Project Report

Sign Language Recognition Using Neural Network

CSE498R.01
Summer 2020

M.J. Darad, Rubayet Zaman
ID# 1620520042, 1620684042
[m.j.darad, rubayet.zaman]@northsouth.edu

Advisor: Mohammad Ashrafuzzaman Khan

Department of Computer Science and Engineering,
North South University

GitHub https://github.com/juneju-darad/CSE498R

Contents

List Of Images

1

Abstract

According to the statistics of the World Federation of the Deaf and the World Health Organization, approximately 70 million people in the world are deaf– mute. 32 million of these individuals are children. Sign language is the main language used by the deaf and mute to communicate with others. Many physically sound people don't understand sign language. When deaf and mute people try to communicate with us, they face difficulties communicating with us.Therefore, We prepared an application which will eliminate the difficulties of deaf and mute. By using machine learning algorithms, we build a system which can recognize sign language from signed hand gestures from any person. We used text to speech API to deliver the letter/sentence.

2

Introduction

2.1 Motivation

Communication is one of the important requirements for survival in society. Deaf and dumb people communicate among themselves using sign language but normal people find it difficult to understand their language. Many physically sound people don't know sign language. Our project aims to eliminate the difficulties. Our project will elminate the communication gap between normal people and deaf and dumb people using sign language. Effective extension of this project to words and common expressions may not only make the deaf and dumb people communicate faster and easier with the outer world, but also provide a boost in developing autonomous systems for understanding and aiding them.

2.2 Project Details

Communication is very important to human beings for living, as it allows us to express ourselves to other people. We communicate through speech, visual aids, body language, reading, writing or gesture. Speech is one of the most commonly used media to communicate with others among them. According to the statistics of the World Federation of the Deaf and the World Health Organization, approximately 70 million people in the world are deaf–mute. 32 million of these individuals are children. Sign language is the main language used by the deaf and mute to communicate with others. Unfortunately, for the deaf and dumb people, there is a communication gap with us. Visual or an interpreter, are used for communicating with them. However, these methods can't be used in an emergency. Sign Language mainly uses manual communication to deliver meaning to the deaf and dumb people. This process requires simultaneously combining hand shapes, orientations and movement of the hands, arms or body to express the speaker's thoughts to the deaf and mute people. Sign Language is consists of finger spelling, this spells out words character by character, and word level association which involves hand gestures that deliver the word meaning to the the deaf and dumb people. Finger-spelling/hand gesture is a main tool in sign language, as it enables the communication of names, addresses and other words that do not carry a meaning in word level association. But it is a matter of sorrow that finger spelling is not widely used and it is challenging to understand and difficult to use for physically sound people. Moreover, there is no universal sign language and very few people know the sign language. When deaf and mute people try to communicate with us, they face difficult situations to communicate with us.. A system for sign language that recognition hand gesture/finger spelling can solve this problem. Our Project for deaf and dumb people to eliminate the difficulties. We use machine learning. By using machine learning algorithms, we build a system which can recognize sign language from signed hand gestures from any person.

2.3 Project Goal

Our Project is Sign language Recognition using Neural Network. Sign language plays a great role for people with hearing difficulties as communication media. This language tool is made for overcoming problems in communication with deaf people. Deaf people use sign language to communicate with each other. They face difficulties when they are communicating with physically sound people. Our project goal is to eliminate this difficulty

3

Literature Review

3.1 Machine learning

In our project machine learning is needed. We will use machine learning to build a model which can recognize the Sign language performed by the deaf dumb people. We will use machine learning algorithms to build models.

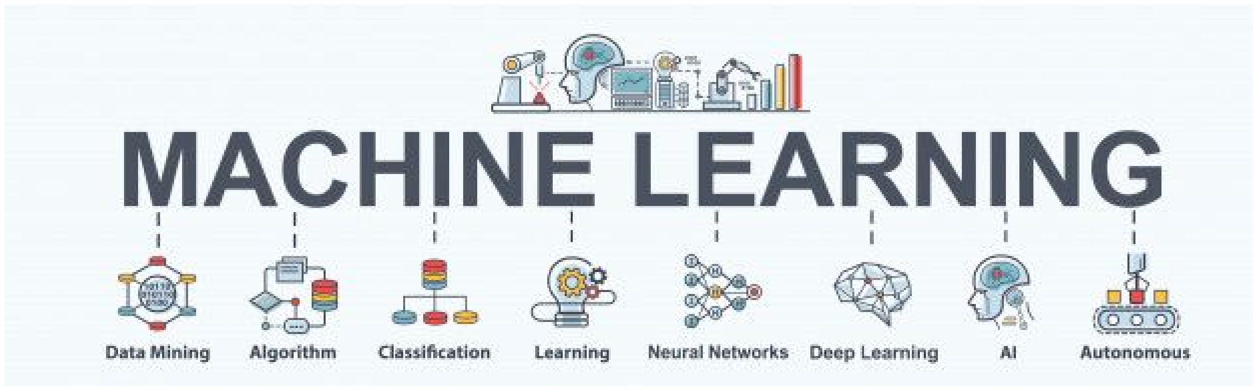


Figure 1. Machine Learning

3.2 Neural Networks

Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated. Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabeled data according to similarities among the example inputs, and they classify data when they have a labeled dataset to train on. (Neural networks can also extract features that are fed to other algorithms for clustering and classification; so you can think of deep neural networks as components of larger machine-learning applications involving algorithms for reinforcement learning, classification and regression.)

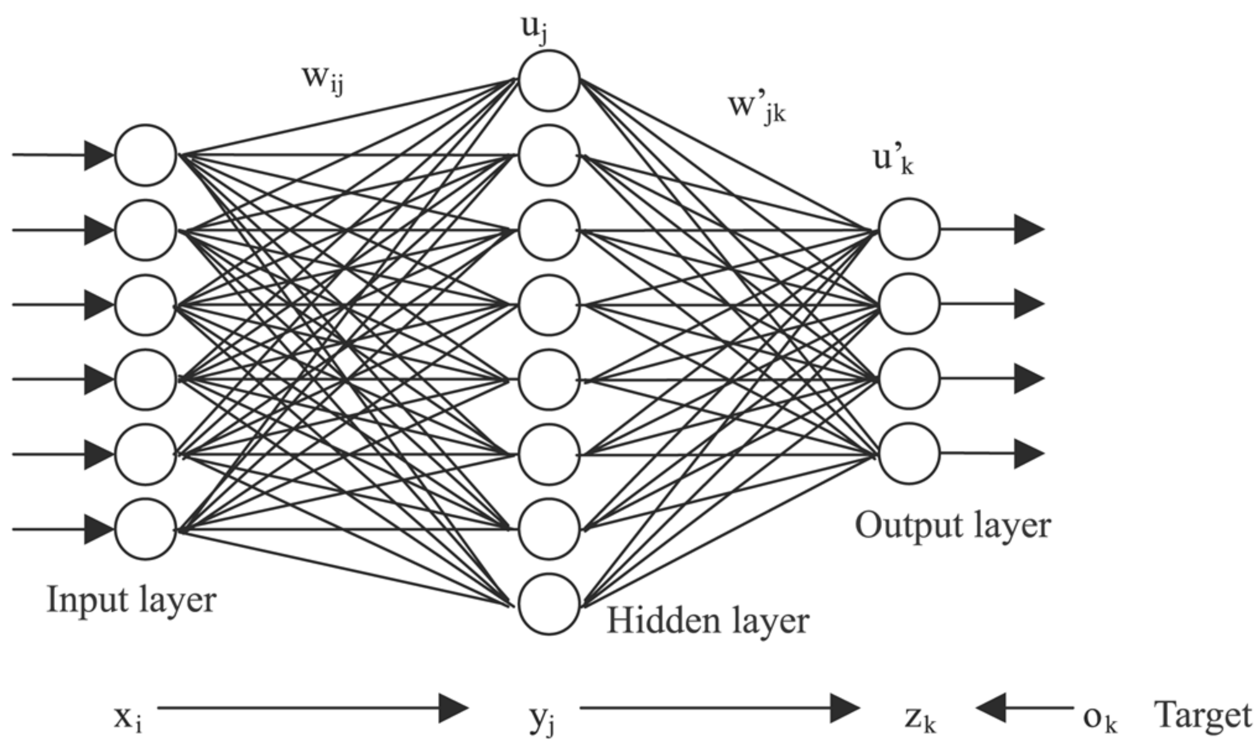


Figure 2. Neural Network

3.3 Python

Python is a programming language Which we will use to develop a model by machine learning algorithm. We will also use python for development of our web application. Python is a powerful programming language which is needed in our project.

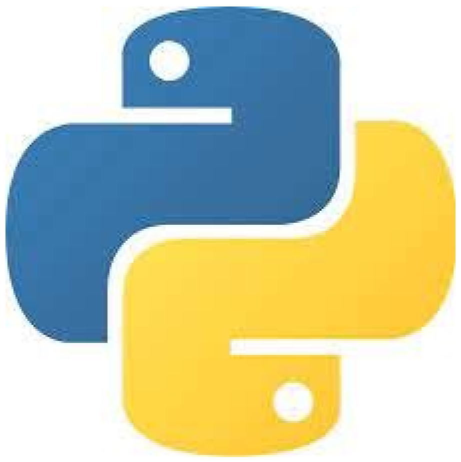


Figure 3. Python

3.4 Libraries

Pandas:

Pandas is an open-source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

NumPy:

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object and tools for working with these arrays.

Scikit-learn:

It is a free software machine learning library for the Python programming language that provides many unsupervised and supervised learning algorithms. It's built upon some of the technology you might already be familiar with, like NumPy, pandas, and Matplotlib.

Jupyter Notebook:

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.

Uses include data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

3.5 TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks.

Figure 4. Tensorflow

It is a symbolic math library, and is also used for machine learning applications such as neural networks.

3.6 Flask web framework

Flask is a micro web framework written in Python. We will use flask web framework for development of web applications. By using a flask web framework, we can build a reliable, scalable and maintainable web application. So, we need good knowledge about the Flask web framework for this project.

Figure 5. Flask

3.7 JavaScript

JavaScript is a high-level interpreted programming language, it just compiled in time and multi-paradigm. JavaScript programming language has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

Figure 6. JavaScript

In our web application we have text to speech, input from camera, output from model etc. These works should be in real time So, JavaScript is mandatory. So, we need to learn JavaScript.

4

Why the problem is interesting

Our project is Sign language recognition using conventional neural network. Sign language plays great role in deaf & dump communities. According to the world health Organization around 466 million people around the world have hearing loss. Deaf and dumb people around the world uses sign language for their communication. This language made for overcoming problem in communication with deaf & dumb people. Deaf people use sign language to communicate with each other. They face difficulties when they are communicating with physically sound people. Because Physically sound people don’t know the sign language. Physically sound people have their own language to communicate with other. Our project goal is to eliminate this difficulty. Our project will help the deaf & dumb people to communicate with us. In our project we have a web application. By using web application deaf people can translate their sign language to English language easily. It will eliminate the problem of communication. Our project is an eco friendly project. Our project is online based project, any deaf & dumb people can access our web application through Internet. The project doesn’t have any impact on air, soil, or water. Our project doesn’t have any harmful effect to any human or human health. For the use of our project user need Internet accessibility. Without good internet connection to enter website. There can be occur another problem. We can face overload problem. Too many people can visit website at a time. We should able to handle overload problem of website. We need to consider the typing speed of user is greater than the sign language speed of user. If our system took too many times to give output then we need to consider this problem. We use text to speech api in our project we need to ensure that our text to speech api working properly. Our project is for the benefit of human specially for the deaf & dumb people.

5

Methodology

5.1 Technical Design: Module Level

Functional block diagram demonstrates the external and internal components of our project. The flow of the functions start from left most block as input, upper block is for software processing the input, right most block is used to output for the input and take input from microphone, which is processed in lower block and finally back to left most block to display the audio from microphone. For every block the name of the component before the comma is used for input and the name of the component after the comma is for output. The left most and right most blocks are external devices and upper and lower blocks are used for internal software.

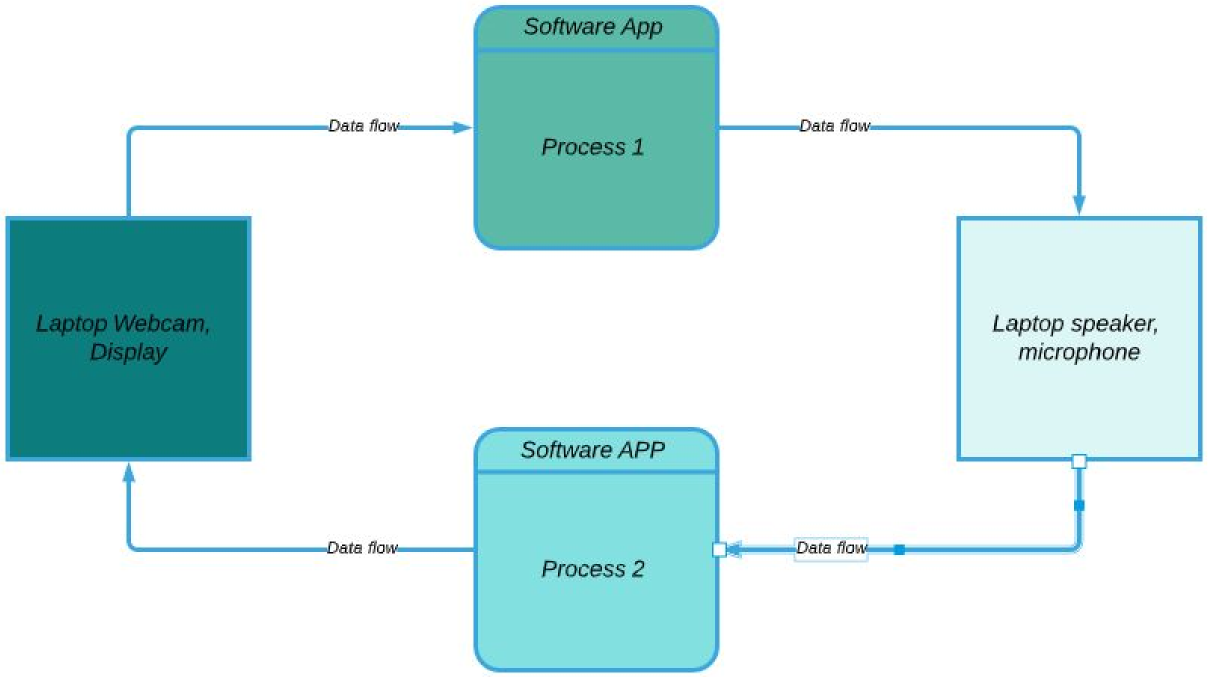


Figure 7. Functional Block Diagram

5.2 Technical Design: System Level

In the code flow chart, the codes done in the Machine Learning Part is illustrated. At first, we import libraries and then input the dataset. The dataset is manually included. There are two different options, such that 1. Preprocessing 2. Model Implementation. In preprocessing, we firstly take grayscale images and secondly take RGB(Red Green Blue contrast) images and resize them. Then we clean the data. For example, unlabeled datasets are labeled. Increasing the dataset by rotating the image slightly. Can use contrast increase and decrease also to increase the amount of dataset. It's then stored in HDD.

In Machine Learning implementation, we first, use the unprocessed data to get a base accuracy which will help us to set the base accuracy. Which will be very low. Then we use preprocessed data to train models like YOLO, CNN etc. From preprocessed data we used 80% for training and 20 % for testing. We test the model with a testing dataset and then tweak the model by changing the model by changing the parameters. We conclude at maximum accuracy obtained.

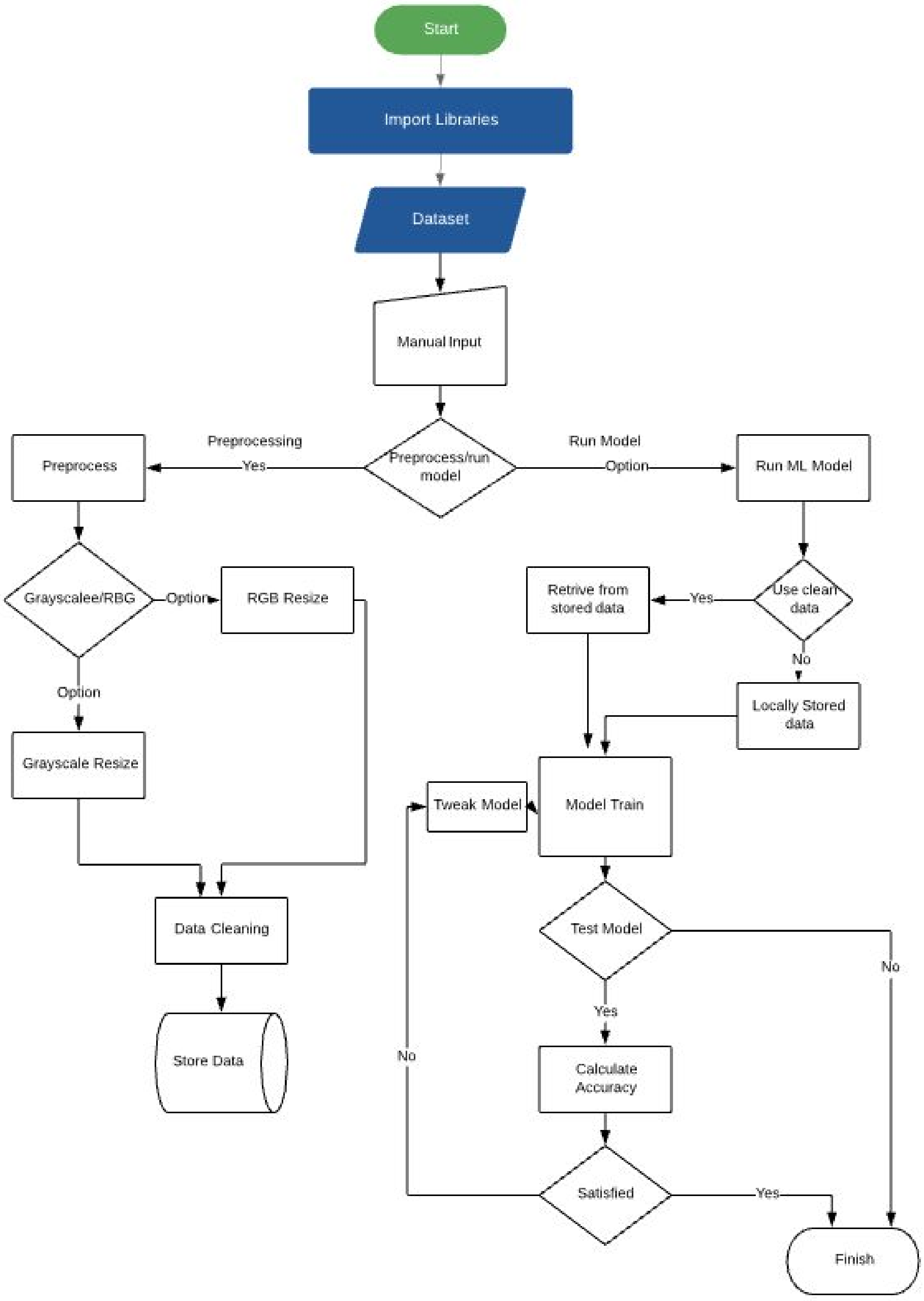


Figure 8. Code Flow Chart

5.3 Software Development Life Cycle

In the Software Development Life Cycle (SDLC), we have six stages. Ours is a hybrid project in the sense that we have to take input through hardware, process it using software and give output using hardware. So, the software will require necessary analysis before approaching to design it. The Analysis of its necessity, real life implementation, longevity must be taken into consideration before designing it. After the analysis has been done, we will jump to the design phase. Design phase will include all the substantial charts and software and hardware tools that are needed to complete the project. Using all of these, Development will be done. After that, adequate testing must be done before deploying the product. Testing will be done to remove faults and bugs in the system. After finishing the testing phase and coming up with a full deployable project, the project will be implemented in real life scenario. Simultaneously, documentation will be carried out to keep track of the project and to avoid any unnecessary data loss. Lastly, Evaluation will be done in a real-life scenario.

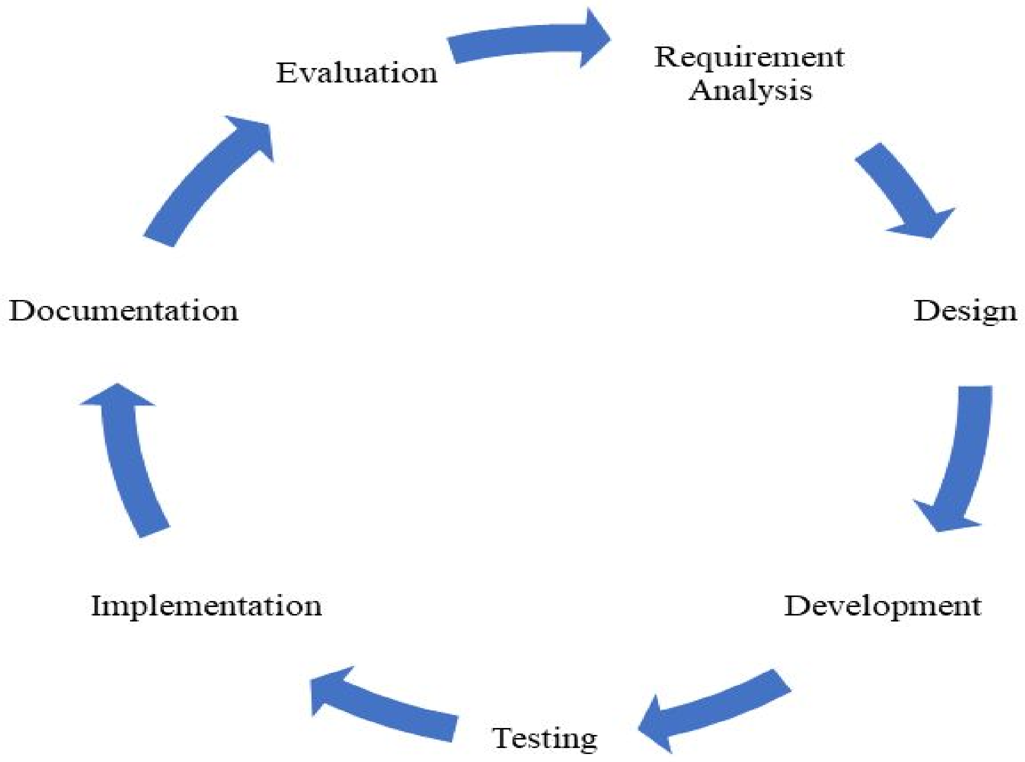


Figure 9. SDLC

5.4 Use Case Diagram

Use Case Diagram gives us a clear view on the usage of the project. There are two users, first one being the person with the hearing disability and the second one being the person communicating with the first person. The first person's use case consists of input sign language, converting sign to text and seeing the text on display. The other user will use a speaker to listen to the speech which is converted from sign, input speech through a microphone and listen to speech which was generated from sign language.

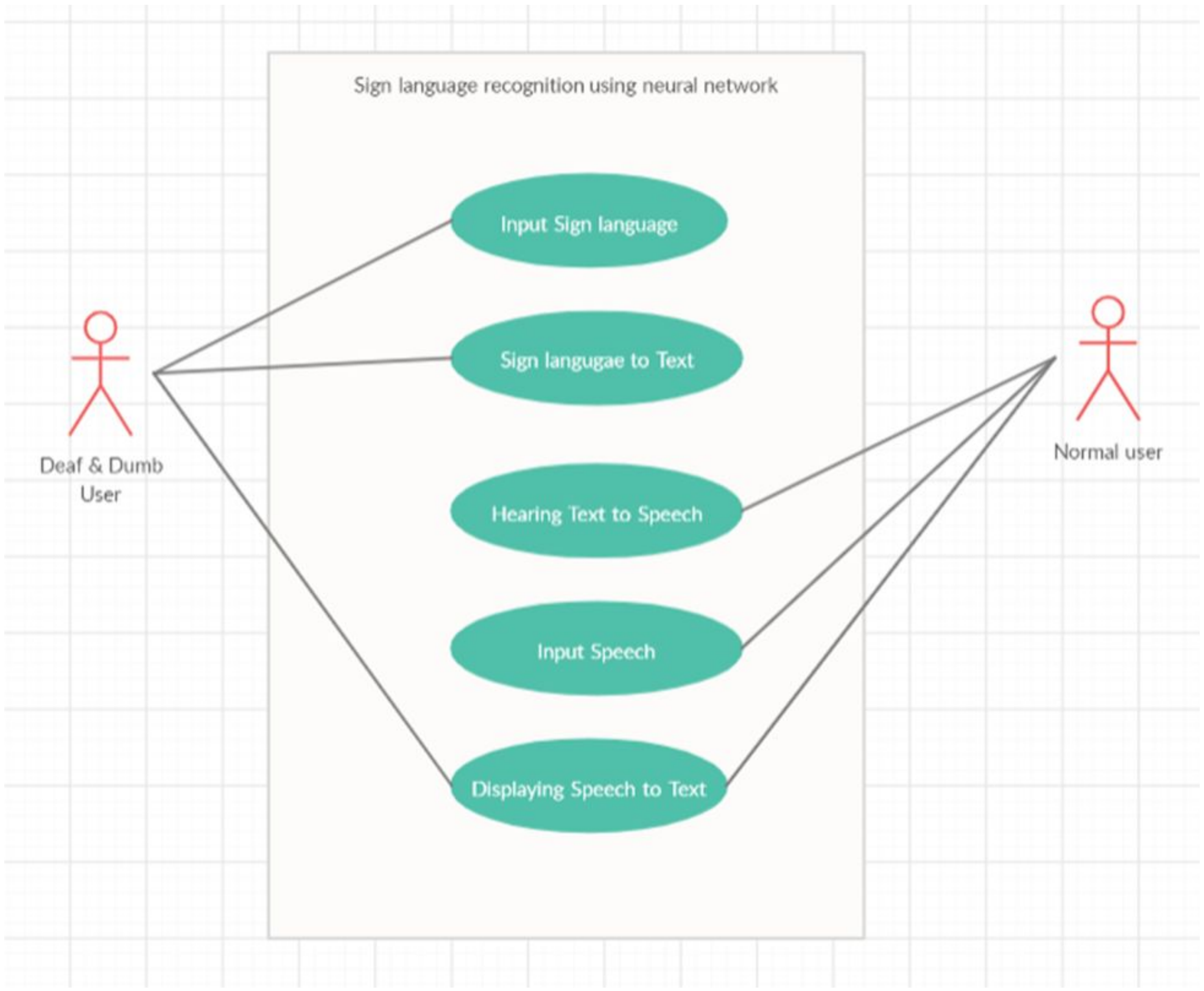


Figure 10. Use Case Diagram

5.5 Sequence Diagram

In the sequence diagram, the user will give a sign. This input is taken and a method will be created in software which will check if the sign is valid or not valid. If valid then it will call another method which will convert the text into speech. In the alternative block there is an if else condition applied which will continue to ask for valid sign until the user stops signing. In the meantime the text that we get from machine learning recognition will be converted into speech using API method.

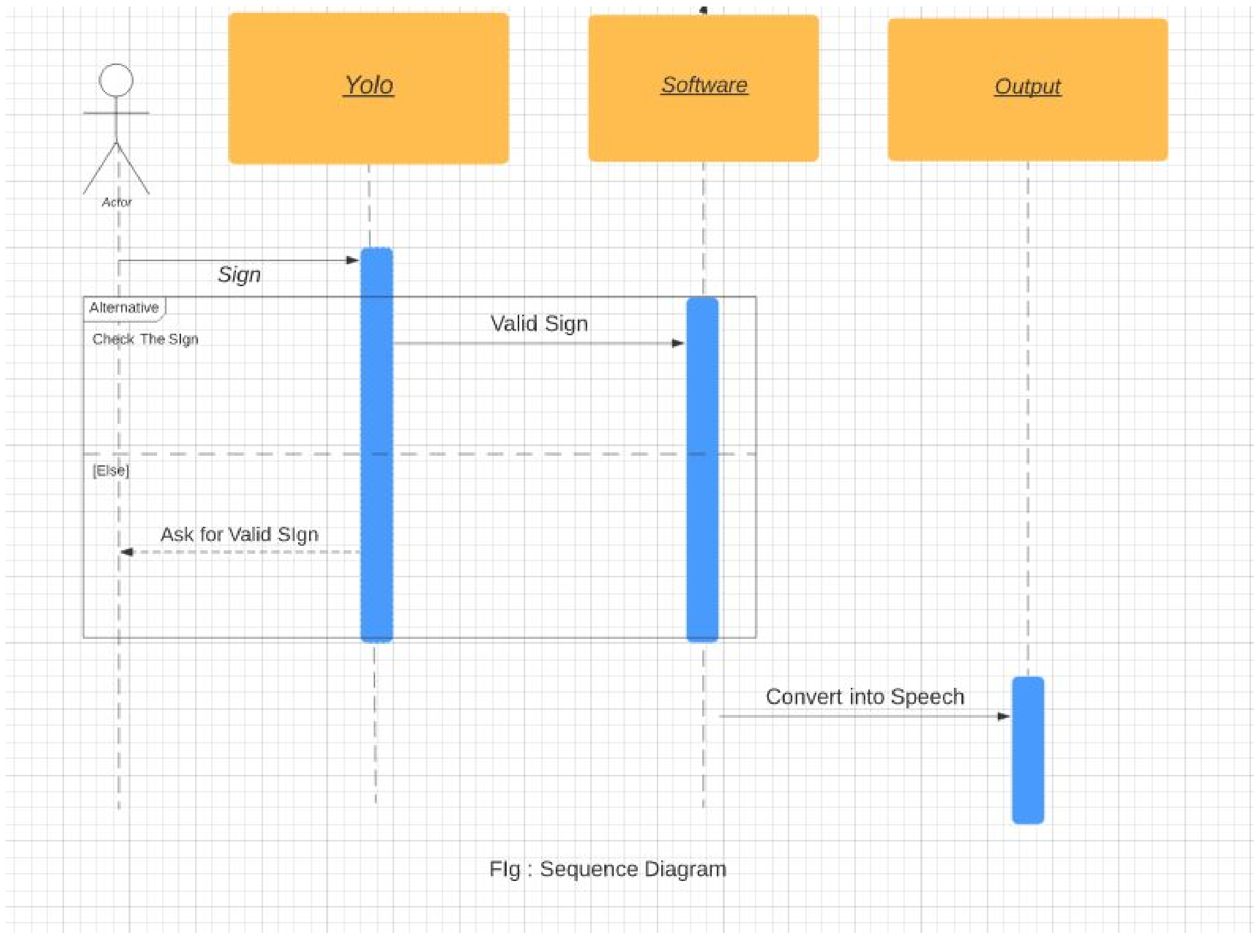


Figure 11. Sequence Diagram

5.6 Activity Diagram

This is an activity diagram for the software part of our project. It describes the User Interface Software's flow functions. After starting the application user will have to choose from either Speak module or Listen module.

For the Speak module, the webcam will record sign language shown by the user. The input will be converted into text and using API(text to speech), the speech will be generated for the listener. For the Listen module, the microphone will be used to take audio input and using API(speech to text), it will be converted into text. Which will be visible on display for the person who has hearing and speaking disability.

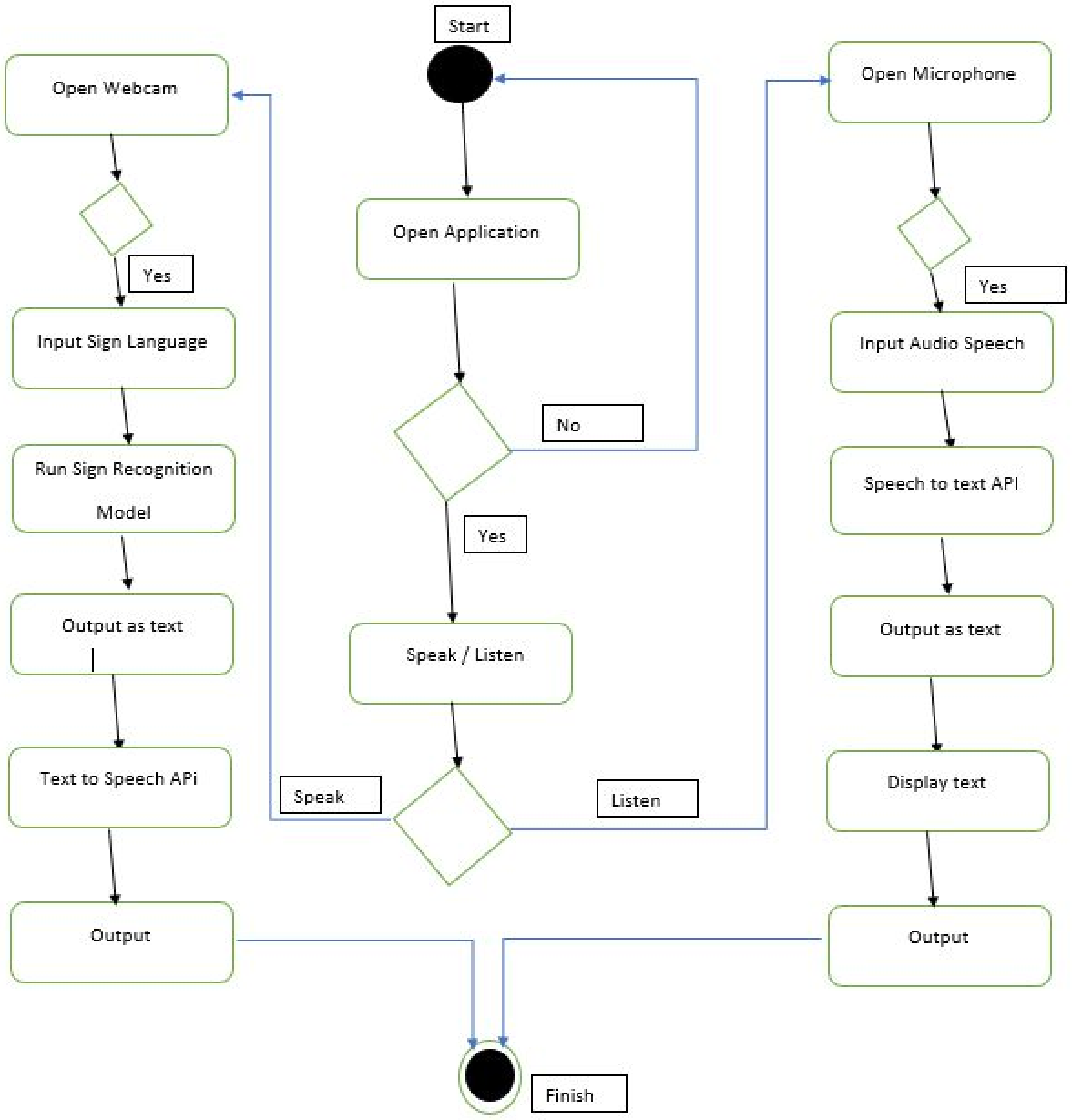


Figure 12. Activity Diagram

6

Work Done

Time	Task
March, 2020	Collecting Dataset
March, 2020	Preprocessing the data
March, 2020	Dataset Labelling
April, 2020	Resizing the Dataset Images
April, 2020	Loading the dataset
May, 2020	Training the dataset using MobileNet
May, 2020	Solving Problems of overfitting and underfitting
June, 2020	Using Softmax as activation function
June, 2020	Using Softmax as activation function
June, 2020	Using Softmax as activation function
July, 2020	Plotting the graphs using Matplotlib
July, 2020	Test the model with testing dataset
July, 2020	Look for ways to improve accuracy.
August, 2020	Loading the pretrained model in the website
August, 2020	Opening the camera option using OpenCV
September, 2020	Login and registration form created
September, 2020	Add features to the existing base block of UI
September, 2020	Implementing Text to Speech API
September, 2020	Testing Project & Finding Error

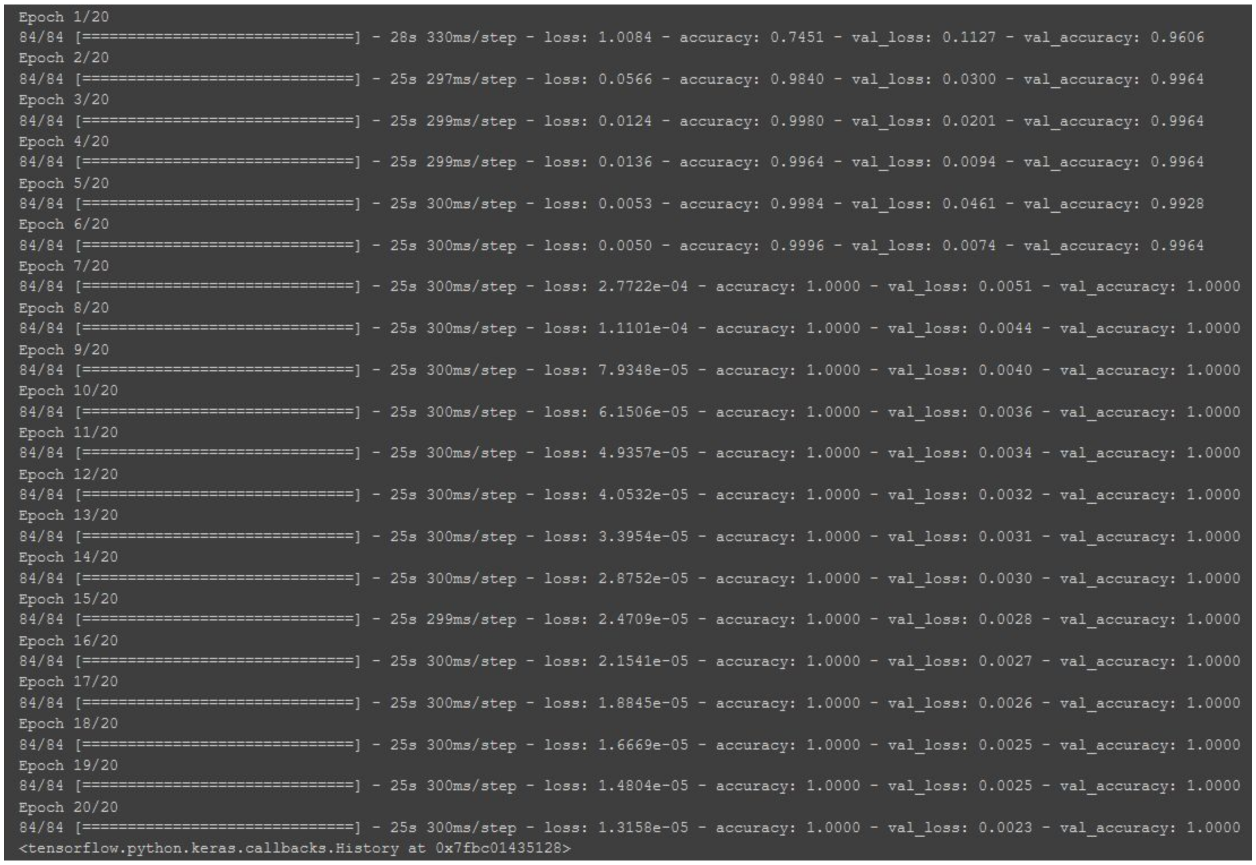
Table 1. Work Structure

7

Results

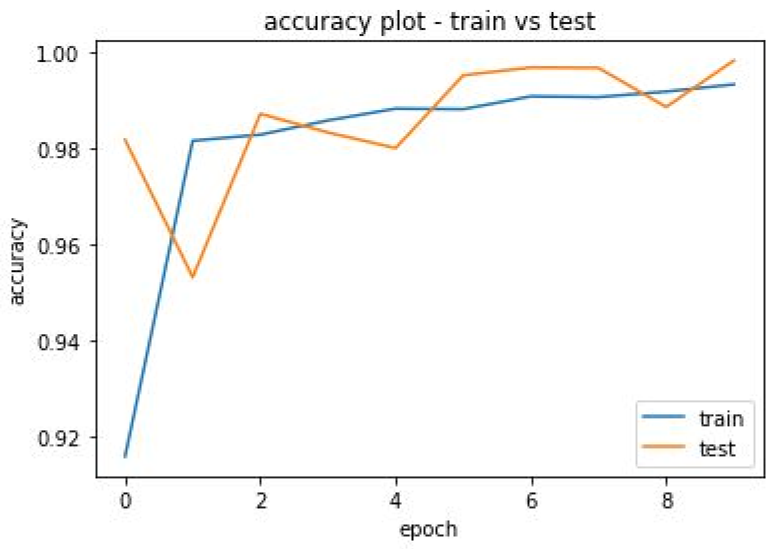
We have worked with multiple architectures because which model will be suitable for our project was ambiguous. We first tried with a general approach with normal Sequential models and created layers while stacking them up. The results were poor and we got accuracy around 97% but the models overfitted. It was not generalizing properly and due to the lack of high quality dataset we couldn't just rely on scraped models.

The predefined models from keras such as VGG16, ResNet, Inception, Xception etc were trained on the IMAGENET dataset and were quite reliable if they were manipulated with shrewd perception. The pre-trained networks were very large and contained many layers and millions of parameters. So, one optimization technique was showing promising implications by freezing most of the layers and using the technique of transfer learning. Since we freezed most of the layers and trained only a couple of layers, our training time was reduced significantly. Still we had other issues. The size of the networks are excessively large and they require lots of time to load into the system which is not very efficient. Our system works in real-time and it's highly inefficient to use large networks such as VGG16,ResNet etc. We then switched to MobileNet and finally settled with it because in comparison to VGG16, MobileNet has 4.2 million parameters whereas, VGG16 has 17 million parameters. Also the size of the network in memory is proportionally smaller than VGG16. It loses a little bit of edge in the accuracy department but compared to its fast and lightweight nature that is acceptable. Fig# 'X' shows the training results alongside the validation loss and Fig# 'Y' gives us the loss and accuracy graph.



```
Epoch 1/20
84/84 [==============================] - 28s 330ms/step - loss: 1.0084 - accuracy: 0.7451 - val_loss: 0.1127 - val_accuracy: 0.9606
Epoch 2/20
84/84 [==============================] - 25s 297ms/step - loss: 0.0566 - accuracy: 0.9840 - val_loss: 0.0300 - val_accuracy: 0.9964
Epoch 3/20
84/84 [==============================] - 25s 299ms/step - loss: 0.0124 - accuracy: 0.9980 - val_loss: 0.0201 - val_accuracy: 0.9964
Epoch 4/20
84/84 [==============================] - 25s 299ms/step - loss: 0.0136 - accuracy: 0.9964 - val_loss: 0.0094 - val_accuracy: 0.9964
Epoch 5/20
84/84 [==============================] - 25s 300ms/step - loss: 0.0053 - accuracy: 0.9984 - val_loss: 0.0461 - val_accuracy: 0.9928
Epoch 6/20
84/84 [==============================] - 25s 300ms/step - loss: 0.0050 - accuracy: 0.9996 - val_loss: 0.0074 - val_accuracy: 0.9964
Epoch 7/20
84/84 [==============================] - 25s 300ms/step - loss: 2.7722e-04 - accuracy: 1.0000 - val_loss: 0.0051 - val_accuracy: 1.0000
Epoch 8/20
84/84 [==============================] - 25s 300ms/step - loss: 1.1101e-04 - accuracy: 1.0000 - val_loss: 0.0044 - val_accuracy: 1.0000
Epoch 9/20
84/84 [==============================] - 25s 300ms/step - loss: 7.9348e-05 - accuracy: 1.0000 - val_loss: 0.0040 - val_accuracy: 1.0000
Epoch 10/20
84/84 [==============================] - 25s 300ms/step - loss: 6.1506e-05 - accuracy: 1.0000 - val_loss: 0.0036 - val_accuracy: 1.0000
Epoch 11/20
84/84 [==============================] - 25s 300ms/step - loss: 4.9357e-05 - accuracy: 1.0000 - val_loss: 0.0034 - val_accuracy: 1.0000
Epoch 12/20
84/84 [==============================] - 25s 300ms/step - loss: 4.0532e-05 - accuracy: 1.0000 - val_loss: 0.0032 - val_accuracy: 1.0000
Epoch 13/20
84/84 [==============================] - 25s 300ms/step - loss: 3.3954e-05 - accuracy: 1.0000 - val_loss: 0.0031 - val_accuracy: 1.0000
Epoch 14/20
84/84 [==============================] - 25s 300ms/step - loss: 2.8752e-05 - accuracy: 1.0000 - val_loss: 0.0030 - val_accuracy: 1.0000
Epoch 15/20
84/84 [==============================] - 25s 299ms/step - loss: 2.4709e-05 - accuracy: 1.0000 - val_loss: 0.0028 - val_accuracy: 1.0000
Epoch 16/20
84/84 [==============================] - 25s 300ms/step - loss: 2.1541e-05 - accuracy: 1.0000 - val_loss: 0.0027 - val_accuracy: 1.0000
Epoch 17/20
84/84 [==============================] - 25s 300ms/step - loss: 1.8845e-05 - accuracy: 1.0000 - val_loss: 0.0026 - val_accuracy: 1.0000
Epoch 18/20
84/84 [==============================] - 25s 300ms/step - loss: 1.6669e-05 - accuracy: 1.0000 - val_loss: 0.0025 - val_accuracy: 1.0000
Epoch 19/20
84/84 [==============================] - 25s 300ms/step - loss: 1.4804e-05 - accuracy: 1.0000 - val_loss: 0.0025 - val_accuracy: 1.0000
Epoch 20/20
84/84 [==============================] - 25s 300ms/step - loss: 1.3158e-05 - accuracy: 1.0000 - val_loss: 0.0023 - val_accuracy: 1.0000
<tensorflow.python.keras.callbacks.History at 0x7fbc01435128>
```

Figure 13. Training and Validation logs.



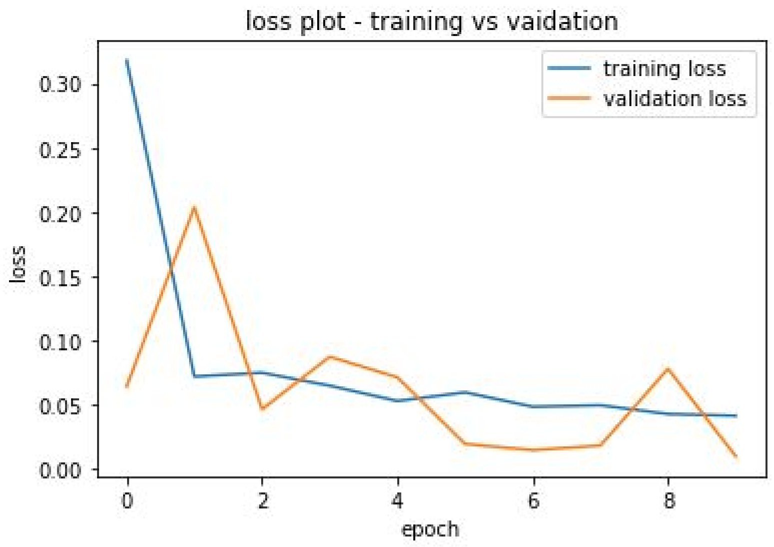


Figure 14. Accuracy and Loss in training, testing and validation

MobileNet is outperforming our hand scratched models and doing almost as good as VGG16 and ResNet. In training and validation we have gotten satisfactory results. But the problem lies elsewhere. As our system takes input from the laptops webcam, there is lots of background noise in the image frames and that creates problems in real-time predictions. Though we have built it to the best of our knowledge it lacks necessary adaptation that is required to handle real-time generated problems.

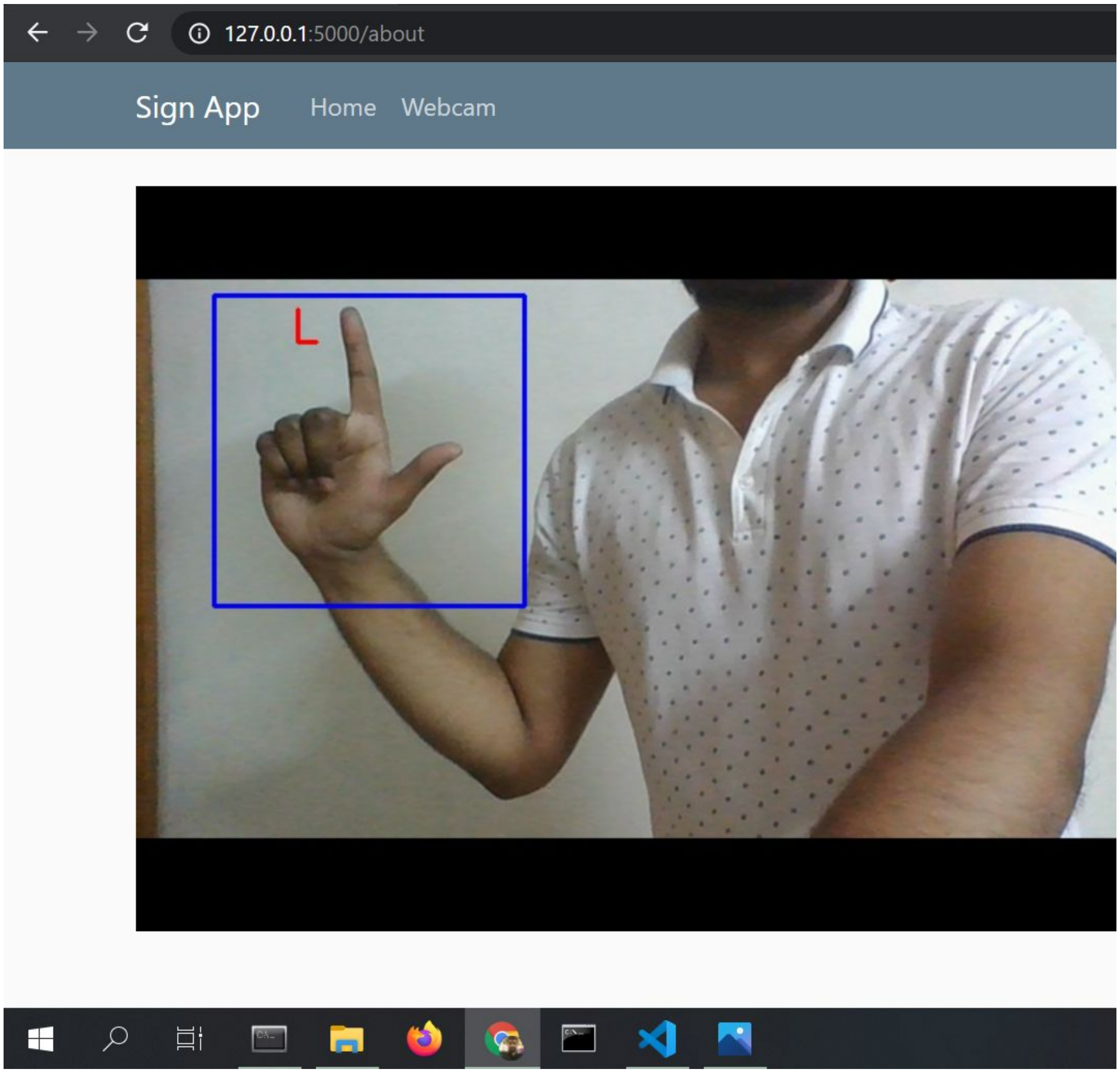


Figure 15. Sign language of ‘L’

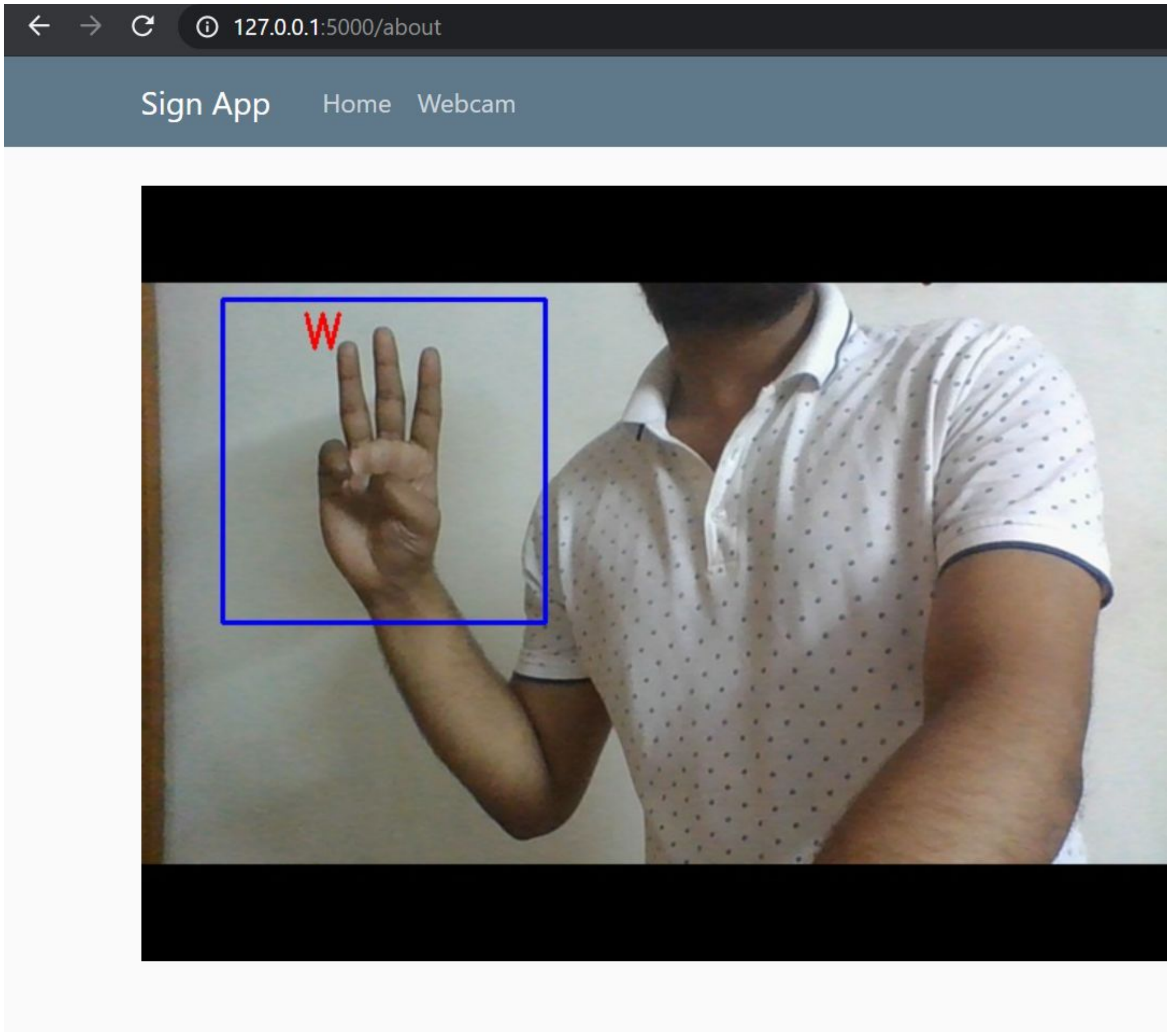
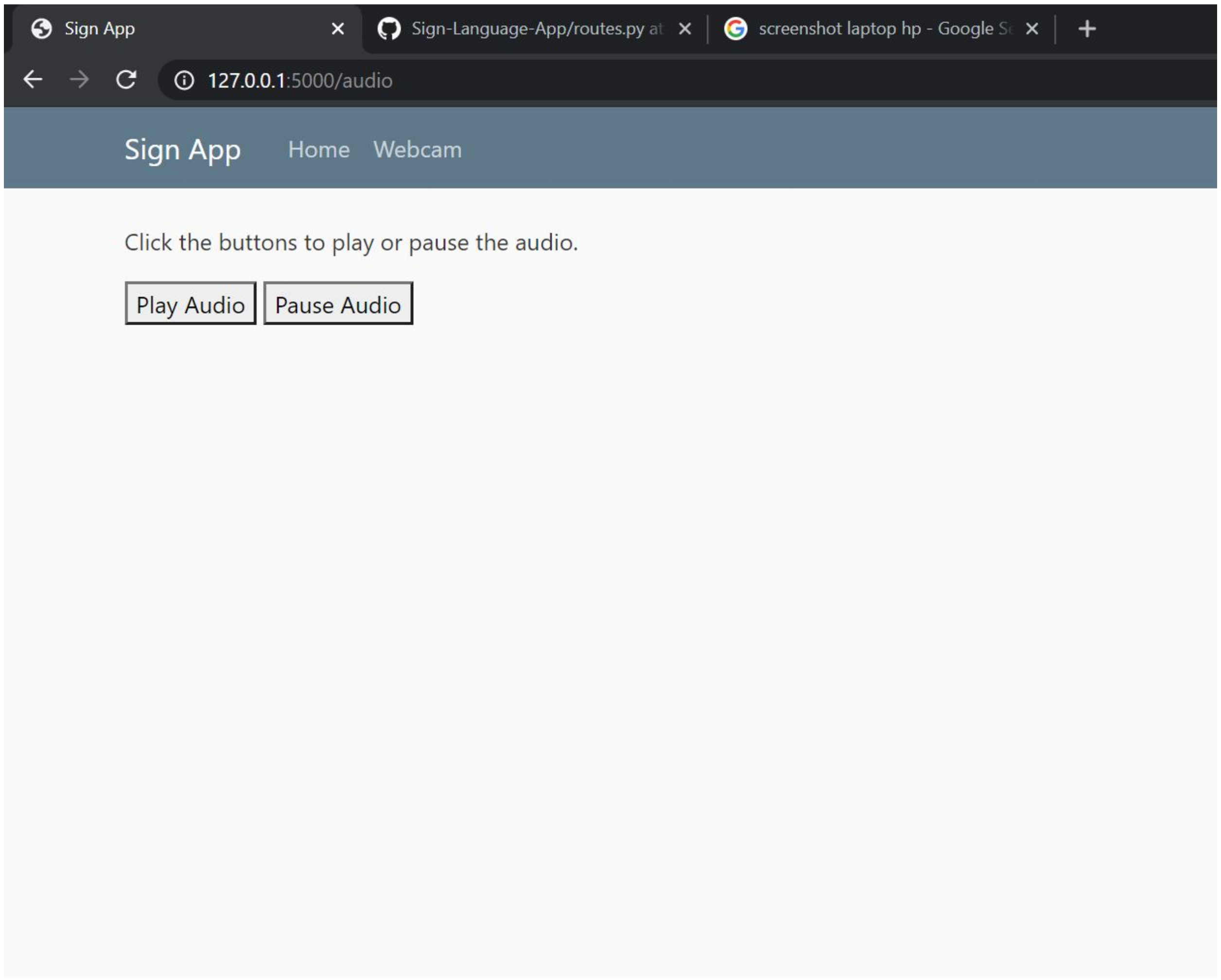


Figure 16. Sign of 'W'



Click the buttons to play or pause the audio.

Figure 17. Option to play the sign

8

Future works

What are the steps before you take this project into any competition or exhibition etc. By using Our project deaf people can translate their sign language to English language easily. They can translate sign language from anywhere over internet service. It will eliminate the problem of communication. Our project goal is making an application which is easy to access and will eliminate the problem of communication with physically sound people. In Future we can improve our project. We can improve the accuracy of our project. This is a software based project. This project can't be 100% accurate. A dataset with more variation and a higher quality van really boost the accuracy of our current project. We can make our web application more user friendly & more easy to use. Now our project is online based. In the future we can make it a mobile device. Today's day most people use smartphones. We can make our application for mobile devices..We can make it for android & ios. By releasing mobile version of our project we can closer to deaf and dumb people. For our project there is a lot of maintenance, server cost etc. For cost reduction we can arrange sponsors and we can also give advertisements on websites for the cost reduction.

9

Conclusion

Communication is one of the important requirements for survival in society. Sign language is the main communication medium for the deaf and mute people. American sign language(ASL) is the most used sign language in the world. Many physically sound people don't know sign language. Our project aims to eliminate the difficulties. Our project will elminate the communication gap between normal people and deaf and dumb

people using sign language. Effective extension of this project to words and common expressions may not only make the deaf and dumb people communicate faster and easier with the outer world, but also provide a boost in developing autonomous systems for understanding and aiding them.

10

References

[1] Charayaphan C. and Marble A. (1992) Image Processing System forInterpreting Motion in American Sign Language, Journal of BiomedicalEngineering, Vol. 14, pp. 419–425.

[2] Huang X., Ariki Y., and Jack M. (1990) Hidden Markov Models forSpeech Recognition, Edinburgh University Press, Edinburgh.

[3] H.Ushiyama, K.Hirota, and K.Murakami, "Hand Gesture Interpretation Using Neural Networks," 40th Information Processing Conference Pro-ceedings ,VO1.4, 1990, pp,152.

[4] K.Maruyama, Sign language hand book ,Dynamic Sellers Publishing Co,1981.

[5] Tamura S. and Kawasaki S. (1988) Recognition of Sign LanguageMotion Images, Pattern Recognition, Vol. 21, pp. 343–353.