North South University
Department of Electrical & Computer Engineering

SENIOR DESIGN PROJECT

Spring 2020

Obhoy Kothon: Sign Language Recognition Using Neural Network

Submitted To:
Mohammad Rezaul Islam
Senior Lecturer

Department of Electrical and Computer Engineering
North South University

Submitted By:

Nayeem Mohammad	1611553042
Zubayer Ahmed	1611840042
Rubayet Zaman	1620684642

LETTER OF TRANSMITTAL

4th July, 2020
To
Dr. Mohammad Rezaul Bari
Associate Professor and Chairman
Department of Electrical and Computer Engineering

Through

Mohammad Rezaul Islam
Senior Lecturer
Department of Electrical and Computer Engineering North
South University
Dhaka, Bangladesh

Subject: Submission of Senior Project Design Report on "Obhoy Kothon: Sign Language Recognition Using Neural Network".

Dear Sir,

With due respect, we would like to submit our Capstone Project Report on "Obhoy Kothon: Sign Language Recognition Using Neural Network" as a part of our BSc program. The report mentions all the technical details about our work. This project was very much valuable to us as it helped us gain experience in this unique field of research that very few scientists have attempted to work on. We tried to the greatest competence to meet all the dimensions required from this report. We will be highly obliged if you kindly receive this report and provide your valuable judgment. It will be our immense pleasure if you find this report useful and informative to have a clear perspective on the issue.

Sincerely,

....

Nayeem Mohammad
1611553042
North South University

....

Zubayer Ahmed
1611840042
North South University

....

Rubayet Zaman
1620684642
North South University

DECLARATION

This is to certify that this Project is our original work. No part of this work has been submitted elsewhere partially or fully for the award of any other degree or diploma. Any material reproduced in this project has been properly acknowledged.

Declared By:

....
Name: Nayeem Mohammad
ID: 1611553042

....
Name: Zubayer Ahmed
ID: 1611840042

....
Name: Rubayet Zaman
ID: 1620684642

APPROVAL

Nayeem Mohammad (ID 1611553042), Zubayer Ahmed (ID 1611840042) and Rubayet Zaman (ID 1620684642) from Electrical and Computer Engineering Department of North South University, have worked on the Senior Design Project titled “Obhoy Kothon: Sign Language Recognition Using Neural Network” under the supervision of Mohammad Rezaul Islam partial fulfillment of the requirement for the degree of Bachelor Science in Engineering and has been accepted as satisfactory.

Approved By:

Supervisor:

....

Mohammad Rezaul Islam
Senior Lecturer
Department of Electrical and Computer Engineering
North South University, Dhaka, Bangladesh

Chairman:

....

Dr. Mohammad Rezaul Bari
Associate Professor and Chairman
Department of Electrical and Computer Engineering
North South University, Dhaka, Bangladesh

Acknowledgements

We are profoundly grateful to Mohammad Rezaul Islam for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.

At last we must express our sincere heartfelt gratitude to all the staff members of the Electrical & Computer Engineering Department who helped us directly or indirectly during this course of work.

Nayeem Mohammad
Zubayer Ahmed
Rubayet Zaman

Abstract

In this era, machines are continuously striving to imitate the human learning process in order to alleviate our struggles. Using the advancement of technology in the sector of human machine interaction and with the help of Convolutional Neural Networks, machines are now able to solve complex problems of image recognition. One of the main reasons why deaf people can’t lead a normal life is because there are so few people who possess the ability to perform sign language who aren’t deaf. To obviate this interaction barrier, we have come up with an elegant and innovative solution. Web app that we are developing will aid a deaf person to communicate with any person without the help of an intermediate interpreter. Our system will allow the user with the hearing disabilities to make signs and turn the signs into corresponding letters to form speech.

Table of Contents

List of Figures:

CHAPTER 1
INTRODUCTION

1.1 Project Details:

Communication is very important to human beings for living, as it allows us to express ourselves to other people in a more intuitive fashion. We communicate through speech, visual aids, body language, reading, writing or gestures. Speech is one of the most commonly used mediums 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 motor driven 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.

1.2 Background and Motivation:

Background:

Over the last decade, people are trying to work on communications and many researchers as well as humanitarians work on the improvement and also invention of various techniques of using sign language to express one's feelings. The Braille system is still an effective technique of using sign language. A paper titled **'A cost effective electronic braille for visually impaired individuals'[1]** published in

February 2018 on IEEE conference tried to ease the suffering of people who are not well off to use expensive braille techniques.

However, even though the braille system is an effective choice to use it for sign language, researchers and companies have been trying to divert it to more sophisticated techniques and with the rise of machine learning and Artificial intelligence people are getting more attracted to develop some techniques. Papers titled '**A comparison of Machine Learning Algorithms applied to hand gesture recognition**'**[2]** published in August 2012 and '**Machine learning model for sign language interpretation using webcam images'[3]** published in june 2014 tried to answer the preliminary questions about using machine learning algorithms to recognize basic hand gesture.

The communication must be made clear, comfortable and easy to understand.Therefore, scientists are still working on machine learning, deep learning techniques to make the gesture recognition more accurate, fast and robust.So researchers have been making neural networks to recognize the signs. Papers such as **'Deep convolutional neural networks for sign language recognition'[4]**, '**Sign language Recognition using 3D convolutional neural networks' published in January 2018 on IEEE[5]** and **'Hand Gesture Recognition with Generalized Hough Transform and DC-CNN Using RealSense'[6].** These three papers are directly associated with neural network techniques. To be very specific, both use CNN(convolutional neural network).Lastly, we have smart gloves which can be used to express in sign language. These mainly use flex sensors and are pretty good with the signs.

All of these have their own perks in their relative works. What we want to achieve is that people who are unable to talk or hear not only can afford a technique that will eradicate their problem, they can also access it from anywhere in the world and can use it with ease without thinking how they can express their feelings.

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.

1.3 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. The project goal is to eliminate this difficulty

CHAPTER 2

TECHNICAL DESIGN

2.1 Existing Solution:

There are many research papers that come with the solution of removing the difficulties faced by deaf and mute people.

As previously we mentioned the papers in background(section 1.2):

Paper 1:

Title : **A cost effective electronic braille for visually impaired individuals**

Author: Md. Ehtesham Adnan ; Noor Muhammad Dastagir ; Jafrina Jabin ; Ahmed Masud Chowdhury ; Mohammad Rezaul Islam

Publisher: IEEE

Date of publication:12 February 2018

Description:

This paper proposes a prototype of an affordable Braille display for the blind people to read when an input is given through a computer. It uses electromagnetic solenoids controlled by Arduino Uno to capture the vertical movement of the dots of the braille. Bringing the cost up to $100 dollars was one of the challenges.

1. Input from the computer is processed by Arduino Uno.
2. Arduino Uno controls the switching circuit which is connected to the solenoids.
3. Components: i) Microcontroller: Arduino Uno ii) MOSFET iii) Solenoid

Paper 2:

Title: **A comparison of Machine Learning Algorithms applied to hand gesture recognition**

Author: Paulo Trigueiros ; Fernando Ribeiro ; Luís Paulo Reis

Publisher: IEEE

Date of Publication:31 August 2012

Description:

- Hand orientation was used with image intensity detection.
- Hand segmentation and feature extracted from the segment hand.
- An Orientation Histogram was generated to see the shape of the fit line.

Pixel intensities can be sensitive to lighting variations, which leads to classification problems within the same gesture under different light conditions.

Paper 3:

Title: **Machine learning model for sign language interpretation using webcam images**

Author: Kanchan Dabre ; Surekha Dholay

Publisher: IEEE

Date of publication:19 June 2014

Description:

-Using web camera, it takes the image of the sign and then process it to make an audio output

Paper 4:

Title: **Deep convolutional neural networks for sign language recognition**

Author: G. Anantha Rao ; K. Syamala ; P. V. V. Kishore ; A. S. C. S. Sastry

Publisher: IEEE

Date of publication: 4-5 Jan. 2018

Description:

This paper proposes the recognition of sign language gestures using a powerful artificial intelligence tool, convolutional neural networks (CNN). They designed a multi stage CNN model. The model is constructed with an input layer, four convolutional layers, five rectified linear units (ReLu), two stochastic pooling layers, one dense and one SoftMax output layer. Figure 1 shows the proposed system architecture.

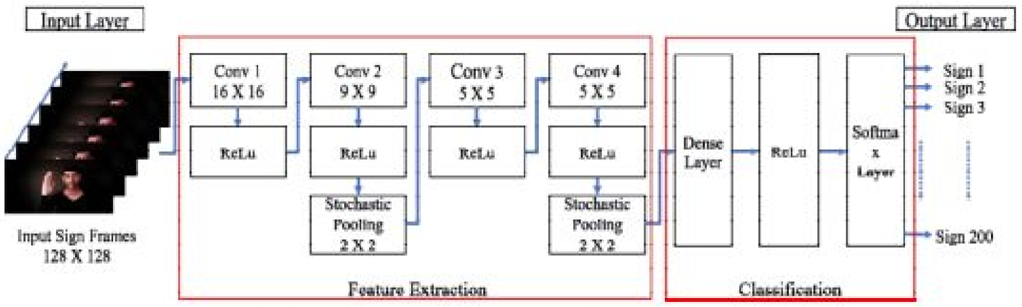


Fig.1. Proposed Deep CNN architecture

Advantage:

The CNN architecture is designed with four convolutional layers. Each convolutional layer with different filtering window sizes is considered which improves the speed and accuracy in recognition. They achieved 92.88% recognition rate compared to other classifier models reported on the same dataset. Their system was also a low-cost model.

Paper 5:

Title : **Sign language Recognition using 3D convolutional neural networks**

Author:Jie Huang, wengang Zhoue, Houqiang Li, Weiping Li

Publisher: IEEE

Date of publication:06 August 2015

Description:

- Applying 2D CNNs in speech domain

-making the local receptive field to move to the next layer in each unit from the prior layer.

-Gaussian Blur, Gaussian Mixture Model-Hidden Markov model (GMM-HMM)

Paper 5:

Title: **Hand Gesture Recognition with Generalized Hough Transform and DC-CNN Using RealSense**

Author: Bo Liao ; Jing Li ; Zhaojie Ju ; Gaoxiang Ouyang

Publisher: IEEE

Date of publication: 09 August 2018

Description:

This paper proposes a hand motion recognition method based on data collected by Intel RealSense Front-Facing Camera SR300. Considering that the pixels in the depth images obtained by RealSense are not one-to-one with those in the color images, the recognition system maps the depth images to the color images based on the generalized Hough transform in order to segment the hands from a complex backdrop to the color images using the depth information. It then recognizes the various hand movements of a novel, two-channel, convolutional neural network comprising two input channels, color images and depth images. As for experimental hardware, Intel(R) Core (TM) i7-6800 K CPU, NVIDIA GeForce GTX 1080 Ti,16 GB RAM were chosen.

Paper 6:

Title: **Smart glove with gesture recognition ability for the hearing and speech impaired**

Author: Tushar Chouhani; Ankit Panse ; Anvesh Kumar Voona ; S. M. Sameer

Publisher: IEEE

Date of publication:01 December 2014

Description:

- The hand talk glove senses the movements through the flex sensors which detect different patterns of motion.
- The device can sense carefully each resistance and each movement by hand. Flex Sensors fixed with the glove pick up the gestures made by the individual and then with the help of Arduino, that analog input is converted into digital for various gestures.

Circuit Operations:

1. Transmitter Section (Hand Glove)
2. Receiver section (display module)

The Heartbeat module is used for heart rate monitoring of the patient. If there is any problem with the heart rate then a message is sent to a doctor or family members also.

All these solutions tried to fight the problem of communication on behalf of deaf and mute people.

2.2 Proposed Solution:

To tackle the communication gap between a deaf person and a normal person our designed solution promises to eliminate the need for both parties being a sign language user. The learning curve of sign language is almost as high as a natural language and so we don’t see many people around who know sign language. According to recent studies done by the world health organization (WHO), 466 million people around the globe have hearing disabilities and they rely heavily on sign language to communicate with everyone around them.
As mentioned in the proposed goal section, we are producing a system that will have three components and it will work as a united front. The first component will be a hand sign recognition module and it will take input from a user through a webcam or camera that is associated with the input system. The input system will be connected to the second module of our system. It will connect to a convolutional neural network from which we get a prediction. The convolutional neural network will take input of a frame from the live feed of the webcam and pass it through various layers to use the machine learning process of image processing and recognition. We will get output from our architecture that will recognize the image. That model will generate a confidence score indicating which letter of American Sign Language (ASL) it represents.

Our third module will encapsulate the first two modules into a whole package. It will be a fully functional web-app that will host the deep learning module inside its backend. A user who has registered to the website can log in and use the application to communicate with a non-sign language user. In this section, the output of the neural network will be stored, and then google text-to-speech API will be used to turn it into speech.

Our solution will be efficient in its usage as it will open a new way in the communication field. Deaf people with access to a laptop and a good internet connection will be benefited from this proposed solution.

2.3 Solution Assessment:

We have digged through many academic papers based on ideas similar to our proposed idea. We have found interesting solutions that were intended to help the person with hearing disabilities. Some of the key extinguishable features of our solution are that our’s will be lightweight and efficient in time. Its cost will be significantly miniscule and its user friendliness will further simplify the problems associated with real-time systems. One thing to mention that none of the existing solutions were using a web application alongside their machine learning models. Since we integrated with a website, our solution represents a level of uniqueness that is non existent in other solutions.

2.4 Technical Design: Module Level

Functional Block Diagram:

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.

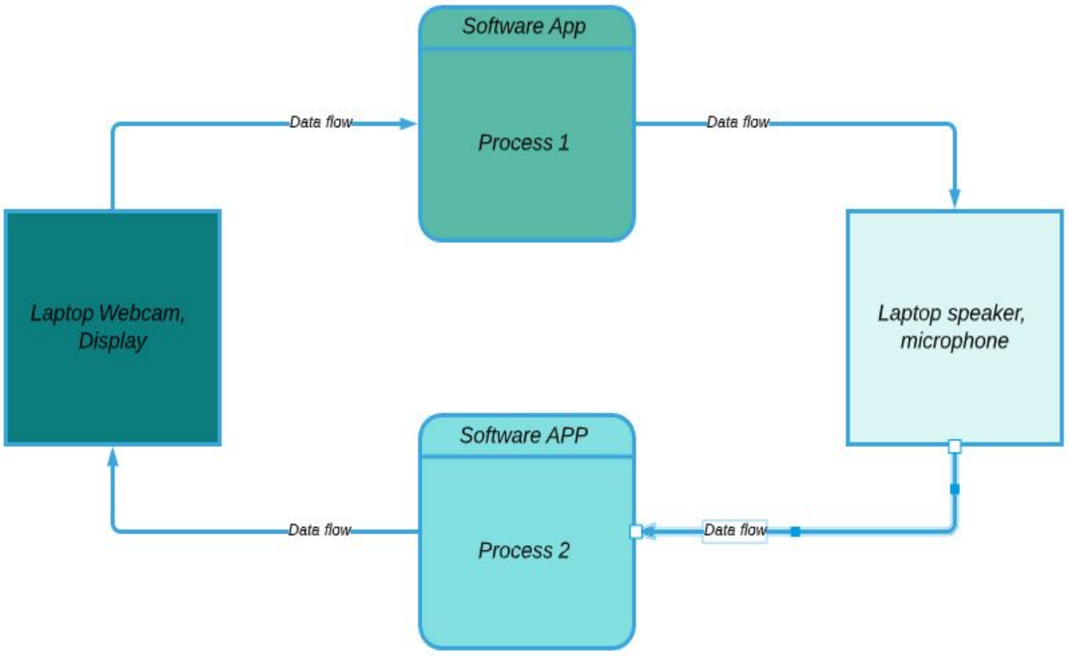


Figure #2: Functional Block Diagram

2.5 Technical Design: System Level

Code Flow Chart:

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 VGG16, MobileNet, 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.

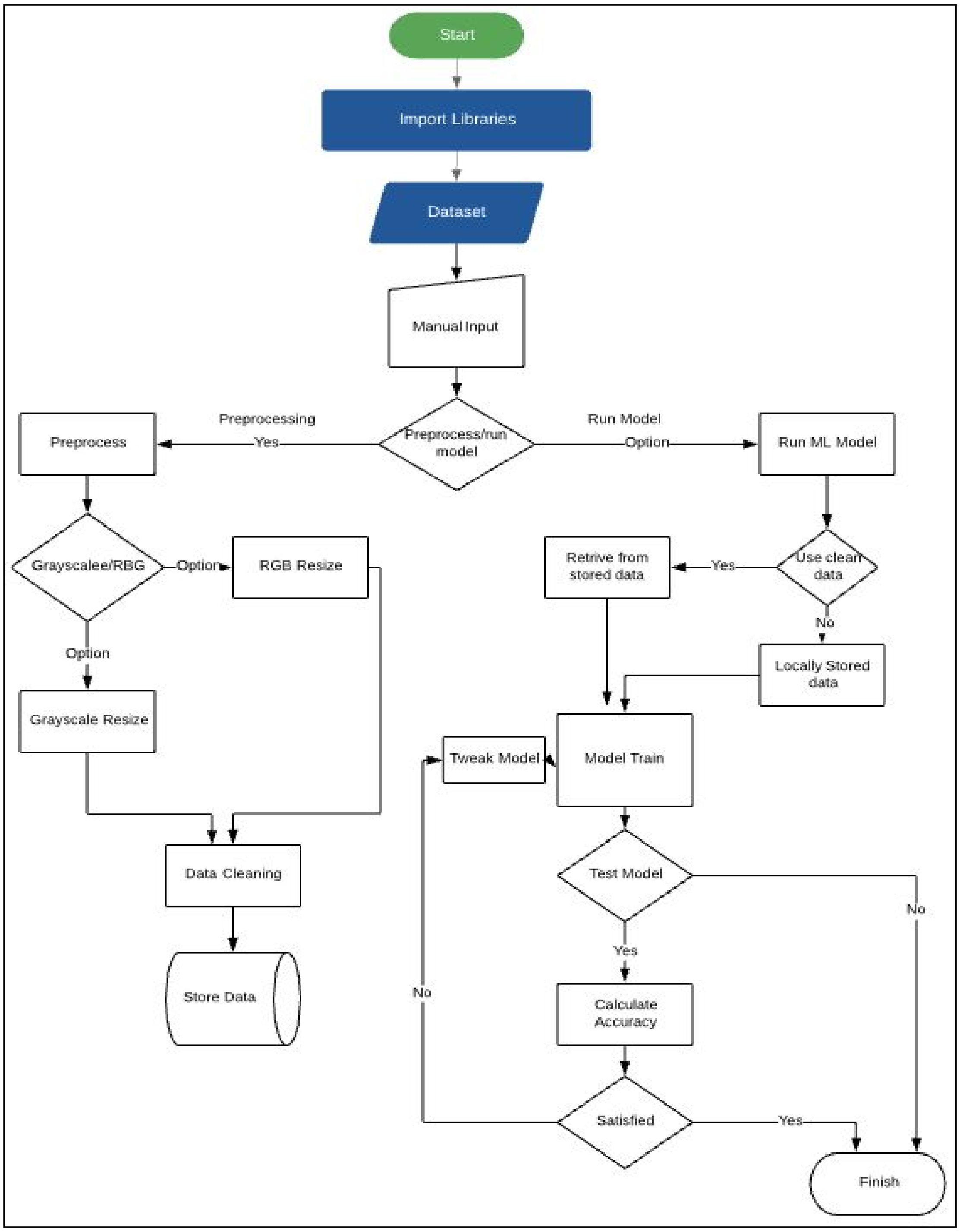


Figure #3: Code Flow Chart

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.

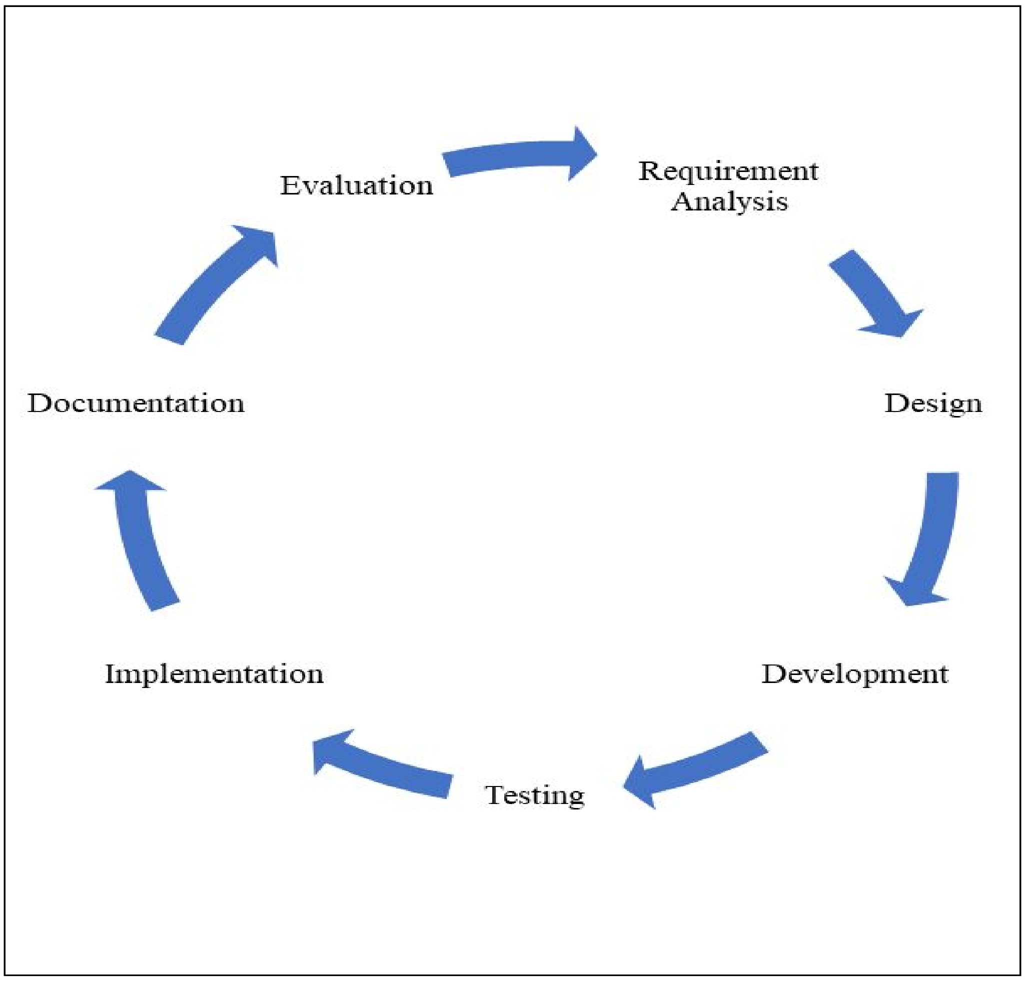


Figure #4: SDLC of the project

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.

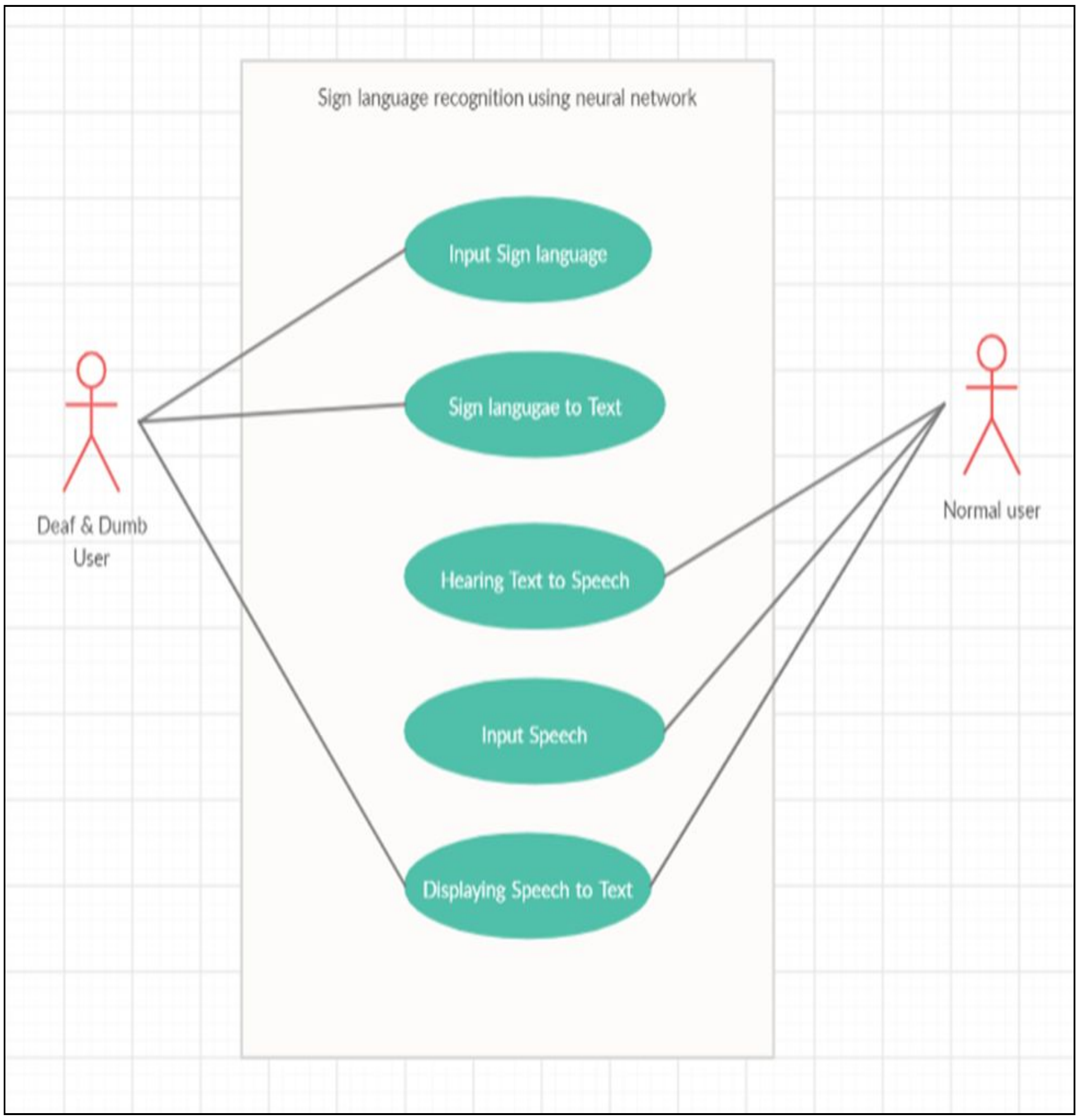


Figure #5: Use Case Diagram

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.

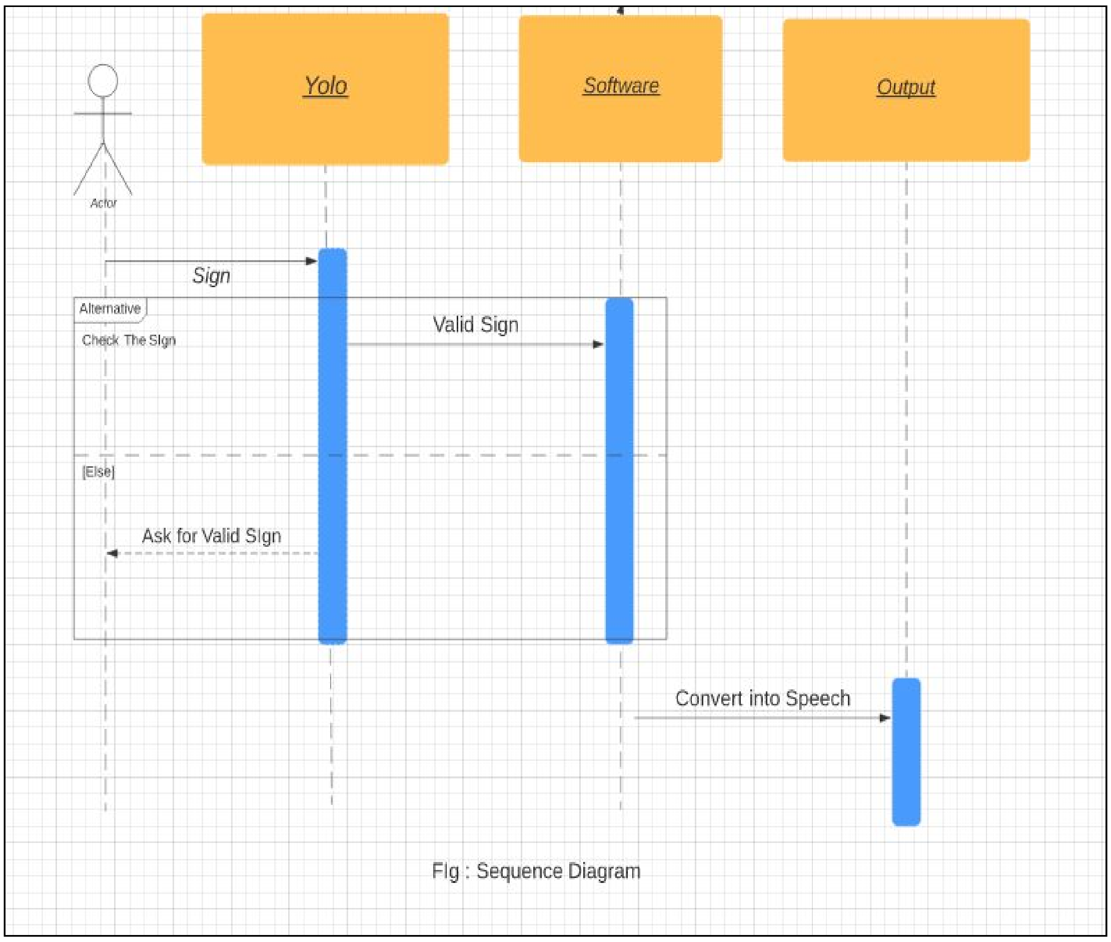


Figure #6:Sequence Diagram

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.

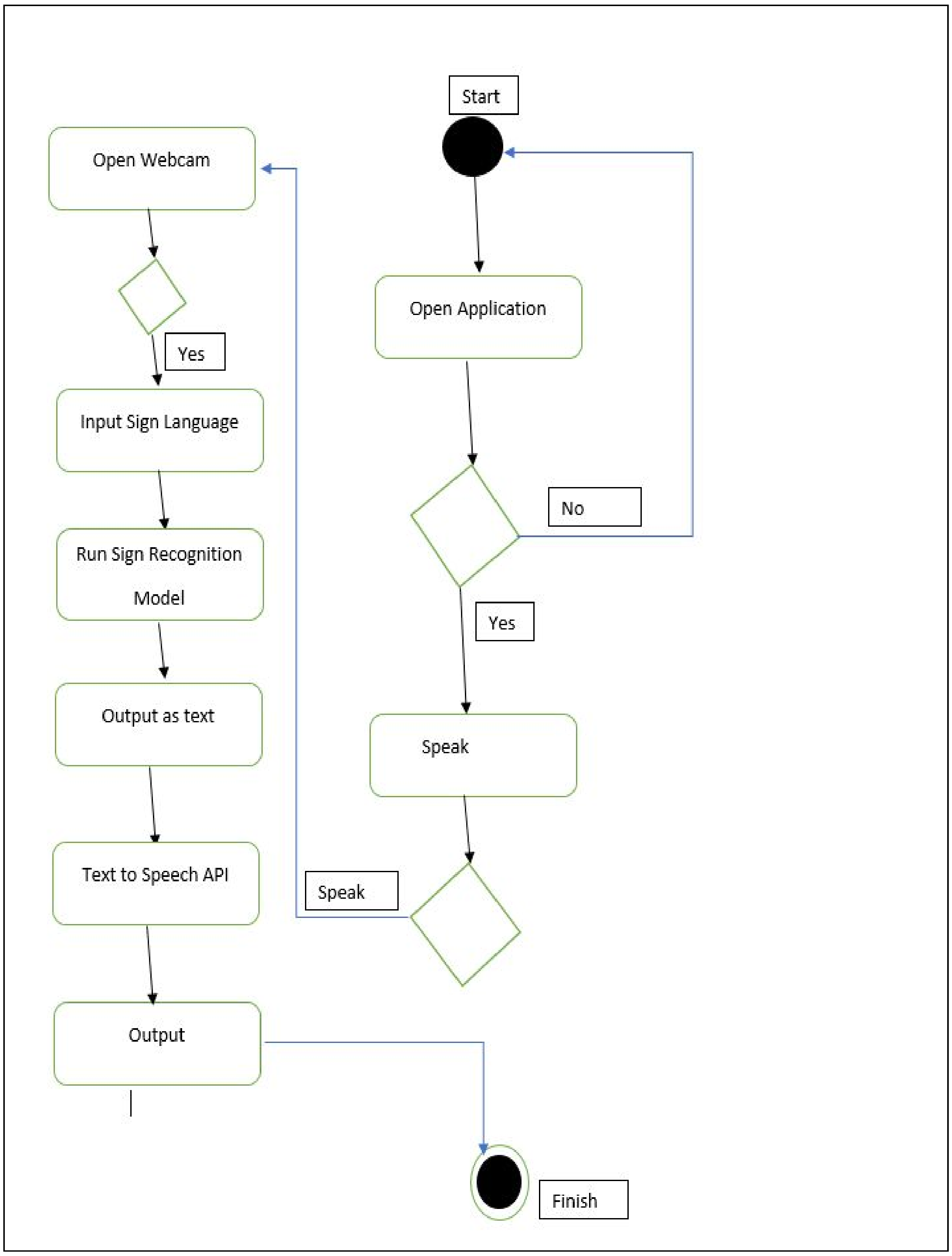


Figure #7: Activity Diagram

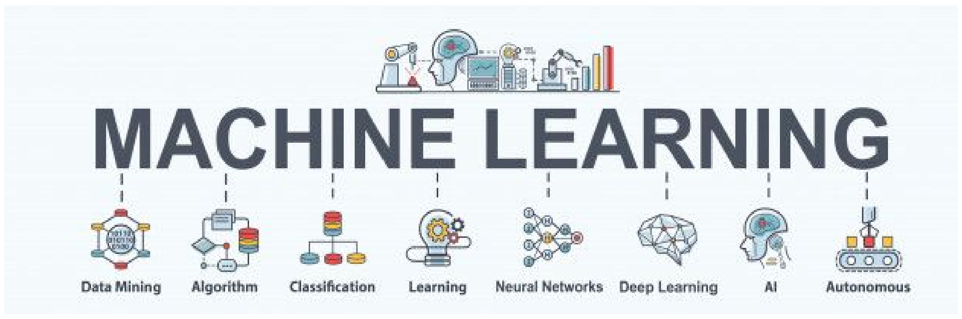
2.6 Required Skills:

Our project is “Sign language Recognition using neural networks”. Every member of our team doesn’t have the required skills to complete all the stages of this project. We should learn these skills to complete our project. Required skills for complete the project given below:

1. Machine learning.
2. Python.
3. Flask web framework
4. JavaScript.
5. OpenCV
6. Keras
7. Kaggle

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 algorithm to build model. Machine learning is an artificial intelligence (AI) technique that allows devices to automatically learn from their experience without being specially trained. Machine learning focuses on developing computer systems that access knowledge and use it for their own purposes.



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.

Flask web framework:

Flask, written in Python, is a micro web framework. We are going to use the flask web platform for web application creation. We can create stable, scalable and maintainable web applications with the use of a flask web framework. So, for this project, we need good information about the Flask web framework.

JavaScript:

JavaScript is a programming language interpreted at a high level, only compiled in time and multi-paradigm. The JavaScript programming language includes curly-bracket notation, dynamics, prototype-based object-orientation and first-class functions.

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.

OpenCV:

OpenCV is a programming library open source. It is an image of a computer and a library of machines. OpenCV is mainly for computer vision in real time. Intel Corporation originally developed OpenCV. The library of OpenCV is cross-platform and BSD permission free. Being a BSD product makes using and modifying the code easy for companies.

OpenCV was created to provide a shared infrastructure for computer vision applications and to accelerate the use of machine perception in consumer products.

In our project we worked with OpenCV.

Keras:

Keras is an open-source library which provides artificial neural networks with a Python interface. Keras serves as an interface for the TensorFlow library. With a focus on allowing fast experimentation, it was developed.It is important to do good research to be able to go from idea to outcome as quickly as possible.

Kaggle:

Kaggle is a subsidiary of Google LLC. Kaggle allows users to find and publicize data sets and models, work with other data scientists and machine teaching engineers in a web-based information science environment and to compete in solve problems .

kaggle

Kaggles is also an online group of data scientists, Data science challenges.

CHAPTER 3
ESSENTIAL PARTS AND DEVICES

3.1 Description of Components:

In this section, we will briefly discuss the tools we have used. Our entire project is written in Python version 3.6. Based on type, we can split the tools into three categories: Library, Framework and Cloud Services

1. Libraries

Pandas:

Pandas is a fast, powerful, scalable and easy to use open source data analysis and management platform, based upon an overlay of Python programming language. It offers data structures and processes, in particular for the management of number tables and time series. Pandas is a free, BSD-licensed library that offers highly-performing and easily usable BSD data structures and data analysis tools for the Python programming language.

NumPy:
Numpy is an array-processing package for general purposes. It offers a high-performance object and software multidimensional array to work with these arrays.

We have lists in Python that serve the purpose of arrays, but are slow to process. NumPy tries to provide an object with an array that is up to 50 times faster than standard Python lists. In data science, arrays are very commonly used, where speed and resources are very important.

Scikit-learn:

It is a free software machine learning library that offers many unsupervised and supervised learning algorithms for the Python programming language. Some of the technology you might already be familiar with, like NumPy , pandas, and Matplotlib, is based on it.

Jupyter Notebook:

The Jupyter Notebook is a software application that helps you to create and exchange live code, calculations, visualizations and texts.

It uses data cleaning and conversion, numerical modelling, mathematical modeling , data visualization, machine education and much more.

2. **Frameworks**

In this section, we discuss the major python frameworks we have used in our project.

TensorFlow:

It is a symbolic math library that is used often for computer study programs like neural networks. TensorFlow is a free and open-source data flow and distinguishable programming software library for a variety of activitie

It has a sturdy, flexible ecosystem of software, databases and community services, which enables researchers to create and deploy applications powered by ML rapidly to advance the latest in ML and developers.

Flask:

Flask is a micro web application written in Python. It is known as a microframework because it does not need specific tools or libraries. It does not provide an abstract storage layer, a type validation or any other compounds where pre-existing third-party libraries have basic functions. However, Flask allows plugins that can incorporate framework features as if implemented in Flask itself.

3. Cloud Services

For training we had to use different cloud services. Here is a brief summary of them.

Google Cloud Platform:

Google Google Cloud Platform (GCP) is a cloud computing suite that runs on the same infrastructure that Google uses for its end user apps internally. Google Cloud Platform is a network infrastructure provider for the deployment and execution of web applications.

Its specialty is to provide people and companies with a place to create and run software, and it uses the web to connect to the software's users.

Google Colab:

Google Colab is essentially a Google-supplied Jupyter Notebook platform for students and researchers. It comes with libraries to easily access different Google services..

It's a Jupyter notebook system that needs no configuration to use and runs entirely in the cloud. Colab offers 25 GB of RAM, so you can load all your data into memory, including for large data sets. Around 2.5x, with the same data generation steps, was found to be the speed up

4. Hardware Component:

For the project, the components we need is :

1. Computer
2. Camera
3. Headphone/Speaker
4. Microphone

Computer:

Requirement of a computer in the development phase of the project is very essential. To accomplish the task of image recognition i.e sign language, requires a high functioning computer/laptop. A computer with high processing power and an adequate graphics memory is a must to run various architectures such as VGG16, RestNet, MobileNet. When the model is under development, it will be in a desktop computer environment. Later on in testing mode, we will transfer all programs into a laptop to use the live feed of the webcam.

Camera:

Camera is required to capture the sign language made by the user. As the system is giving real time feedback, therefore, a camera will capture the motion as well as the video and the live feed will be processed in the computer to recognize the sign and give the output.

Headphone/Speaker:

Microphone is used to change the sign to speech. However, it is not possible to change the sign to speech directly. First, it will take the sign from the live feed. Secondly, it will process the sign and come up with the text. Lastly, using the Text to Speech API it will give output on a speech.

Microphone:

Microphone can be used if the person on the other end tries to communicate with the dumb person. He/she will say what he/she wants to convey on the microphone and the microphone will receive it and the computer will process it using a Speech to Text API and show the message in a text format on the screen.

3.2 Test Requirements:

Required Components/ Softwares/ Modules	**Usage in testing**
Kaggle/Colab	● The entire code base will reside here ● They provide access to GPU
TensorFlow, Keras	● Machine learning and deep learning models and dataset manipulation ● To test uniformness of dataset and unbiasedness ● Models performance is also tested with these modules
Pandas, Numpy, Scikit-learn	● Implementing dataset clean up and augmentation ● Testing the image processing results
Flask Framework	● Debugging server side problem ● Handling database ● Stability assessment of the developing system

CHAPTER 4

WORKING SHEETS

4.1 Work Breakdown Structure:

WBS (Work Breakdown Structure) is a deliverable-oriented division of work which will provide a clear understanding of the procedure that we are going to follow for a 24 weeks time period. It will manage the team's work into manageable sections. It is used to manage the budget, timelines, work division of each member and identifying potential risks during and after the project completion to avoid.

Here we used the Gantt chart to make a WBS of our capstone design project.

Week	Task/Activity	Members involved
1-4	Group Forming, Project Selection, Project Proposal, Presentation	Nayeem Mohammad Rubayet Zaman Zubayer Ahmed
5	Creating the python environment, anaconda, jupyter…………Nayeem Mohammad	
	Setting Git Repo, create .gitignore directory………………………Rubayet Zaman	
	Dataset collection…………………………………………..Nayeem Mohammad	
	Research Software Development…………………………...…Zubayer Ahmed	
6	Preprocessing the data……………………………….….Nayeem Mohammad	
	Augmenting the dataset……………………………….………Rubayet Zaman	
	Research Software Development…………………….………Zubayer Ahmed	
	Preprocessing the data………………………………………..…Rubayet Zaman	

7	Running TensorFlow and Keras module on dataset…………Nayeem Mohammad Image Processing..……………………………………….……...Rubayet Zaman User Interface Language and Platform selection……………….....Zubayer Ahmed
8	Train Machine Learning Model(CNN, YOLO, VGG16)......,..Nayeem Mohammad Train Machine Learning Model(Random forest, ……………..….Rubayet Zaman Multilayer Pereptron) Learning Flask……….……………………………..……..Zubayer Ahmed
9-10	Solving Problems of overfitting and underfitting…………....Nayeem Mohammad Comparing Accuracy and Representing them in graph or table….Rubayet Zaman Creating basic structure block of the UI.exe program …………....Zubayer Ahmed
11-13	Test the model with testing(unseen) dataset…..……………...Nayeem Mohammad and look for ways to improve accuracy Add features to the existing base block of UI & Learn Flask…..…Rubayet Zaman Adding new features to the UI & Learn Flask…..………………...Zubayer Ahmed
14	Test the model with testing(unseen) dataset………………….Nayeem Mohammad Learn Flask……………………….……………………………...Rubayet Zaman

15-18	Website development: Home/Login Frontend…………………………………………….……Rubayet Zaman Sign Language Recognition Page…………………….……..….Nayeem Mohammad Text-to-speech API Create Database……………………………………….…………..Zubayer Ahmed
19-21	Website development: Integrate website with the trained model & website hosting....……...Rubayet Zaman Integrate website with the trained ML model………………..Nayeem Mohammad Host the Complete Website…………………………………..………Zubayer Ahmed
22-24	Test & Improvement Testing Project & Finding Error……………………….....…...Nayeem Mohammad Solving Problems of overfitting and underfitting………...……….....Rubayet Zaman Add features to the existing base block of UI…………….....……..Zubayer Ahmed

Table No 1: WBS of The Project

4.2 Financial Plan and Costs:

This is a research based project.Therefore, the financial bearing cannot be measured .

Things we can look into:

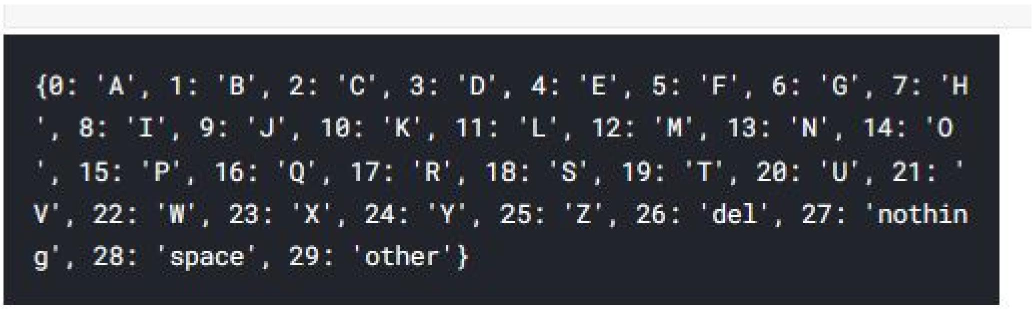
1. We need a PC that must a webcam of higher megapixel
2. We need to invite/hire a professional/expert in sign language who will communicate with the program.
3. We need to commute to various places.

CHAPTER 5
PROJECT SUMMARY

5.1 Result and Discussion:

5.1.1 Dataset

As per our dataset, we had in total 29 classes. After the image load using TensorFlow the following dictionary was created to represent those classes by mapping them with numerical integer values ranging from 0-28.



```
{0: 'A', 1: 'B', 2: 'C', 3: 'D', 4: 'E', 5: 'F', 6: 'G', 7: 'H
', 8: 'I', 9: 'J', 10: 'K', 11: 'L', 12: 'M', 13: 'N', 14: 'O
', 15: 'P', 16: 'Q', 17: 'R', 18: 'S', 19: 'T', 20: 'U', 21: '
V', 22: 'W', 23: 'X', 24: 'Y', 25: 'Z', 26: 'del', 27: 'nothin
g', 28: 'space', 29: 'other'}
```

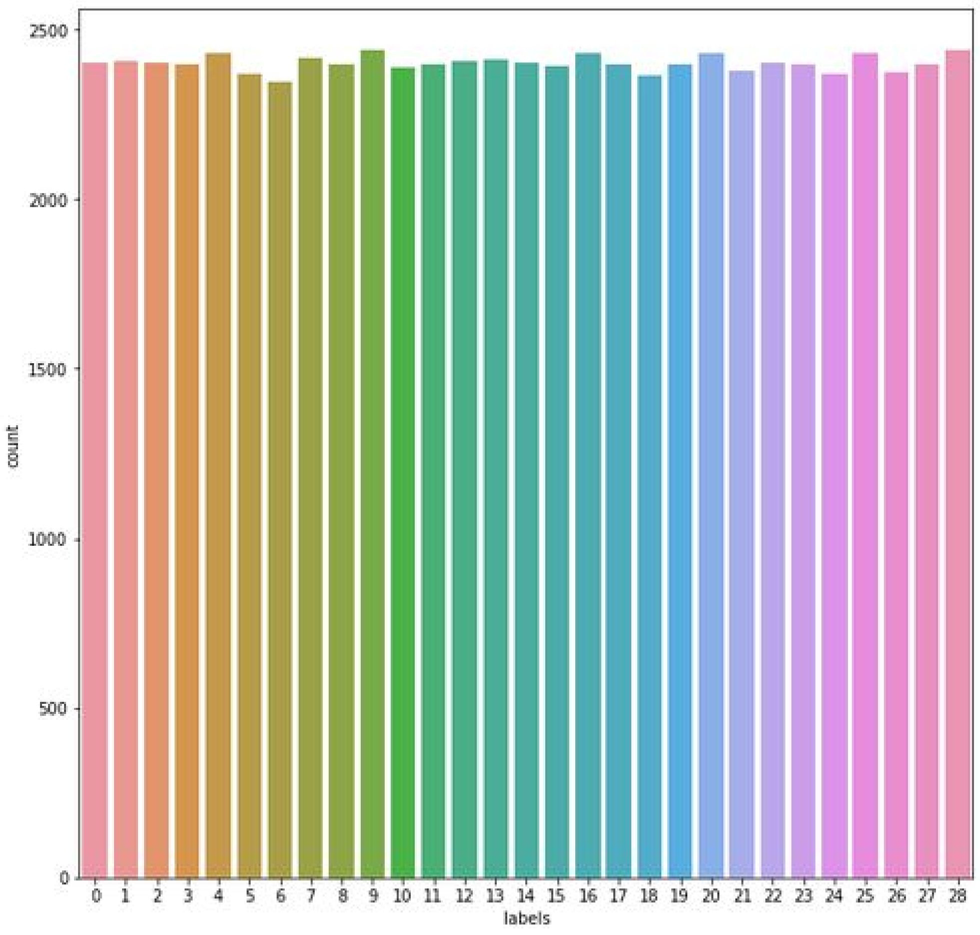
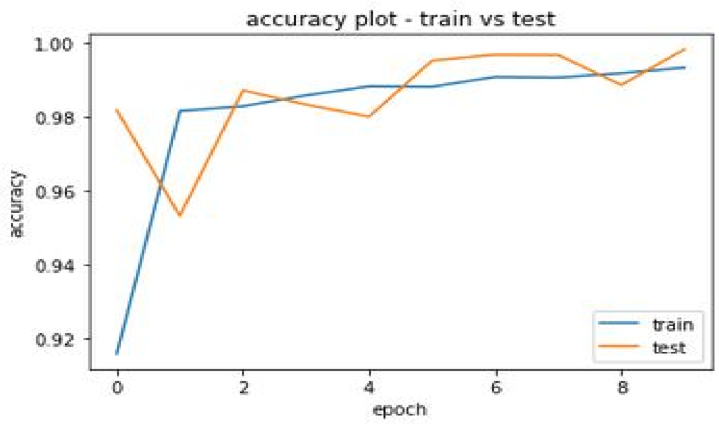


Figure #8: Distribution of the entire dataset in bar graph

The bar graph that is illustrated above represents the distribution of the entire dataset. Since we had 29 classes, we can see that all the classes (x-axis) had approximately the same number of images. They are all ranging around 2400. This shows that our dataset was uniformly distributed and we didn't have any class having a disproportionate amount to representation in the dataset.

5.1.2 VGG16 Architecture results and performance

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.



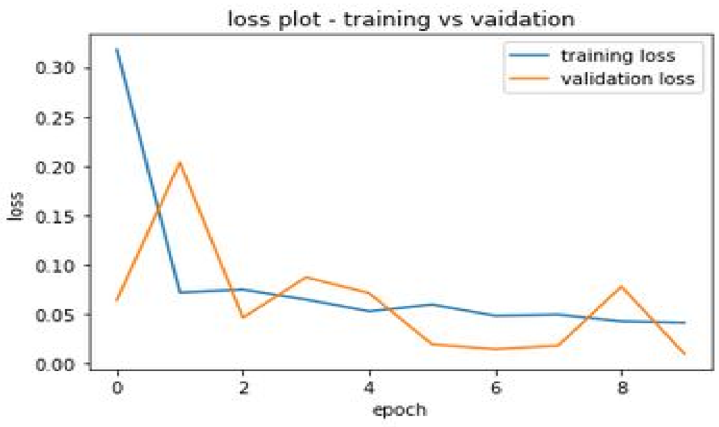


Figure #9: Accuracy and Loss Plot

The above depicted graph indicates that for sequential models it learned very fast in training (accuracy plot) period but on validation it went up and down even at the very end. Same results were evident in the loss line where training loss was steadily declining but the validation loss was rising and plummeting frantically. This was a clear indication that it was ineffective.

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.

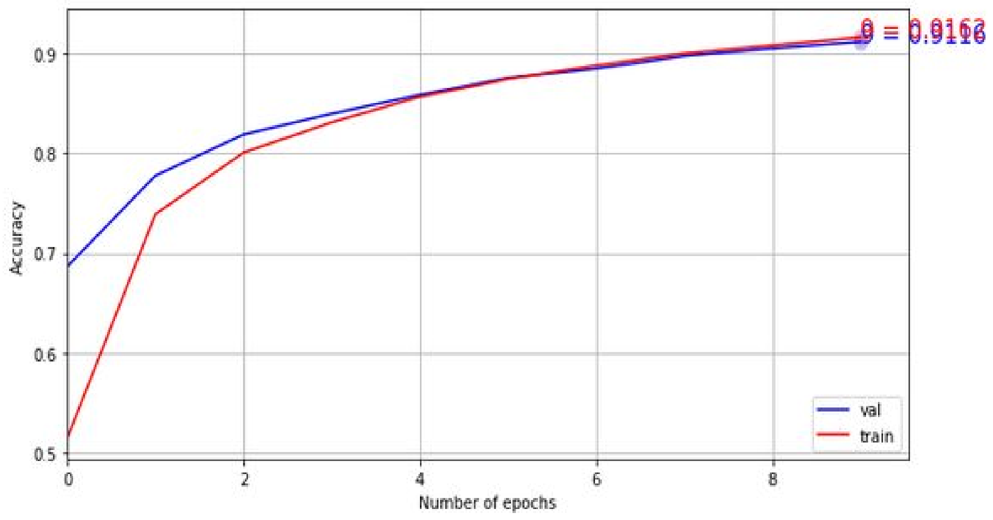


Figure #10: Results of VGG16(Accuracy vs No. of epochs)

This graph illustrates the results of VGG16 training with steady increase as the epochs increase. The accuracy reached well over 90% and following its training the confusion matrix in testing proved signs of the model being overfitted.

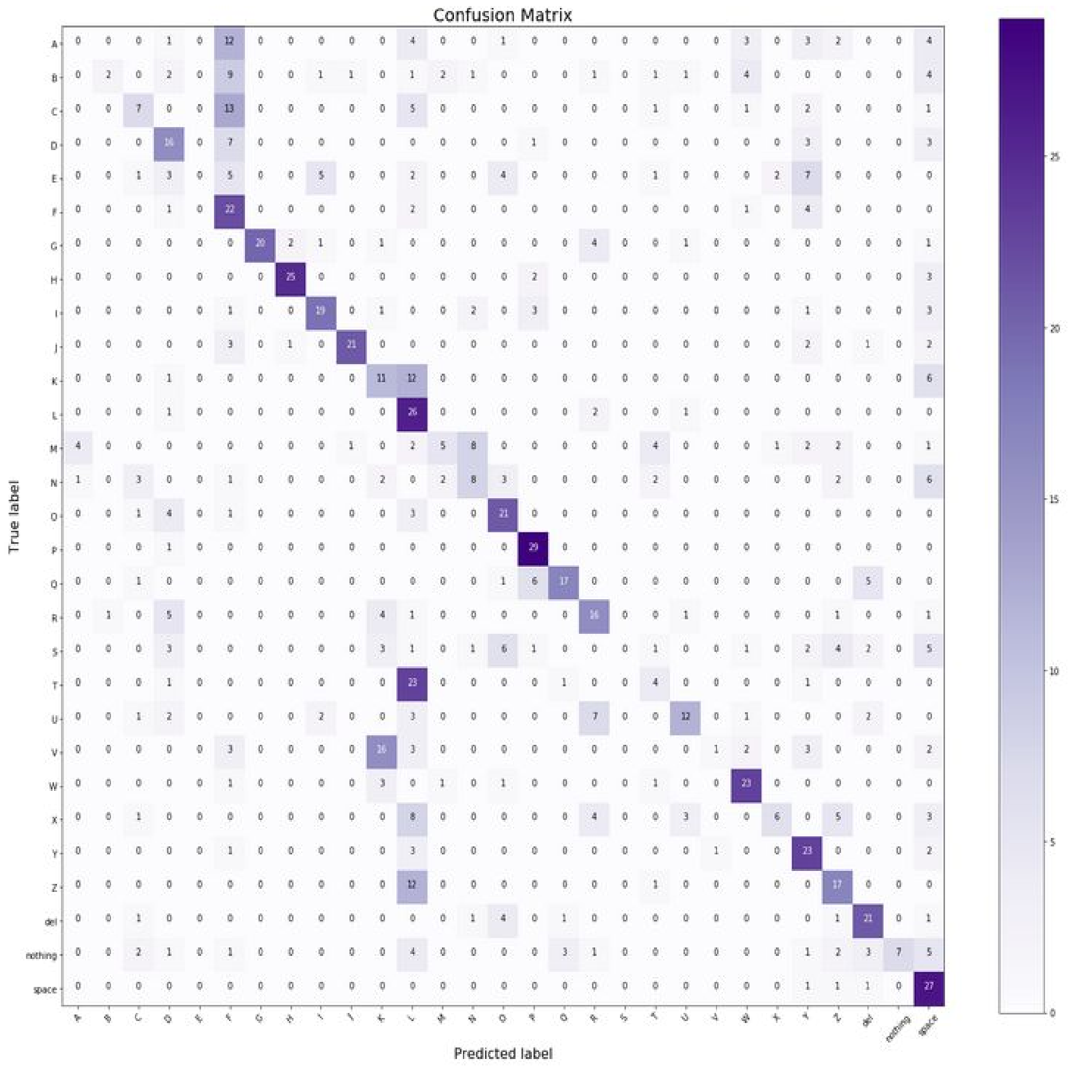
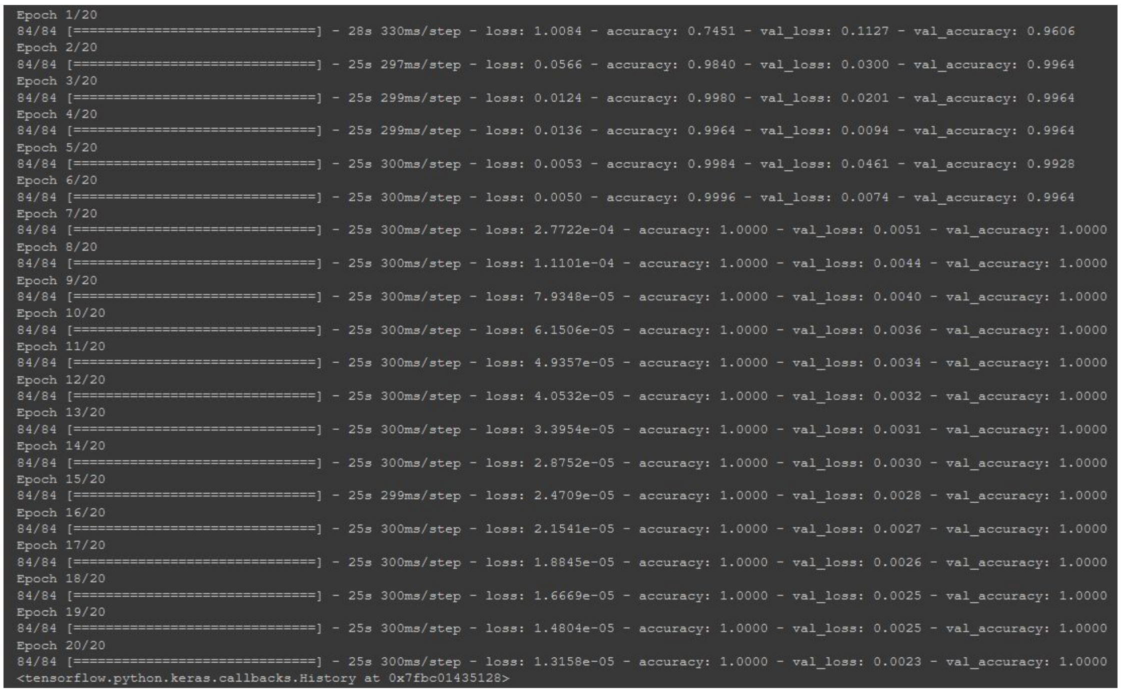


Figure #11: Confusion Matrix using vgg16

The confusion matrix reveals that when the model tried to predict on unseen dataset it wasn't able to generalize properly and the missing high contrast of purple along the diagonal indicates that it predicted wrong on certain classes.

5.1.3 MobileNet Architecture

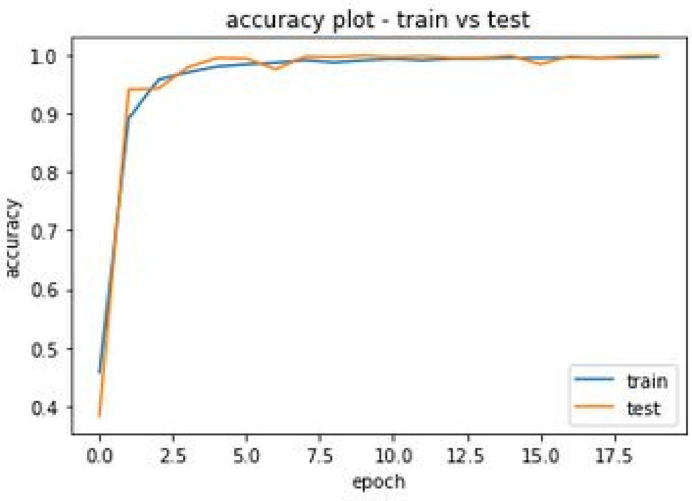
We then switched to MobileNet and finally settled with it because in comparison to VGG16, MobileNet has 4.2 million parameters whereas, VGG16 has 17 million parameters. Also the size of the network in memory is proportionally smaller than VGG16. It loses a little bit of edge in the accuracy department 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>
```

Fig# ‘12’ : Training and Validation logs.

The two graphs that are placed under, were the results of MobileNet model training on our dataset. We split our dataset into training and testing. When the architecture was training it set aside 20% images from the training dataset to create a validation set. In every epoch the architecture tried to iterate the process of learning new features and cross checking its learning with the validation set. This was calibrated in such a way that all the weights and biases were either incrementing or decrementing according to the feature they were trying to learn.



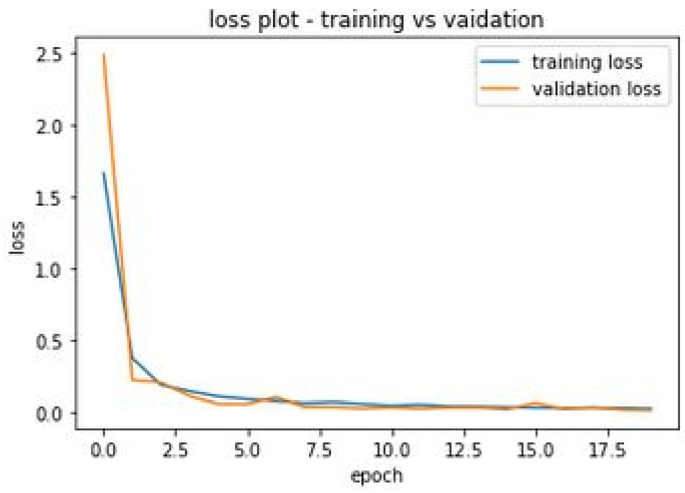


Figure #13: Accuracy and loss in training, testing and validation of MobileNet

The MobileNet was first using its IMAGENET weights and biases. But we realized that using those weights and biases had a downside. The downside was very crucial in the sense that the pretrained model that was using those biases was good at performing the task of recognizing IMAGENET classes. We had to take apart the weights and biases from the MobileNet architecture. Unfreezing the layers and making the layer's parameters trainable was the course of action that seemed to get the best output. The curves below are the training with all the layers and it is evident from the graph that since we unfreezed the layers, the curve took more time to gradually reach a stable form. Whereas, the curve above had a very steep curve indicating erroneous information being learned by the architecture. The graphs depicted below are very smooth and its steady rise indicates that overfitting was being minimized along the training.

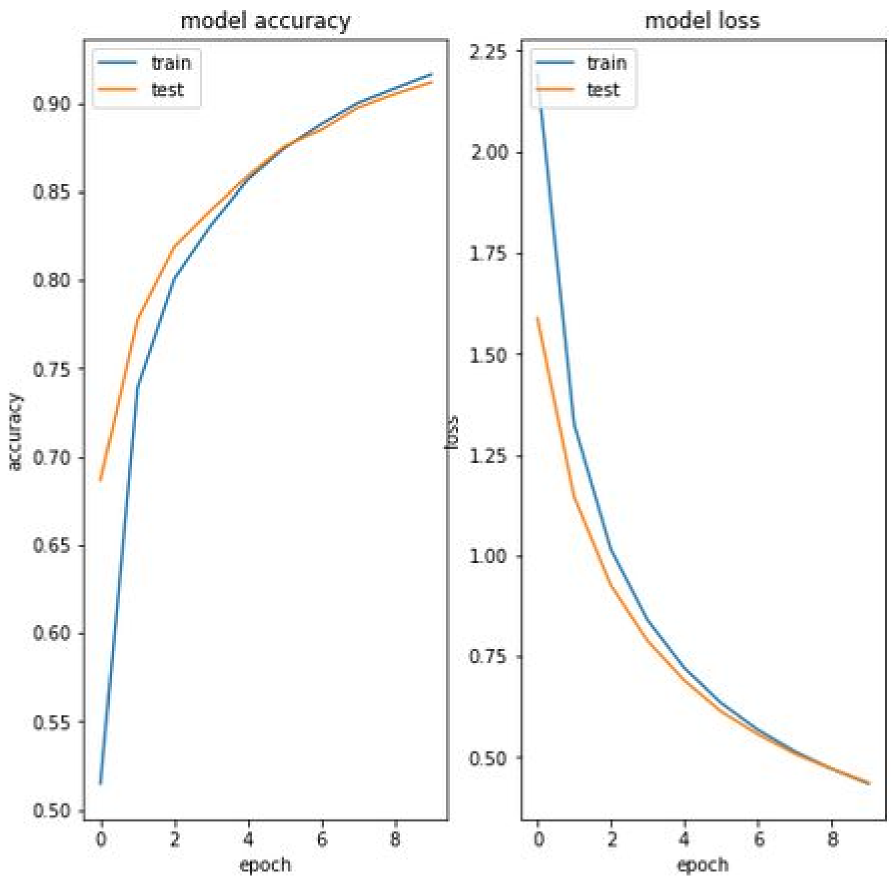


Figure #14: Accuracy and loss against epoch of MobileNet model

MobileNet computation was similar in steps with the VGG16 model. But its black box layers had parameter reduction optimization embedded in it. Those layers intuitively learned complex features without increasing the number of features. That results in the model size shrinking and it's one of the most elegant optimizations we can work on. The confusion matrix that was produced had uniform diagonal only minor mistakes in training. The model was performing very aptitude wise and its error was very insignificant. The confusion matrix below was produced during training and the difference in VGG16's confusion matrix and MobileNet's confusion matrix is unambiguous. Furthermore, in real time prediction MobileNet architecture had significant prediction skill upgrade than VGG16 architecture.

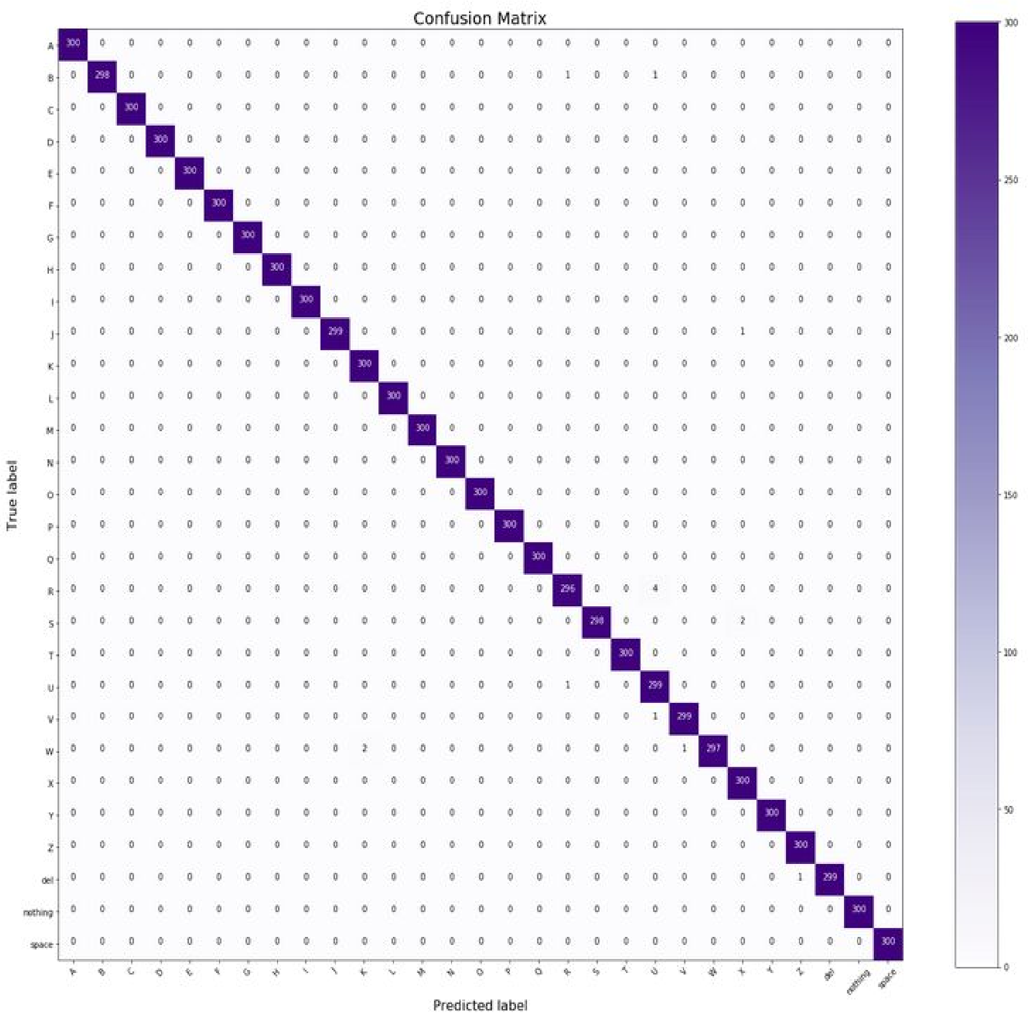


Figure #15: Confusion Matrix using MobileNet

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.

5.2 Feasibility Study:

Identify and validate the impact of environmental considerations:

Sign language recognition using a conventional neural network in our project. Sign language plays a large part in Deaf & Dump cultures. Around 466 million people worldwide have hearing loss, says the World Health Organization. Deaf and dumb people all over the globe use sign language to reach them. In communication with deaf & dumb people, this language was created to overcome problems. In order to communicate with each other, deaf people use sign language. When they interact with physically sound persons, they face difficulties. Because people who are physically healthy don't know sign language. To connect with others, physically sound individuals have their own language. Our aim for the project is to eradicate this challenge. Our project will help us connect with deaf and dumb people. In our project we have a web application. By using web applications 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 a 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, users need Internet accessibility. Without a good internet connection to enter websites. There can be another problem. We can face an overload problem. Too many people can visit websites at a time. We should be able to handle the overload problem of websites. We need to consider the typing speed of the user is greater than the sign language speed of the 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 and we need to ensure that our text to speech api works properly. Our project is for the benefit of human specially for the deaf & dumb people.

Usability:

Our System is Sign language recognition using convolutional neural networks. Our System has a web application & Server where sign language will recognize. Deaf & dumb individuals are the main users of our system. When they interact with physically sound persons, they face difficulties. Our System will eliminate this difficulty. Users can access web applications over the internet. The user will give sign language through the camera module as input in a web application which he wants to translate for communication. Then the web application will send the data to the server to process the data using a data model which uses conventional neural networks. After then recognized letters or words or sentences will be sent to the user end. Then the text will be displayed and text will be converted into speech using google text to speech api. For the use of the website users should have working internet connection. To give input of hand gesture, the user should give his/her hand gesture in camera viewpoint. Otherwise input given from the user will not count as input. There are notification system in our website. If you do wrong input or anything wrong then it will give us notification. The User interface of our website is very user friendly. Users will easily understand how to do tasks. Our system is very user friendly and easy to use.

Manufacturability:

For deaf and dumb people, our system is easy to use. Our system contains a web application & Server where sign language will recognize. There is a web application that can translate the sign language of the ASL into the English language and then turn it into speech. Our system is easy to manufacture. The Manufacturing cost is not that much of our system. In our project we have a machine learning model which will recognize sign language through web application. We need a Web host server that can run our system. Host server needs to run our machine learning model. So our system needs high GPU & CPU power for sign language recognition. That is an online version. In the future our system can also implement the offline version. It can be desktop application & mobile application.

Sustainability:

According to the world health Organization around 466 million people around the world are deaf & dumb. 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. This project will break language barrier between deaf & dumb people. Our system will give an easy communication medium with another person who is physically sound. This will be very beneficial for them. Users can use our system everyday.

Non Technical Constraints:

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 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. In our project development phase, there are no Non Technical Constraints of our project. Our project is for the benefit of our society. There are no political, legal issues. We are using an open source api/tool for our project. So, there are no economic issues. Our application will be for the benefit of deaf people. Our project has no health concerns for them and it is safe to use.

Ethics:

Engineering is the development of an effective process, which uses scarce capital to speed and simplify work with the aid of technology. Ethics are social values, and they also correspond to the moral standards of human beings. An honest engineer would best serve humanity. The analysis of engineering ethics, in which engineers introduce a certain amount of engineering ethics, is therefore essential for the benefit of humanity. The study of the decisions , policies and values that Engineers are supposed to possess the highest degree of integrity and competence as members of this discipline. Engineering has a direct and important effect on the quality of living for all individuals. The services provided by engineers thus require honesty, impartiality, fairness and equity, and must be committed to public health, protection and welfare. Engineers must operate in compliance with a professional ethical framework that requires a dedication to the highest level of ethical behaviour. The ethical standard of engineering practice and study is Engineering Ethics. Throughout the world's various tech firms and charter companies, the fundamental principles of the Code of Ethics are basically the same. These principles of engineering ethics:

1. Engineers are paramount to protecting the public, health and well-being in the fulfillment of their professional duties and aspiring to meet the principles of sustainable development.

2. Engineers will only perform services in the areas of their competence.

3. Engineers will only make public statements in an objective and genuine manner.

4. In professional matters, engineers act as loyal agents or trustees of each employer or client and avoid conflicts of interest.

5. Engineers build their professional prestige on the merits of their services and should not contend arbitrarily with other engineers.

6. Engineers shall behave in such a manner as not to tolerate misconduct, theft, and abuse that the honors, integrity, and reputation of the engineering profession will be maintained and enhanced.

7. During their technical lives, engineers continue their professional growth and, under their supervision, give these engineers potential for professional development.

8. Fair treatment by engineers of all individuals is given, regardless of gender or gender identity, race, national origins, nationality , religion, age , sexual

orientation and disability, political affiliation or family, marital or economic status, in all matters related to the occupation of each citizen.

9. Unless authorized or required in compliance with law or this Code, engineers shall not reveal facts , data or data without the consumer and/or employer 's prior consent.

10. Engineers shall not reveal, without permission, proprietary details relating to the corporate affairs or technical processes of any current or former client or employer or public entity to which they serve.

11. Engineers shall not pursue jobs or advancement or professional dedication by unfairly criticizing other engineers or by other inappropriate or dubious methods.

12. Engineers shall give credit to those to whom credit is owed for engineering work, and shall consider the ownership rights of others.

In our project we practiced Engineering Ethics. Practice Engineering Ethics is part of our mission. Our Sign Language Recognition initiative uses neural networks for society's deaf and dumb people. Our plan is for deaf and dumb people to remove the challenges they encounter when interacting with people who are physically sound. Our goods are safe to use and have no health issues. The client / user will maintain their trust in us. Our project does not harm society. We are using open source development software for our project. There are no conflicting interests in our project. Members are not unfair to each other, are honest, respectful to each other, demonstrate professional conduct.

5.3 Problem Faced and Solutions:

5.3.1 Weighting the pros and cons of Kaggle Dataset and MNIST Dataset

At the very onset of our project, we faced myriad challenges, and we obviated them one by one. The first challenge was finding the right dataset as the neural network models heavily depend on the quality of the dataset. We focused on mainly two datasets from kaggle. One was a really popular MNIST Sign Language dataset and the other was a private dataset. As we started to work using these two datasets we realized that the models that learned from those datasets were heavily biased. The

MNIST Sign Language dataset consisted of images of size 28x28. The frame that we were going to extract from in the real application was of size 480x640. The only viable solution was to convert the images into the shape of 28x28. But then the images will lose a lot of features and subsequently a lot of contexts will be erased. The second dataset didn't have this problem. It had images that were of a size 200x200 pixels. Our feed will generate frames of size 480x640 and we will crop a portion of the image to match the similar size of the second dataset. The private dataset had one flaw that really put a lot of trouble in our path and that is, the images were augmented images. Meaning they have augmented variations of hue, saturation, rotation, zoom, the brightness of a handful of images. A handful of images were actually taken and then augmented using an image generator to make the dataset large. This leads to a problem of overfitting. The models were not able to generalize properly and they were not really learning different features. They were learning to recognize only those specific images and not focusing on the key features. This particular problem was solved very late because it was evident to us at the very end of our project.

5.3.2 Model overfitting and lack of robustness in performance

We could have kept on trying to fit our model but the model always overfitted. The dataset was simply not diverse enough to train a model to be robust. The real-time application that we were striving to develop had too much noise that was not even possible to reduce through image processing methods. In comparison, we choose the model that was trained on the second dataset because of the high pixel values in them.

5.3.3 Pretrained model VGG16's enormous size slowed the training and prediction process significantly

We first started working with VGG16 and stumbled into an array of unexpected results staring at our faces. The complexity of VGG16 is remarkable in the sense that it has 21 blocks of convolution and pooling layers and they make up to 17 million parameters. Its weights and biases were more than 500 MB in size. This type of network was not compatible with the hardware resources we had and it became unfeasible to run this type of network on a simple laptop. We tried to do the training in our local machine but it proved to be impossible as the training estimated time was more than hundreds of hours. We needed to search for other resources that were available that we could use. One way to resolve the issue was to use kaggle or colab. Both of these two online platforms are known for their limited access to a powerful GPU. The pros of using kaggle were that you could turn on GPU whenever you wanted to use GPU. Since the dataset was present in kaggle, the loading time of the dataset was fairly minuscule. Whereas, colab didn't allow you to switch on GPU in the middle of running a session. It had to start all over from the top every time we needed a GPU. And the dataset had to be imported from kaggle which took hours in colab. In the end, we used kaggle to train our VGG16 model. Still, the amount of GPU access that kaggle gave us was not enough to train the VGG16 model. And our results were not up to our expectations and we had to come up with an alternative approach.

5.3.4 MobileNet architecture problems and speed vs accuracy tradeoff

In response to our problem, we switched our training model. It was MobileNet Convolutional Network and its key feature was it was super lightweight and fast to be on a mobile device. MobileNet had many layers like the VGG16 model but it used pooling to reduce its size with a high compression method. Its weights and biases were only 62 MB in size and it was implemented as a proper replacement of VGG16.

Though we used MobileNet but we couldn't use its IMAGNET weights and biases. Everytime we freezed most of the layers and trained the last 20 layers it didn't do well in the prediction. We had to discard all the IMAGENET weights and biases and train from input layer to output layer. It increased our model training period but the results were an improvement on its predecessors.

5.3.5 Module versions compatibility issues

When the website was being developed various modules were added. Jinja, SQLalchemy, SQlite, Bcrypt, opencv, tensorflow to name a few. This modules were not compatible with their latest release. It was compatible with their stable releases. In some cases we had to use modules that were intended for non-GPU platforms to develop the program in our local machine. Most of the times we needed to check the version of the modules that we are going to use and if that module is compatible with the existing modules.

5.4 Future Development:

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 have many sponsors and we can also give advertisements on websites for the cost reduction.

5.5 Conclusion:

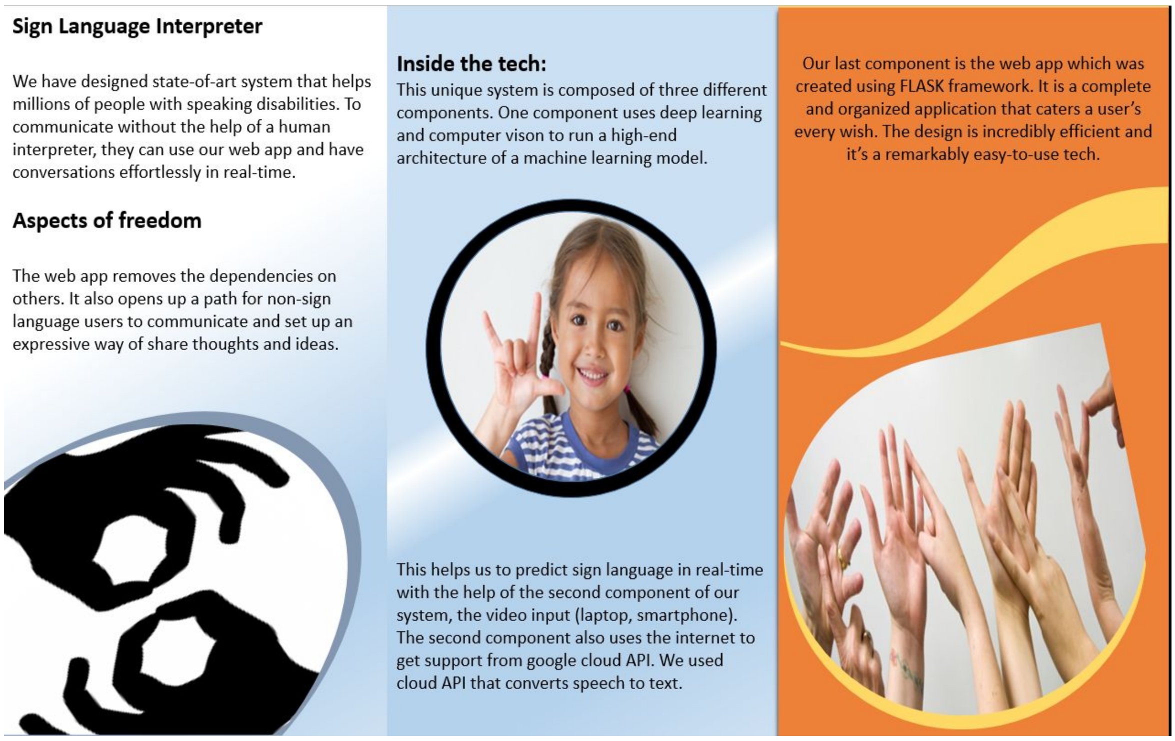
Communication is one of the important requirements for survival in society. Sign language is the main communication medium for the deaf and mute people. American sign language(ASL) is the most used sign language in the world. Many physically sound people don’t know sign language. Our project aims to eliminate the difficulties. Our initiative would close the contact distance between normal people and deaf and ignorant people through sign language. The successful expansion of this project to words and everyday phrases will not only make it simpler and simpler for deaf and stupid people to communicate with the outside world, but it may also promote the development of autonomous networks of understanding and assistance.

5.6 Project Demonstration Review:

During our demonstration we went through a vigorous fact checking test.The points we explained in the demonstration is mentioned below:

1. Firstly, we explained briefly about how we trained the machine learning model and the code behind it.
2. Then we gave a clear overview of how the website works after integrating with the pretrained model
3. Lastly, we demonstrated by giving a hand gesture on the website and converted it into a speech.
4. Finally, we wrapped up our demonstration by answering the several questions asked by our supervisor.

5.7 Brochure:



Sign Language Interpreter

We have designed state-of-art system that helps millions of people with speaking disabilities. To communicate without the help of a human interpreter, they can use our web app and have conversations effortlessly in real-time.

Aspects of freedom

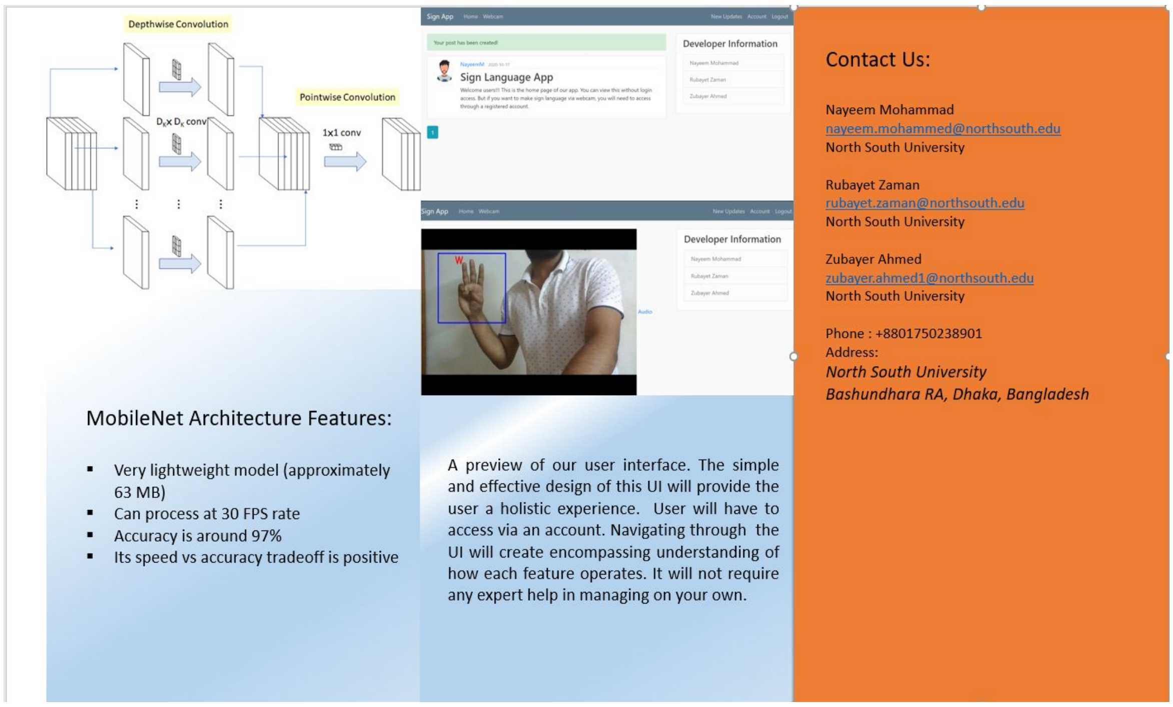
The web app removes the dependencies on others. It also opens up a path for non-sign language users to communicate and set up an expressive way of share thoughts and ideas.

Inside the tech:

This unique system is composed of three different components. One component uses deep learning and computer vison to run a high-end architecture of a machine learning model.

This helps us to predict sign language in real-time with the help of the second component of our system, the video input (laptop, smartphone). The second component also uses the internet to get support from google cloud API. We used cloud API that converts speech to text.

Our last component is the web app which was created using FLASK framework. It is a complete and organized application that caters a user's every wish. The design is incredibly efficient and it's a remarkably easy-to-use tech.


Depthwise Convolution
$D_K x D_K$ conv
Pointwise Convolution
1x1 conv
Sign App
Home
Webcam
New Updates
Account
Logout
1
W
Audio

Your post has been created!

NayeemM 2020-10-17

Sign Language App

Welcome users!!! This is the home page of our app. You can view this without login access. But if you want to make sign language via webcam, you will need to access through a registered account.

Developer Information

Nayeem Mohammad

Rubayet Zaman

Zubayer Ahmed

MobileNet Architecture Features:

- Very lightweight model (approximately 63 MB)
- Can process at 30 FPS rate
- Accuracy is around 97%
- Its speed vs accuracy tradeoff is positive

A preview of our user interface. The simple and effective design of this UI will provide the user a holistic experience. User will have to access via an account. Navigating through the UI will create encompassing understanding of how each feature operates. It will not require any expert help in managing on your own.

Contact Us:

Nayeem Mohammad
nayeem.mohammed@northsouth.edu
North South University

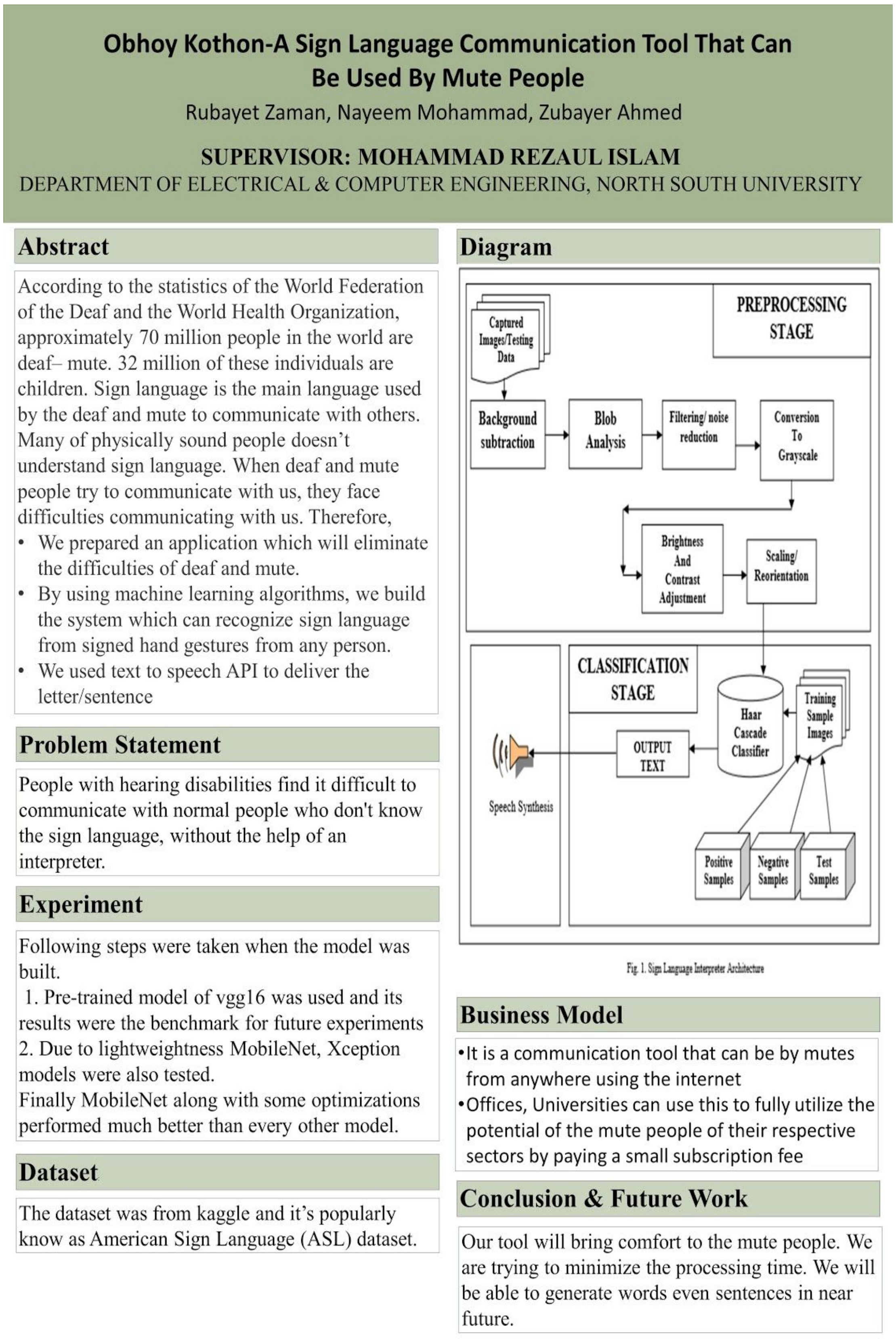
Rubayet Zaman
rubayet.zaman@northsouth.edu
North South University

Zubayer Ahmed
zubayer.ahmed1@northsouth.edu
North South University

Phone : +8801750238901
Address:
North South University
Bashundhara RA, Dhaka, Bangladesh

Figure #16: Brochure

5.8 Poster:



Obhoy Kothon-A Sign Language Communication Tool That Can Be Used By Mute People

Rubayet Zaman, Nayeem Mohammad, Zubayer Ahmed

SUPERVISOR: MOHAMMAD REZAUL ISLAM

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING, NORTH SOUTH UNIVERSITY

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 of physically sound people doesn'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 the system which can recognize sign language from signed hand gestures from any person.
- We used text to speech API to deliver the letter/sentence

Problem Statement

People with hearing disabilities find it difficult to communicate with normal people who don't know the sign language, without the help of an interpreter.

Experiment

Following steps were taken when the model was built.

1. Pre-trained model of vgg16 was used and its results were the benchmark for future experiments
2. Due to lightweightness MobileNet, Xception models were also tested.

Finally MobileNet along with some optimizations performed much better than every other model.

Dataset

The dataset was from kaggle and it's popularly know as American Sign Language (ASL) dataset.

Diagram

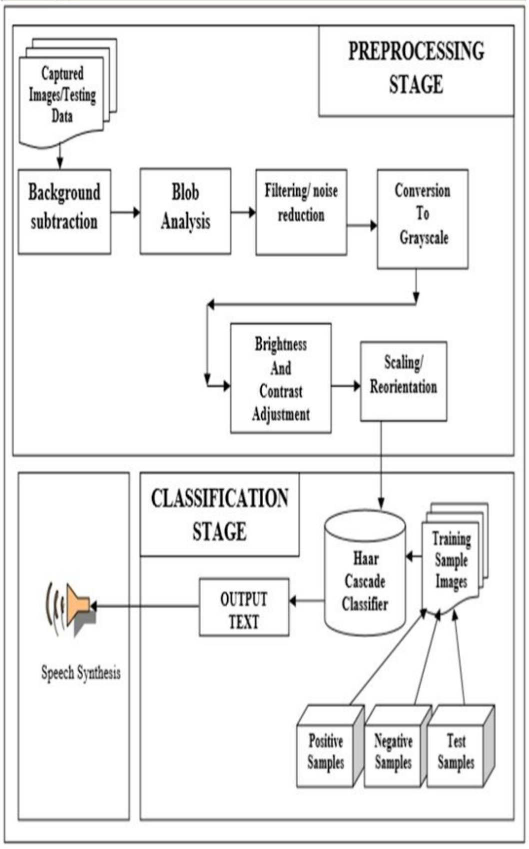


Fig. 1. Sign Language Interpreter Architecture

Business Model

- It is a communication tool that can be by mutes from anywhere using the internet
- Offices, Universities can use this to fully utilize the potential of the mute people of their respective sectors by paying a small subscription fee

Conclusion & Future Work

Our tool will bring comfort to the mute people. We are trying to minimize the processing time. We will be able to generate words even sentences in near future.

Figure#17 : Poster

5.9 IEEE format paper:

Obhoy Kothon - A Sign Language Communication Tool That Can Be Used By Mute People

1st Nayeem Mohammad
Electrical and Computer Engineering
North South University (of Aff.)
Dhaka, Bangladesh
nayeemmohammed@norhtsouth.edu

2nd Rubayet Zaman
Electrical and Computer Engineering
North South University
Dhaka, Bangladesh
rubayet.zaman@northsouth.edu

3rd Zubayer Ahmed
Electrical and Computer Engineering
North South University
Dhaka, Bangladesh
zubayer.ahmed@northsouth.edu

4th Mohammad Rezaul Islam
Lecturer, Electrical and Computer Engineering
North South University
Dhaka, Bangladesh
mohammad.rezaul@northsouth.edu

Abstract—**In this era, machines are continuously striving to imitate the human learning process in order to alleviate our struggles. Using the advancement of technology in the sector of human machine interaction and with the help of Convolutional Neural Networks, machines are now able to solve complex problems of image recognition. One of the main reasons why deaf people can't lead a normal life is because there are so few people who possess the ability to perform sign language who aren't deaf. To obviate this interaction barrier, we have come up with an elegant and innovative solution. Web app that we are developing will aid a deaf person to communicate with any person without the help of an intermediate interpreter. Our system will allow the user with the hearing disabilities to make signs and turn the signs into corresponding letters to form speech.**

Index Terms—**Sign Language, American Sign Language (ASL), Deep Learning, Computer Vision, Artificial Intelligence, Machine Learning, Flask Web-framework**

I. INTRODUCTION

Sign Language mainly uses motor driven 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.

II. PROPOSED SOLUTION

Our project is Sign language recognition using conventional neural network. Sign language plays great role in deaf amp; dump communities. According to the world health Organization around 70 million people around the world have hearing loss and mute. Deaf and dumb people around the world uses sign language for their communication. This language made for overcoming problem in communication with deaf amp; 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 amp; dumb people to communicate with us. In our project we build a model using conventional neural network which can detect word or sentence from Hand gesture of American sign language. We worked with American sign language because it is the most used sign language in the world. In our project we have a web application. We design this web application with the model that can recognize American Sign Language(ASL). By using web applications 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.

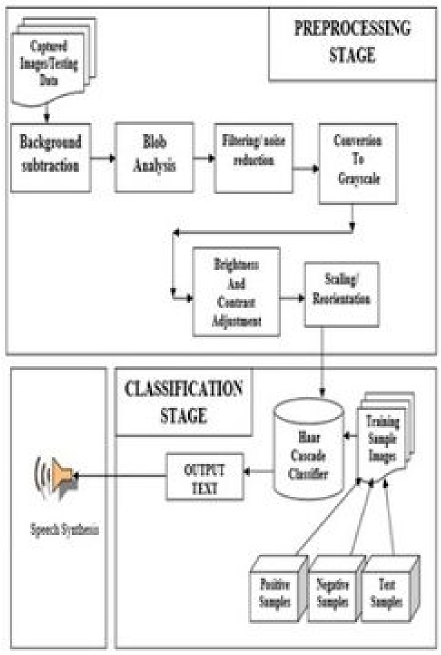


Fig. 1. Diagram of the full program

A. Solution Assessment

We have digged through many academic papers based on ideas similar to our proposed idea. We have found interesting solutions that were intended to help the person with hearing disabilities. Some of the key extinguishable features of our solution are that our's will be lightweight and efficient in time. Its cost will be significantly miniscule and its user friendliness will further simplify the problems associated with real-time systems. One thing to mention that none of the existing solutions were using a web application alongside their machine learning models. Since we integrated with a website, our solution represents a level of uniqueness that is non existent in other solutions.

III. Dataset

We had to consider data from two datasets as a source. One was the MNIST Sign Language dataset. It has images of 28x28 pixel values and that proved to be difficult to train. The other one was the American Sign Langauge dataset with images of 200x200 pixels. The high resolution proved to vital in training as it had much more information stored in the huge number of pixels. We still had to use image augmentaion and other method of image processing on the dataset to increase is usability.

IV. VGG16 Architecture and 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 data set we couldn't just rely on scraped models.

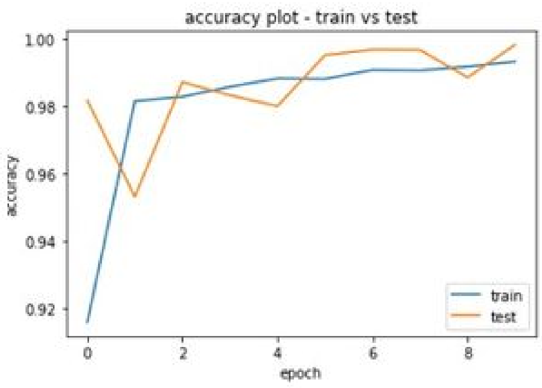


Fig. 2. Accuracy plot of VGG16

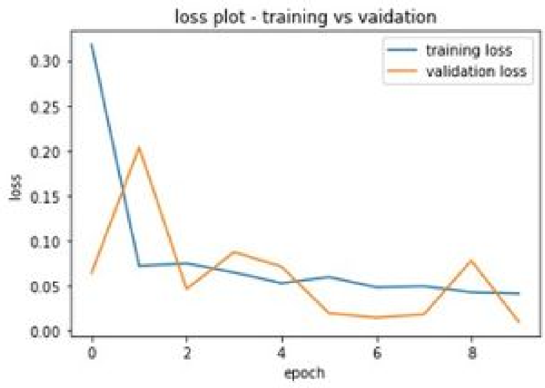


Fig. 3. Loss plot of VGG16

The above depicted graph indicates that for sequential models it learned very fast in training (accuracy plot) period but on validation it went up and down even at the very end. Same results were evident in the loss line where training loss was steadily declining but the validation loss was rising and plummeting frantically. This was a clear indication that it was ineffective.

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 lot time to load into the system which is not very efficient. system works in real-time and it's highly inefficient to large networks such as VGG16,ResNet etc.

V. MobileNet Architecture

This section is dedicated to the explanation of core la that MobileNet is built on. MobileNet usese the com method of depthwise separarble filters. It has embedded model shrinking hyper-parameters known as width multi -| and resolution multiplier.

A. *Depthwise Separable Feature Design*

Depthwise separable convolutions, which has the natur factorized convolutions, can separate features from noise very robust manner. The filters are 1x1 convolutions kn as pointwiese convolution. In the depthwise convolutioı mobilenet, it applies 1x1 convolution so that it can transf...... the input channels from the very beginning of operations. There are multiple blocks of convolutions that has sets of maxpooling layers, dropout layers. This helps to reduce the size of the network significantly and in turns makes the mo achieve remarkable speed in predicting objects.

VI. Results and Analysis

We have worked with multiple architectures because wh model will be suitable for our project was ambiguous. ' first tried with a general approach with normal Sequen models and created layers while stacking them up. The resı were poor and we got accuracy around in percentage but the models overfitted. It was not generalizing prope and due to the lack of high quality dataset we couldn't j rely on scraped models. The predefined models from ke such as VGG16, ResNet, Inception, Xception etc were trair on the IMAGENET dataset and were quite reliable if tl were manipulated with shrewd perception. The pre-trair networks were very large and contained many layers ȧ millions of parameters. So, one optimization technique v showing promising implications by freezing most of the lay and using the technique of transfer learning. Since we freezeu 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.

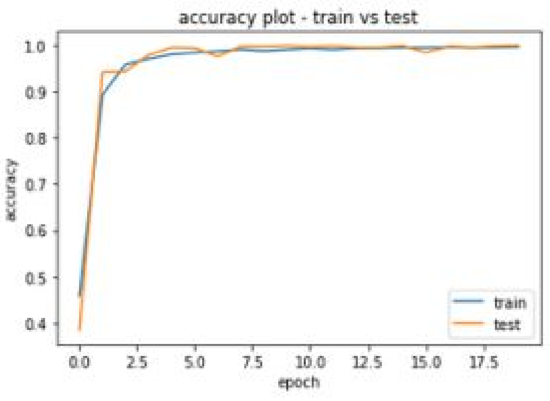


Fig. 4. Accuracy plot.

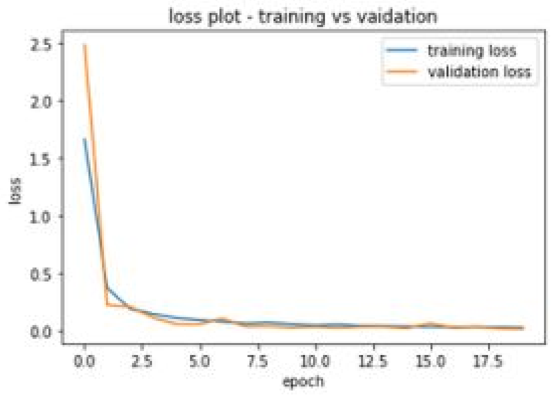


Fig. 5. Loss plot

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.

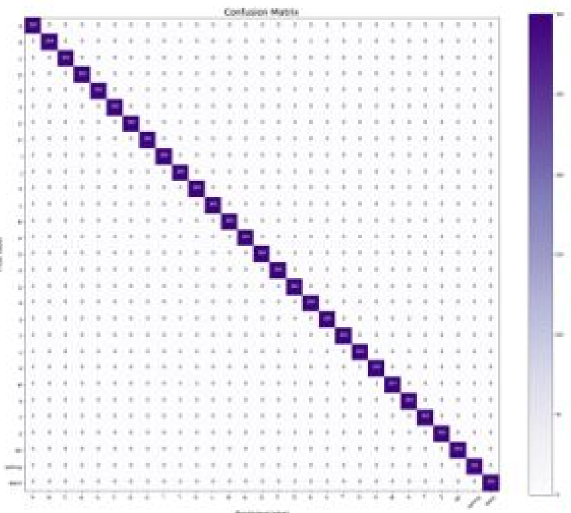


Fig. 6. Confusion Matrix of MobileNet

Though we have built it to the best of our knowledge it lacks necessary adaptation that is required to handle real-time generated problems.

DISCUSSION

This research based project has proven to be extremely challenging. We had to power through many obstacles in order to finalize our work. The limited resources issue was crippling at the beginning of the project until we figured out solutions that will enable us to complete our work. One of the most daunting problems staring down on our face was the dataset. There were not many resources from which we could choose and so we had make do with the one we had. The model's accuracy and prediction in the training was excellent. But there is an array of differences between a real-time image and dataset. Which is the reason why our model was not robust enough to make perfect predictions. We experimented with multiple models and we choose the model that was good in accuracy and also not very heavy in size.

FUTURE WORK

There are couple of new ideas and approaches that we couldn't try out during the development because of the lack of resources. For example, we could use models that can learn while predicting in test run. Reinforced Learning is a whole new field in the deep learning world and it is being used in autonomous vehicle operation with huge positive feedback. We could try to increase the the robustness by using a single color glove in the users hand and then extract the hand only to use learn from it. This are the promising aspects that has come to our perception and we would love to work in the future on them.

REFERENCES

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

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

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

[4] A. S. Nikam and A. G. Ambekar, "Sign language recognition using image based hand gesture recognition techniques," 2016 Online International Conference on Green Engineering and Technologies (IC-GET), Coimbatore, 2016, pp. 1-5, doi: 10.1109/GET.2016.7916786

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

[6] Efthimiou, Eleni Fotinea, Stavroula-Evita Vogler, Christian Hanke, Thomas Glauert, John Bowden, Richard Braffort, Annelies Collet, Christophe Maragos, Petros Segouat, Jérémie. (2009). Sign Language Recognition, Generation, and Modelling: A Research Effort with Applications in Deaf Communication. 21-30. 10.1007/978-3-642-02707-9-3.

[7] K. Dabre and S. Dholay, "Machine learning model for sign language interpretation using webcam images," 2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA), Mumbai, 2014, pp. 317-321, doi: 10.1109/CSCITA.2014.6839279.

[8] Suharjito, H. Gunawan, N. Thiracitta and A. Nugroho, "Sign Language Recognition Using Modified Convolutional Neural Network Model," 2018 Indonesian Association for Pattern Recognition International Conference (INAPR), Jakarta, Indonesia, 2018, pp. 1-5, doi: 10.1109/INAPR.2018.8627014.

[9] G. A. Rao, K. Syamala, P. V. V. Kishore and A. S. C. S. Sastry, "Deep convolutional neural networks for sign language recognition," 2018 Conference on Signal Processing And Communication Engineering Systems (SPACES), Vijayawada, 2018, pp. 194-197, doi: 10.1109/SPACES.2018.8316344.

[10] A. S. Nikam and A. G. Ambekar, "Bilingual sign recognition using image based Hand gesture technique for hearing and speech impaired people," 2016 International Conference on Computing Communication Control and automation (ICCUBEA), Pune, 2016, pp. 1-6, doi: 10.1109/ICCUBEA.2016.7860057.

Reference Appendix A

1. Jie Huang, Wengang Zhou, Houqiang Li and Weiping Li, "Sign Language Recognition using 3D convolutional neural networks," *2015 IEEE International Conference on Multimedia and Expo (ICME)*, Turin, 2015, pp. 1-6, doi: 10.1109/ICME.2015.7177428.

   Available on :

   https://ieeexplore.ieee.org/document/7177428

2. A. S. Nikam and A. G. Ambekar, "Sign language recognition using image based hand gesture recognition techniques," *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, Coimbatore, 2016, pp. 1-5, doi: 10.1109/GET.2016.7916786.

   Available on :

   https://ieeexplore.ieee.org/document/7916786

3. M. E. Adnan, N. M. Dastagir, J. Jabin, A. M. Chowdhury and M. R. Islam, "A cost effective electronic braille for visually impaired individuals," *2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, Dhaka, 2017, pp. 175-178, doi: 10.1109/R10-HTC.2017.8288932.

   Available on:
   https://ieeexplore.ieee.org/document/8288932

4. P. Trigueiros, F. Ribeiro and L. P. Reis, "A comparison of machine learning algorithms applied to hand gesture recognition," *7th Iberian Conference on Information Systems and Technologies (CISTI 2012)*, Madrid, 2012, pp. 1-6.

   Available on:
   https://ieeexplore.ieee.org/document/6263058

5. Chouhan, Tushar & Panse, Ankit & Voona, Anvesh & M, Sameer. (2014). Smart Glove With Gesture Recognition Ability For The Hearing And Speech Impaired. 10.1109/GHTC-SAS.2014.6967567.

Available on:

https://www.researchgate.net/publication/281066609_Smart_Glove_With_Gesture_Recognition_Ability_For_The_Hearing_And_Speech_Impaired

6. K. Dabre and S. Dholay, "Machine learning model for sign language interpretation using webcam images," *2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA)*, Mumbai, 2014, pp. 317-321, doi: 10.1109/CSCITA.2014.6839279.

Available on:

https://ieeexplore.ieee.org/document/6839279?denied=

7. B. Liao, J. Li, Z. Ju and G. Ouyang, "Hand Gesture Recognition with Generalized Hough Transform and DC-CNN Using Realsense," 2018 Eighth International Conference on Information Science and Technology (ICIST), Cordoba, 2018, pp. 84-90, doi: 10.1109/ICIST.2018.8426125.

Available on:

https://ieeexplore.ieee.org/abstract/document/8426125

8. Efthimiou, Eleni & Fotinea, Stavroula-Evita & Vogler, Christian & Hanke, Thomas & Glauert, John & Bowden, Richard & Braffort, Annelies & Collet, Christophe & Maragos, Petros & Segouat, Jérémie. (2009). Sign Language Recognition, Generation, and Modelling: A Research Effort with Applications in Deaf Communication. 21-30. 10.1007/978-3-642-02707-9_3.

Available on:

https://www.researchgate.net/publication/221099416_Sign_Language_Recognition_Generation_and_Modelling_A_Research_Effort_with_Applications_in_Deaf_Communication

9. K. Dabre and S. Dholay, "Machine learning model for sign language interpretation using webcam images," 2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA), Mumbai, 2014, pp. 317-321, doi: 10.1109/CSCITA.2014.6839279.

Available on:

https://ieeexplore.ieee.org/document/6839279

10. H. Muthu Mariappan and V. Gomathi, "Real-Time Recognition of Indian Sign Language," 2019 International Conference on Computational Intelligence in Data Science (ICCIDS), Chennai, India, 2019, pp. 1-6, doi: 10.1109/ICCIDS.2019.8862125.

Available on:

https://ieeexplore.ieee.org/document/8862125

11. Suharjito, H. Gunawan, N. Thiracitta and A. Nugroho, "Sign Language Recognition Using Modified Convolutional Neural Network Model," *2018 Indonesian Association for Pattern Recognition International Conference (INAPR)*, Jakarta, Indonesia, 2018, pp. 1-5, doi: 10.1109/INAPR.2018.8627014.

Available on:

https://ieeexplore.ieee.org/document/8627014

12. G. A. Rao, K. Syamala, P. V. V. Kishore and A. S. C. S. Sastry, "Deep convolutional neural networks for sign language recognition," 2018 Conference on Signal Processing And Communication Engineering Systems (SPACES), Vijayawada, 2018, pp. 194-197, doi: 10.1109/SPACES.2018.8316344.

Available on:

https://ieeexplore.ieee.org/document/8316344

13. M. Xie and X. Ma, "End-to-End Residual Neural Network with Data Augmentation for Sign Language Recognition," 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, 2019, pp. 1629-1633, doi: 10.1109/IAEAC47372.2019.8998073.

Available on:

https://ieeexplore.ieee.org/document/8998073

14. K. Bantupalli and Y. Xie, "American Sign Language Recognition using Deep Learning and Computer Vision," 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018, pp. 4896-4899, doi: 10.1109/BigData.2018.8622141.

    Available on:

    https://ieeexplore.ieee.org/document/8622141

15. A. S. Nikam and A. G. Ambekar, "Bilingual sign recognition using image based Hand gesture technique for hearing and speech impaired people," 2016 International Conference on Computing Communication Control and automation (ICCUBEA), Pune, 2016, pp. 1-6, doi: 10.1109/ICCUBEA.2016.7860057.

    Available on:

    https://ieeexplore.ieee.org/document/7860057

16. D. Naglot and M. Kulkarni, "Real time sign language recognition using the leap motion controller," 2016 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, 2016, pp. 1-5, doi: 10.1109/INVENTIVE.2016.7830097.

    Available on:

    https://ieeexplore.ieee.org/document/7830097

17. S. He, "Research of a Sign Language Translation System Based on Deep Learning," 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM), Dublin, Ireland, 2019, pp. 392-396, doi: 10.1109/AIAM48774.2019.00083.

    Available on:

    https://ieeexplore.ieee.org/document/8950864

18. TensorFlow Documentation

    Available : https://www.tensorflow.org/api_docs/python/tf

19. Flask Documentation

Available: https://flask.palletsprojects.com/en/1.1.x/

20. Google Cloud API

Available: https://cloud.google.com/apis/design/documentation

21. OpenCV Documentation

Available: https://docs.opencv.org/

22. Keras Documentation

Available: https://keras.io/api/

23. Matplotlib Documentation

Available: https://matplotlib.org/3.3.2/contents.html

24. Kaggle Documentation

Available: https://www.kaggle.com/docs

25. Colab Documentation

Available: https://colab.research.google.com/notebooks/intro.ipynb#scrollTo=GJBs_flRovLc

26. Numpy Documentation

Available: https://numpy.org/doc/

27. How to Use The Pre-Trained VGG Model to Classify Objects in Photographs - Machine Learning Mastery

Available on:

https://machinelearningmastery.com/use-pre-trained-vgg-model-classify-objects-photographs/

28. Transfer Learning Using MobileNet and Keras

Available on:

https://towardsdatascience.com/transfer-learning-using-mobilenet-and-keras-c75daf7ff299

29. Keras Python Deep Learning API

Available: https://deeplizard.com/learn/video/OO4HD-1wRN8

30. Tensorflow and Image Augmentation

Available:
https://towardsdatascience.com/tensorflow-and-image-augmentation-3610c6c243a2

Codes Appendix B

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.layers import Dense, Activation,Conv2D, MaxPool2D, Flatten, Dropout, Batc
hNormalization,InputLayer
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.metrics import categorical_crossentropy
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.applications import imagenet_utils

from sklearn.model_selection import train_test_split

import cv2
```

```
train_dir = '../input/asl-alphabet/asl_alphabet_train/asl_alphabet_train'

test_dir = '../input/asl-alphabet/asl_alphabet_test/asl_alphabet_test'
```

```
labels_dict = {'A':0,'B':1,'C':2,'D':3,'E':4,'F':5,'G':6,'H':7,'I':8,'J':9,'K':10,'L':11,'M':1
2,
                   'N':13,'O':14,'P':15,'Q':16,'R':17,'S':18,'T':19,'U':20,'V':21,'W':22,'X':2
3,'Y':24,
                   'Z':25,'space':26,'del':27,'nothing':28}

def load_data():
    """
    Loads data and preprocess. Returns train and test data along with labels.
    """
    images = []
    labels = []
    size = 64,64
    print("LOADING DATA FROM : ",end = "")
    for folder in os.listdir(train_dir):
        print(folder, end = ' | ')
        i = 0
        for image in os.listdir(train_dir + "/" + folder):
            temp_img = cv2.imread(train_dir + '/' + folder + '/' + image)
            temp_img = cv2.resize(temp_img, size)
            images.append(temp_img)
            labels.append(labels_dict[folder])


    images = np.array(images)
    images = images.astype('float32')

    labels = keras.utils.to_categorical(labels)

    X_train, X_test, Y_train, Y_test = train_test_split(images, labels, test_size = 0.15)

    print()
    print('Loaded', len(X_train),'images for training,','Train data shape =',X_train.shape)
    print('Loaded', len(X_test),'images for testing','Test data shape =',X_test.shape)

    return X_train, X_test, Y_train, Y_test
    return images, labels
```

```
X_train, X_test, Y_train, Y_test = load_data()
```

```
LOADING DATA FROM : N | I | U | P | Y | Z | H | C | B | del | D | T | M | K | Q | F | G | no
thing | E | X | S | L | A | V | space | W | R | J | O |
Loaded 73950 images for training, Train data shape = (73950, 64, 64, 3)
Loaded 13050 images for testing Test data shape = (13050, 64, 64, 3)
```

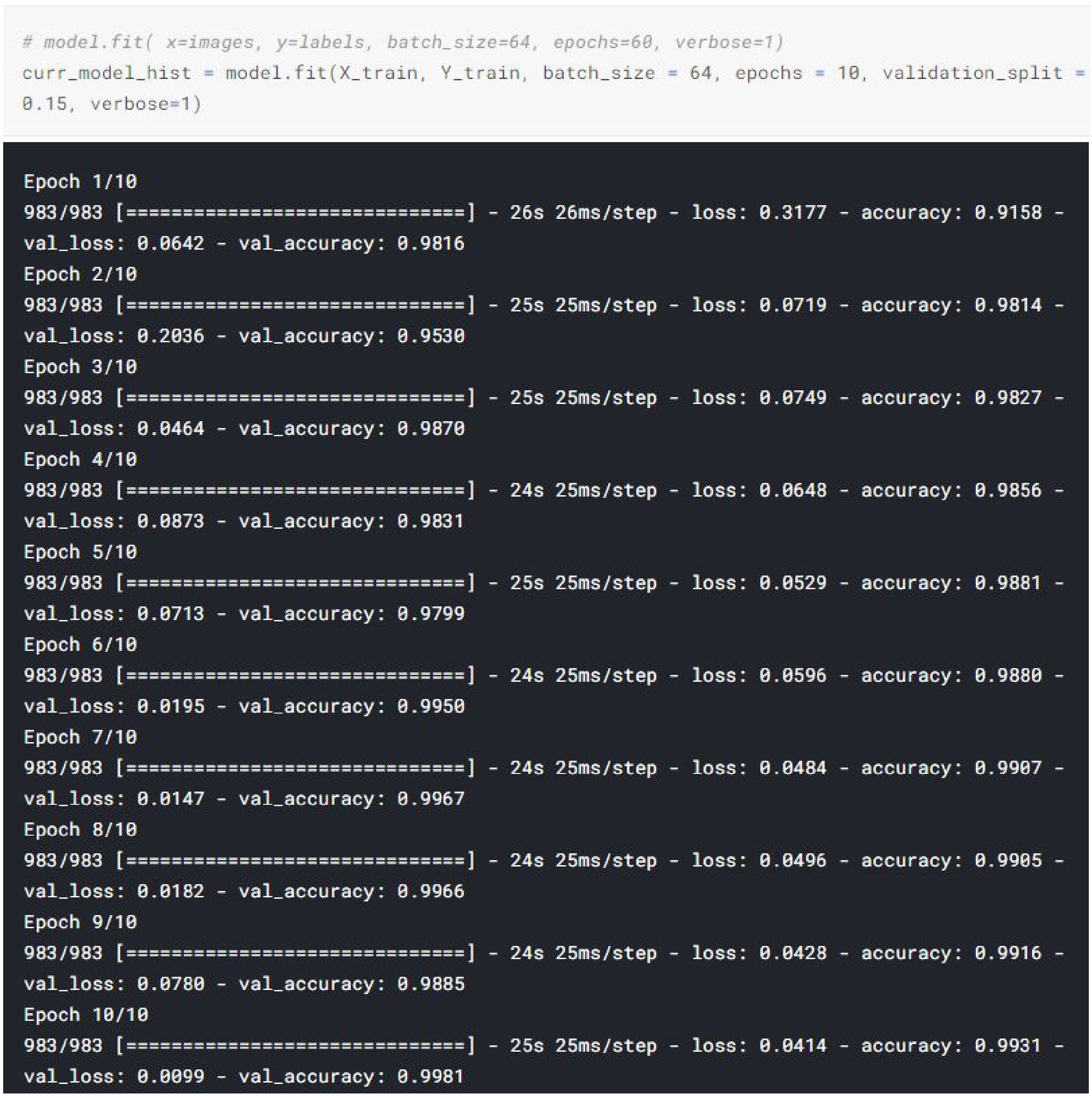
```
X_train = tf.keras.applications.mobilenet.preprocess_input(X_train)
```

```
mobilenet = tf.keras.applications.MobileNet(include_top=False, weights='imagenet')
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilene
t/mobilenet_1_0_224_tf_no_top.h5
17227776/17225924 [==============================] - 1s 0us/step
```

```
model = Sequential()
model.add(InputLayer(input_shape=(64,64,3)))
for layer in mobilenet.layers:
    model.add(layer)
```

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=["accuracy"])
```



```
# model.fit( x=images, y=labels, batch_size=64, epochs=60, verbose=1)
curr_model_hist = model.fit(X_train, Y_train, batch_size = 64, epochs = 10, validation_split =
0.15, verbose=1)
```

```
Epoch 1/10
983/983 [==============================] - 26s 26ms/step - loss: 0.3177 - accuracy: 0.9158 -
val_loss: 0.0642 - val_accuracy: 0.9816
Epoch 2/10
983/983 [==============================] - 25s 25ms/step - loss: 0.0719 - accuracy: 0.9814 -
val_loss: 0.2036 - val_accuracy: 0.9530
Epoch 3/10
983/983 [==============================] - 25s 25ms/step - loss: 0.0749 - accuracy: 0.9827 -
val_loss: 0.0464 - val_accuracy: 0.9870
Epoch 4/10
983/983 [==============================] - 24s 25ms/step - loss: 0.0648 - accuracy: 0.9856 -
val_loss: 0.0873 - val_accuracy: 0.9831
Epoch 5/10
983/983 [==============================] - 25s 25ms/step - loss: 0.0529 - accuracy: 0.9881 -
val_loss: 0.0713 - val_accuracy: 0.9799
Epoch 6/10
983/983 [==============================] - 24s 25ms/step - loss: 0.0596 - accuracy: 0.9880 -
val_loss: 0.0195 - val_accuracy: 0.9950
Epoch 7/10
983/983 [==============================] - 24s 25ms/step - loss: 0.0484 - accuracy: 0.9907 -
val_loss: 0.0147 - val_accuracy: 0.9967
Epoch 8/10
983/983 [==============================] - 24s 25ms/step - loss: 0.0496 - accuracy: 0.9905 -
val_loss: 0.0182 - val_accuracy: 0.9966
Epoch 9/10
983/983 [==============================] - 24s 25ms/step - loss: 0.0428 - accuracy: 0.9916 -
val_loss: 0.0780 - val_accuracy: 0.9885
Epoch 10/10
983/983 [==============================] - 25s 25ms/step - loss: 0.0414 - accuracy: 0.9931 -
val_loss: 0.0099 - val_accuracy: 0.9981
```

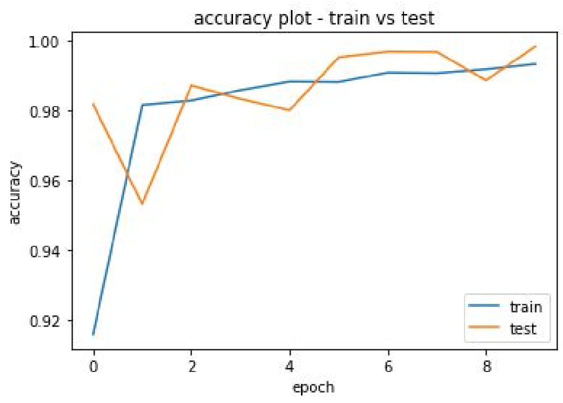
```
import matplotlib.pyplot as plt
```

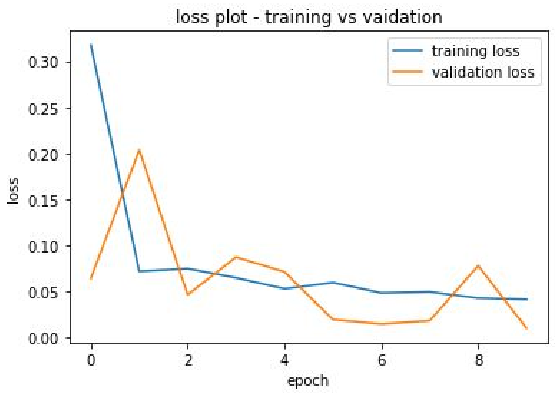
```
curr_model_hist
```

```
<tensorflow.python.keras.callbacks.History at 0x7f0f0c07c390>
```

```
plt.plot(curr_model_hist.history['accuracy'])
plt.plot(curr_model_hist.history['val_accuracy'])
plt.legend(['train', 'test'], loc='lower right')
plt.title('accuracy plot - train vs test')
plt.xlabel('epoch')
plt.ylabel('accuracy')
plt.show()

plt.plot(curr_model_hist.history['loss'])
plt.plot(curr_model_hist.history['val_loss'])
plt.legend(['training loss', 'validation loss'], loc = 'upper right')
plt.title('loss plot - training vs vaidation')
plt.xlabel('epoch')
plt.ylabel('loss')
plt.show()
```





```
X_test = tf.keras.applications.mobilenet.preprocess_input(X_test)
```

```
score = model.evaluate(x = X_test, y = Y_test, verbose = 0)
print('Accuracy for test images:', round(score[1]*100, 3), '%')
```

```
Accuracy for test images: 99.831 %
```

```
#model.save('asl3.h5')
```

Survey Questionnaires Appendix C

1) Do you need to use sign language in your daily life?
2) How hard do you find to use sign language to express what you want to say?
3) Do you think all people should know sign language?
4) Is being unable to express themselves properly makes the mute and deaf people frustrated? Does it abstinate them from sharing their full potential?
5) If there is a website for the deaf and mute people to use sign language to communicate, will it help them a lot?
6) Do you think the website should be free to use?