Vision based American Sign Language Recognition

A Report submitted

in partial fulfillment of the Degree of

Bachelor of Technology

In

Computer Engineering

By

ANAS INTEKHAB ALAM (18BCS003)

MASUMA AKTHER (18BCS010)

SHAZIA ANSARI (17BCS066)

Under the guidance

of

Prof. Mohammad Amjad

Department of Computer Engineering

F/O Engineering & Technology

Jamia Millia Islamia New Delhi –110025

MAY, 2022

جامعہ ملیّہ اسلامیہ
जामिया मिल्लिया इस्लामिया

Jamia Millia Islamia
A Central University
(NAAC Accredited A++ Grade)

CERTIFICATE

This is to certify that the project report entitled *Vision based American Sign Language Recognition* submitted by Anas Intekhab Alam, Masuma Akther, and Shazia Ansari to the Department of Computer Engineering, F/O Engineering & Technology, Jamia Millia Islamia New Delhi – 110025 in partial fulfillment for the award of the degree of B. Tech in (Computer Engineering) is a bona fide record of project work carried out by them under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institution or University for the award of any degree.

Prof. Mohammad Amjad

Supervisor

Department of Computer Engineering

Jamia Millia Islamia New Delhi – 110025

16/06/2022

DECLARATION

We declare that this project report titled *Vision based American Sign Language Recognition* submitted in partial fulfillment of the degree of B. Tech in (Computer Engineering) is a record of original work carried out by us under the supervision of Prof. Mohammad Amjad, and has not formed the basis for the award of any other degree, in this or any other Institution or University. In keeping with the ethical practice in reporting scientific information, due acknowledgements have been made wherever the findings of others have been cited.

ANAS INTEKHAB ALAM
18BCS003

MASUMA AKTHER
18BCS010

SHAZIA ANSARI
17BCS066

New Delhi – 110025
16/06/2022

ACKNOWLEDGMENTS

In the accomplishment of completion of my project on *Vision based American Sign Language Recognition* we would like to convey my special gratitude to Prof. Mohammad Amjad, of the department of computer engineering JMI and as well as Dr. Bashir Alam Head of the department of computer engineering JMI.

Your valuable guidance and suggestions helped me in various phases of the completion of this project. We will always be thankful to you in this regard. My thanks and appreciations also go to my group member for their support and coordination, we hope we will achieve more in our future endeavors.

Finally, we would like to thank our parents and friends who helped us a lot in finishing this project.

ANAS INTEKHAB ALAM

MASUMA AKTHER

SHAZIA ANSARI

ABSTRACT

The communication barrier between the hearing majority and the deaf community is an undeniable issue. Our work aims to break this barrier through the use of automatic sign language recognition. In our method, we have developed a method that combines neural networks for finger-based sign language which is basically uses the hand gesture. Hand gesture is one of the most prominent ways of communication since the beginning of the human era. Hand gesture recognition extends human-computer interaction (HCI) more convenient and flexible. Therefore, it is important to identify each character correctly for calm and error-free HCI. Literature survey reveals that most of the existing hand gesture recognition (HGR) systems have considered only a few simple discriminating gestures for recognition performance. This paper applies deep learning-based VGG16 and RESNET50 modeling of static signs in the context of sign language recognition. In this work, VGG16 and RESNET50 are employed for HGR where alphabets of ASL is considered. The pros and cons of VGG16 and RESNET50 used for HGR are also highlighted. 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. The proposed method was evaluated based on accuracy, Model loss using a dataset comprising 4608 images of sign images which is ASL of the English alphabet and numbers from 1 – 10 which have spread of 40 class labels assigned to them. Each class label is a set of sign images of the English alphabet and numbers.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1. Overview

American sign language is a predominant sign language Since the only disability Deaf & Dumb people have been 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, behavior and visuals. Deaf and dumb people make use of their hands to express different gestures to express their ideas with other people. Gestures are the nonverbally 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:

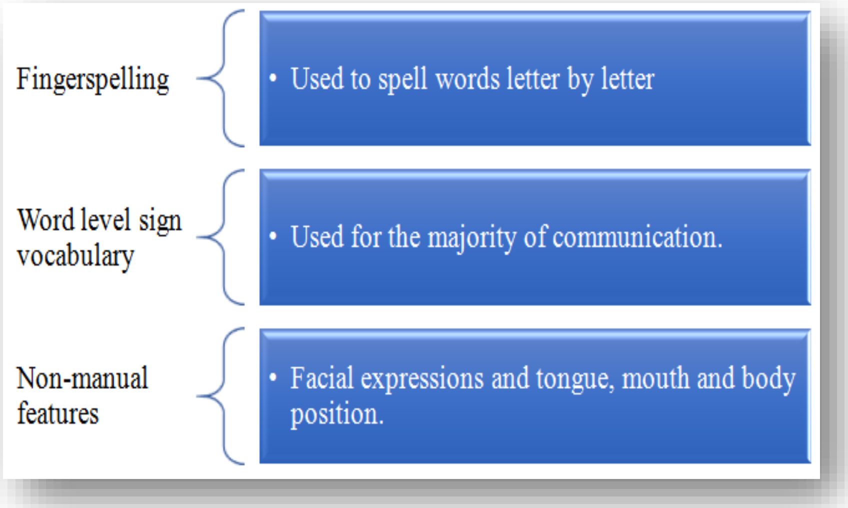

Fingerspelling
• Used to spell words letter by letter
Word level sign vocabulary
• Used for the majority of communication.
Non-manual features
• Facial expressions and tongue, mouth and body position.

Figure 1.1: Components of sign language

In our project we basically focus on producing a model which can recognize 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. We use some models they are VGG16 and RESNET50.

1.2. Motivation of the study

Sign language is learned by deaf and dumb, and usually it is not known to normal people, so it becomes a challenge for communication between a normal and hearing impaired person. Its strike to our mind to bridge the gap between hearing impaired and normal people to make the communication easier. Sign Language Recognition (SLR) system takes an input expression from the hearing impaired person gives output to the normal person in the form of text or voice.

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 interfaces (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.

1.3. Literature survey

In the recent years there has been tremendous research done on the hand gesture recognition. With the help of literature survey done we realized the basic steps in hand gesture recognition are: -

- Data acquisition
- Data preprocessing
- Feature extraction
- Gesture classification

1.3.1 Keywords and definitions

- **Feature Extraction and Representation:** The representation of an image as a 3D matrix having dimension as of height and width of the image and the value of each pixel as depth (1 in case of Grayscale and 3 in case of RGB). Further, these pixel values are used for extracting useful features using CNN.
- **Artificial Neural Networks:** Artificial Neural Network is a connection of neurons, replicating the structure of human brain. Each connection of neuron transfers information to another neuron. Inputs are fed into first layer of neurons which processes it and transfers to another layer of neurons called as hidden layers. After processing of information through multiple layers of hidden layers, information is passed to final output layer.

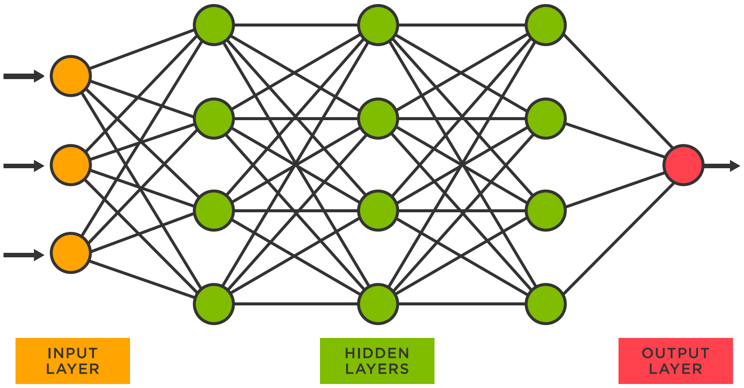


Figure 1.2: Artificial Neural Network

They are capable of learning and they have to be trained. There are different learning strategies:

1. Unsupervised Learning

2. Supervised Learning

3. Reinforcement Learning

- **Convolution 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.

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.

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

3. 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.

4. 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.

- **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.
- **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.

CHAPTER 2

MODELS AND CONCEPTS USED

In our project we have used two different models they are:-

- VGG16
- RESNET50

2.1. VGG16

VGG16 is a simple and widely used Convolutional Neural Network (CNN) Architecture used for ImageNet, a large visual database project used in visual object recognition software research and VGG-16 is 16 layers deep also. VGG16 is used in many deep learning image classification techniques and is popular due to its ease of implementation. VGG16 is extensively used in learning applications due to the advantage that it has. VGG16 is a CNN Architecture, which was used to win the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2014. It is still one of the best vision architectures to date.

VGG16 architecture

During training, the input to the convnets is a fixed-size 224 x 224 RGB image. Subtracting the mean RGB value computed on the training set from each pixel is the only pre-processing done here. The image is passed through a stack of convolutional (conv.) layers, where filters with a very small receptive field: 3 × 3 (which is the smallest size to capture the notion of left/right, up/down, center and has the same effective receptive field as one 7 x 7), is used. It is deeper, has more non-linearities, and has fewer parameters. In one of the configurations, 1 × 1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity), are also utilized. The convolution stride and the spatial padding of conv. layer input is fixed to 1 pixel for 3 x 3 convolutional layers, which ensures that the spatial resolution is preserved after convolution. Five max-pooling layers, which follow some of the convolutional layers, helps in spatial pooling. Max-pooling is performed over a 2×2-pixel window, with stride 2. There are three Fully-Connected (FC) layers that follow a stack of convolutional layers (these have different depths in different architectures): the first two have 4096 channels each, the third performs 1000-way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. The configuration of the fully connected layers is the same in all networks. The 16 layer VGG architecture was the best performing, and it achieved a top-5 error rate of 7.3% (92.7% accuracy) in ILSVRC — 2014, as mentioned above. VGG16 had significantly outperformed the previous generation of models ILSVRC — 2012 and ILSVRC — 2013 competitions.

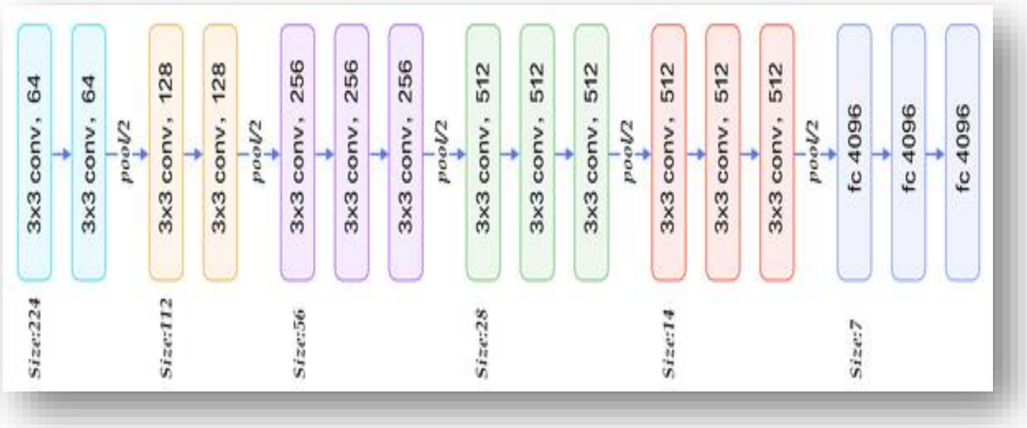


Figure 2.1: VGG16 Architecture

2.2. RESNET50

ResNet50 is a variant of ResNet model which has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer. It has 3.8 x 10^9 Floating points operations. It is a widely used ResNet model and we have explored ResNet50 architecture in depth.

RESNET50 architecture

We can see the Resent 50 architecture contains the following element:

- A convolution with a kernel size of 7 * 7 and 64 different kernels all with a stride of size 2 giving us 1 layer.

- Next, we see max pooling with also a stride size of 2.
- In the next convolution there is a 1 * 1,64 kernel following this a 3 * 3,64 kernel and at last a 1 * 1,256 kernel, these three layers are repeated in total 3 time so giving us 9 layers in this step.
- Next, we see kernel of 1 * 1,128 after that a kernel of 3 * 3,128 and at last a kernel of 1 * 1,512 this step was repeated 4 time so giving us 12 layers in this step.
- After that there is a kernel of 1 * 1,256 and two more kernels with 3 * 3,256 and 1 * 1,1024 and this is repeated 6 time giving us a total of 18 layers.
- And then again, a 1 * 1,512 kernel with two more of 3 * 3,512 and 1 * 1,2048 and this was repeated 3 times giving us a total of 9 layers.
- After that we do a average pool and end it with a fully connected layer containing 1000 nodes and at the end a SoftMax function so this gives us 1 layer.

  We don't actually count the activation functions and the max/ average pooling layers.

So, totaling this it gives us a 1 + 9 + 12 + 18 + 9 + 1 = 50 layers Deep Convolutional network.

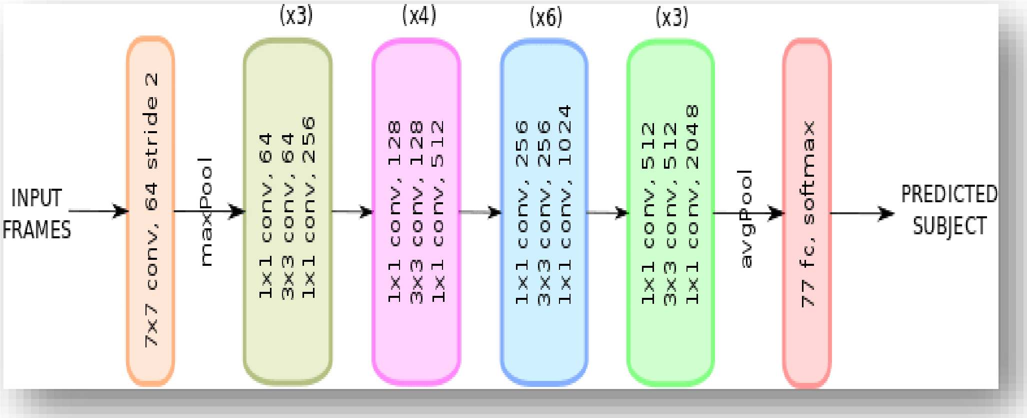


Figure 2.2: RESNET50 Architecture

CHAPTER 3

EXPERIMENTAL WORK

3.1. Materials and Procedure

In this section, we briefly describe the approach used to achieve the objectives of the study. The diagram of the proposed method is represented in Figure 3.1.

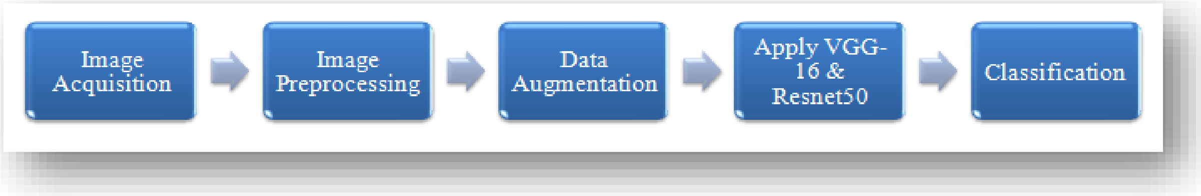


Figure 3.1: Block diagram of the proposed method.

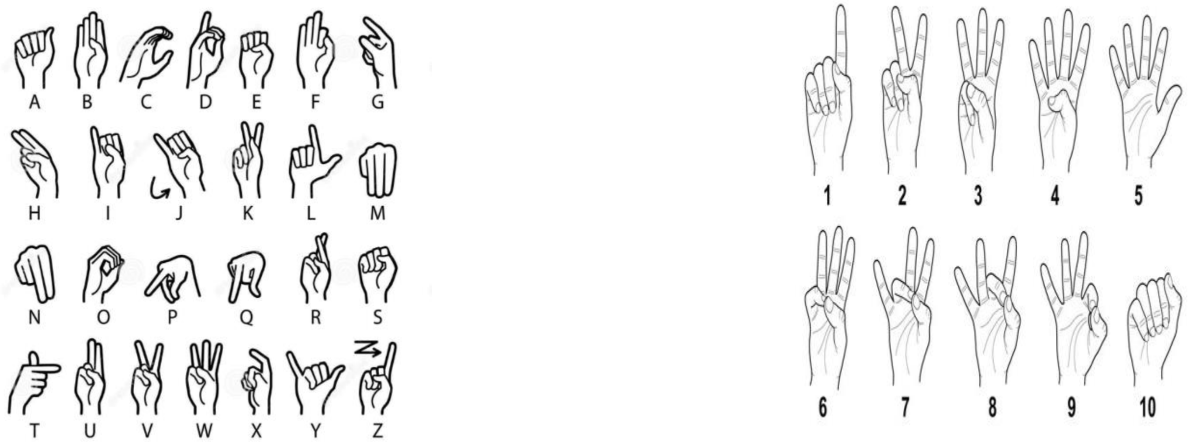
3.1.1 Dataset and Preprocessing

- The Sign language image dataset was downloaded from the given web site[1]. From this dataset we analyze 4608 images of sign images which is ASL of the English alphabet and numbers from 1 – 10 which have spread of 40 class labels assigned to them. Each class label is a set of sign images of the English alphabet and numbers.
- Below figures shows an example from every class of sign images dataset.

Data	ASL
Train	6584
Test	1384

Table 3.1: Summarized dataset for training and testing

[1] https://www.kaggle.com/datasets/grassknoted/asl-alphabet



(a) English alphabet

(b) Numbers

Figure 3.2: Samples of sign language Images used in this project

CHAPTER 4

TOOLS AND TECHNOLOGIES USED

- **Platform used for training and prediction: -**

  Jupyter notebook

- **Programming Language: -**

  Python 3

- **Libraries Used: -**

  TensorFlow 1.11.0

  Cv2

  NumPy 1.15.3

  Matplotlib 3.0.0

  Keras 2.2.1

CHAPTER 5
METHODOLOGY

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

There are several goals that we will accomplish in this chapter.

• Examine the process of extracting images from gesture videos.

• Obtain the results of the CNN with predicted labels and the output of the pool layer.

• Predict results.

5.1. Data Set Generation

For the project we decided to use a dataset that is available online on Kaggle link of the same is mentioned in page number 9.

- We analyze 6584 images of sign images which is ASL of the English alphabet, which have a spread of 40 class labels assigned to them. Each class label is a set of sign images of the English alphabet and numbers.
- All the images are resized to 640 x 480 pixels, and we perform both the model optimization and predictions on these downscaled images
-

5.2. Gesture Classification

Our approach uses layer of algorithm to predict the final symbol of the user.

Algorithm Layer 1:

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.

5.3. Training and Testing

In practice it’s hard to train a CNN from scratch, because it is rare to find a dataset to train it to the appropriate parameters. A solution to which is using a pretrained CNN as a starting point and use transfer learning. Several transfer learning scenarios exist such as:

1. CNN as fixed feature extractor: Using a CNN which is pretrained and removing the last fully-connected layer, then use the rest of the CNN as a fixed feature extractor for the new dataset. Once we retrain the CNN, we then use a classifier for the new dataset.
2. Fine-Tuning the CNN: The second strategy is to not only retrain the CNN but also modify the weights for the new dataset. It is possible to retrain every layer and
modify or keep existing layers. This is used when earlier training of the model has generic features and we need to fine tune for specific features in the new dataset.
3. Pretrained model: Since newer models of CNN take several weeks to train across GPU’s, it is common to use checkpoints for finetuning. Deciding which transfer learning to use depends on several factors such as size of the new dataset and its similarity to the original dataset. There are several rules which help you decide:

1. If the new dataset is small, you should not fine tune the CNN to avoid overfitting. It is a better idea to use higher level features instead of finer details.
2. If the new dataset is large and like the original training dataset then the CNN will have accuracy with good accuracy.
3. If the new dataset is small but also different from the original, using a linear classifier is the best idea, since CNN’s pick-up dataset specific terms.
4. If the new dataset is large and different, we might have to retrain the CNN from scratch.

We convert our input images (RGB) into grayscale. We feed the input images after preprocessing to our model for training and testing after applying all the operations mentioned above. The prediction layer estimates how likely the image will fall under one of the classes. So, the output is normalized between 0 and 1 and such that the sum of each value in each class sums to 1. We have achieved this using SoftMax function. At first the output of the prediction layer will be somewhat far from the actual value. To make it better we have trained the networks using labeled data. The cross-entropy is a performance measurement used in the classification. It is a continuous function which is positive at values which is not same as labeled value and is zero exactly when it is equal to the labeled value. Therefore, we optimized the cross-entropy by minimizing it as close to zero. To do this in our network layer we adjust the weights of our neural networks. TensorFlow has an inbuilt function to calculate the cross entropy. As we have found out the cross-entropy function, we have optimized it using Gradient Descent in fact with the best gradient descent optimizer is called Adam Optimizer.

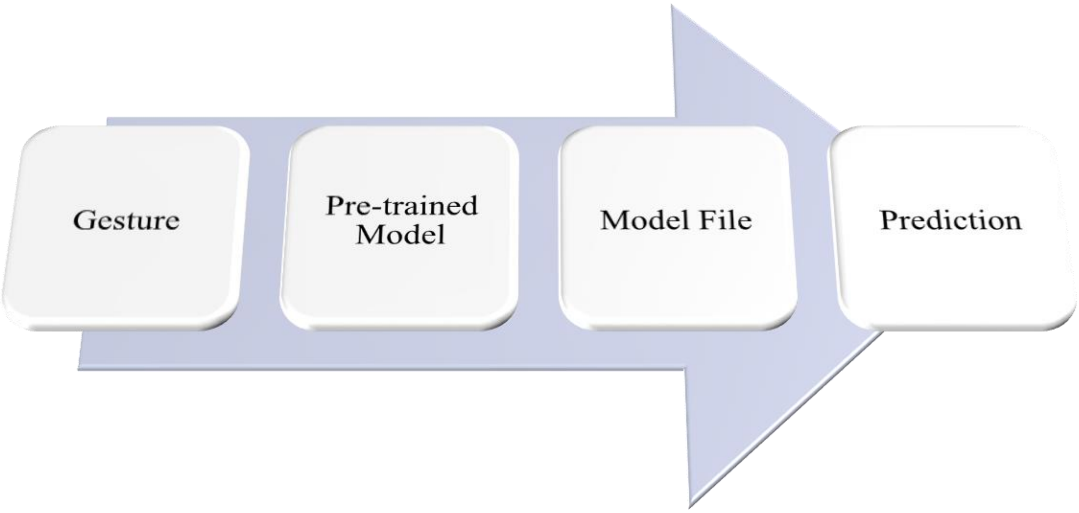


Figure 5.1: Model for predicting labels

CHAPTER 6

EXPERIMENTAL RESULTS

6.1. Results

In this project all the experiments are done using Python programming language on jupyter notebook with an AMD Ryzen 5 processor and 8 GB RAM on 64-bit windows 11 operating system.

6.1.1 VGG16

On training our model on 5 epochs we get –

- Model accuracy
  - Accuracy: 94.281%
- Model loss
  - Loss: 0.2021

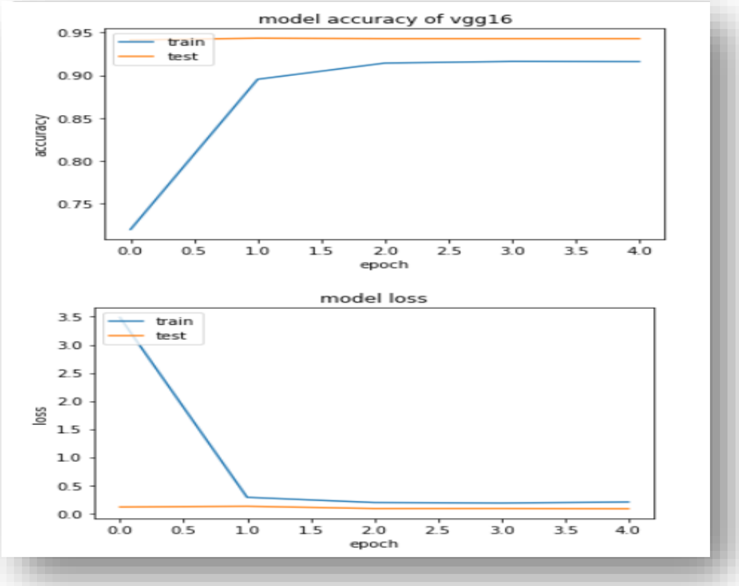


Figure 6.1: VGG16 Model Accuracy and Loss Curve

6.1.2 RESNET50

On training our model on 5 epochs we get –

- Model Accuracy
  - Accuracy: 93.269%
- Model loss
  - Loss: 0.4981

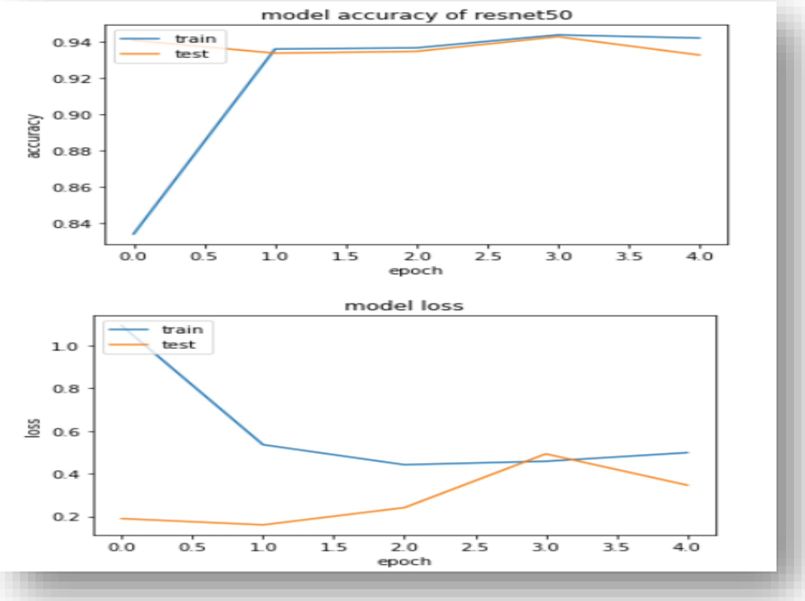


Figure 6.2: RESNET50 Model Accuracy and Loss

In our project, the VGG16 model is outperformed other models in the aspects of accuracy and loss.

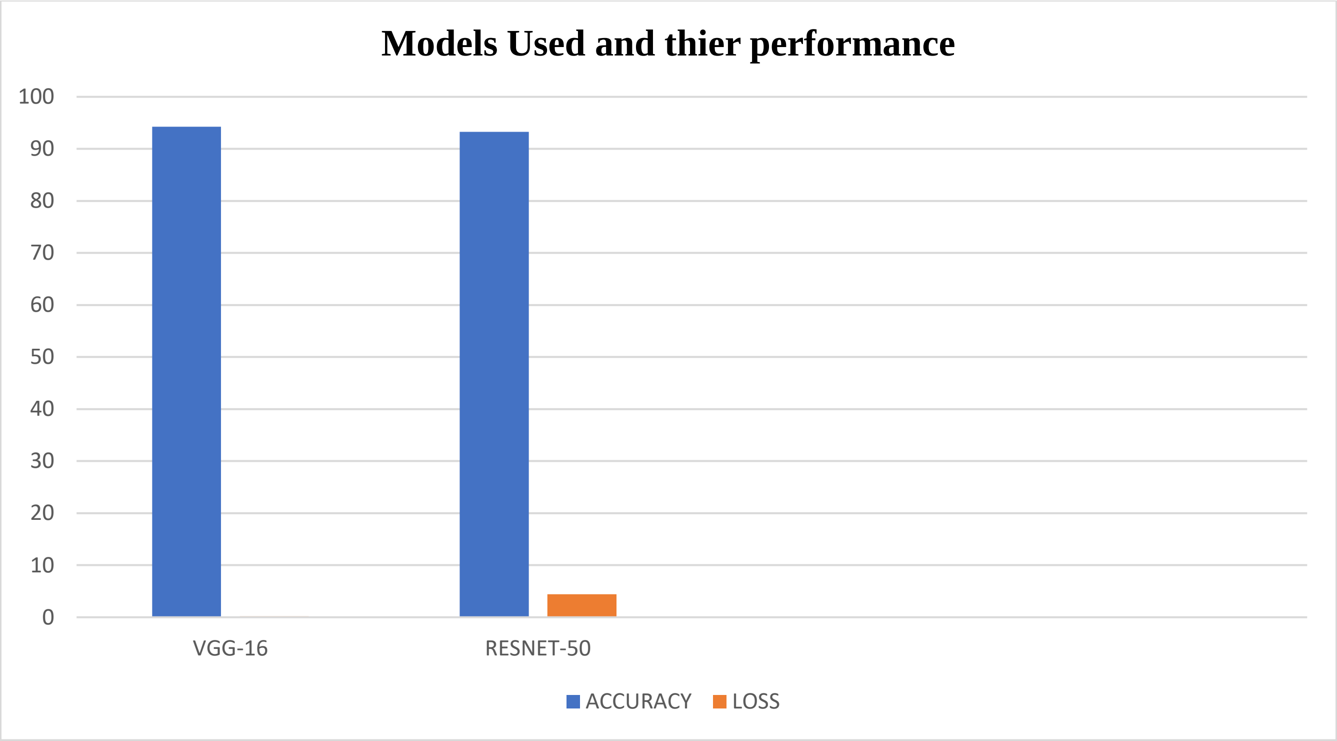


Figure 6.3: Performance of the used models

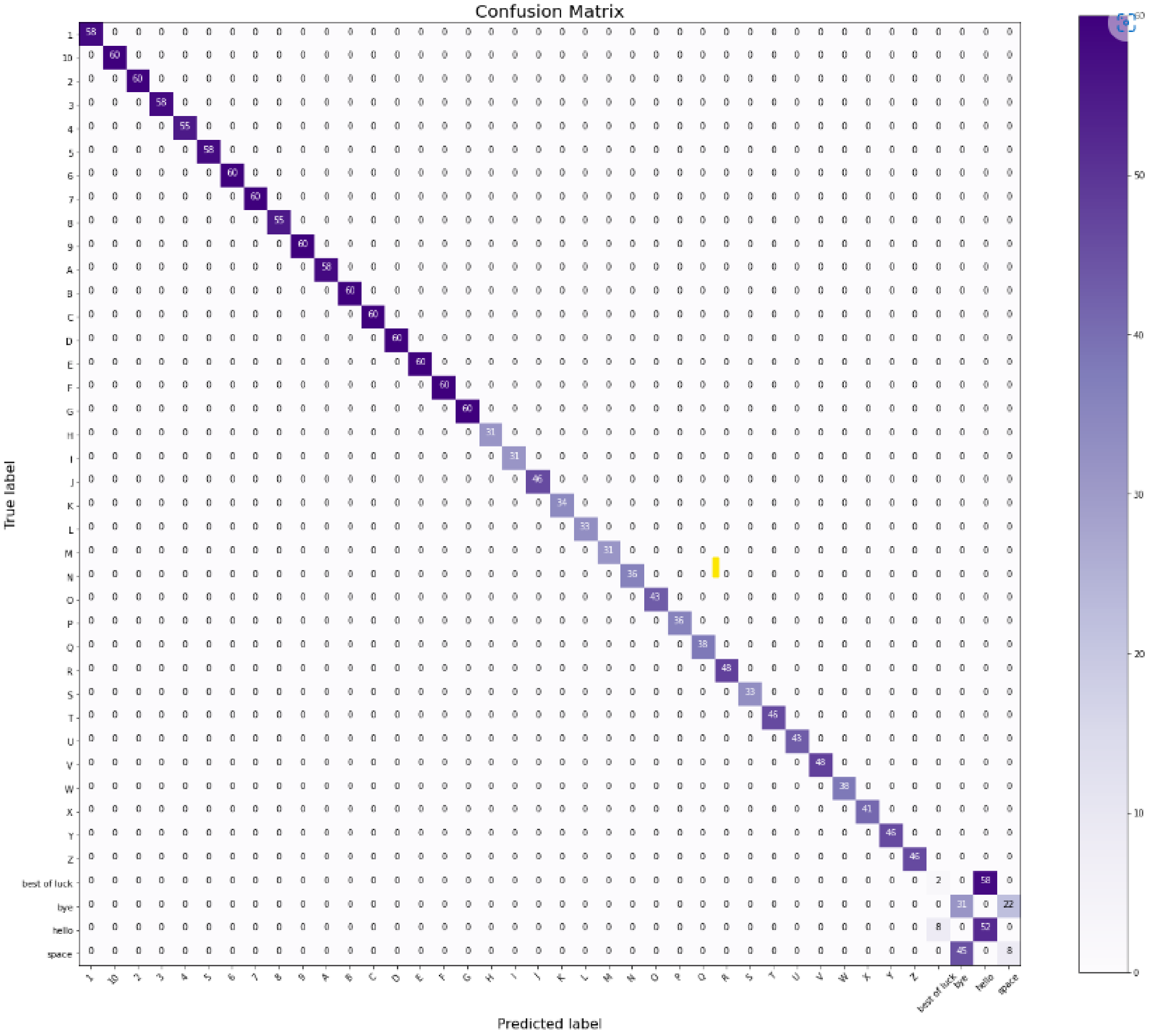


Figure 6.4: Confusion Matrix

CHAPTER 7

CONCLUSION

In this project, a functional real time vision based American sign language recognition for D&M people have been developed for asl alphabets and numbers. This proposed work ensures the accuracy of 94.281% using VGG-16 and 93.269% using Resnet50 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. In this project, we proposed an idea for feasible communication between hearing impaired and normal person with the help of deep learning approach.

REFERENCES

1. T. Yang, Y. Xu, and “A. , Hidden Markov Model for Gesture Recognition”, CMU-RI-TR-94 10, Robotics Institute, Carnegie Mellon Univ.,Pittsburgh,PA, May 1994.
2. Pujan Ziaie, Thomas M ̈uller , Mary Ellen Foster , and Alois Knoll“A Na ̈ive Bayes Munich,Dept. of Informatics VI, Robotics and Embedded Systems,Boltzmannstr. 3, DE-85748 Garching, Germany.
3. https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_bilateral_filter/gausian_median_blur_bilateral_filter.html
4. Mohammed Waleed Kalous, Machine recognition of Auslan signs using PowerGloves: Towards large-lexicon recognition of sign language.
5. aeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/
6. http://www-i6.informatik.rwth-aachen.de/~dreuw/database.php
7. 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
8. A. K. Jaiswal, P. Tiwari, S. Kumar, D. Gupta, A. Khanna, and J. J. P. C. Rodrigues, “Identifying pneumonia in chest X-rays: a deep learning approach,” Measurement, vol. 145, pp. 511– 518, 2019.