Sign Language Translator
(ML Application)

A Project Report

Submitted in Partial fulfilment of the

Requirements for the award of the Degree

of

BACHELOR OF SCIENCE (COMPUTER SCIENCE)

By

Deepak Jaygopal Gond

Seat No: ________

Under the esteemed guidance of

Prof. Javed Pathan

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE

RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

(Affiliated to University of Mumbai)

MUMBAI – 400053
MAHARASHTRA
2022-2023

RIZVI COLLEGE OF ARTS SCIENCE AND COMMERCE
(Affiliated to University of Mumbai)
MUMBAI MAHARASHTRA-400050

DEPARTMENT OF COMPUTER SCIENCE

<u>CERTIFICATE</u>

This certify that the project entitled," **Sign Language Translator" (ML Application), is** bonafide work of **Deepak Jaygopal Gond** Bearing **Seat No**: ___________ **Roll No: __________**submitted in partial fulfilment of the requirement for the award of degree of Bachelor of Science in Computer Science from University of Mumbai.

Project Guide HOD

External Examiner

Date …………. College Seal

ACKNOWLEDGEMENT

I owe special thanks to the Department of Computer Science of **Rizvi College of Arts, Science & Commerce** for giving me a chance to prepare this project dissertation I thank the Principal, **Professor Anjum Ara Ahmed** for her leadership and management.

I thank the Coordinator and Head of the Department **Professor Arif Patel** for providing us the required facilities and guidance throughout the course which culminated into this thesis.

Last and not the least to the project guide this semester, **Professor Javed Pathan**. Deep gratitude to the staff and the faculty of Rizvi College for their help and support.

And also, my beloved Parent and Classmate for their infinite Support and Love.

ABSTRACT

Sign Language is mainly used by deaf (hard hearing) and Dum people to exchange information between their own community and with other people. It is a language where people use their hand gestures to communicate as they can’t speak or hear.

Sign Language Recognition (SLR) deals with recognizing the hand gestures acquisition and continues till text or speech is generated for corresponding hand gestures. Here hand gestures for sign language can be classified as static and dynamic. However, static hand gesture recognition is simpler than dynamic hand gesture recognition, but both recognition is important to the human community. We can use Deep Learning Computer Vision to recognize the hand gestures by building Deep Neural Network architectures (Convolution Neural Network Architectures) where the model will learn to recognize the hand gestures images over an epoch. Once the model Successfully recognizes the gesture the corresponding English text is generated and then text can be converted to speech. This model will be more efficient and hence communicate for the deaf (hard hearing) and dump people will be easier. In this paper, we will discuss how Sign Language Recognition is done using Deep Learning.

DECLARATION

I hereby declare that the project entitled, **“Sign Language Translator (ML Application)”** done at **Rizvi College of Arts, Science and Commerce**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfilment of the requirements for the award of degree of **BACHELOR OF SCIENCE (COMPUTER SCIENCE)** to be submitted as final year; Fifth semester project as part of our curriculum.

Deepak Jaygopal Gond

Table Of Contents

CHAPTER 1. INTRODUCTION

1.1 Introduction to the system

American sign language is a predominant sign language Since the only disability D&M people have is communication related and they cannot use spoken languages hence the only way for them to communicate is through sign language. Communication is the process of exchange of thoughts and messages in various ways such as speech, signals, behaviour and visuals. Deaf and dumb(D&M) people make use of their hands to express different gestures to express their ideas with other people. Gestures are the non-verbally exchanged messages and these gestures are understood with vision. This nonverbal communication of deaf and dumb people is called sign language.

Sign language is a visual language and consists of 3 major components [6]:

Fingerspelling	Word Level sign vocabulary	Non-Manual Features
Used to spell Word Letter by Letter	Used for majority of communication	Facial expressions And tongue and mouth and body position

In our project we basically focus on producing a model which can recognise Fingerspelling based hand gestures in order to form a complete word by combining each gesture. The gestures we aim to train are as given in the image below.

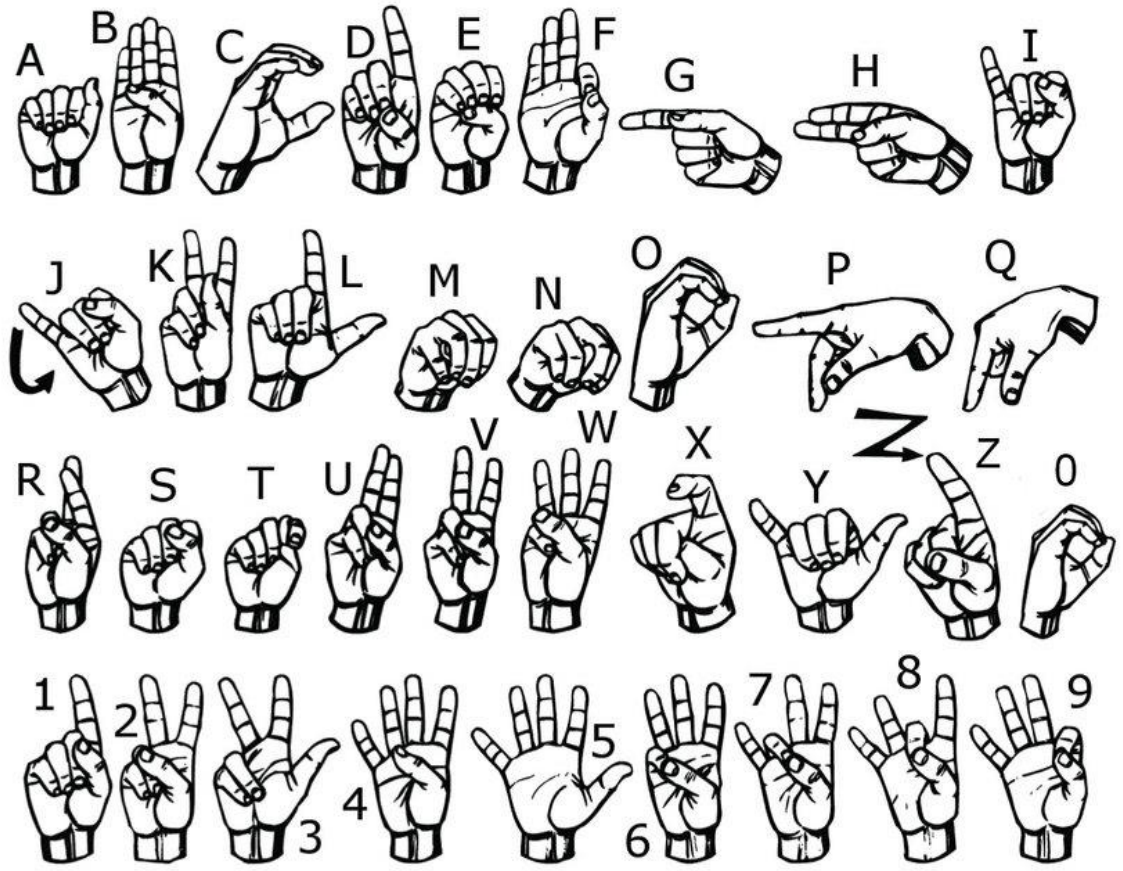


Fig 1.ASL Sign Standard

1.2 Problem Definition

Sign language uses lots of gestures so that it looks like movement language which consists of a series of hands and arms motions. For different countries, there are different sign languages and hand gestures. Also, it is noted that some unknown words are translated by simply showing gestures for each alphabet in the word. In addition, sign language also includes specific gestures to each alphabet in the English dictionary and for each number between 0 and 9. Based on these sign languages are made up of two groups, namely static gesture, and dynamic gesture. The static gesture is used for alphabet and number representation, whereas the dynamic gesture is used for specific concepts. Dynamic also includes words, sentences, etc. The static gesture consists of hand gestures, whereas the latter includes motion of hands, head, or both. Sign language is a visual language and consists of 3 major components, such as finger-spelling, word-level sign vocabulary, and non-manual features. Finger-spelling is used to spell words letter by letter and convey the message whereas the latter is keyword-based. But the design of a sign language translator is quite challenging despite many research efforts during the last few decades. Also, even the same signs have significantly different appearances for different signers and different viewpoints. This work focuses on the creation of a static sign language translator by using a Convolutional Neural Network. We created a lightweight network that can be used with embedded devices/standalone applications/web applications having fewer resources.

1.3 Aim

For interaction between normal people and D&M people a language barrier is created as sign language structure which is different from normal text. So, they depend on vision-based communication for interaction.

If there is a common interface that converts the sign language to text the gestures can be easily understood by the other people. So, research has been made for a vision-based interface system where D&M people can enjoy communication without really knowing each other's language.

The aim is to develop a user-friendly human computer interface (HCI) where the computer understands the human sign language. There are various sign languages all over the world, namely American Sign Language (ASL), French Sign Language, British Sign Language (BSL), Indian Sign language, Japanese Sign Language and work has been done on other languages all around the world.

1.4 Objective

The main objectives of this project are to contribute to the field of automatic sign language recognition and translation to text or speech. In our project, we focus on static sign language hand gestures. This work focused on recognizing the hand gestures which includes 26 English alphabets (A-Z) and 10 digits (0-9) using Deep Neural Networks (DNN). We created a convolution neural networks classifier that can classify the hand gestures into English alphabets and digits. We have trained the neural network under different configurations and architectures like LeNet-5 [2], MobileNetV2 [3], and our own architecture. We used the horizontal voting ensemble technique to achieve the maximum accuracy of the model. We have also created an application using Python to test our results from a live camera.

1.5 Goal

The goal of this project was to build a neural network able to classify which letter of the American Sign Language (ASL) alphabet is being signed, given an image of a signing hand. This project is a first step towards building a possible sign language translator, which can take communications in sign language and translate them into written and oral language. Such a translator would greatly lower the barrier for many deaf and mute individuals to be able to better communicate with others in day-to-day interactions. This goal is further motivated by the isolation that is felt within the deaf community. Loneliness and depression exist in higher rates among the deaf population, especially when they are immersed in a hearing world [1]. Large barriers that profoundly affect life quality stem from the communication disconnect between the deaf and the hearing. Some examples are information deprivation, limitation of social connections, and difficulty integrating in society [2]. Most research implementations for this task have used depth maps generated by depth camera and high-resolution images. The objective of this project was to see if neural networks are able to classify signed ASL letters using simple images of hands taken with a personal device such as a laptop webcam. This is in alignment with the motivation as this would make a future implementation of a real time ASL-to-oral/written language translator practical in an everyday situation.

1.6 Need of System

Speech impaired people use hand signs and gestures to communicate. Normal people face difficulty in understanding their language. Hence there is a need of a system which recognizes the different signs, gestures and conveys the information to the normal people. It bridges the gap between physically challenged people and normal people.

CHAPTER 2. REQUIREMENT SPECIFICATION

2.1 Introduction

Deaf (hard hearing) and dumb people use Sign Language (SL) [1] as their primary means to express their ideas and thoughts with their own community and with other people with hand and body gestures. It has its own vocabulary, meaning, and syntax which is different from the spoken language or written language. Spoken language is a language produced by articulate sounds mapped against specific words and grammatical combinations to convey meaningful messages. Sign language uses visual hand and body gestures to convey meaningful messages. There are somewhere between 138 and 300 different types of Sign Language used around globally today. In India, there are only about 250 certified sign language interpreters for a deaf population of around 7 million. This would be a problem to teach sign language to the deaf and dumb people as there is a limited number of sign language interpreters exits today.

2.2 System Environment

1. Use of sensory devices

It uses electromechanical devices to provide exact hand configuration, and position. Different glove-based approaches can be used to extract information. But it is expensive and not user friendly.

2. Vision based approach

In vision-based methods computer camera is the input device for observing the information of hands or fingers. The Vision Based methods require only a camera, thus realizing a natural interaction between humans and computers without the use of any extra devices. These systems tend to complement biological vision by describing artificial vision systems that are implemented in software and/or hardware. The main challenge of vision-based hand detection is to cope with the large variability of human hand's appearance due to a huge number of hand movements, to different skin-colour possibilities as well as to the variations in viewpoints, scales, and speed of the camera capturing the scene.

2.3 Software Requirement

Operating System: Windows, Mac, Linux
SDK:

OpenCV:
OpenCV (Open-Source Computer Vision) is an open-source library of programming functions used for real-time computer-vision. It is mainly used for image processing, video capture and

analysis for features like face and object recognition. It is written in C++ which is its primary interface, however bindings are available for Python, Java, MATLAB/OCTAVE.

TensorFlow:
TensorFlow is an open-source software library for numerical computation. First, we define the nodes of the computation graph, then inside a session, the 13 actual computation takes place. TensorFlow is widely used in Machine Learning.

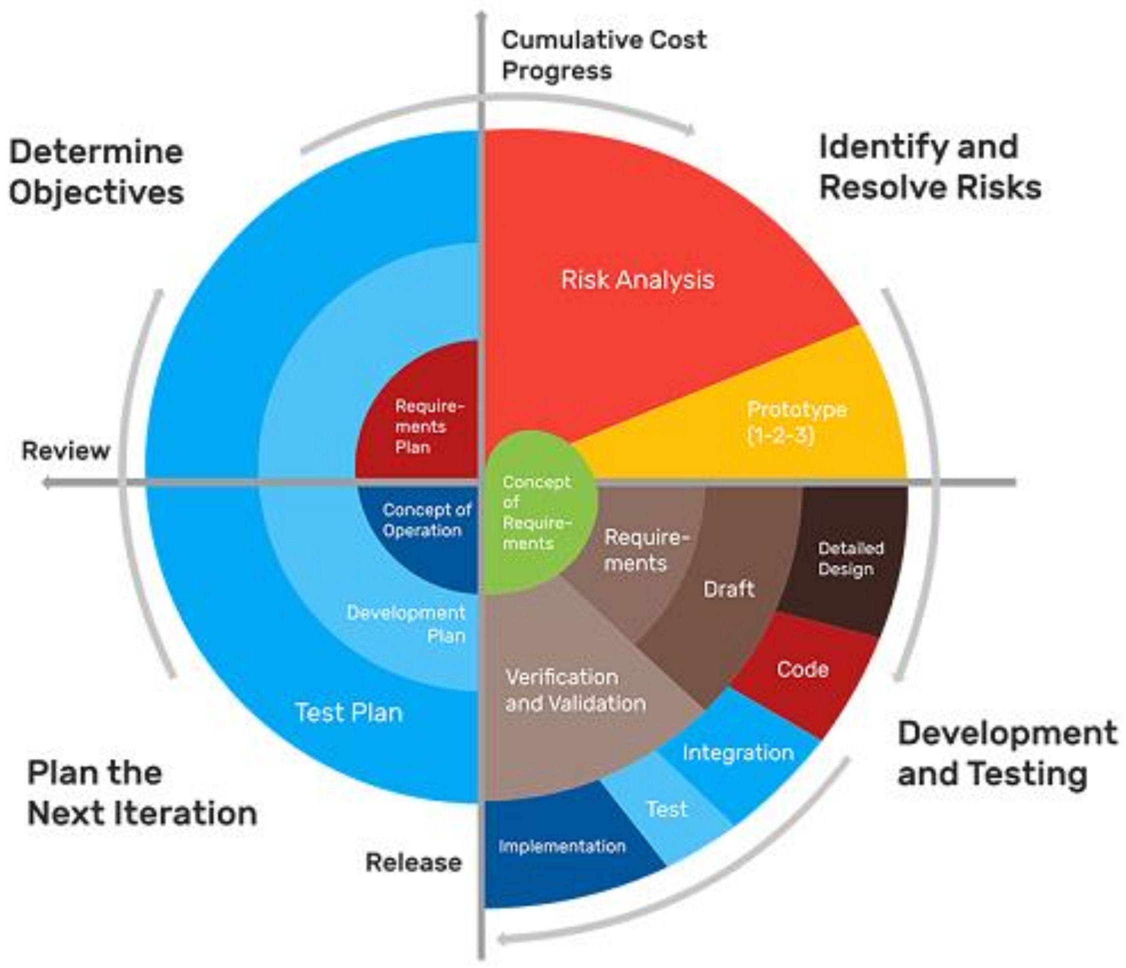
Keras:
Keras is a high-level neural networks library written in python that works as a wrapper to TensorFlow. It is used in cases where we want to quickly build and test the neural network with minimal lines of code. It contains implementations of commonly used neural network elements like layers, objective, activation functions, optimizers, and tools to make working with images and text data easier.

2.4 Hardware Requirement

The Hardware Interfaces Required are:
Camera: Good quality,3MP
Ram: Minimum 8GB or higher
GPU: 4GB dedicated
Processor: Intel Pentium 4 or higher
HDD: 10GB or higher
Monitor: 15" or 17" colour monitor
Mouse: Scroll or Optical Mouse or Touch Pad
Keyboard: Standard 110 keys keyboard

2.5 Methodology

In order to achieve goals and planned results within a defined schedule and a budget, a methodology in used in a project. Regardless of which field or which trade, there are assortments of methodologies to help managers at every stage of a project from the initiation to implementation to the closure. A methodology is a model, which project managers employ for the design, planning, implementation and achievement of their project objectives. There are different project management methodologies to benefit different projects.



Spiral Model Methodology and its Phases

The spiral model, initially proposed by Boehm, is an evolutionary software process model that couples the iterative feature of prototyping with the controlled and systematic aspects of the linear sequential model. It implements the potential for rapid development of new versions of the software. Using the spiral model, the software is developed in a series of incremental releases. During the early iterations, the additional release may be a paper model or prototype. During later iterations, more and more complete versions of the engineered system are produced.

Each cycle in the spiral is divided into four parts:

1. Objective setting: Each cycle in the spiral starts with the identification of purpose for that cycle, the various alternatives that are possible for achieving the targets, and the constraints that exists.

2. Risk Assessment and reduction: The next phase in the cycle is to calculate these various alternatives based on the goals and constraints. The focus of evaluation in this stage is located on the risk perception for the project.

3. Development and validation: The next phase is to develop strategies that resolve uncertainties and risks. This process may include activities such as benchmarking, simulation, and prototyping.

4. Planning: Finally, the next step is planned. The project is reviewed, and a choice made whether to continue with a further period of the spiral. If it is determined to keep, plans are drawn up for the next step of the project. The development phase depends on the remaining risks. For example, if performance or user-interface risks are treated more essential than the program development risks, the next phase may be an evolutionary development that includes developing a more detailed prototype for solving the risks. The risk-driven feature of the spiral model allows it to accommodate any mixture of a specification-oriented, prototype-oriented, simulation-oriented, or another type of approach. An essential element of the model is that each period of the spiral is completed by a review that includes all the products developed during that cycle, including plans for the next cycle. The spiral model works for development as well as enhancement projects. task performed by every user once the software has been delivered to the customer, installed, and operational.

The steps for Spiral Model can be generalized as follows:

- The new system requirements are defined in as much details as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.

- A preliminary design is created for the new system.

- A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product

- A second prototype is evolved by a fourfold procedure:

1. Evaluating the first prototype in terms of its strengths, weakness, and risks.
2. Defining the requirements of the second prototype.
3. Planning a designing the second prototype.
4. Constructing and testing the second prototype.

- At the customer option, the entire project can be aborted if the risk is deemed too great. Risk factors might involve development cost overruns, operating-cost miscalculation, or any other factor that could, in the customer’s judgment, result in a less-than-satisfactory final product.

- The existing prototype is evaluated in the same manner as was the previous prototype, and if necessary, another prototype is developed from it according to the fourfold procedure outlined above.

- The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.

- The final system is constructed, based on the refined prototype.

- The final system is thoroughly evaluated and tested. Routine maintenance is carried on a continuing basis to prevent large scale failures and to minimize down time.

CHAPTER 3. SYSTEM ANALYSIS

3.1 Introduction

In System Analysis we have gone through other similar works that are implemented in the domain of sign language recognition. The summaries of each of the project works are mentioned below

3.2 System Analysis

3.2.1 Analysis of Existing System

A Survey of Hand Gesture Recognition Methods in Sign Language Recognition

Sign Language Recognition (SLR) system, which is required to recognize sign languages, has been widely studied for years. The studies are based on various input sensors, gesture segmentation, extraction of features and classification methods. This paper aims to analyze and compare the methods employed in the SLR systems, classifications methods that have been used, and suggests the most promising method for future research. Due to recent advancement in classification methods, many of the recent proposed works mainly contribute on the classification methods, such as hybrid method and Deep Learning. This paper focuses on the classification methods used in prior Sign Language Recognition system. Based on our review, HMMbased approaches have been explored extensively in prior research, including its modifications. This study is based on various input sensors, gesture segmentation, extraction of features and classification methods. This paper aims to analyze and compare the methods employed in the SLR systems, classifications methods that have been used, and suggests the most reliable method for future research. Due to recent advancement in classification methods, many of the recently proposed works mainly contribute to the classification methods, such as hybrid method and Deep Learning. Based on our review, HMM-based approaches have been explored extensively in prior research, including its modifications. Hybrid CNN-HMM and fully Deep Learning approaches have shown promising results and offer opportunities for further exploration.

A System for Recognition of Indian Sign Language for Deaf People using Otsu’s Algorithm

In this paper we proposed some methods, through which the recognition of the signs becomes easy for peoples while communication. And the result of those symbols signs will be converted into the text. In this project, we are capturing hand gestures through webcam and convert this image into gray scale image. The segmentation of gray scale image of a hand gesture is performed using Otsu thresholding algorithm. Total image level is divided into two classes one is hand and other is background. The optimal threshold value is determined by computing the ratio between class variance and total class variance. To find the boundary of hand gesture in image Canny edge detection technique is used. In Canny edge detection we used edge-based segmentation and threshold-based segmentation. Then Otsu’s algorithm is used because of its simple calculation and stability. This algorithm fails, when the global distribution of the target and background vary widely.

Sign Language Interpreter using Image Processing and Machine Learning

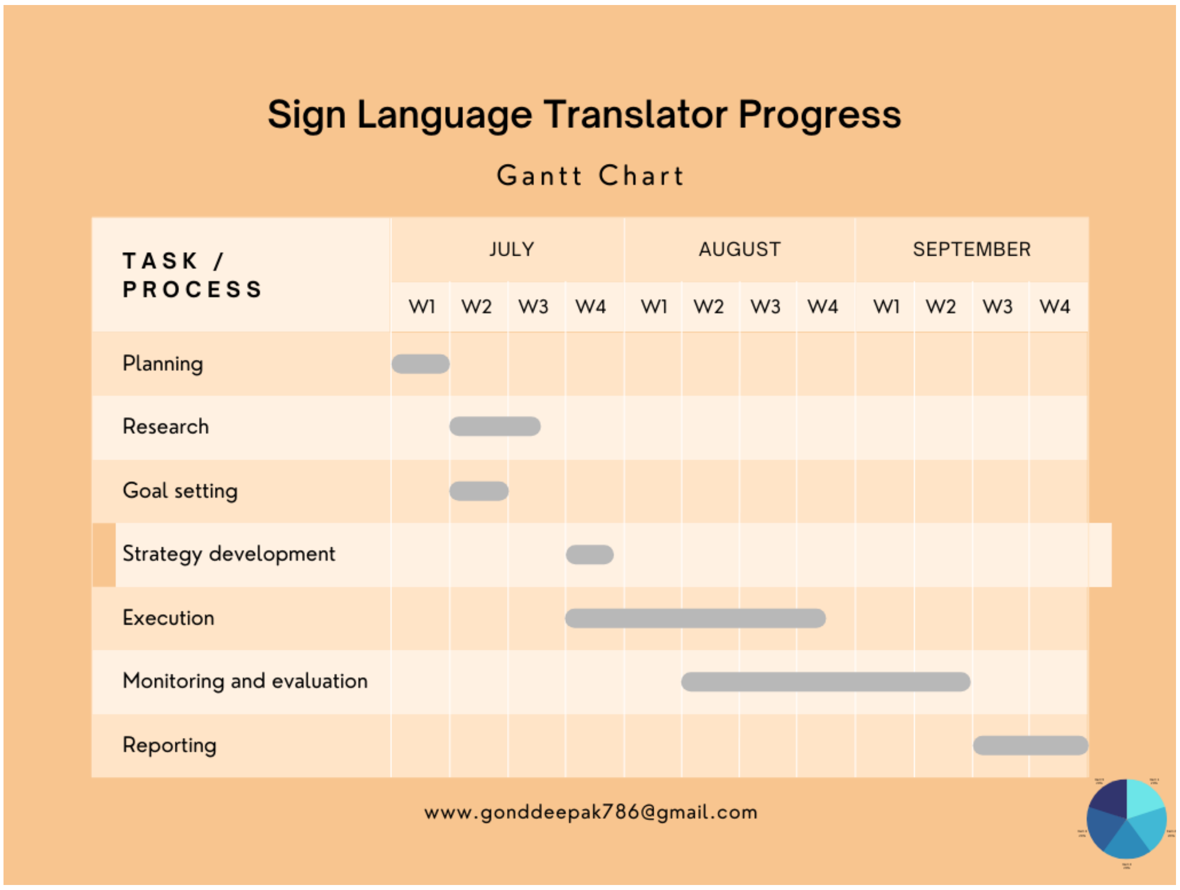
Speech impairment is a disability which affects one’s ability to speak and hear. Such individuals use sign language to communicate with other people. Although it is an effective form of communication, there remains a challenge for people who do not understand sign language to communicate with speech impaired people. The aim of this paper is to develop an application which will translate sign language to English in the form of text and audio, thus aiding communication with sign language. The application acquires image data using the webcam of the computer, then it is pre-processed using a combinational algorithm and recognition is done using template matching. The translation in the form of text is then converted to audio. The database used for this system includes 6000 images of English alphabets. We used 4800 images for training and 1200 images for testing. The system produces 88% accuracy.

3.2.2 Analysis of Purposed System

Our proposed system is sign language recognition system using convolution neural networks which recognizes various hand gestures by capturing video and converting it into frames. Then the hand pixels are segmented and the image it obtained and sent for comparison to the trained model. Thus, our system is more robust in getting exact text labels of letters.

3.3 Gantt Chart

A Gantt chart is a type of bar chart that illustrates a project schedule. This chart lists the tasks to be performed on the vertical axis, and time intervals on the horizontal axis. The width of the horizontal bars in the graph shows the duration of each activity. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements constitute the work breakdown structure of the project. Modern Gantt charts also show the dependency relationships between activities. Gantt charts can be used to show current schedule status using percent-complete shadings and a vertical "TODAY" line as shown here. Gantt charts are sometimes equated with bar charts. Gantt charts are usually created initially using an early start time approach, where each task is scheduled to start immediately when its prerequisites are complete. This method maximizes the float time available for all tasks.:



Sign Language Translator Progress

Gantt Chart

TASK / PROCESS	JULY				AUGUST				SEPTEMBER			
	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4
Planning	■											
Research		■	■									
Goal setting		■										
Strategy development				■								
Execution				■	■	■	■	■				
Monitoring and evaluation						■	■	■	■	■		
Reporting											■	■

www.gonddeepak786@gmail.com

CHAPTER 4. SURVEY OF TECHNOLOGY

TensorFlow: TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. Features: TensorFlow provides stable Python (for version 3.7 across all platforms) and C APIs; and without API backwards compatibility guarantee: C++, Go, Java, JavaScript and Swift (early release). Third-party packages are available for C#, Haskell Julia, MATLAB, R, Scala, Rust, OCaml, and Crystal. “New language support should be built on top of the C API. However, not all functionality is available in C yet." Some more functionality is provided by the Python API. Application: Among the applications for which TensorFlow is the foundation, are automated image-captioning software, such as Deep Dream.

OpenCV: OpenCV (Open-Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision.[1] Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel [2]). The library is cross-platform and free for use under the open-source BSD license.
OpenCV's application areas include:

- 2D and 3D feature toolkits
- Ego motion estimation
- Facial recognition system
- Gesture recognition
- Human–computer interaction (HCI)
- Mobile robotics
- Motion understanding
- Object identification
- Segmentation and recognition Stereopsis stereo vision: depth perception from 2 cameras
- Structure from motion (SFM).
- Motion tracking
- Augmented reality

Keras: Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet also is the author of the XCeption deep neural network model. Features: Keras contains numerous implementations of commonly used neural network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel.

Pre-trained models

Trained model consists of two parts model Architecture and model Weights. Model weights are large file so we have to download and extract the feature from ImageNet database. Some of the popular pre-trained models are listed below,

- ResNet
- VGG16
- Mobile Net
- InceptionResNetV2
- InceptionV3

Convolutional Neural Network:

Unlike regular Neural Networks, in the layers of CNN, the neurons are arranged in 3 dimensions: width, height, depth. The neurons in a layer will only be connected to a small region of the layer (window size) before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would have dimensions (number of classes), because by the end of the CNN architecture we will reduce the full image into a single vector of class scores

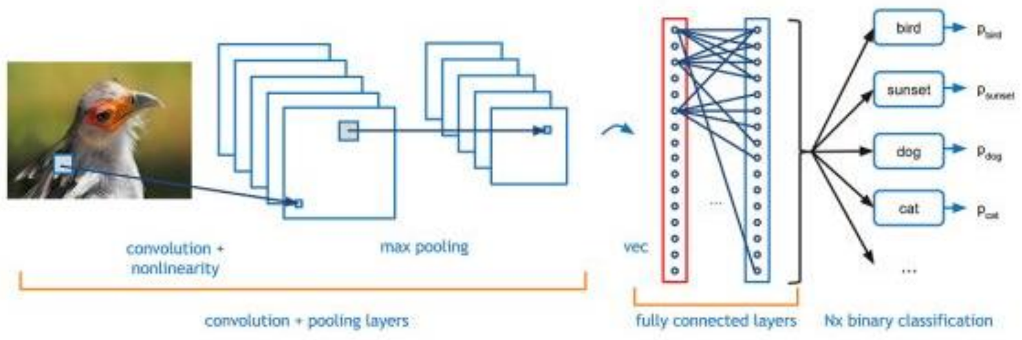


Figure 5.2: Convolution neural networks

1.Convolution Layer: In convolution layer we take a small window size [typically of length 5*5] that extends to the depth of the input matrix. The layer consists of learnable filters of window size. During every iteration we slid the window by stride size [typically 1], and compute the dot product of filter entries and input values at a given position. As we continue this process well create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position. That is, the network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some color.

2. Pooling Layer: We use pooling layer to decrease the size of activation matrix and ultimately reduce the learnable parameters. There are two types of pooling:

a) **Max Pooling:** In max pooling we take a window size [for example window of size 2*2], and only take the maximum of 4 values. Well lid this window and continue this process, so well finally get a activation matrix half of its
original Size.

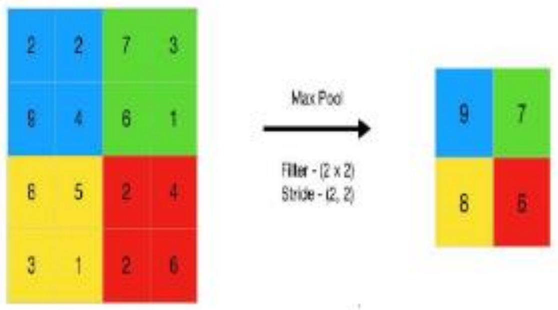


Fig. 7. Max Pooling

b) **Average Pooling:** In average pooling we take average of all values in a window.

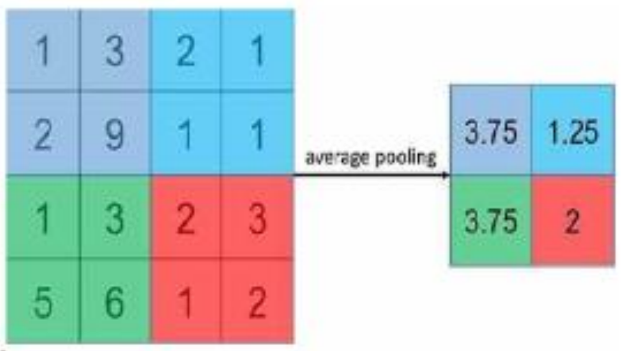


Fig. 8. Average Pooling

3.Flatten: The obtained resultant matrix will be in muti-dimension. Flattening is converting the data into a 1-dimensional array for inputting the layer to the next layer. We flatten the convolution layers to create a single feature vector.

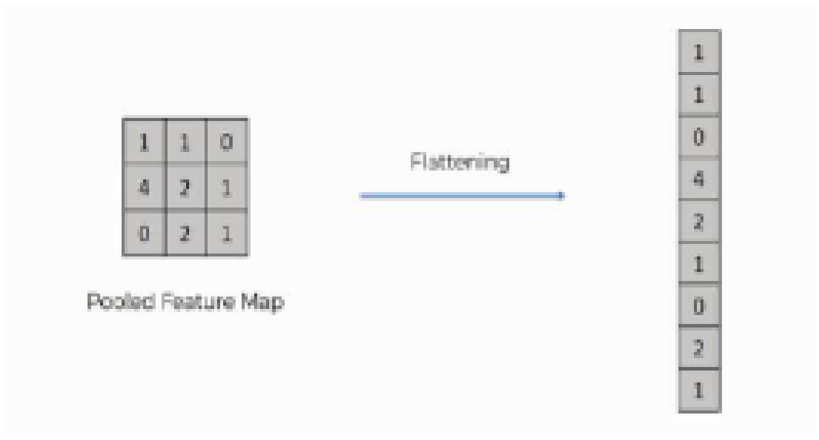


Fig. 9. Flatten

4. Fully Connected Layer: In convolution layer neurons are connected only to a local region, while in a fully connected region, well connect the all the inputs to neurons.

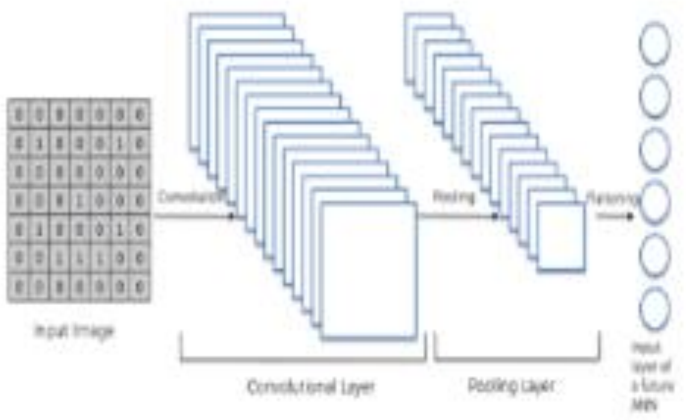


Fig. 10. Full Connection

5. Final Output Layer: After getting values from fully connected layer, well connect them to final layer of neurons [having count equal to total number of classes], that will predict the probability of each image to be in different classes

CHAPTER 5. SYSTEM DESIGN

5.1 Introduction

This Module Describe the Basic Design of the Sign Language Translator.

5.2 System Architecture

The system is a vision-based approach. All the signs are represented with bare hands and so it eliminates the problem of using any artificial devices for interaction.

Data Set Generation For the project we tried to find already made datasets but we couldn't find dataset in the form of raw images that matched our requirements. All we could find were the datasets in the form of RGB values. Hence, we decided to create our own data set. Steps we followed to create our data set are as follows.

We used Open computer vision (OpenCV) library in order to produce our dataset. Firstly, we captured around 2500 images of each of the symbol in ASL for training purposes and around 1000 images per symbol for testing purpose. First, we capture each frame shown by the webcam of our machine. In each frame we define a region of interest (ROI) which is denoted by a blue bounded square as shown in the image below.

The dataset consists of 35 different hand sign gestures which include A-Z alphabet gestures, 1-9 number gestures, and also a gesture for space which means how the deaf (hard hearing) and dumb people represent space between two letters or two words while communicating. Each gesture has 2500 images which are 64x64 pixels, so altogether there are 35 gestures which means there 87,500 images for all gestures. Convolutional Neural Network (CNN) is well suited for this dataset for model training purposes and gesture prediction.

Data Pre-processing An image is nothing more than a 2-dimensional array of numbers or pixels which are ranging from 0 to 255.Typically, 0 means black, and 255 means white. Image is defined by mathematical function f (x, y) where 'x' represents horizontal and 'y' represents vertical in a coordinate plane. The value of f (x, y) at any point is giving the pixel value at that point of an image.

Image Pre-processing is the use of algorithms to perform operations on images. It is important to Pre-process the images before sending the images for model training. For example, all the images should have the same size of 200x200 pixels. If not, the model cannot be trained.

The steps we have taken for image Pre-processing are:

- Read Images.
- Resize or reshape all the images to the same

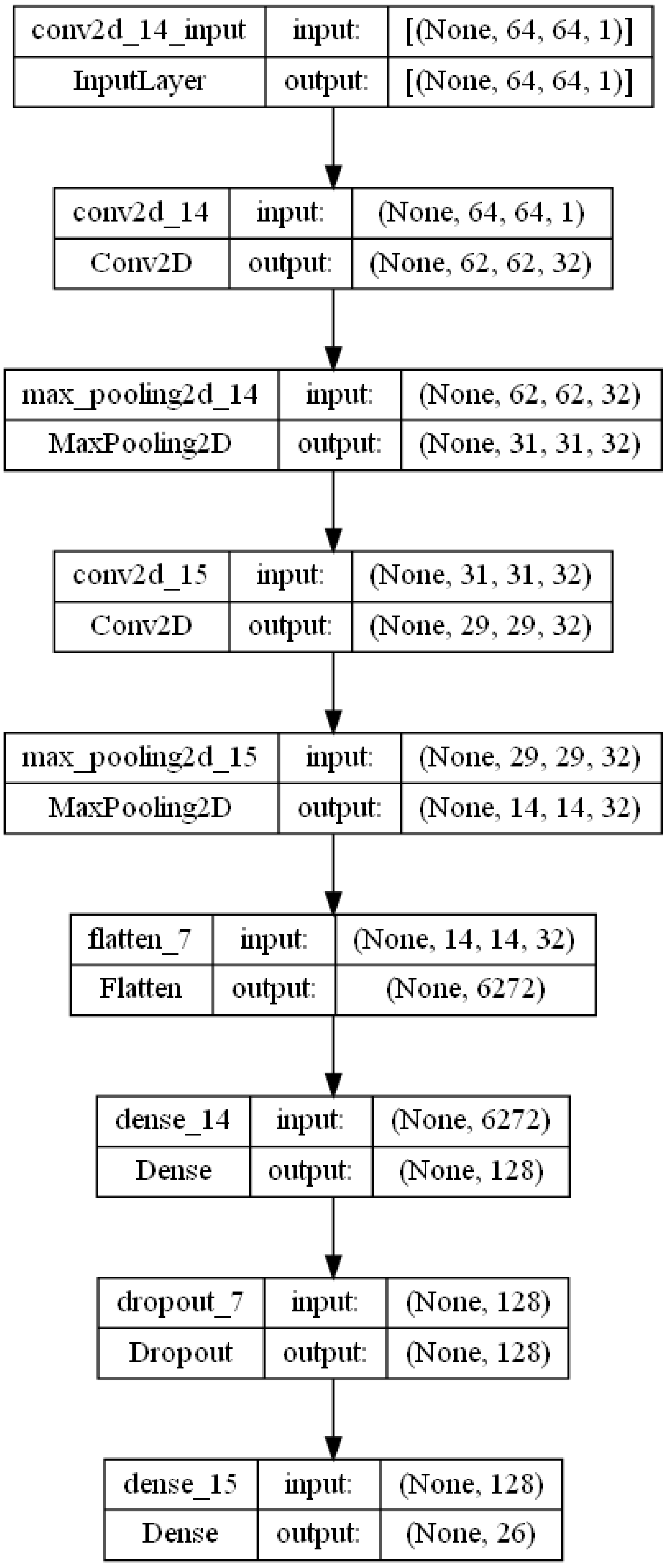
- Remove noise.
- All the image pixels arrays are converted to 0 to 255 by dividing the image array by 255.

Training module: Supervised machine learning: It is one of the ways of machine learning where the model is trained by input data and expected output data. To create such model, it is necessary to go through the following phases:
1. model construction
2. model training
3. model testing
4. model evaluation

Model construction: It depends on machine learning algorithms. In this projects case, it was neural networks. Such an algorithm looks like: 1. Begin with its object: model = Sequential () 2. Then consist of layers with their types: model. Add (type_of_layer ()) 3. After adding a sufficient number of layers, the model is compiled. At this moment Keras communicates with TensorFlow for construction of the model. During model compilation it is important to write a loss function and an optimizer algorithm. It looks like: model. Compile (loss='name_of_loss_function',optimizer= 'name_of_opimazer_alg') The loss function shows the accuracy of each prediction made by the model. Before model training it is important to scale data for their further use.

Model training: After model construction it is time for model training. In this phase, the model is trained using training data and expected output for this data. It's look this way: model.fit(training_data, expected_output). Progress is visible on the console when the script runs. At the end it will report the final accuracy of the model.

Model Testing: During this phase a second set of data is loaded. This data set has never been seen by the model and therefore it's true accuracy will be verified. After the model training is complete, and it is understood that the model shows the right result, it can 35 be saved by: model. Save("name_of_file.h5"). Finally, the saved model can be used in the real world. The name of this phase is model evaluation. This means that the model can be used to evaluate new data.


conv2d_14_input | input: | [(None, 64, 64, 1)]
InputLayer | output: | [(None, 64, 64, 1)]
conv2d_14 | input: | (None, 64, 64, 1)
Conv2D | output: | (None, 62, 62, 32)
max_pooling2d_14 | input: | (None, 62, 62, 32)
MaxPooling2D | output: | (None, 31, 31, 32)
conv2d_15 | input: | (None, 31, 31, 32)
Conv2D | output: | (None, 29, 29, 32)
max_pooling2d_15 | input: | (None, 29, 29, 32)
MaxPooling2D | output: | (None, 14, 14, 32)
flatten_7 | input: | (None, 14, 14, 32)
Flatten | output: | (None, 6272)
dense_14 | input: | (None, 6272)
Dense | output: | (None, 128)
dropout_7 | input: | (None, 128)
Dropout | output: | (None, 128)
dense_15 | input: | (None, 128)
Dense | output: | (None, 26)

GESTURE CLASSIFICATION

The approach which we used for this project is: Our approach uses two layers of algorithm to predict the final symbol of the user.

Algorithm Layer 1:

1. Apply gaussian blur filter and threshold to the frame taken with OpenCV to get the processed image after feature extraction.

2. This processed image is passed to the CNN model for prediction and if a letter is detected for more than 50 frames then the letter is printed and taken into consideration for forming the word.

3. Space between the words are considered using the blank symbol.

Algorithm Layer 2:

1. We detect various sets of symbols which show similar results on getting detected.

2. We then classify between those sets using classifiers made for those sets only.

Layer 1:

CNN Model:

1. 1st Convolution Layer: The input picture has resolution of 64x64 pixels. It is first processed in the first convolutional layer using 32 filter weights (3x3 pixels each). This will result in a 62X62 pixel image, one for each Filter-weights.

2. 1st Pooling Layer: The pictures are down sampled using max pooling of 2x2 i.e., we keep the highest value in the 2x2 square of array. Therefore, our picture is down sampled to 31 x 31 pixels.

3. 2nd Convolution Layer: Now, these 31 x 31 from the output of the first pooling layer is served as an input to the second convolutional layer. It is processed in the second convolutional layer using 32 filter weights (3x3 pixels each). This will result in a 29 x 29-pixel image.

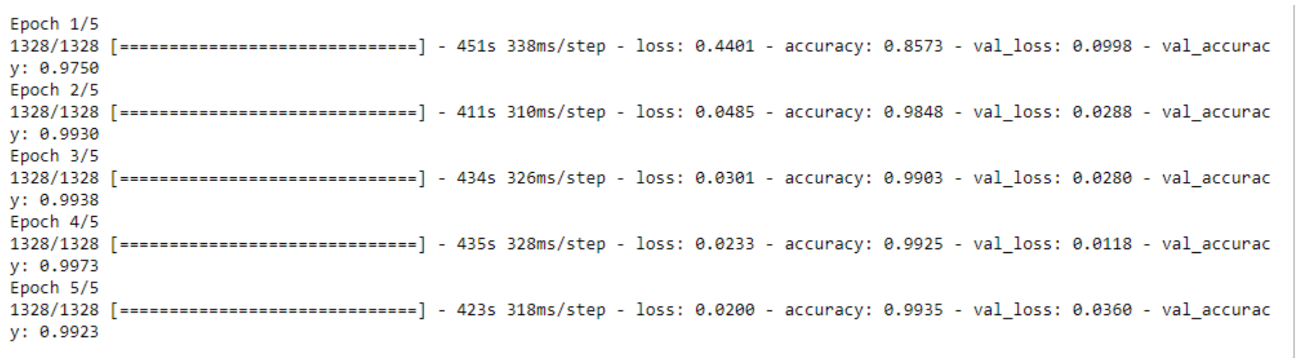
4. 2nd Pooling Layer: The resulting images are down sampled again using max pool of 2x2 and is reduced to 14 x 14 resolution of images.

5. 1st Densely Connected Layer: Now these images are used as an input to a fully connected layer with 128 neurons and the output from the second convolutional layer is reshaped to an array of 14x14x32 =6272 values. The input to this layer is an array of 6272 values. The output of these layer is fed to the 2nd Densely Connected Layer. We are using a dropout layer of value 0.2 to avoid overfitting.

6. Final layer: The output of the 2nd Densely Connected Layer serves as an input for the final layer which will have the number of neurons as the number of classes we are classifying (alphabets).

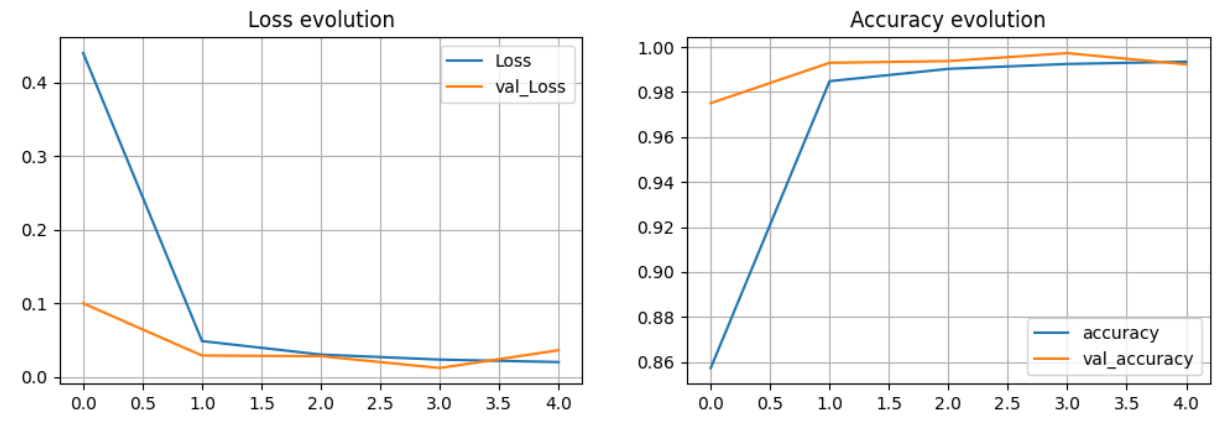
Layer 2:

We are using two layers of algorithms to verify and predict symbols which are more similar to each other so that we can get us close as we can get to detect the symbol shown. In our testing we found that following symbols were not showing properly and were giving other symbols also:
1. For K: B, F, V, K and M
2. For U: B, V, K, D and I
3. For V: F, K, and B
So, to handle above cases we made three different classifiers for classifying these sets:
1. {B, F, V, K, M}
2. {B, V, K, D, I}
3. {F, K, B}



```
Epoch 1/5
1328/1328 [==============================] - 451s 338ms/step - loss: 0.4401 - accuracy: 0.8573 - val_loss: 0.0998 - val_accurac
y: 0.9750
Epoch 2/5
1328/1328 [==============================] - 411s 310ms/step - loss: 0.0485 - accuracy: 0.9848 - val_loss: 0.0288 - val_accurac
y: 0.9930
Epoch 3/5
1328/1328 [==============================] - 434s 326ms/step - loss: 0.0301 - accuracy: 0.9903 - val_loss: 0.0280 - val_accurac
y: 0.9938
Epoch 4/5
1328/1328 [==============================] - 435s 328ms/step - loss: 0.0233 - accuracy: 0.9925 - val_loss: 0.0118 - val_accurac
y: 0.9973
Epoch 5/5
1328/1328 [==============================] - 423s 318ms/step - loss: 0.0200 - accuracy: 0.9935 - val_loss: 0.0360 - val_accurac
y: 0.9923
```

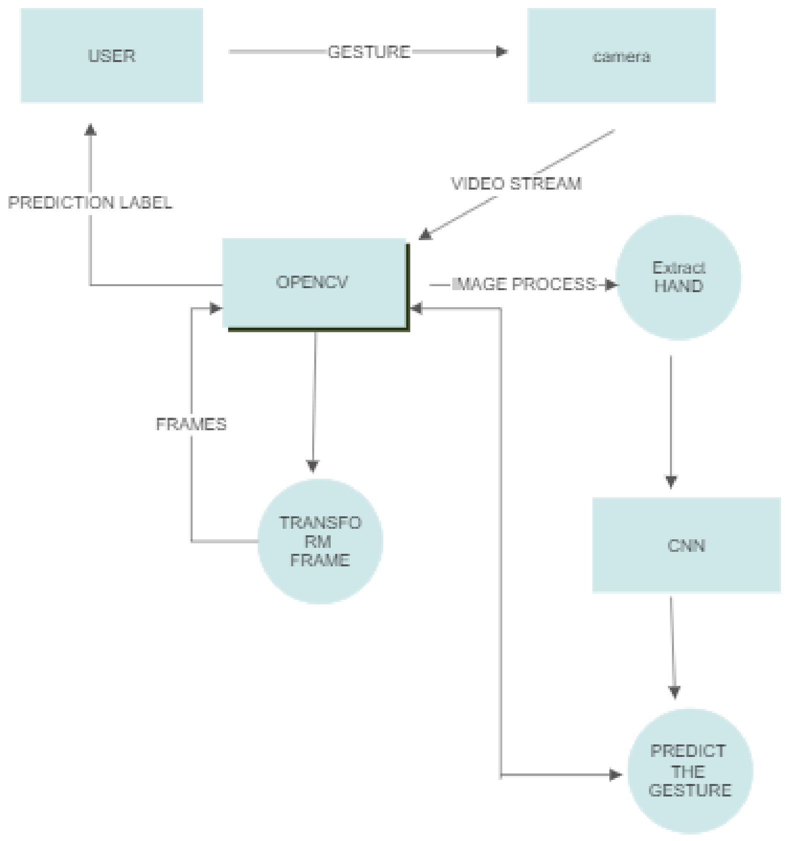
```
532/532 [==============================] - 54s 100ms/step - loss: 0.0360 - accuracy: 0.9923
Test Score:  0.036013953387737274
Test accuracy:  0.9922941327095032

Text(0.5, 1.0, 'Accuracy evolution')
```



5.3 Data Flow Diagram

The DFD is also known as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data, and the output data is generated by the system. It maps out the flow of information for any process or system, how data is processed in terms of inputs and outputs. It uses defined symbols like rectangles, circles and arrows to show data inputs, outputs, storage points and the routes between each destination. They can be used to analyse an existing system or model of a new one. A DFD can often visually “say” things that would be hard to explain in words and they work for both technical and non- technical. There are four components in DFD:

1. External Entity
2. Process
3. Data Flow
4. data Store


USER
GESTURE
camera
VIDEO STREAM
PREDICTION LABEL
OPENCV
IMAGE PROCESS
Extract HAND
FRAMES
TRANSFO RM FRAME
CNN
PREDICT THE GESTURE

CHAPTER 6. SYSTEM IMPLEMENTATION

6.1 Introduction

This Module, Reference to the Implementation of the project functionality and working.

6.2 Flowchart

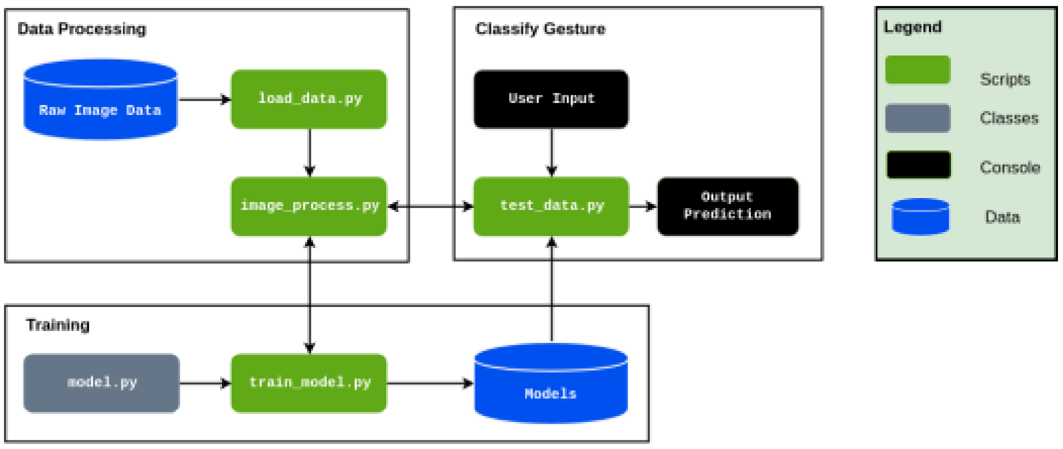


Figure 1: Block Diagram of Software

6.3 Code:

load_data.py:

```
import cv2
import os

def makedir(directory):
  if not os.path.exists(directory):
    os.makedirs(directory)
    return None
  else:
      pass


cap = cv2.VideoCapture(0)
i = 0
image_count = 0
```

```
while i<108:
  ret,frame = cap.read()
  frame = cv2.flip(frame,1)
  roi = frame[100:400,320:620]
  cv2.imshow('roi',roi)
  gray = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
  blur = cv2.GaussianBlur(gray,(5,5),2)
  th3 =
cv2.adaptiveThreshold(blur,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_
BINARY_INV,11,2)
  ret, test_image = cv2.threshold(th3, 70, 255,
cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
  test_image = cv2.resize(test_image,(64,64))
  cv2.imshow('roi scaled and gray',test_image)
  copy = frame.copy()
  cv2.rectangle(copy,(320,100),(620,400),(255,0,0),5)
  if i == 0:
    image_count=0
    cv2.putText(copy,'Hit Enter to record when ready to record A
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 1:
    image_count+=1
    cv2.putText(copy,'Record A gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/A/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 2:
    image_count+=1
    cv2.putText(copy,'Record B gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/A/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 3:
    cv2.putText(copy,'Hit Enter to record when ready to Record B
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 4:
    image_count+=1
    cv2.putText(copy,'Record B gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/B/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 5:
    image_count+=1
    cv2.putText(copy,'Record B gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/B/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 6:
    cv2.putText(copy,'Hit Enter to record when ready to Record C 
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 7:
    image_count+=1
    cv2.putText(copy,'Record C gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/C/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 8:
    image_count+=1
    cv2.putText(copy,'Record C gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/C/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 9:
    cv2.putText(copy,'Hit Enter to record when ready to Record D
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 10:
    image_count+=1
    cv2.putText(copy,'Record D gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/D/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 11:
    image_count+=1
    cv2.putText(copy,'Record D gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/D/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 12:
    cv2.putText(copy,'Hit Enter to record when ready to Record E
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 13:
    image_count+=1
    cv2.putText(copy,'Record E gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/E/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 14:
    image_count+=1
    cv2.putText(copy,'Record E gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/E/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 15:
    cv2.putText(copy,'Hit Enter to record when ready to Record F
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 16:
    image_count+=1
    cv2.putText(copy,'Record F gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/F/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 17:
    image_count+=1
    cv2.putText(copy,'Record F gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/F/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 18:
    cv2.putText(copy,'Hit Enter to record when ready to Record G
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 19:
    image_count+=1
    cv2.putText(copy,'Record G gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/G/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 20:
    image_count+=1
    cv2.putText(copy,'Record G gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
```

```
    gesture_one ='./handgesture/test/G/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 21:
    cv2.putText(copy,'Hit Enter to record when ready to Record H
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 22:
    image_count+=1
    cv2.putText(copy,'Record H gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/H/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 23:
    image_count+=1
    cv2.putText(copy,'Record H gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/H/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 24:
    cv2.putText(copy,'Hit Enter to record when ready to Record I
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 25:
    image_count+=1
    cv2.putText(copy,'Record I gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/I/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 26:
    image_count+=1
    cv2.putText(copy,'Record I gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
```

```
cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/I/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 27:
    cv2.putText(copy,'Hit Enter to record when ready to Record J
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 28:
    image_count+=1
    cv2.putText(copy,'Record J gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/J/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 29:
    image_count+=1
    cv2.putText(copy,'Record J gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/j/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 30:
    cv2.putText(copy,'Hit Enter to record when ready to Record K
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 31:
    image_count+=1
    cv2.putText(copy,'Record K gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/K/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 32:
    image_count+=1
    cv2.putText(copy,'Record K gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/K/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 33:
    cv2.putText(copy,'Hit Enter to record when ready to Record L 
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 34:
    image_count+=1
    cv2.putText(copy,'Record L gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/L/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 35:
    image_count+=1
    cv2.putText(copy,'Record L gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/L/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 36:
    cv2.putText(copy,'Hit Enter to record when ready to Record M 
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 37:
    image_count+=1
    cv2.putText(copy,'Record M gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/M/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 38:
    image_count+=1
    cv2.putText(copy,'Record M gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/M/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 39:
    cv2.putText(copy,'Hit Enter to record when ready to Record N
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 40:
    image_count+=1
    cv2.putText(copy,'Record N gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/N/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 41:
    image_count+=1
    cv2.putText(copy,'Record N gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/N/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 42:
    cv2.putText(copy,'Hit Enter to record when ready to Record O
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 43:
    image_count+=1
    cv2.putText(copy,'Record O gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/O/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 44:
    image_count+=1
    cv2.putText(copy,'Record O gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/O/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 45:
    cv2.putText(copy,'Hit Enter to record when ready to Record P
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 46:
    image_count+=1
    cv2.putText(copy,'Record P gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/P/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 47:
    image_count+=1
    cv2.putText(copy,'Record P gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/P/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 48:
    cv2.putText(copy,'Hit Enter to record when ready to Record Q
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 49:
    image_count+=1
    cv2.putText(copy,'Record Q gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
```

```
cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/Q/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 50:
    image_count+=1
    cv2.putText(copy,'Record Q gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/Q/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 51:
    cv2.putText(copy,'Hit Enter to record when ready to Record R
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 52:
    image_count+=1
    cv2.putText(copy,'Record R gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/R/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 53:
    image_count+=1
    cv2.putText(copy,'Record R gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/R/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 54:
    cv2.putText(copy,'Hit Enter to record when ready to Record S
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 55:
    image_count+=1
    cv2.putText(copy,'Record S gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/S/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 56:
    image_count+=1
    cv2.putText(copy,'Record S gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/S/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 57:
    cv2.putText(copy,'Hit Enter to record when ready to Record T 
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 58:
    image_count+=1
    cv2.putText(copy,'Record T gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/T/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 59:
    image_count+=1
    cv2.putText(copy,'Record T gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/T/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 60:
    cv2.putText(copy,'Hit Enter to record when ready to Record U 
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
```

```
  if i == 61:
    image_count+=1
    cv2.putText(copy,'Record U gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/U/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 62:
    image_count+=1
    cv2.putText(copy,'Record U gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/U/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 63:
    cv2.putText(copy,'Hit Enter to record when ready to Record V
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 64:
    image_count+=1
    cv2.putText(copy,'Record V gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/V/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 65:
    image_count+=1
    cv2.putText(copy,'Record V gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/V/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 66:
    cv2.putText(copy,'Hit Enter to record when ready to Record W 
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 67:
    image_count+=1
    cv2.putText(copy,'Record W gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/W/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 68:
    image_count+=1
    cv2.putText(copy,'Record W gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/W/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 69:
    cv2.putText(copy,'Hit Enter to record when ready to Record X 
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 70:
    image_count+=1
    cv2.putText(copy,'Record X gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/X/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 71:
    image_count+=1
    cv2.putText(copy,'Record X gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/X/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 72:
    cv2.putText(copy,'Hit Enter to record when ready to Record Y
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 73:
    image_count+=1
    cv2.putText(copy,'Record Y gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/Y/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 74:
    image_count+=1
    cv2.putText(copy,'Record Y gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/Y/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 75:
    cv2.putText(copy,'Hit Enter to record when ready to Record Z
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 76:
    image_count+=1
    cv2.putText(copy,'Record Z gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/Z/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 77:
    image_count+=1
    cv2.putText(copy,'Record Z gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
```

```
    gesture_one ='./handgesture/test/Z/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 78:
    cv2.putText(copy,'Hit Enter to record when ready to Record 0
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 79:
    image_count+=1
    cv2.putText(copy,'Record 0 gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/0/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 80:
    image_count+=1
    cv2.putText(copy,'Record 0 gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/0/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 81:
    cv2.putText(copy,'Hit Enter to record when ready to Record 1
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 82:
    image_count+=1
    cv2.putText(copy,'Record 1 gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/1/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 83:
    image_count+=1
    cv2.putText(copy,'Record 1 gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
```

```
cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/1/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 84:
    cv2.putText(copy,'Hit Enter to record when ready to Record 2
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 85:
    image_count+=1
    cv2.putText(copy,'Record 2 gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/2/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 86:
    image_count+=1
    cv2.putText(copy,'Record 2 gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/2/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 87:
    cv2.putText(copy,'Hit Enter to record when ready to Record 3
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 88:
    image_count+=1
    cv2.putText(copy,'Record 3 gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/3/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 89:
    image_count+=1
    cv2.putText(copy,'Record 3 gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/3/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 90:
    cv2.putText(copy,'Hit Enter to record when ready to Record 4 
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 91:
    image_count+=1
    cv2.putText(copy,'Record 4 gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/4/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 92:
    image_count+=1
    cv2.putText(copy,'Record 4 gesture - 
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/4/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 93:
    cv2.putText(copy,'Hit Enter to record when ready to Record 5 
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 94:
    image_count+=1
    cv2.putText(copy,'Record 5 gesture - 
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/5/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 95:
    image_count+=1
    cv2.putText(copy,'Record 5 gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/5/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 96:
    cv2.putText(copy,'Hit Enter to record when ready to Record 6
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 97:
    image_count+=1
    cv2.putText(copy,'Record 6 gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/6/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 98:
    image_count+=1
    cv2.putText(copy,'Record 6 gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/6/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 99:
    cv2.putText(copy,'Hit Enter to record when ready to Record 7
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 100:
    image_count+=1
    cv2.putText(copy,'Record 7 gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/7/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 101:
    image_count+=1
    cv2.putText(copy,'Record 7 gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/7/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 102:
    cv2.putText(copy,'Hit Enter to record when ready to Record 8
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 103:
    image_count+=1
    cv2.putText(copy,'Record 8 gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/8/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 104:
    image_count+=1
    cv2.putText(copy,'Record 8 gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/8/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 105:
    cv2.putText(copy,'Hit Enter to record when ready to Record 9
gesture',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  if i == 106:
    image_count+=1
    cv2.putText(copy,'Record 9 gesture -
Train',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
```

```
cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/train/9/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 107:
    image_count+=1
    cv2.putText(copy,'Record 9 gesture -
Test',(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)

cv2.putText(copy,str(image_count),(400,400),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0
),1)
    gesture_one ='./handgesture/test/9/'
    makedir(gesture_one)
    cv2.imwrite(gesture_one+str(image_count)+".jpg",test_image)
  if i == 108:
    cv2.putText(copy,"Hit Enter to
Exit",(100,100),cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),1)
  cv2.imshow('frame',copy)

  if cv2.waitKey(1) == 13:
    image_count=0
    i+=1

cap.release()
cv2.destroyAllWindows()
```

Train_model.py:

```
import tensorflow.keras
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import  Dense,Dropout,Activation,Flatten,BatchNormalization
from tensorflow.keras.layers import Conv2D,MaxPooling2D
import os

num_classes = 35
img_rows,img_cols = 64,64
batch_size = 32
train_data_dir = './handgesture/train'
validation_data_dir = './handgesture/test'

train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range = 30,
    shear_range = 0.2,
    zoom_range = 0.2,
    horizontal_flip = True,
    fill_mode = 'nearest')

validation_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size = (img_rows,img_cols),
    color_mode = 'grayscale',
    class_mode = 'categorical')

validation_generator = validation_datagen.flow_from_directory(
    validation_data_dir,
    target_size = (img_rows,img_cols),
    batch_size = batch_size,
    color_mode = 'grayscale',
    class_mode = 'categorical')

model = Sequential()
model.add(Conv2D(32,kernel_size=(3,3),activation='relu',input_shape=(64,64,1)))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(32,kernel_size=(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dense(128,activation='relu'))
model.add(Dropout(0.20))

model.add(Dense(35,activation='softmax'))
print(model.summary())

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

nb_train_samples = 65000
nb_validation_samples = 25966
epochs = 10
```

```
history = model.fit(
    train_generator,
    steps_per_epoch = nb_train_samples // batch_size,
    epochs=epochs,
    validation_data = validation_generator,
    validation_steps = nb_validation_samples // bstch_size)


model_json = model.to_json()
with open("model-bw.json", "w") as json_file:
    json_file.write(model_json)
print('Model Saved')
model.save_weights('model-bw.h5')
print('Weights saved')
```

App.py:

```
from keras.models import load_model,model_from_json
import cv2 as cv2
import numpy as np
import operator
import sys, os

json_file = open("model-bw.json", "r")
model_json = json_file.read()
json_file.close()
loaded_model = model_from_json(model_json)
loaded_model.load_weights("model-bw.h5")
print("Loaded model from disk")
#categories = {1: 'ONE', 2: 'TWO', 3: 'THREE', 4: 'FOUR', 5:
'FIVE',6:'SIX',7:'SEVEN',8:'EIGHT',9:'NINE',A:'A',B:'B',C:'C',D:'D',E:'E',F:'F',G:'G',H:'H',I:'
I',J:'J',K:'K',L:'L',M:'M',N:'N',O:'O',P:'P',Q:'Q',R:'R',S:'S',T:'T',U:'U',V:'V',W:'W',X:'X',Y:'Y',Z
:'Z'}

cap = cv2.VideoCapture(0)

while True:
  ret,frame = cap.read()
  frame = cv2.flip(frame,1)
  roi = frame[100:400,320:620]
  cv2.imshow('roi',roi)
  gray = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
  blur = cv2.GaussianBlur(gray,(5,5),2)
  th3 = cv2.adaptiveThreshold(blur,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY_INV,11,2)
  ret, test_image = cv2.threshold(th3, 70, 255, cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
  test_image = cv2.resize(test_image,(64,64))
  cv2.imshow('roi scaled and grey',test_image)
  copy = frame.copy()
  cv2.rectangle(copy,(320,100),(620,400),(255,0,0),5)
  roi=roi/255
  result = loaded_model.predict(test_image.reshape(1,64,64,1))
  prediction={'ONE':result[0][1],
            'TWO':result[0][2],
            'THREE':result[0][3],
            'FOUR':result[0][4],
            'FIVE':result[0][5],
            'SIX':result[0][6],
            'SEVEN':result[0][7],
            'EIGHT':result[0][8],
            'NINE':result[0][9],
            'A':result[0][10],
            'B':result[0][11],
            'C':result[0][11],
            'D':result[0][12],
            'E':result[0][13],
            'F':result[0][14],
            'G':result[0][15],
            'H':result[0][16],
            'I':result[0][17],
            'J':result[0][18],
            'K':result[0][19],
            'L':result[0][20],
            'M':result[0][21],
            'N':result[0][22],
            'O':result[0][23],
            'P':result[0][24],
            'Q':result[0][25],
            'R':result[0][26],
            'S':result[0][27],
            'T':result[0][28],
            'U':result[0][29],
            'V':result[0][30],
            'W':result[0][31],
            'X':result[0][32],
            'Y':result[0][33],
            'Z':result[0][34]}
  prediction=sorted(prediction.items(),key=operator.itemgetter(1),reverse=True)
  current_predict = prediction[0][0]

cv2.putText(copy,current_predict,(300,100),cv2.FONT_HERSHEY_COMPLEX,2,(0,255,0,
2))
  cv2.imshow('frame',copy)

  if cv2.waitKey(1) == 13:
    break

cap.release()
cv2.destroyAllWindows()
```

6.4 Testing Approach

The purpose of testing is to discover errors. Testing is a process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. Software testing is an important element of the software quality assurance and represents the ultimate review of specification, design and coding. The increasing feasibility of software as a system and the cost associated with the software failures are motivated forces for well planned through testing.

Testing Objectives:

There are several rules that can serve as testing objectives they are:
- Testing is a process of executing program with the intent of finding an error.
- A good test case is the one that has a high probability of finding an undiscovered error.

To determine whether our pre-processing of images actually results in a more robust model, we verified on a test set comprised of images from the original dataset, and our own collected image data. The performance of the models on the test set is shown in Table 2. We see that the model trained on pre-processed images performs much better than the model trained on the original images, likely due to the former’s lack of overfitting.

Types of Testing:

In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at different phases of software development are :

Unit Testing: Unit testing is done on individual models as they are completed and becomes executable. It is confined only to the designer's requirements. Unit testing is different from and should be preceded by other techniques, including:

- Inform Debugging
- Code Inspection Black Box testing in this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program. This testing has been used to find error in the following categories: Incorrect or missing functions
- Interface errors
- Errors in data structures are external database access
- Performance error
- Initialisation and termination of errors
- In this testing only the output is checked for correctness
- The logical flow of data is not checked

White Box testing

In this the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases. It has been used to generate the test cases in the following cases:

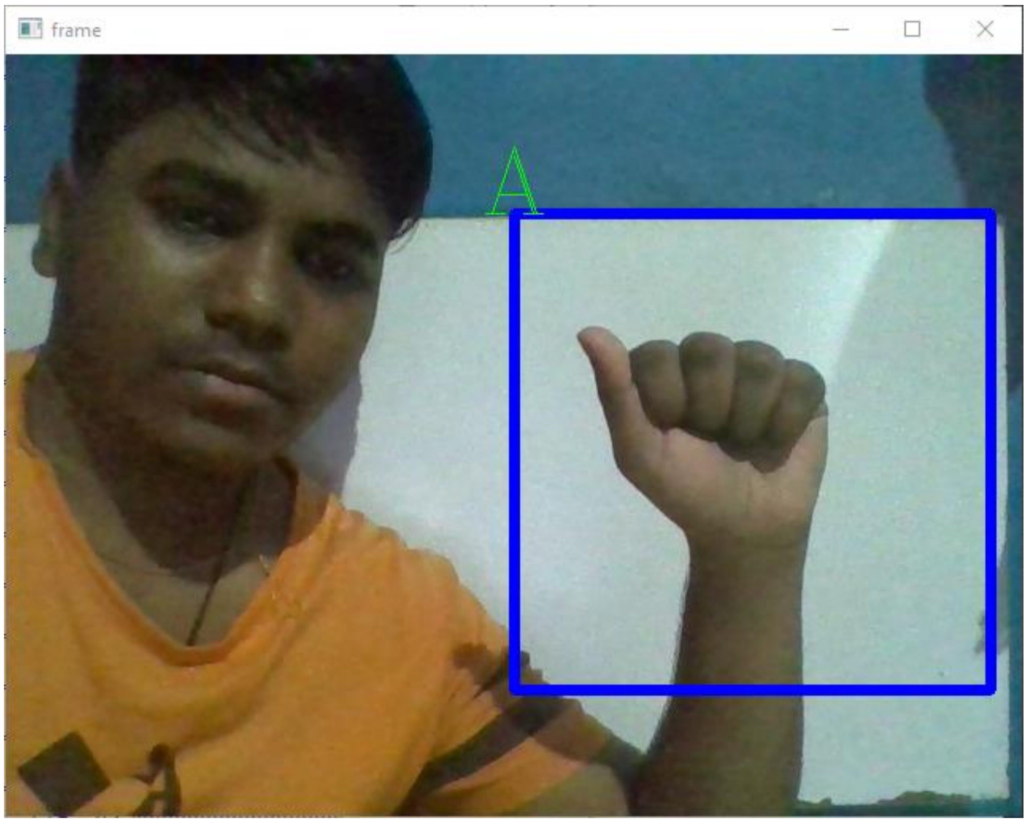
- Guarantee that all independent paths have been executed
- Execute all loops at their boundaries and within their operational bounds.
- Execute internal data structures to ensure their validity.

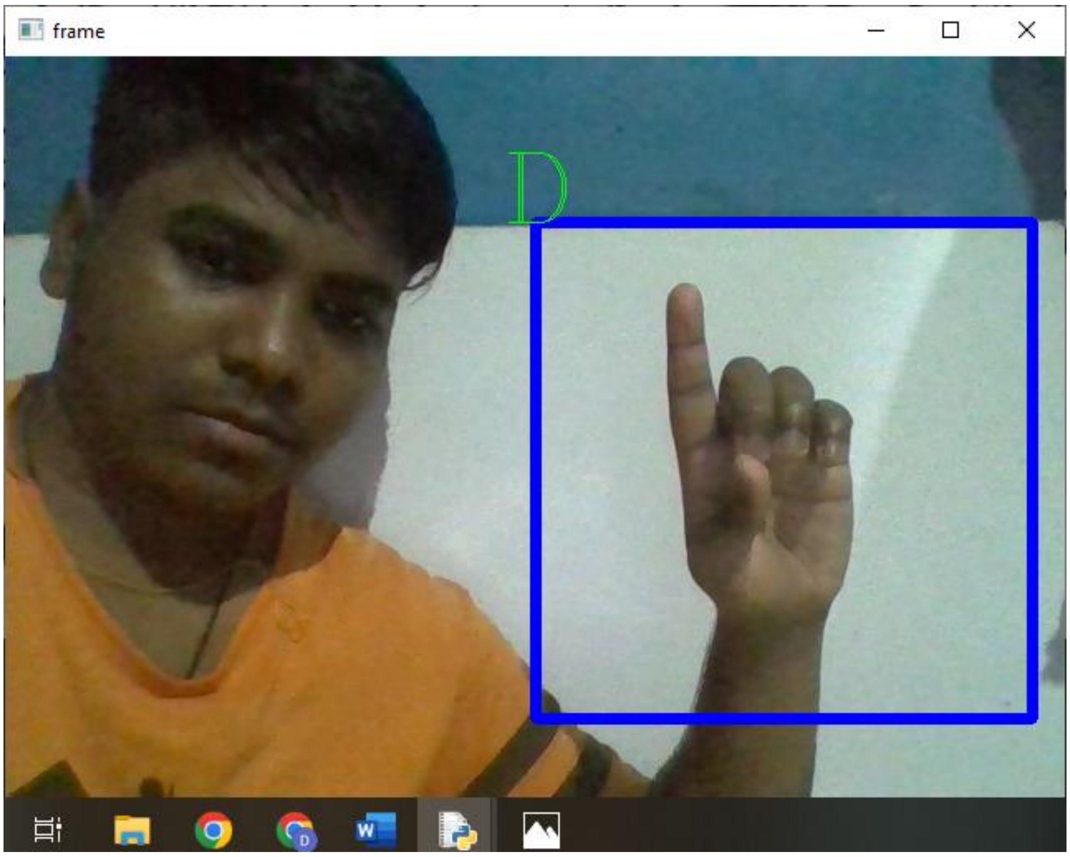
6.5 Testing Case

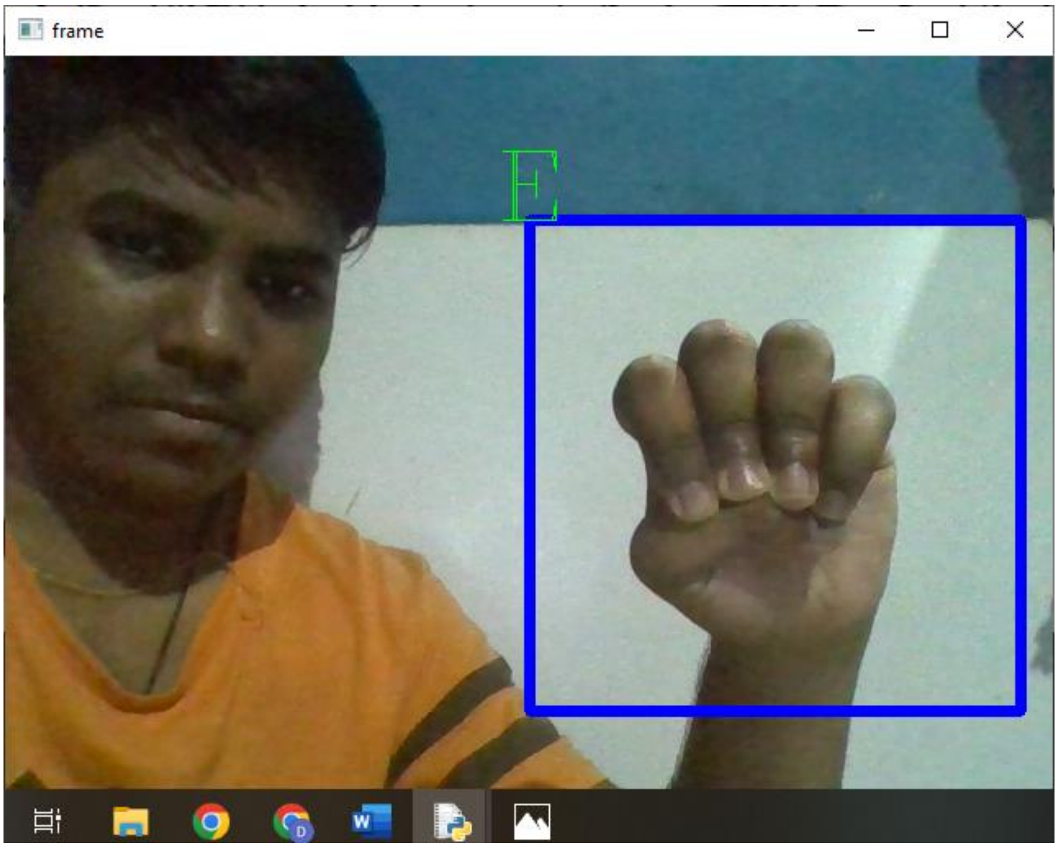
ID	Test Case	Input Description	Expected Output	Test Status
1	Loaded Model	Initializing trained model and load it into ON	Loaded Model without Error	Pass
2	Converting Video to Frame	Capturing video and converting it into frames	Image frame of Capture video	Pass
3	Recognize Hand Gesture	Image frame that contains hand object	label	Pass

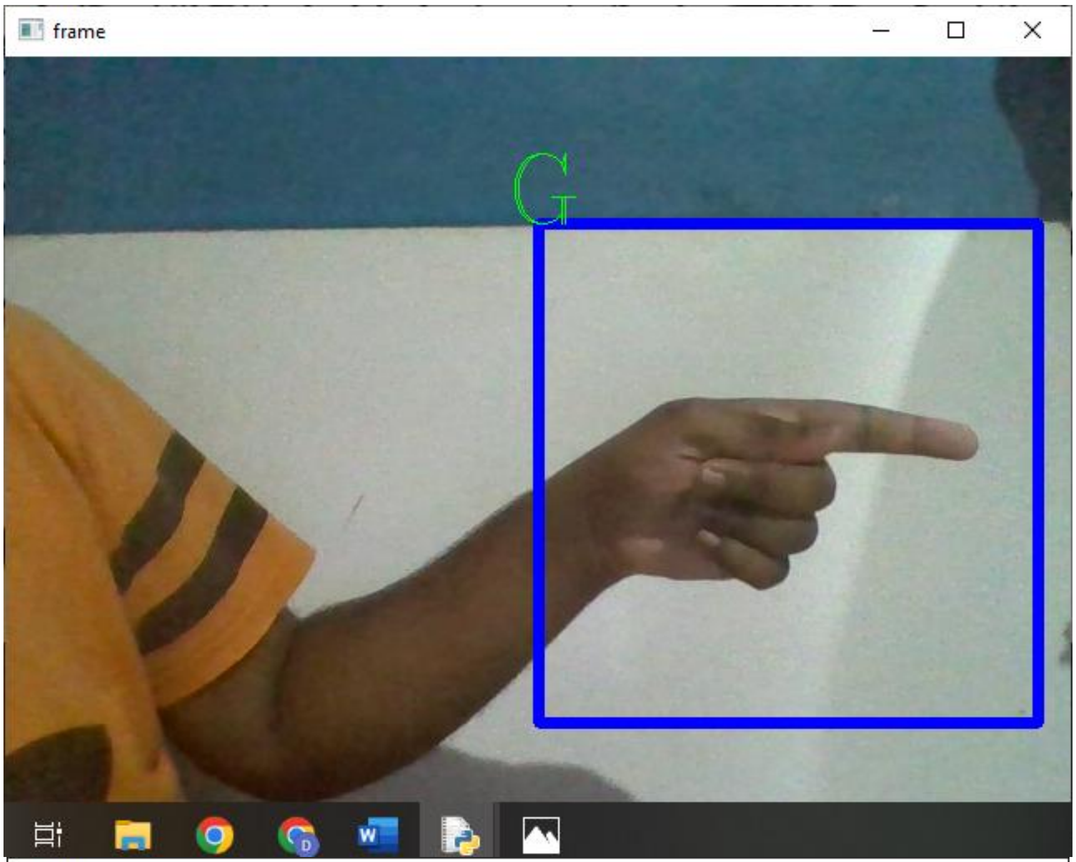
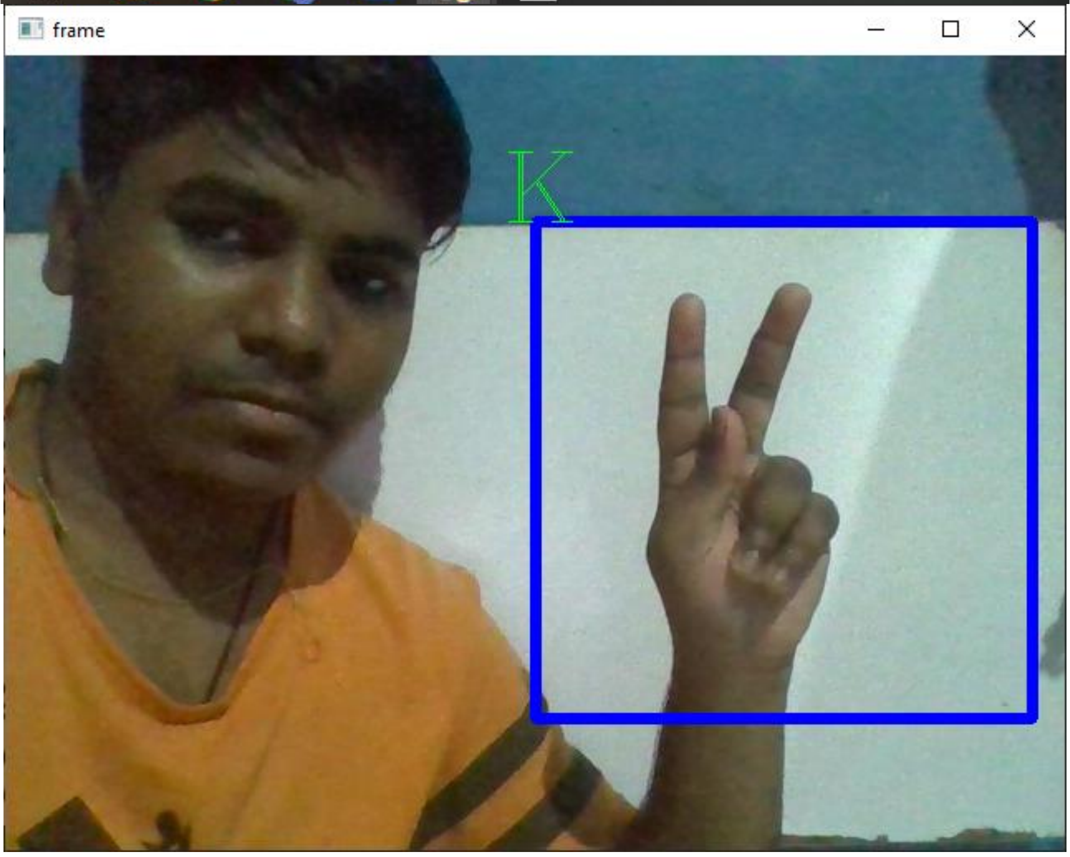
CHAPTER 7. RESULTS

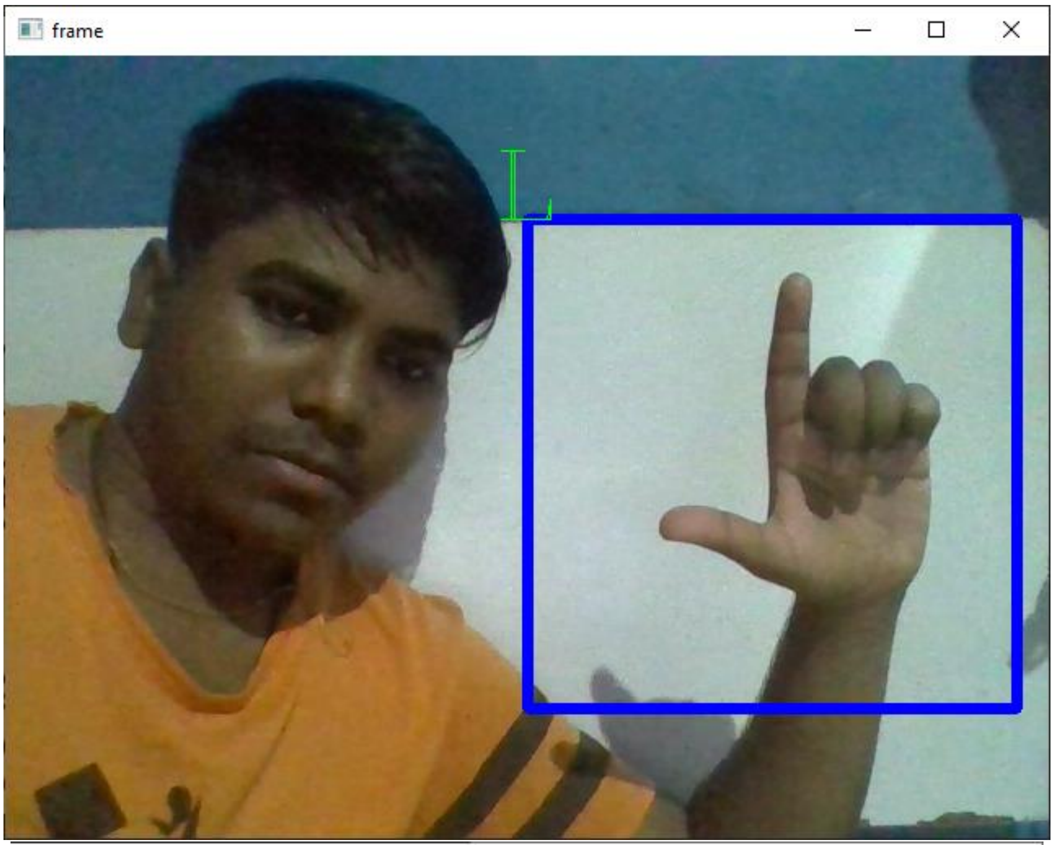
We have achieved an accuracy of 99.8% in our model using only layer 1 of our algorithm, and using the combination of layer 1 and layer 2 we achieve an accuracy of 98.0%, which is a better accuracy then most of the current research papers on American sign language. Most of the research papers focus on using devices like Kinect for hand detection. In [7] they build a recognition system for Flemish sign language using convolutional neural networks and Kinect and achieve an error rate of 2.5%. In [8] a recognition model is built using hidden Markov model classifier and a vocabulary of 30 words and they achieve an error rate of 10.90%. In [9] they achieve an average accuracy of 86% for 41 static gestures in Japanese sign language. Using depth sensors map [10] achieved an accuracy of 99.99% for observed signers and 83.58% and 85.49% for new signers. They also used CNN for their recognition system. One thing should be noted that our model doesn't uses any background subtraction algorithm whiles some of the models present above do that. So, once we try to implement background subtraction in our project the accuracies may vary. On the other hand, most of the above projects use Kinect devices but our main aim was to create a project which can be used with readily available resources. A sensor like Kinect not only isn't readily available but also is expensive for most of audience to buy and our model uses a normal webcam of the laptop hence it is great plus point.

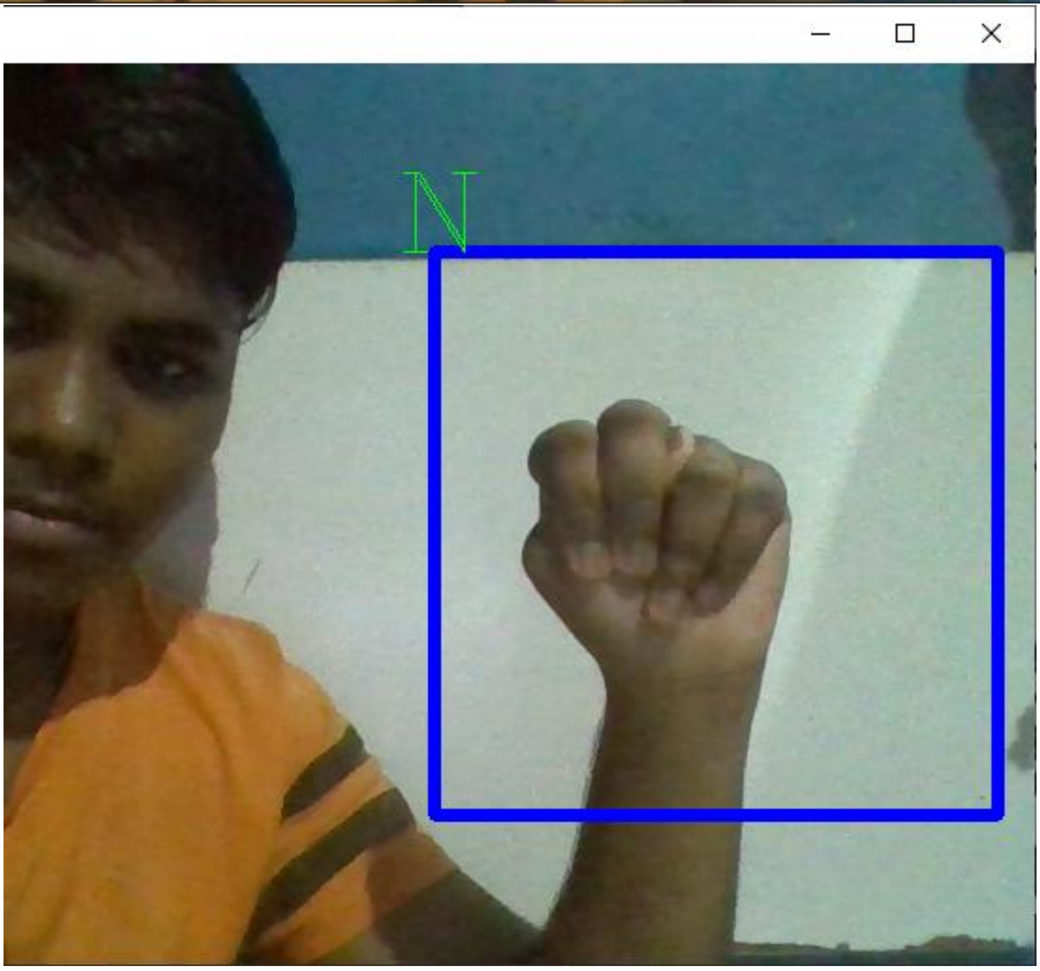


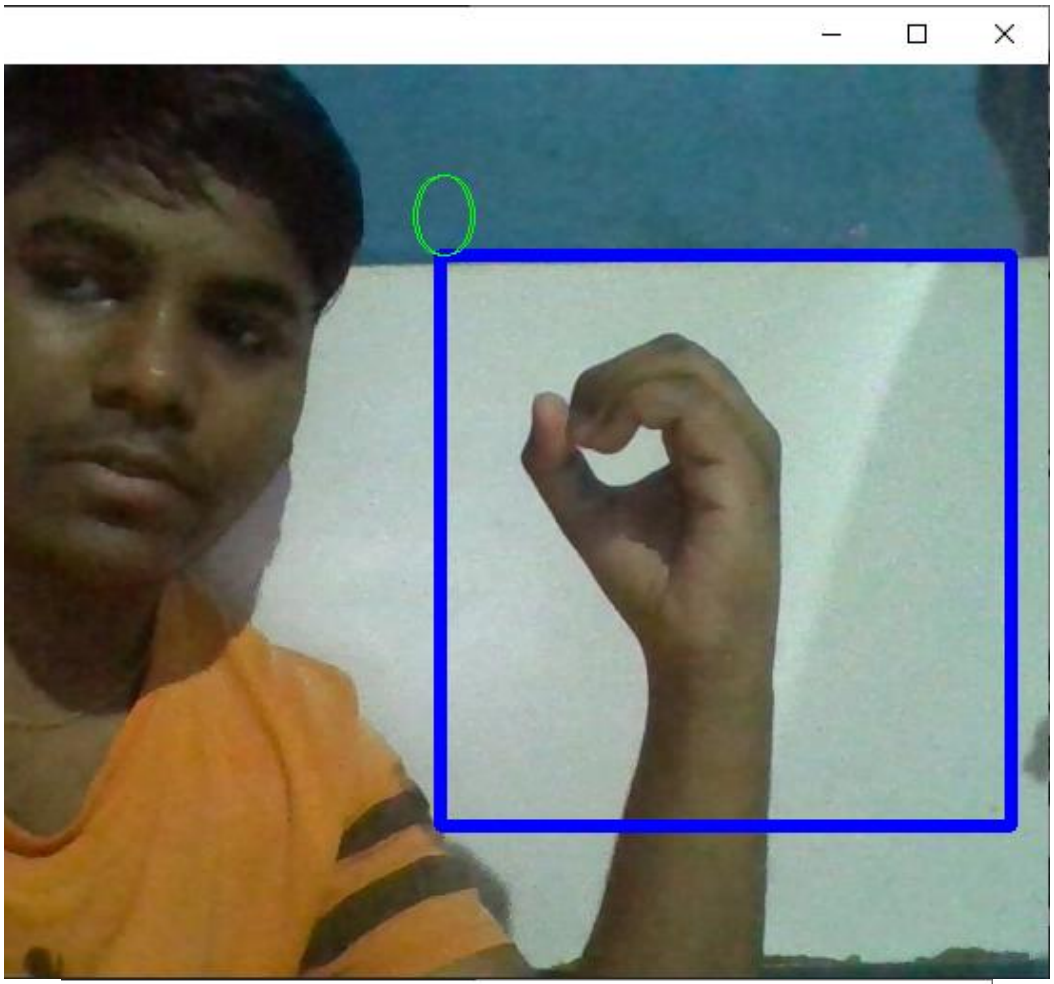

frame
D

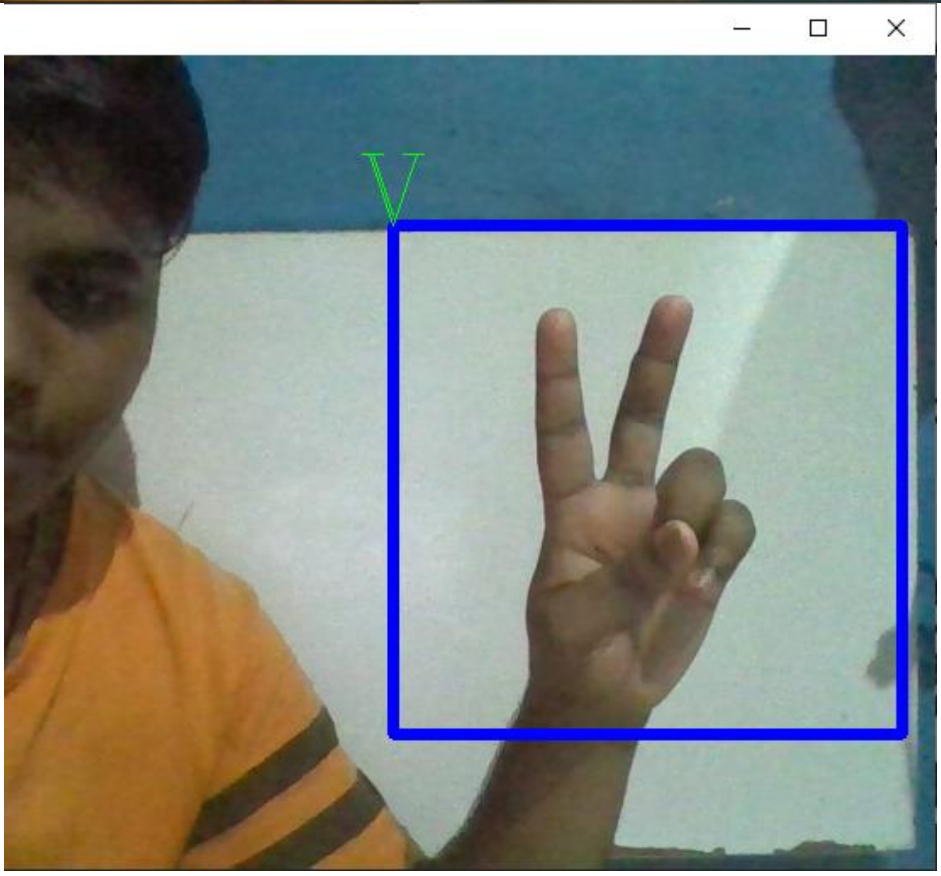

frame
E


frame
G

frame
K


frame
L


N


O


V

CHAPTER 8. CONCLUSION AND FUTURE SCOPE

8.1 Conclusion

In this report, a functional real time vision based American sign language recognition for D&M people have been developed for asl alphabets. We achieved final accuracy of 98.0% on our dataset. We are able to improve our prediction after implementing two layers of algorithms in which we verify and predict symbols which are more similar to each other. This way we are able to detect almost all the symbols provided that they are shown properly, there is no noise in the background and lighting is adequate.

There were many challenges faced by us during the project. The very first issue we faced was of dataset. We wanted to deal with raw images and that too square images as CNN in Keras as it was a lot more convenient working with only square images. We couldn’t find any existing dataset for that hence we decided to make our own dataset. Second issue was to select a filter which we could apply on our images so that proper features of the images could be obtained and hence then we could provide that image as input for CNN model. We tried various filter including binary threshold, canny edge detection, gaussian blur etc. but finally we settled with gaussian blur filter. More issues were faced relating to the accuracy of the model we trained in earlier phases which we eventually improved by increasing the input image size and also by improving the dataset.

8.1.1 Advantage

1. The system in this report has the image processing module which can directly process the image in the blur threshold necessary for the training process.

2. The Training Process directly can process the Image for Image Augmentation and train the model for Maximum Accuracy.

3.The Gesture Detection module convert video frame to image frame and input in the model and get the Prediction of the Maximum Label.

8.1.2 Limitation

1. The Model in the report cannot process the Background the noise which create the inaccuracy in the model.

2.The Model Required Ambient Light condition for Gesture classification.

8.2 Future Improvement

We are planning to achieve higher accuracy even in case of complex backgrounds by trying out various background subtraction algorithms. We are also thinking of improving the pre-processing to predict gestures in low light conditions with a higher accuracy

The proposed sign language recognition system used to recognize sign language letters can be further extended to recognize gestures facial expressions. Instead of displaying letter labels it will be more appropriate to display sentences as more appropriate translation of language. This also increases readability. The scope of different sign languages can be increased. More training data can be added to detect the letter with more accuracy. This project can further be extended to convert the signs to speech. We are also trying to develop this model into low size to be applied in Edge device, web frameworks and Mobile Application.

CHAPTER 9. REFERENCE

9.1 Reference and Biography

1. https://www.youtube.com/watch?v=7HPwo4wnJeA Title: Image classification using CNN
2. https://www.youtube.com/watch?v=zfiSAzpy9NM Title: Simple explanation of convolutional neural network
3. https://data-flair.training/blogs/sign-language-recognition-python-ml-opencv/
4. https://data-flair.training/blogs/convolutional-neural-networks-tutorial/
5. https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html
6. https://www.analyticsvidhya.com/blog/2020/08/image-augmentation-on-the-fly-using-keras-imagedatagenerator/
7. https://data-flair.training/blogs/tensorflow-image-recognition/
8. https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_b ilateral_filter/gausian_median_blur_bilateral_filter.html
9. Pigou L., Dieleman S., Kindermans PJ., Schrauwen B. (2015) Sign Language Recognition Using Convolutional Neural Networks. In: Agapito L., Bronstein M., Rother C. (eds) Computer Vision - ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8925. Springer, Cham
10. https://arxiv.org/pdf/2201.01486
11. https://www.analyticsvidhya.com/blog/2021/06/sign-language-recognition-for-computer-vision-enthusiasts/
12. https://link.springer.com/chapter/10.1007/978-3-319-16178-5_40
13. https://ieeexplore.ieee.org/abstract/document/9399594