which channel has been entered to the computer. The chunk works like a buffer that works collecting maximum of 1024 bytes, then re-initialize to release excessive on the computer processors.

Figure 7 Import Library Pyaudio

```
In [3]: import pyaudio
        import wave
        import os
        import matplotlib.pyplot as plt
        from IPython.display import Audio
        import numpy as np
        import scipy
        from pydub import AudioSegment
        from pydub.silence import split_on_silence
        FORMAT = pyaudio.paInt16
        CHANNELS = 1
        RATE = 24100  # 24100 is the frequency max which is in Khz human only hear between 20,000 and 22,000 Hz
        CHUNK = 1024
        RECORD_SECONDS = 50
        WAVE_OUTPUT_FILENAME = "file.wav"
```

As we seen in Figure 8 below we started recording audio for every 1024 bytes in the chunk and append those frames are appended in every loop until the Record Seconding finish. Furthermore, we converted also from Speech form into digital form, so the computer can recognize. The final step we need to make is trying to save the audio into a .wav format to be chunked as their any silence.

Figure 8 Start Recording audio

```
In [8]: audio = pyaudio.PyAudio()

In [9]: # start Recording
        stream = audio.open(format=FORMAT, channels=CHANNELS,
                            rate=RATE, input=True,
                            frames_per_buffer=CHUNK)
        print("recording...")
        frames = []

        for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
            data = stream.read(CHUNK)
            frames.append(data)
        print("finished recording")
        DATA_PATH = "./data/"
        stream.stop_stream()
        stream.close()
        audio.terminate()
        # stop Recording

        recording...
        finished recording
```

To save the wave format. We need to know where we want to save it. However, as we see in Figure 9 the file.wav audio and outputted the length of frames = 4971 which means is the numbers of frames that has been detected and saved as I was recording the word " Stop " .Then, to trim into chunks of audio "word :- Stop" as I recorded in this example, at Figure 9 ln[5] we used "split on silence" library which contain 3 parameters condition to cut at this point(1- The minimum silence required= 300 MS, 2- silence threshold (sound level)=-47 dB 3- Keep Silence=300ms) the silence threshold in negative because the digital wave from contain values both positive and negative values which means the threshold must be in negative value. However, in parameter 3 Keep silence is made to not trim the audio immediately, but too keep extra 300ms for a better recognition.

Figure 9 Saving and Trimming audio

In [4]:
```
#start Saving

waveFile = wave.open(WAVE_OUTPUT_FILENAME, 'wb')
waveFile.setnchannels(CHANNELS)
waveFile.setsampwidth(audio.get_sample_size(FORMAT))
waveFile.setframerate(RATE)
waveFile.writeframes(b''.join(frames))
waveFile.close()
sound_file = AudioSegment.from_wav("file.wav")
print(len(sound_file))

#Saved File.wav
```

```
4971
```

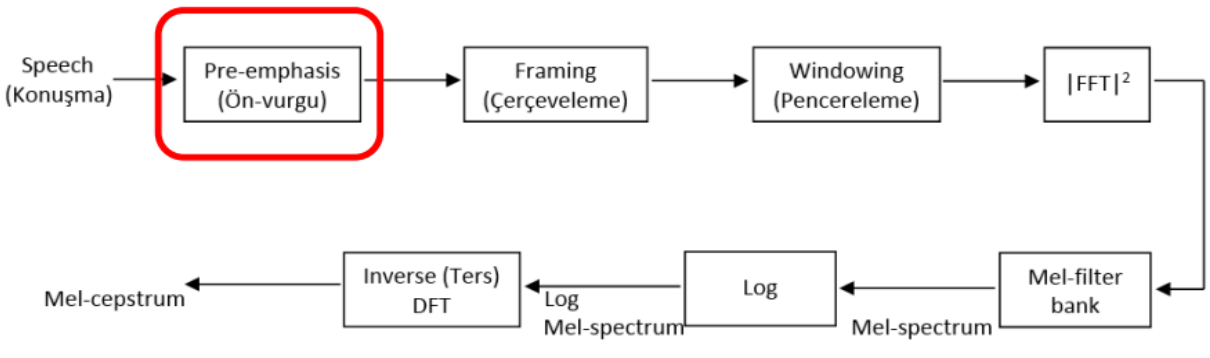
In [5]:
```
#Start Chunking
audio_chunks = split_on_silence(sound_file,
                        # must be silent for 220 ms
                        min_silence_len=300,

                        # consider it silent if quieter than -60 dBFS
                        silence_thresh=-47,
                        # keep silence for 200 and save it
                        keep_silence=300
                        )

print(len(audio_chunks))
for i, chunk in enumerate(audio_chunks):
    out_file = "Split_audio/chunk{0}.wav".format(i)
    print("exporting", out_file)
    chunk.export(out_file, format="wav")
#Stopped Trimming audio
```

```
1
exporting Split_audio/chunk0.wav
```

4.2 Extract voice features by MFCC

MFCC is [10] an approximate behaviour of a human auditory system, which make it one of the most popular methods of voice feature extraction techniques in speech

Figure 10 MFCC Steps of Extraction



MFCC steps in speech analysis

recognition. This technique as shown in Figure 10 is considered as frequency domain features which is more accurate than time domain features. Mel Frequency Cepstral Coefficient (MFCC) is a representation of the real cepstral of a windowed short time signal which is derived from the Fast Fourier Transform (FFT) that shows the power spectrum of the speech. The Mel scale is based on the human ear scale that can extracts parameters from the speech like the ones that are used by the human hearing system.

4.2.1 Applying Pre-emphasis

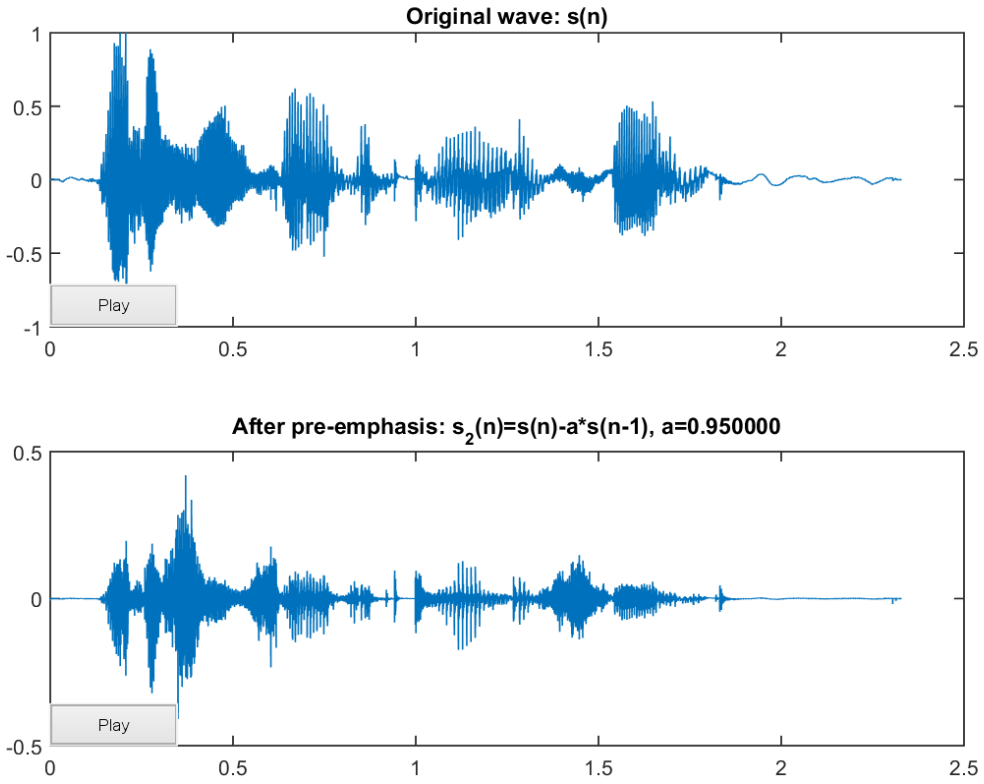
The goal of the pre-emphasis is to compensate the high-frequencies in the audio that was excessively supressed during the sound production mechanism of humans.

The speech signal is sent to a high pass filter to supress the high frequencies.: -

$$s2(n) = s(n) - a*s(n-1)$$

The s2(n) is the value of the wav after applying pre-emphasis algorithm [10] which is important because it tries to implement how the human ear system is more sensitive to lower frequencies than to higher frequencies sensitives, also to remove the background noises or any distortion of the sound that can affects the accuracy of the acoustic model.

Figure 11 Pre-emphasis flow

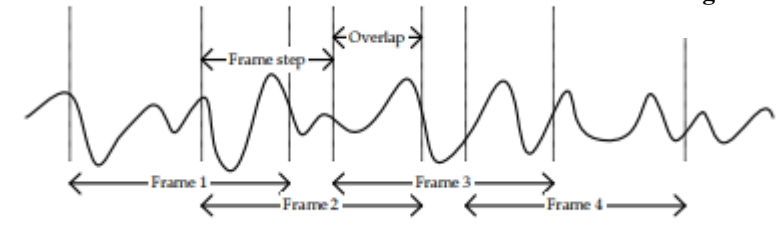


As shown in Figure 11 is a frequency (y-axis) to time (x-axis) presentation of the sound a typical digital speech wave form that shows before and after applying pre-emphasis that decreases the high frequency to lower frequency as shown in Figure 11.

4.2.2 Frame Blocking

In speech processing it is advantageous to divide the signal into frames to achieve stationarity. Because it is necessary to consider how to treat each frame with its own processing to avoid overlapping frames as we can see in Figure 12.

Figure 12 Framing



In Figure 12 shows how, a speech signal is divided into frames. Which each frame shares the first part [10] with the overlapping of 30ms of frame before it to not to lose any important information's from the speech signal. To calculate the time for each frame can be calculated as follows the *TFS* indicates the duration between the start time of the frame and the end of it. Then the *To* is the time overlapping between each frame. Length of the frame *TF1*=

$$\mathbf{TF1} = \boldsymbol{tfs} + \boldsymbol{To}$$

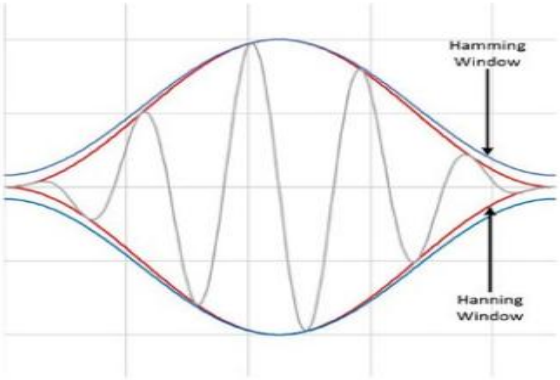
Finally, we will use reframing to combine all the separated frames into a desirable combined frame back to speech signal, to apply more algorithms afterwards.

4.2.3 Hamming and Windowing

We need to use windowing function like hamming. Which is a type of enhancement to the digital wave after using ADC (Analogue to Digital Converter).

As you know when you use the ADC on the whole signal wave is down sampled into a 32 sample or 22 sample, according to the specification you have applied. That causes a lot of leakage of information from the original signal wave. So, if we applied FFT (Fast Fourier Transformation) on the sample digital wave form, before using the windowing function it will causes a lot of wrong power spectrum frequencies. Because of having spike waves in the middle that increase the rate of errors in applying FFT (Fast Fourier Transformation) "It will be explained in the next section".

As we can see in the Figure 13 below the blue line represents the digital wave **before** applying hamming window, which shows the large spike in the wave structure of the inputted signal wave. Furthermore, at Figure 13 how the windowing functions affects in the red line the wave by stretching the spike to have a smoother and more decreased gradient across time domain to have a better feature extraction in the FFT (Fast Fourier Transformation) in the next phase.



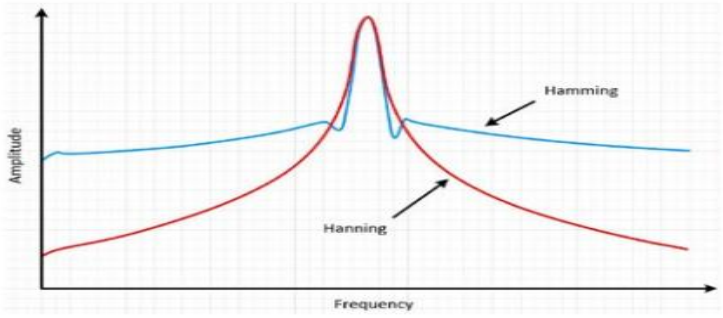


Figure 13 Hamming and Windowing

4.2.4 Fast Fourier Transformation (FFT)

Fourier transformation is a mathematical method of transforming a time domain into frequency domain, which is very useful for analysis of time dependent phenomena. This type of mathematical application is very useful for speech analysis implementation. It is also, important to recognize the frequency distribution of power spectrum in the sound because the human ear exercise that capacity in the hearing processes as well.

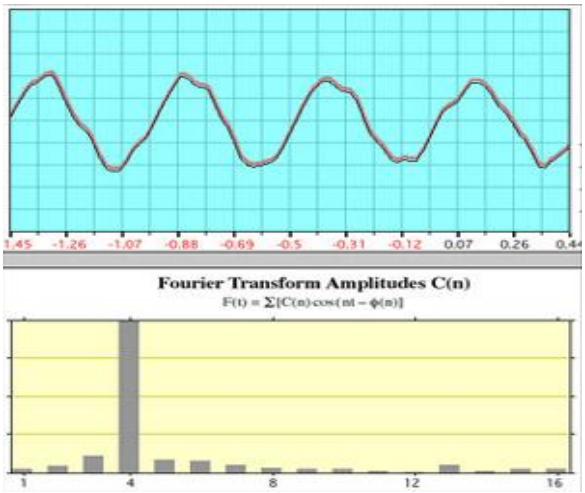
Figure 14 FFT equation

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx$$

As shown in Figure 14 this equation to apply Fast Fourier Transformation (FFT) where f(x) represent the time domain and x is the time of space along the x-axis which brings every f(x).

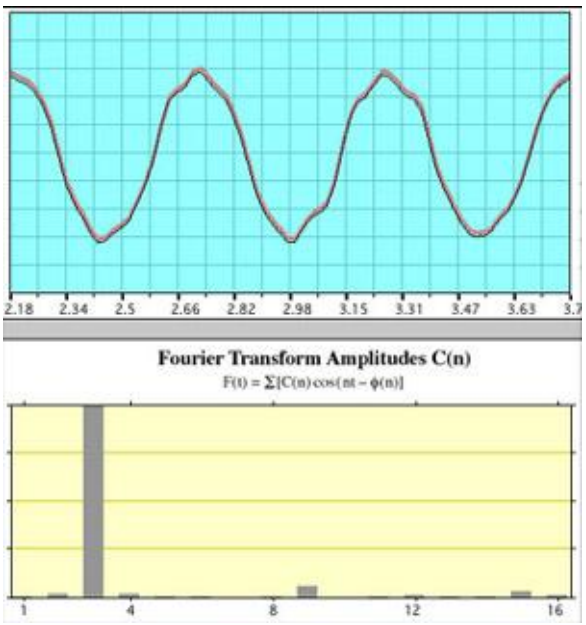
To give an example below a wave format of two whistles one is **A** and one is **B**. To see how different the frequency domain between the two sounds, but the same time domain.

Figure 15 example sound of Whistle A



To analyse the shown in Figure 15 is that when we whistle in a specific sound the time domain gives us the representation of how as the time changes the frequency is intervalley changing up and down. However, after applying FFT (Fast Fourier Transformation) we can analyse better that this sound has a large power spectrum at frequency 4 KHz.

Figure 16 example sound of Whistle B



In Figure 16 we whistle with a different sound that a human ear system can compile the difference between the two sounds easily, but the computer only compiles by numbers and figures. So, if compared the difference between the two waves in time domain we can see that there is very low difference in values as the two waves has the shape visually.

However, after applying FFT (power spectrum) we can classify easily between the two easily that one shows power energy. At Figure 15 large power spectrum is at frequency 4 KHz. However, in Figure 16 the power spectrum is at 3 KHz. Which can easily see the difference as a values and classification between the two for a computer system and making it preview the difference like human perception to different frequencies.

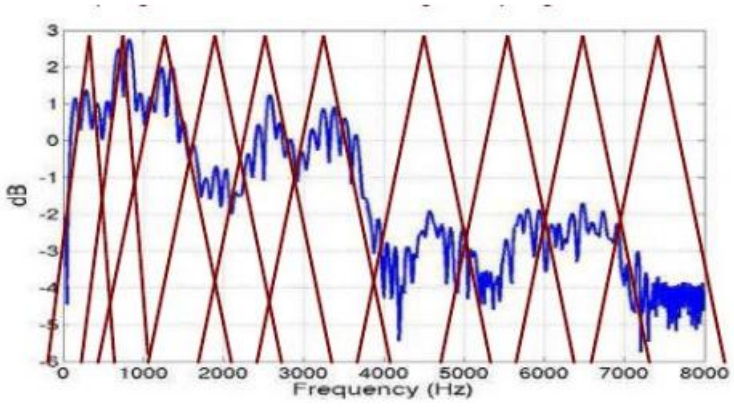
4.2.5 Mel Filter Bank

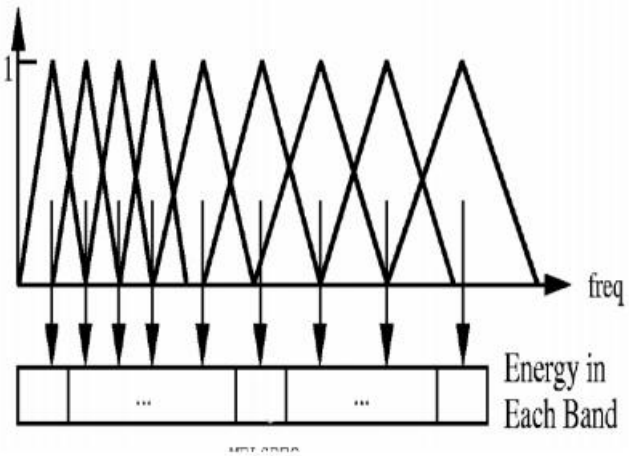
The Mel filter bank is important to the following reasons:

It applies the Mel-frequency scaling, which is perceptual to the human ears and how it functions in a more complicated way. It also, corresponds to better resolution at low frequencies and less importance to the high frequencies as we said earlier human ears are more sensitive to low frequencies.

Utilizing the triangular filter-bank as shown in Figure 17 makes a difference to capture the vitality at each basic band and gives a harsh guess of the range shape, as well as smooths the consonant structure. In hypothesis you may control on crude DFT bins, but at that point you're not reducing the dimensionality of your highlights - this is often the complete point of doing filter-bank examination, to capture the ghostly envelope.

Figure 17 Mel-Filter Bank Flow





$$F(Mel) = [2595 * \log 10 [1 + f] * 700]$$

Which *f* denotes the real Frequency, and Mel(f) denotes the perceived frequency after applying Mel- Filter Bank.

4.2.6 LOG Energy

Logarithm compresses dynamic range of values. Human response to signal level is logarithmic Humans less sensitive to slight differences in amplitude at high amplitudes than low amplitudes. Makes frequency estimates less sensitive to slight variations in input (power variation due to speaker's mouth moving closer to mike). Phase information not helpful in speech recognition.

To apply log function on the FFT we will use this formula in Figure 18: -

Figure 18 Log Equation to apply

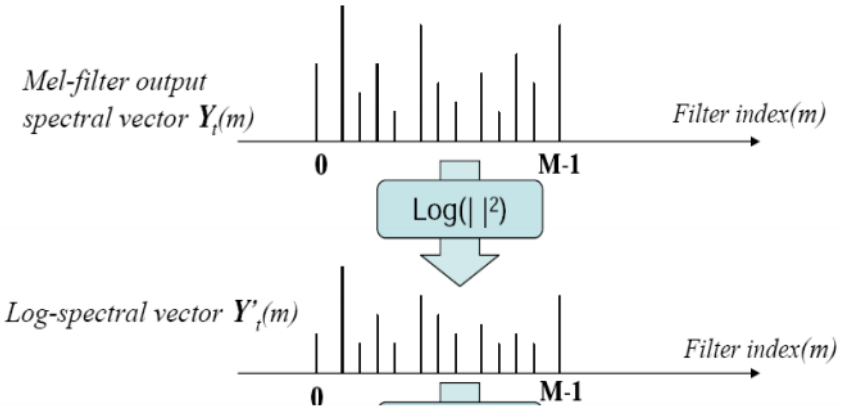
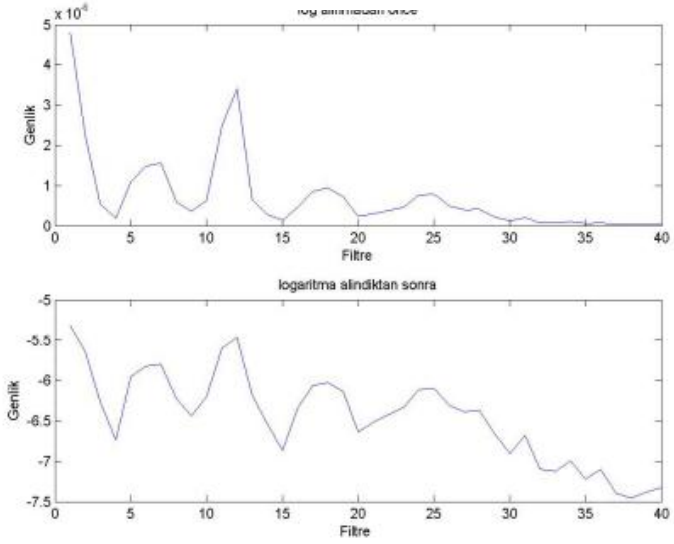


Figure 19 Logarithm applying

To see the difference after applying Logarithm we can see on the Figure 19:

As we can see in the Figure 19 the amplitude on the FFT wave before applying logarithm has



high amplitudes at special frequencies, but after applying the Log (x)^2 we can demonstrate that the low frequencies have a higher amplitude than the high frequencies. We needed to apply this method because of deforming the same system of the human ear that recognize the lower frequencies than the high frequencies.

4.2.7 Inverse Fast Fourier Transformation

Is applying inverse method of the Fast Fourier Transformation to transmit from frequency domain back to time domain by applying this equation in Figure 20.

Figure 20 FFT equation

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{j 2\pi u x} du$$

As we can F(u) is the values from the frequency domain need to be converted back to time domain to give acoustic vector of features which is the MFCC.

That is how to embed your wave signal into features of number to be able to train for the acoustic model using CNN (Convolutional Neural Network).

4.3 Dataset for training the acoustic model

The Dataset is important for making accurate results and training the acoustic model for the speech analysis give an accurate result afterwards.

The 27 words that has been recorded is given below in Table 4 for the on the acoustic model using Convolutional Neural Network (CNN) in the next phase of the implementation.

Table 4 All words recorded for training the model

WORD	NO. OF SAMPLES	NO. OF SPEAKERS
A	40	1 speaker
About	33	1 speaker
Be	36	1 speaker
Break	13	1 speaker
Complete	13	1 speaker
E-learning	30	1 speaker
From	39	1 speaker
Have	12	1 speaker
How	15	1 speaker
Internet	52	1 speaker
Is	31	1 speaker
Issues	31	1 speaker
Lecture	45	1 speaker
Major	38	1 speaker
No	13	1 speaker
On	39	1 speaker
Recognition	44	1 speaker
Security	32	1 speaker
Speech	26	1 speaker

Submission	30	1 speaker
Suffers	33	1 speaker
The	44	1 speaker
There	36	1 speaker
To	11	1 speaker
Today	17	1 speaker
Will	39	1 speaker
Works	25	1 speaker
Total	860	

As we can see in Table 4 above 860 sample of audio. Last phase we need to embed all this audio into one hot encoding of Mel Frequency Cepstral Coefficient (MFCC) encoding and save it as a NumPy file, that holds array of acoustic features for each word. This is made to have a better processing in the training and testing phase of the acoustic model.

Figure 21 Save MFCC Features

```
In [6]: DATA_PATH = "./data/"

In [7]: def get_labels(path=DATA_PATH):
            labels = os.listdir(path)
            label_indices = np.arange(0, len(labels))
            return labels, label_indices, to_categorical(label_indices)

In [8]: def save_data_to_array(path=DATA_PATH, max_len=11):
            labels, _, _ = get_labels(path)
            # for each class extarct it's mfcc features
            for label in labels:
                # Init mfcc vectors
                mfcc_vectors = []

                wavfiles = [path + label + '/' +
                            wavfile for wavfile in os.listdir(path + '/' + label)]
                for wavfile in tqdm(wavfiles, "Saving vectors of label - '{}'".format(label)):
                    mfcc = wav2mfcc(wavfile, max_len=max_len)
                    mfcc_vectors.append(mfcc)
                np.save(label + '.npy', mfcc_vectors)
```

The first thing as we can see in Figure 21 is variable DATA- ". /data" which where will be all the training set that has been saved by. Then, the first we will call is function "save_data_to_array" which first thing call's function "get labels". That has the passing argument =" path". This function is made to retrieve all the labels of each word and count their numbers, that is made as we see. That every dataset for each word is saved in a folder in its own. So, the function gets the name each folder (each word) by for looping in the folder data; in the same time keep counting how many words will be. Furthermore, after getting all the labels (all words) we want to extract the MFCC feature of each one and save all its features as a NumPy array. As we can see in Figure 19 MFCC vectors will append for each audio and then we are finishing this label we save it for that word ex: - "be" as "be.npy" containing all the MFCC features. This how to apply one hot encoding for each word and save it in NumPy array ready to be loaded for training and testing in the next phase.

4.4 Training an acoustic model by CNN (Convolutional Neural Network)

The CNN (Convolutional Neural Network) is a type of types of Neural Network that has the same architecture of any Neural architecture input layer from MFCC extraction, hidden layer to apply some weights of classification, and output layer for classification, but the CNN have some extra layers needed to make more specialization and processing of the data. The visualization of any human being can percept a large amount of information and data the second it recognizes that there is an image or hears a sound from a distance. Making our brains combined Neurons that functions every single one in its own way of calculating weights and features. Each Neuron is connected to the other neurons of the next layer, so they can communicate with each other for pattern recognition and visualization of images. That is how each one processed to cover the whole image for example. The receptive field in the biological vision system

of the humans. Each Neuron are placed in layers to be more arranged for inquiring to fetch visual data, detect simple patterns …etc. A CNN typically has five layers: - Input layer, Convolutional layer, pooling layer, fully connected layer, and Output layer as shown in Figure 22 below. In Table 5 is how the architecture of the acoustic model has been implemented. The implementation using python is made by using Tensor-Flow as backend to compile our acoustic model by the following: -

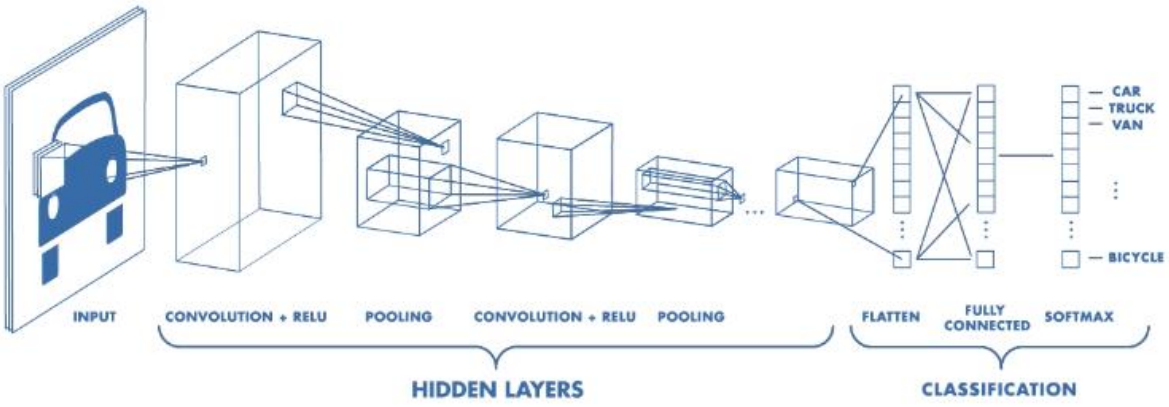


Figure 22 CNN Architecture

Table 5 architecture of the CNN

NAME OF THE LAYER	LAYER NO.	STRIDE	ACTIVATION BY
Input Layer	22 Neuron each contain 11 features.	N/A	N/A
Convolutional Layer	32 filter	3x3	Relu
Convolutional Layer	48 filter	3x3	Relu
Convolutional Layer	120 filter	3x3	Relu
Max-Pooling 2D		2x2	

Fully Connected Layer	1		
Dense Layer	128		Relu
Dropout by Prob=0.25			
Dense	64		Relu
Dropout by Prob=0.4			
Dense Layer	27 = num of classes (words)		SoftMax

4.4.1 Input Layer in the CNN

In the speech signal we arranged the MFCC vector for each audio will have a dimension of 22x11 2D, so the input layer in our architecture will contain 22 Neuron, each Neuron will have 1D array of 11 features to be passed into the convolutional layer.

4.4.2 Convolutional 2D Layer

In the Convolution Neural Network which known as the CNN the most important layer is the convolutional layer as it carries a very important role in how the input data is filtered and minimized.

This layer first thing happens is that have two matrices one is the input features matrix and the another is the kernel matrix. which we will see in the Figure 22 below how it is being applied. The kernel matrix is 2D dimensional smaller than the inputted matrix but must have the same depth. For example, the input matrix is 22x11x1. The kernel matrix is 3x3x1. The kernel can have different width and height but must have the same depth as the input.

However, during the forward pass of the network for every iteration, the kernel project over the whole input matrix vertically and horizontally, but there is a parameter called “Stride” must be initialized first. Because the stride makes the kernel matrix jumps how many pixels vertically or horizontally. As we see in the Figure 22 below the stride is equal to 1. As it only moves one pixel to the right. And when it finishes the whole row, it jumps one pixel vertically.

If we have a speech waveform with an input of size Height x Width x Depth

and number of kernels will be applied with a spatial size of AxA with stride S, then the size of output volume can be determined by the following formula:

Out(width)=Input(width)-Kernel(width)+1

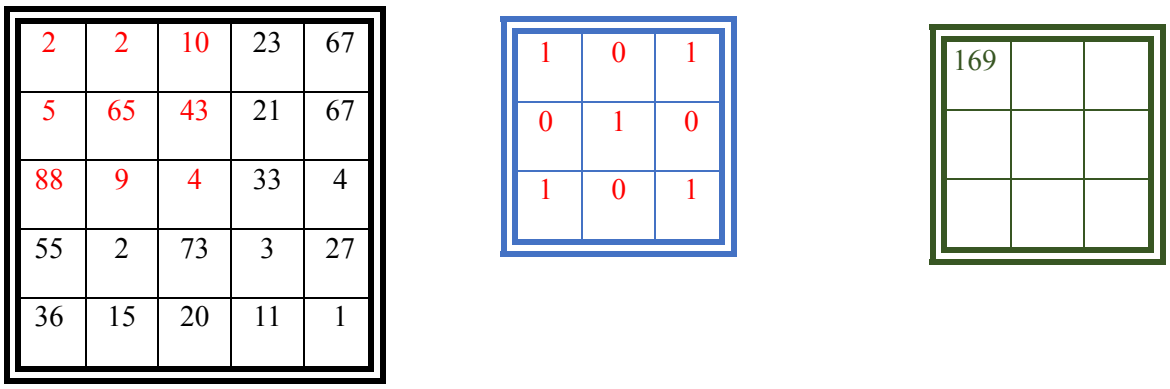
Out(height)=Input(height)-Kernel(height)+1

To give an example below of how to apply the convolutional operation in the Figure 23 below: -

(5-3+1) =3 width | (5-3+1) =3 height is the size of the Feature map = 3x3

2D matrix 5x5 **I** Kernel size 3x3 stride=1**H** Feature Map **F**

Figure 23 example of applying convolutional filter



F1 in the feature map is calculated as = [I11 * H11] + [I12 * H12] + [I13 * H13] + [I21 * H21] + [I22 * H22] + [I23 * H23] + [I31 * H31] + [I32 * H32] + [I33 * H33] = 169.

The same for all the feature maps that has been extracted.

4.4.3 Max Pooling Layer

The pooling layer is a layer that dense and condensate the numbers of feature maps that has been extracted. This helps in decreasing the spatial size of an image from its actual representation, which decreases the number of iterations and layers from the feature maps to decrease also the required amount of propagation and arrays holding features and weights into smaller array. The pooling operation is been applied on every slice of the feature map made by the Convolutional layer output separately.

Figure 24 Rule for the pooling layer

$$\textbf{Out (width) = Input(width) – h(width) +1 / stride}$$

Layer of 3 x 3 with a pooling of 2x2 with a stride =1.

[3-2] +1/1 = 2 width [3-2] +1/1=1 height

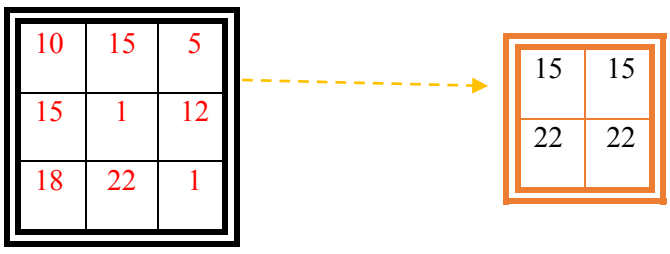


Figure 25 example of Max Pooling

As we can see in the Figure 25 an example shows above after applying max pooling we get the most important features and pass it to the next layer. Taking the maximum value as the max pooling filter passes across the matrix to bring all the important features.

4.4.4 Fully connected layer

Neurons in this layer will be concatenated to have full connectivity with all neurons in the in a one large vector to connect all the features together and can classify afterwards.

The Fully connected layer helps map the representation between the input layer and the output layer because it connects after applying filtering in the convolutional layer as we seen above and Max Pooling layer the flattening all the feature maps into one dimensional vector that holds all the features that has been extracted. Then a normal Neural Network can be implemented by using hidden layers for classification by SoftMax for classifications.

4.4.5 Activation by Relu and SoftMax

Relu activating function known as Rectified Linear Unit has become very popular in the last few years. The way it computes the function is by $f(x)=\max(0,x)$ x is the value of the features (x). In other words, this activation function is cuts off any negative value below zero as it compares them by maximization function.

The Relu functions unfortunately has bad points in implementation is that Relu can be very hustled and weak during training of the CNN model. The gradient can be increased through the learning rate of the CNN model as it can update it in such a way that the neuron doesn't being updated in every iteration and continue to hold the same values every learning iteration (Epoch). However, we can justify this work by adjusting the learning rate of our acoustic model during learning on the dataset. To set a proper learning rate.

The SoftMax layer at last is made to classify the last hidden layer to classification of in our implementation is 27 words which means 27 neurons and who has the high-

est probability is classified to has the most reliable answer given by the acoustic model.

4.5 Animating ASL (American Sign Language) by the Avatar by Unity 3D

In Implementation, we installed “Unity” as its editing includes multiple tools that enable fast editing in our development structures alongside using “C# scripting” because it is easier for development. In addition, the Play mode feature quick previews of our project in minimum time.

- All-in-one space**:** a range of artist-friendly tools for designing 3-dimensional experiences, as well as a strong suite of developer tools for implementing game strategies and high-quality performance.

4.5.1 Download an Avatar from Unity Store

By using the Unity asset store to download the avatar that have a proper structure of joints and human hydro system, so it can animate in a more accurate way of how human system make ASL (American Sign Language).

4.5.2 C# script to listen for any message from the server to receive

By using the TCP client receiver make a code to listen on a specific IP address of the Server (tutor who talks through the microphone).

As we can see in Figure 26 we will receive the text in bytes we will transform those bytes into string, so it can be displayed through Unity 3D. Then will send it to another script to animate each word separately by for-looping the whole sentence that has been received.

When in the condition changed is true it will call the function animation scripty to start reading the whole message (sentence) to animate.

```
private void ListenForIncommingRequests()
{
    try
    {
        // Create listener on localhost port 8052.
        tcpListener = new TcpListener(IPAddress.Parse(this.ip), 8052);
        tcpListener.Start();
        Debug.Log("Server is listening");
        Byte[] bytes = new Byte[1024];
        while (true)
        {
            using (connectedTcpClient = tcpListener.AcceptTcpClient())
            {
                // Get a stream object for reading
                using (NetworkStream stream = connectedTcpClient.GetStre
                {
                    int length;
                    // Read incomming stream into byte arrary.
                    while ((length = stream.Read(bytes, 0, bytes.Length)
                    {
                        var incommingData = new byte[length];
                        Array.Copy(bytes, 0, incommingData, 0, length);
                        // Convert byte array to string message.
                        string clientMessage = Encoding.ASCII.GetString(
                        mess = clientMessage;
                        changed = true;
                    }
                }
            }
        }
    }
```

Figure 26 code of listening client

4.5.3 C# script to animate the avatar

We need to make an animator controller, so it can control how animations can run so when the server sends the word. The script will receive a sentence like “take a break” each word will be split it into w1=” take”, w2=” a”, and w3=” break. Then by using a function called I Enumerator wait time (). Which is a function has yield that wait for seconds as we can see in the Figure 27. Making a wait time between every animation for 1 second.

Figure 27 code for animation

```
public void set_Text(string all_text)
{
    counter = 0;
    all_message = all_text;
    this.all_message_cutted = all_message.Split(' ');
    start_cutting = true;
}
IEnumerator waitTime()
{
    this.waits = true;
        if (!anim_work)
        {
            this.anim_work = true;
            anim.SetTrigger(text_cutted.text);
        }
    yield return new WaitForSeconds(this.tim);
    anim.SetTrigger("idle");
    this.anim_work = false;
    this.waits = false;

}
```

When the wait functions return. It will trigger back to the idle state, so it can be ready for the next animation. That how to animate fully between each animation by using the animator controller.

4.6 Send Broadcast message by TCP Sockets to clients

The most common type of sockets is the client-server connection which the server peer creates its network and waits for the clients to accept on their IP address. Which in the next part we will see how to use it by python.

The first thing is to install socket library by anaconda environment by running the follow command: - “conda install socket” to import all the required libraries.

We first need to run the server, so it can accept client sockets afterwards, that is by using first “socket. socket ()” to identify the type of socket you are using which by sending bytes of data. Then identify all the IP’s and which port they will be receiving you want to send their message to when you start recording audio and classifying your speech. As we can see the full picture of sending a message to clients. We need at last to close the connection at the end of the code for safety causes of your computer processers. As we can see in the Figure 28 below after finishing the message we need to transform it to “utf-8” bytes.

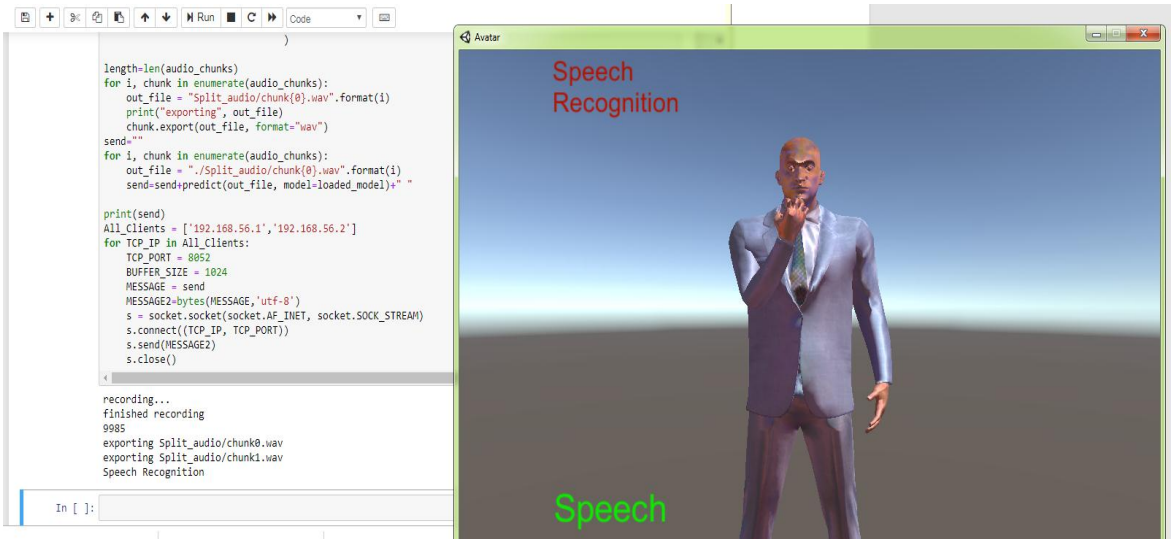
Figure 28 Send TCP Message

```
In [ ]: TCP_IP = '127.0.0.1'
        TCP_PORT = 8052
        BUFFER_SIZE = 1024
        MESSAGE = send
        MESSAGE2=bytes(MESSAGE,'utf-8')
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.connect((TCP_IP, TCP_PORT))
        s.send(MESSAGE2)
        s.close()
```

4.7 Run the Speech To ASL (American Sign Language) system

First, we need to run the client (Unity 3D Avatar) to listen on its own IP address if there any message is sent from the server. We need to make any user wants to see the avatar animation will have to run the unity program. After that is try run the speech recognition to start recording for 10 seconds for the recorder. Furthermore, the trained acoustic model will be inputted all the chunked audio that has been recoded and will predict which class it belongs to until it forms a whole message, then it will send to all clients IP addresses' that has been saved so they can receive the message and the avatar starts to animate to each word has been sent through. In the Figure 29 below, you can see a screenshot of the actual program running as I was recording "Speech Recognition".

Figure 29 screen shot of program running



In the Figure 28 was an example of saying Speech Recognition and how the trimming has been applied and separated into 2 chunks 0 and 1. Then, the acoustic model classified that output from audio chunk 0 as "Speech" and chunk 0 as "Recognition. Finally, on the right side showing the avatar as he animates each word separately.

5 Testing and evaluation

5.1 Testing

We tested the Convolutional Neural Network as we can see in the table below for 1 known speaker to compare the accuracy of the acoustic model how it reacts to new acoustic features enters to the neural model. We tested for 15 sample for each word of we have trained on. In Table 6 the results of the test cases we applied.

Table 6 Testing accuracy for Known Speaker

Word Name	No. of samples for testing	Speaker Dependent (Known speaker's speech were used for testing)	
		No. of Properly Recognized Word	Recognition Rate (%)
a	15	15	100
About	15	15	100
Be	15	15	100
Break	15	15	100
Complete	15	14	93.33
E-learning	15	15	100
From	15	15	100
Have	15	14	93.33
How	15	14	93.33
Internet	15	15	100
Is	15	15	100
Issues	15	14	93.33
Lecture	15	15	100

Major	15	15	100%
No	15	13	86.66
On	15	15	100%
Recognition	15	15	100%
Security	15	15	100%
Speech	15	15	100%
Submission	15	15	100%
Suffers	15	14	93.33%
The	15	13	86.66%
There	15	14	93.33%
To	15	14	93.33%
Today	15	15	100%
Will	15	15	100%
Works	15	15	100%
Total	**405**	**394**	**97%**

5.2 Evaluation

If we compared the result of the unknown speaker accuracy between my results and the related work results as shown in Table 7 we can identify how the accuracy reached better for my samples as it has been tested for any small sample, which if I give a larger sample of training for example 10,000 audio the accuracy and the learning curve can reach a better accuracy.

Table 7 Accuracy comparison between acoustic models

MODEL	**FEATURE EXTRACTION**	**NO. OF SPEAKERS**	**ACCURACY**
HMM (Hidden Markov Model	PLP or MFCC	15 speakers	92.2%
Neural Network to Jawi characters	LPC	10 (male) and 5(females)	92% and 88% for unknown
Neural Network to Bangla Digit	MFCC	5 speakers	95.32% and 91% for the unknown
CNN (Convolutional Neural Network) **Speech To ASL**	MFCC	1 speaker	97% for the known

The CNN model has been for its fast training of models with high precision, but as we can see in the related work each has large datasets maximum of 10,000 samples….
Etc. and with small classification for each work. In the HMM model it recognized only 10 classes of words, In Jawi implementation it had only 15 words of Jawi to classify with only 92% for known speakers. However, in the Bangla digit recognition it only recognizes 10 numbers, but has large accuracy, but not like our acoustic model of reaching 97% of accuracy with small dataset.

6 Conclusions and Future Work

6.1 Summary

The initial step is recording from the microphone for 10s from the speaker between a range of frequencies from 10,000hz and 24,000hz, due to the human ear can listen only between that range. The next step is saving it as file.wav format file (saving it in a .wav format can make us open and modify it later). Afterwards, we should load the file.wav and trim it into chunks of each word said in the recorded file. Moreover, we will extract the MFCC feature of every chunk audio (containing a word) and enter it as an input for the trained CNN model to transform it into text. Finally, we will send the whole sentence for each client on the network as a broadcast message to be animated into ASL (American Sign Language). We reached an accurate percent of 97% for 27 trained words and 18 animated words has been made by the avatar. This can be viewed as a proper solution for anyone who has a problem in communicating with deaf people; either by learning ASL easily through this tool, or by communicating directly through the microphone.

6.2 Future Work

We need to have a broader type of speakers to generalize the training on the acoustic model. The words can be classified by the acoustic model according to the accent the speaker has as he records. Also, we need to train on larger datasets of sample for each word. For example: we will make each speaker record one word 1,000 times, these will be the audio samples of one speaker, and we will get 10 speakers to do the same giving a total of 10,000 audio sample for each word to be trained properly.

However, the step of trimming function of an audio that is applied needs major improvement because the speaker needs to be silent for 200ms while recording each word. For instance: “Speech” | silence for =200ms | “Recognition”, to trim it properly the whole sentence without concatenating the two audios together in one audio, which will give wrong result from the trained model.

Finally, we need to have a better result for the sentence fragmentation and structure. This can be done by using NLP (Natural Language Processing) to parse syntactically complex sentences. This leads in making the output understandable to represent how the human language has specific grammar structure in every sentence. For example: “Speech Recognition the” has been given from the acoustic model, but as for the user it is misunderstood and wrong according to the grammar structure in the English language [11] .

References

[1] Choudhary, R. (2003, March 30). *The Physics FactBook*. Retrieved from hypertextbook: https://hypertextbook.com/facts/2003/ChrisDAmbrose.shtml

[2] Dave, N. (2013, July 6). *International Journal for Advance Research In Engineering and Technology*. Retrieved from Feature Extraction Methods LPC, PLP and MFCC In Speech Recognition: www.ijarte.org

[3] Gales, M. (2008). The Application of Hidden Markov Models in Speech Recognition. Cambridge: Foundations and Trends.

[4] Hossain, M. A., Rahman, M. M., Prodhan, U. K., & Khan, M. F. (2013). Implementation Of Back-Propagation Neural. *International Journal of Information Sciences and Techniques (IJIST)*, 1-9.

[5] Miller, L. (2017, May 22). *STAT*. Retrieved from 'I was panicked' Deaf patients struggle to get interpreters in medical emergencies: https://www.statnews.com/2017/05/22/deaf-patients-interpreters/

[6] Othman, Z. A., Razak, Z., & Abdullah, N. A. (2009). *Jawi Character Speech-To-Text Engine Using Linear Predictive and Neural.* y, Academy of Islamic Studies, Computer Science. Kuala Lumpur: Third Asia International Confrence.

[7] Rabinar, L. (2004). Automatic Speech Recognition – A Brief History of the Technology. Atlanta: Georgia Institute of Technology,.

[8] Shalabi, M. (2018, March 28). *New Law in Egypt Supports Disabled Students*. Retrieved from Al-Fanar Media:https://www.al-fanarmedia.org/2018/03/new-law-in-egypt-supports-disabled-students/

[9] Woodford, C. (2019, January 31). *Microphones*. Retrieved from ExplainThatStiff: https://www.explainthatstuff.com/microphones.html

[10] Davis, S. Mermelstein, P. (1980) *Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences*. In IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 28 No. 4, pp. 357-366

[11] Berg, G. (2009, June 21). *A Connectionist Parser with Recursive Sentence*. Retrieved from semanticsscholar: https://pdfs.semanticscholar.org/f48d/6233238c9da9cf36cbcdc9e5d00cf6e1b4b0.pdf

[12]. L. R. Rabiner, S. E. Levinson, A. E. Rosenberg and J. G. Wilpon, Speaker Independent Recognition of Isolated Words Using Clustering Techniques, IEEE Trans. Acoustics, Speech and Signal Proc., Vol. Assp-27, pp. 336-349, Aug. 1979.