JES
Journal of Engineering Sciences

Vol 13, Issue 04, APRIL /2022

ISSN NO:0377-9254

Sign language Recognition using Convolution Neural Networks

Surya Samhitha Balusu[1], Anupoju Varaha Narasimha charyulu[2], Abhishek Samuel Daniel[3], Andhavaram Mythrayi[4]

#1#2,#3,#4 Student, Department of CSE, GITAM (deemed to be University), Gandhi nagar Rushikonda Visakhapatnam 530045 Andhra Pradesh, INDIA

ABSTRACT_ Almost 300 million individuals in the world are deaf or deafeningly deafeningly deaf. An American Sign language is a well-structured language that serves as a communication medium for people with disabilities. It is made up of hand motions, each of which represents a different alphabet and number. We created an application that uses this American sign language. Our proposed system employs convolution neural networks to create the program, which is capable of extracting hand motions and converting them into the appropriate symbols.

1.INTRODUCTION

Interacting with persons who are deaf or hard of hearing can be difficult. Deaf and mute persons utilize hand gestures to communicate, making it difficult for non-deaf and silent people to comprehend them. As a result, systems that can recognize a wide range of signs and transmit the data to regular people are necessary.

Sign languages are classified into several kinds:

- American Sign Language (ASL)
- Indian Sign Language (ISL)
- Chinese Sign Language (CSL)
- French Sign Language (LSF)

In our project, we use American Sign Language (ASL), which has 26 alphabets. The American Sign Language (ASL) is a language that differs significantly from English. In addition to all of the basic features of language, it has its own rules for pronunciation, word creation, and word order. In order to communicate properly, we need a model that can help ordinary people interpret the signing correctly. The method for converting sign language to alphabets is straightforward, efficient, and precise. The primary goal of the study is to develop a model that aids in the recognition of hand movements. [3]

Fig 1.1 American -sign language

1.1 Deep learning algorithms for object detetction:

The following are the most often used deep learning models for detecting objects:

R-CNN model family: It stands for Region- based convolution Neural Network.

a. **R-CNN**: It obtains objects from images using a process called selective search. The primary function of this algorithm is to create large regions and work on them concurrently for classification of an object.

b. **Fast R-CNN:** In order the overcome the drawbacks of the R-CNN algorithm, the approach was changed to feed the input image instead of acquiring a feature map, submit region recommendations to CNN. The feature map is used to identify the region suggestions, which are then distorted into squares. Finally, they are reshaped into predetermined sizes using a RoI layer before being delivered to the fully connected layer for categorization.

c. **Faster R-CNN:** The technique of obtaining region ideas differs significantly between Fast R-CNN and Faster R-CNN. The first procedure employs selective search, which is a time-consuming and sluggish process method that hinders performance and efficiency, while the latter uses a separate network to identify region proposals. After obtaining the region proposals the rest of the process is similar to its predecessor.

d. **YOLO:** The way You only look once works is by splitting the image into a one dimensional matrix/grid, and for every grid in the matrix we take a m bounding boxes. Each bounding box is assigned a probability. And the entire grid contains a number of probabilities. To find the item within the image, a bounding box with class probability greater than a given threshold value is utilised. The main advantage of YOLO is that it is much faster (45 frames per second) than the previous classification method.

1.2 Convolution neural networks:

A multilayered neural network (CNN) is mostly utilised in image identification and classification that works on pixels data. It extracts features from the images and to predict hand gestures accurately. It has 3 layers, although it may have one or more convolution layers, dense layer ,max pooling layers, fully connected layers, and so on.[1]

a.Convolution Layer:

The very first layer is the Convolutional layer, which is made up of a variety of learnable filters (kernels or features). In order to determine patterns, these processing techniques are applied to the full input image. Convolution is achieved by dragging the filter over the source image and calculating the filter's dot product and parts of the input picture. Firstly, consider the size of the extracted tiles (typically 3x3 or 5x5 pixels. The second consideration is the level of the output feature map, which is dependent on the number of filters utilised.

During coaching, the CNN "learns" the best filter matrices' parameters, allowing it to recover important features (textures, boundaries, and shapes) from the input feature map. As the filter size (final feature map depth) applied to the input rises, along with the CNN's potential to extract a significant number of features. Yet, because filters account for the vast majority of CNN resources, training time doubles as more filters are applied. Further, each new a network filter has been added offers less additional value more than the prior one, therefore it's ideal to create networks with as few filters as possible to extract the attributes required for correct picture classification.

ReLU (activation function) is executed after every Convolution. Rectified Linear Unit is an abbreviation for Rectified Linear Unit, which is a non-linear operation. ReLU is a pixel-by-pixel method that substitutes all negative feature map pixel values with zero. Because much of the real-world data we'd like our ConvNet to learn is non-linear, ReLU brings non-linearity into ConvNet. Different non-linear functions, including tanh or sigmoid, can be substituted with ReLU, but in most circumstances, ReLU performs better. [1]

b. Max pooling layer:

The very next stage in the CNN is the Pooling layer.The Pooling layer reduces feature maps by summarising sub- regions with operations such taking the average or highest value. Pooling is accomplished by dragging a window across the input and feeding the contents of the window to a pooling function. Pooling has the purpose of reducing the amount of parameters in our network (down-sampling). Max pooling works in the same way as convolution does. We drag a slider across the feature map to obtain a set of tiles of a certain length. For each tile, the highest value is transferred to a new feature map, whilst other entries are deleted.[1]

c. Fully connected layer:

It is completely interconnected with the previous layer's output, which is the pooling layer. Ever other neuron in the preceding layer are linked to every neuron in the completely connected layer (whether fully connected, pooling, or convolutional). It’s job is to classify data using the features collected by convolutions. In the last fully linked layer a soft max input function is commonly used, which offers a statistical significance ranging from 0 to 1 for each of the categorization labels the model is attempting to predict.[1]

e.Output layer:

This layer gives the desired result from the input of the hidden layers. The desired result will be in the form of numerals.[1]

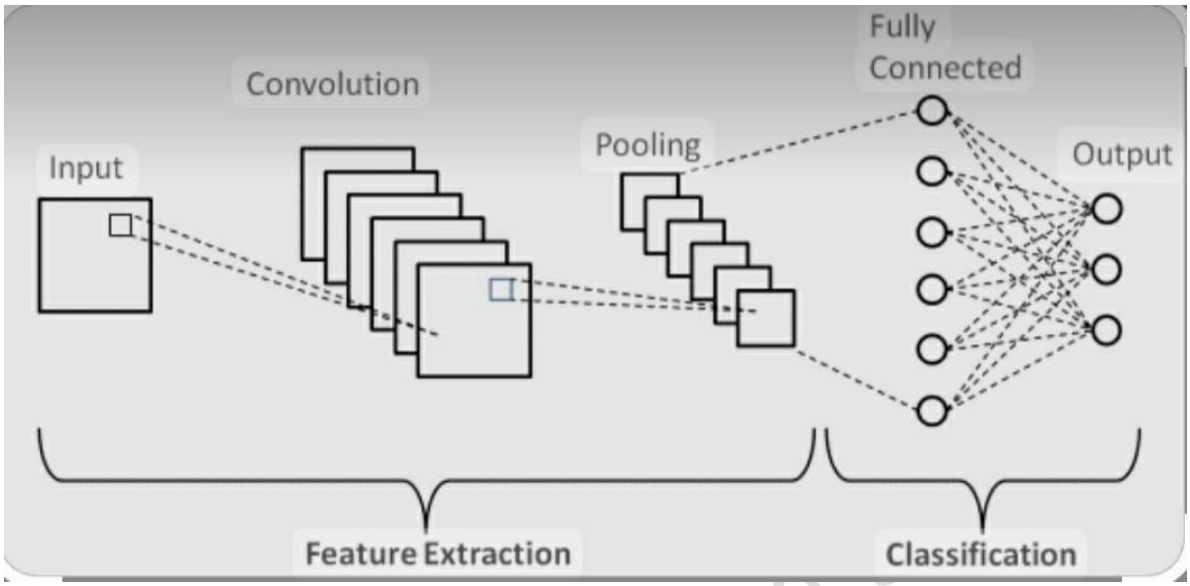


Figure 1.2 CNN Architecture

2.LITERATURE SURVEY

[1] Arvind Sreenivas, Mudit Maheshwari, Saiyam Jain, Shalini Choudhary, and Dr.G.Vadivu accomplished this in this study by recognising individual alphanumeric characters and appending them to make phrases. This paradigm has high scalability, and its future ramifications might be significant.

[2] The goal of this paper by A.Sunitha Nandhini, Shiva Roopan. D, Shiyaam S, and Yogesh. S is to understand the advantages that will be learned by a system known as convolutional neural networks (CNN). It is around 90% accurate.

[3] Md. Mehedi Hasan, Azmain Yakin Srizon,Abu Sayeed, and Md. Al Mehedi Hasan designed a model of deep convolutional neural networks to successfully identify sign language alphabets. They got a total accuracy of 97.62 percent after deploying the model.

[4] Diksha Hatibaruah, Anjan Kumar Talukdar, and Kandarpa Kumar Sarma created the image's histogram using the Back projection Histogram method. After training with CNN, they observed that at 5 epoch, the measure efficiency was 99.89 percent and the validation accuracy was 99.85 percent.

[5] Convolutional Neural Networks were utilised to create Sign Language by Prangon Das, Tanvir Ahmed, and Md. Firoj Ali. To construct and test their model, they utilised the ASL dataset, which comprises 1815 pictures of the 26 English alphabets. The model's precision was determined to be 94.34 percent.

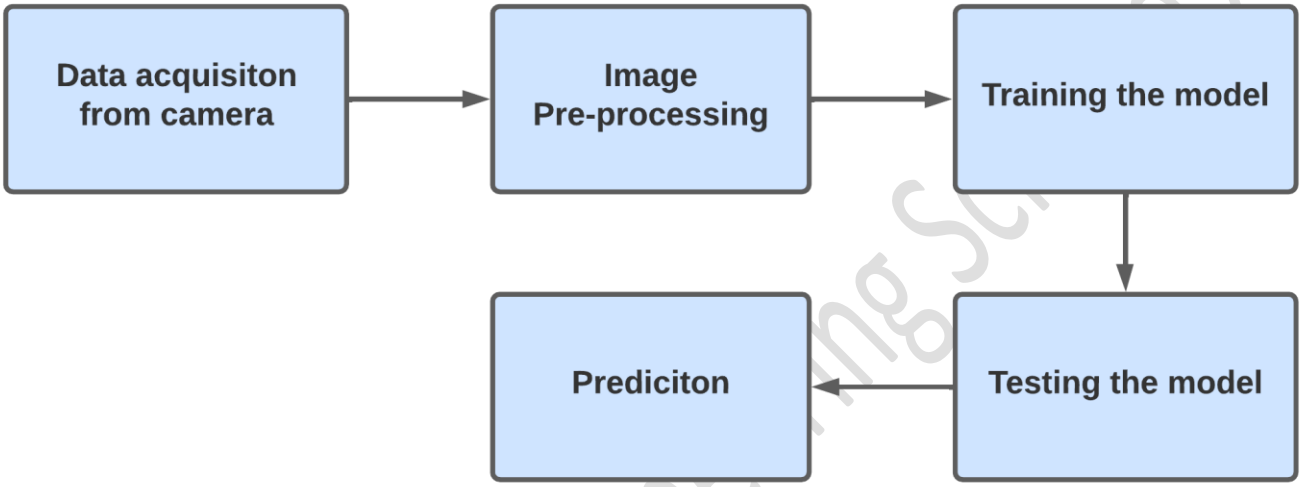
[6] Aditya Das, Shantanu Gawde, Khyati Suratwala, and Dr. Dhananjay Kalbande examine the numerous attempts at sign language identification employing machine learning and depth data of pictures in this work.

[7] Victor Murray, Cristian Amaya recognise sign language, they employ principal component analysis (PCA) and all support vector machines (SVM) classification. More than 80% accuracy in tests is achieved.

3.PROPOSED SYSTEM

In our proposed system, we used a Convolution Neural Network to develop sign language in order to solve difficulties with room lighting and unrelated object exclusion. Block Diagram for our proposed system seen in Figure 3.1.[4]

Figure 3.1 Block Diagram



A.Data Acquisition:

For the implementation of the system, the foremost step is collecting a dataset. For this we created our own dataset of about 1750 images for each alphabets, to avoid the problem of overfitting due to the similarities between the hand gestures. We turn on the web camera to capture as many images of the hand gestures in real time as accurately as possible. After that, each input is saved in its own directory and then processed using a variety of image preprocessing methods. Here we have taken the image size to be (64, 64).[4]

B.Image Preprocessing:

On the data acquired from step 1, skin filtering algorithms are used by the system to differentiate between skin and non-skin regions like the background. We converted the images from RGB scale to HSV. This was done because the HSV colour scheme was less sensitive to lighting changes than the RGB colour space. As we need to consider the illumination in the environment , we have used trackbars to regulate the lights. After that, all of the pictures in the dataset are resized to 64,64 pixels in order to be used as input to the CNN model.[2]

Figure 3.2 Preprocessing output

C.Training the model: The Preprocessing image of size (64,64) from the previous step are sent to the Convolutional Neural Network (CNN) which extracts the most important features from the hand gestures. A maximum of 3 convolutional layers and 3 pooling layers were used in this experiment. In the convolution layer, the ReLU activation function was used. 32 convolutional filters were employed in the initial two convolutional layers, which were increased to 64 in the subsequently one convolutional layer to obtain more deep features in the pictures. In the pooling layer, a 2*2 size filter was applied. Finally, the image characteristic was transferred to the completely linked layer. In the hidden layer, the ReLU activation function was also used. The unseen layer included two fifty six neurons that reimplemented the transfer between the fully connected layer's inputs and the output layer's outputs. As in the classification, the output layer contains a total of twenty four nodes with the softmax activation function. Now we'll train the model and assess how accurate it is by comparing it to the test set.[3]

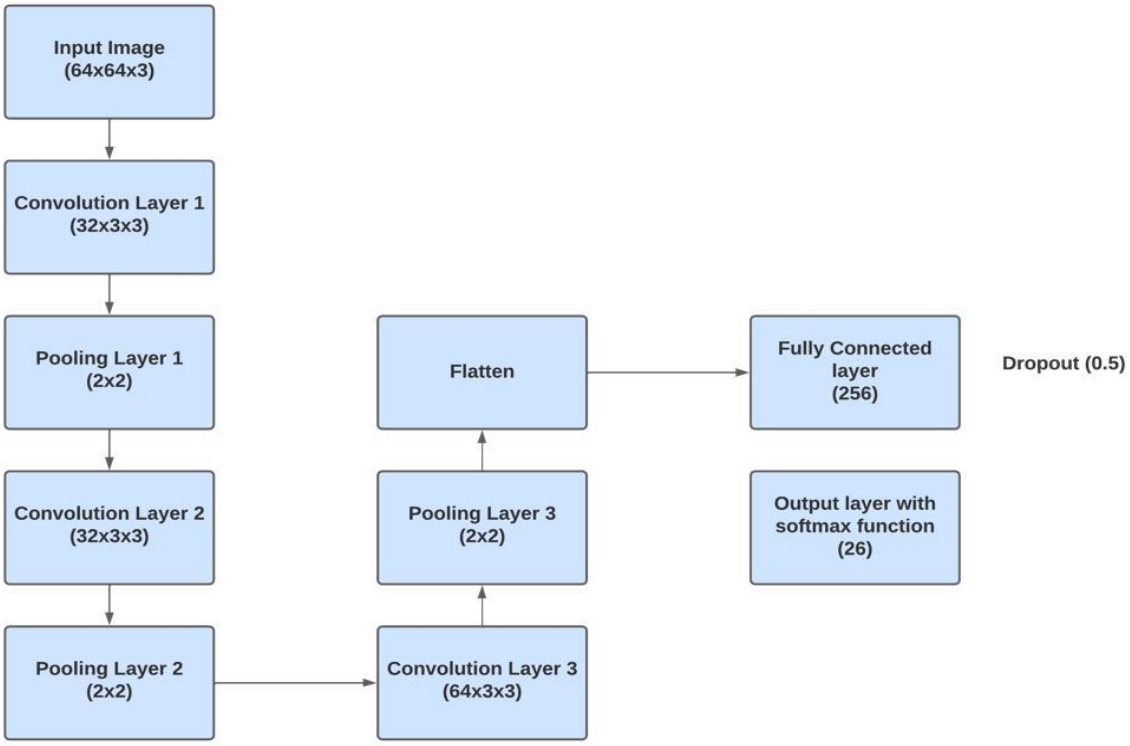


Figure 3.3 CNN model of our proposed system

D.Testing:

Once the training is done, we are testing in real time through web cam in various aspects.

E.Prediction: We give real-time hand gestures and check whether the model predicts the letter to the corresponding hand gesture, and then displays in form of labelled text.

4.EXPERIMENTAL SETUP

We have implemented this application using python and its libraries. The application is set up to collect frames from the web camera's which will be subjected to a variety of image processing algorithms. Next, the images are preprocessed utilising image preprocessing technique such as RGB to HSV (Hue,Saturation,Value) conversion, which eliminates the skin tone's performance fluctuation. Furthermore, the hand phase is retrieved from the image, as well as the hand sign. After that, the images are compared to the taught model. Such captured images are saved in the input folder. The alphabets of hand gestures are included in this dataset. There are around 1750 photos per category, i.e. one for each letter of the alphabet. After that images are then processed using the CNN module of the tensorflow module, with the sequential model. We used 65 epochs in this training. The model creates a file of .h5 that contains a summary of the training output. This file will be used to determine alphabets according to the model. Finally, input picture is sent to the CNN model for prediction, which compares the input and recorded images. In comparison of the input and recorded images, the CNN model produces textual output[2].

5. TESTING:

We have tested the application in numerous scenarios such as :

Test Case 1: Hand gesture detection
Expected outcome: The system should be able to recognise the hand.
Obtainable outcome: The system was able to identify the hand when we moved it in front of the webcam.

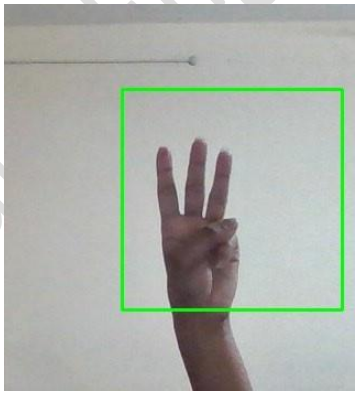


Figure 5.1 Test case for identifying the hand

Test Case 2: To identify all of the alphabetical hand gestures (A-Z):

Expected outcome: The system should be able to recognize all of the alphabets hand motions (A-Z).

Obtainable outcome: We provided with all of the alphabet hand gestures (A-Z), and the system correctly identified a ll of them (A-Z).

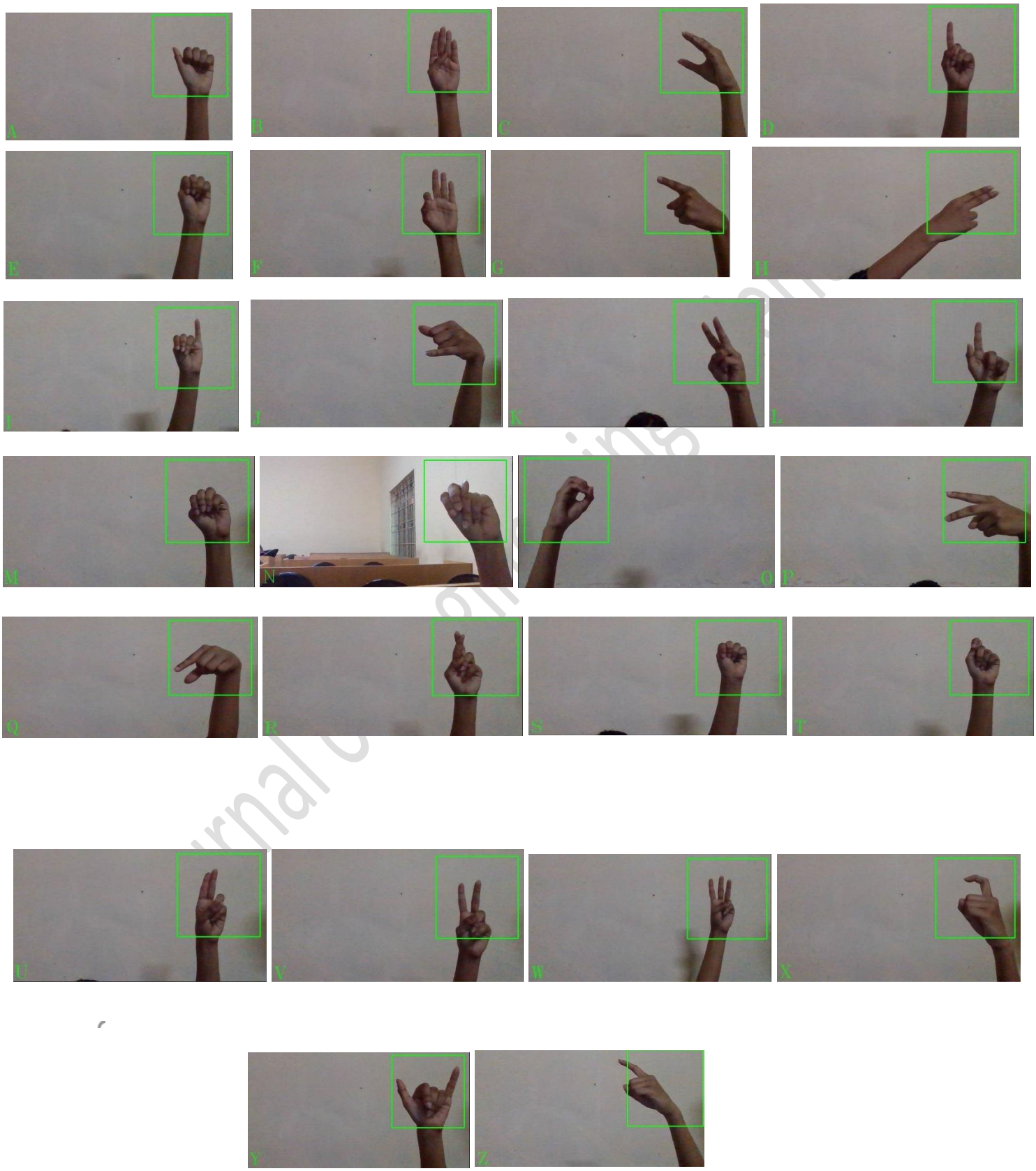


Figure 5.2 Test Case for identifying the alphabet hand gestures

Test Case 3: To detect the hand gesture in a bright light environment:

Expected outcome: The system should be able to identify the hand gestures in a bright light environment

Obtainable outcome: We displayed the hand motions in a bright-light situation, and the system was able to recognize the gestures quickly.

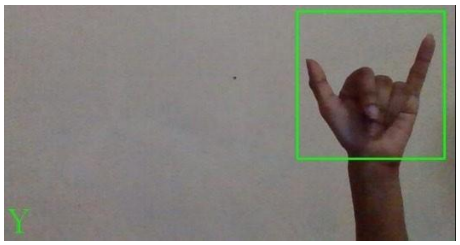
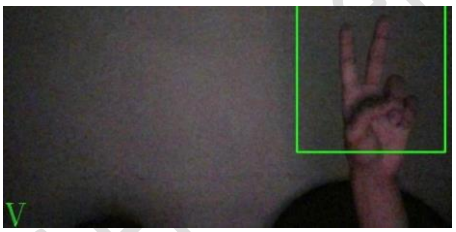


Figure 5.3 Test Case for recognizing hand gestures in a bright light environment.

Test Case 4: To detect the hand gesture in a low light environment.

Expected outcome: The system should be able to identify the hand gestures in a low light environment



Obtainable outcome: We displayed the hand motions in a low-light situation, and the system was able to identify the gestures quickly.

Figure 5.4 Test Case for identifying the hand gesture in a low light environment.

Test Case 5: To detect the presence of a hand while there is an objects/human in the backdrop.

Expected outcome: The application should be able to detect the hand when there is an object/human at the backdrop.

Obtainable outcome: The software recognized the correct gesture once we placed the object/human against the backdrop.

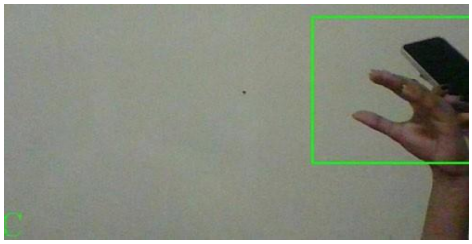


Figure 5.5 Test Case for identifying the hand gesture when an object/human is present at the backdrop

Test Case 6: To detect the hand at a range of distances away from the camera.

Expected outcome: The model should be able to recognise a hand gesture at a range of distances away from the camera.

Obtainable outcome: The hand was placed at several distances from the webcam, and the model was able to detect up to a distance of 0.84 meters approximately.

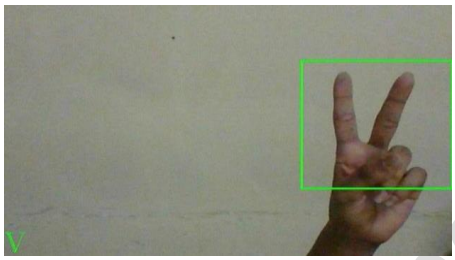


Figure 5.6 Test Case to check whether the system is identifying gestures from multiple distances or not

Test Case 7: To detect the hand at different angles.

Expected outcome: The model should be able to recognise a hand gesture in different angles.

Obtainable outcome: The application did not identify some of the alphabets when seen from various angles.

Figure 5.7 Hand gesture detection in different angles

6.RESULTS ANALYSIS:

The Convolution neural network model for sign language detection has been trained in 65 epochs and obtained an trained accuracy of about 99.4%.[3]

Training accuracy of cnn model	99.4%
Training loss of cnn model	1.7%
Test accuracy of cnn model	99.0%
Test loss of cnn model	0.6%

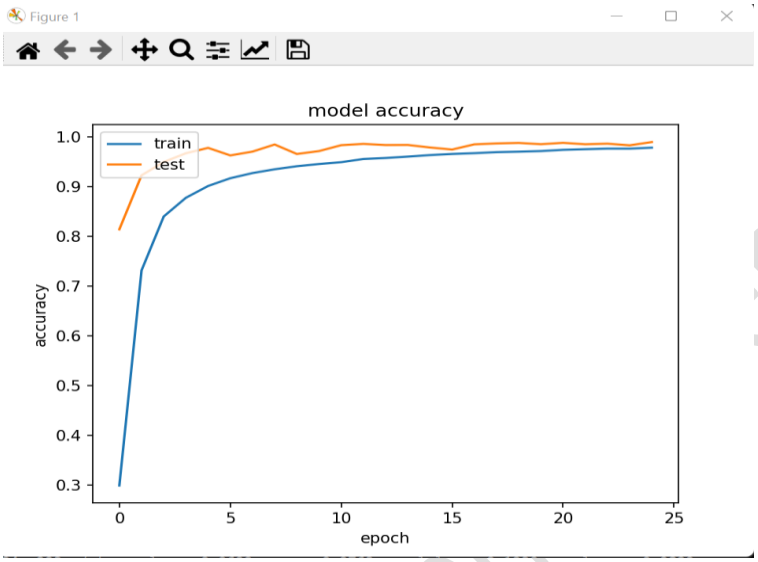


Figure 6.1 Train and validation accuracy of our convolution neural network model

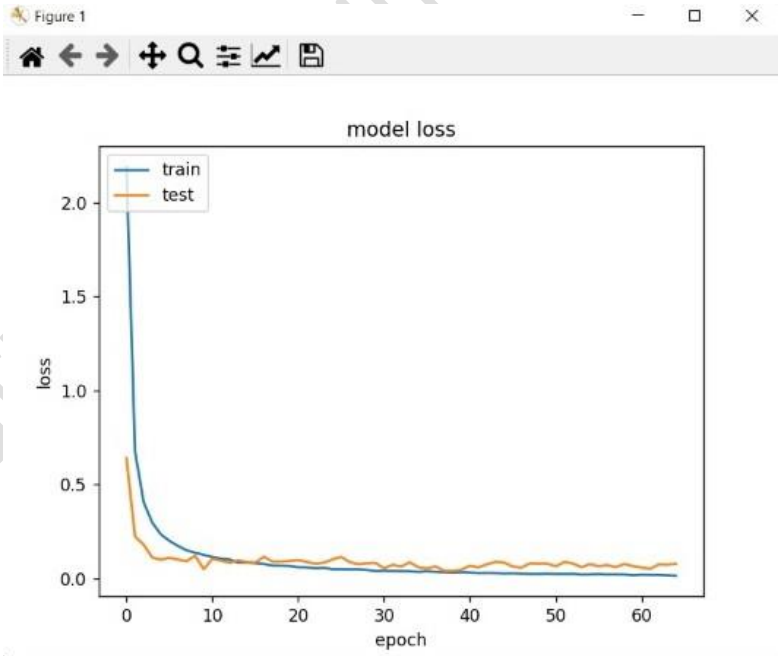


Figure 6.2 Loss of train and validation for our convolution neural network model

7.CONCLUSION

Using this convolution neural networks deep learning approach, we developed a technique for detecting sign languages based on American Sign Language in our proposed system. We built

the application with python and its libraries. The application can capture images and assign them to appropriate alphabets (A-Z). Using this method, we built an image dataset of 1750 photos containing hand gestures. The images are trained using a convolutional neural network model. The application is evaluated in several areas, including hand detection and changes in angle, distance from the view, and illuminance in the surrounding environment. In the majority of cases, we are able to obtain appropriate results.[4]

8.FUTURE SCOPE

The following improvements can be done to our project:
1.Calculating the accuracy values of individual characters.
2. Improvising the detection in terms of angular movements in real time.
3.Inclusion of numerical signs in object detection.
4.Converting a series of hand gestures into natural language sentences.
5.Voice conversion from a series of hand gestures.

9.REFERENCES

[1]Arvind Sreenivas, Mudit Maheshwari, Saiyam Jain, Shalini Choudhary, and Dr.G.Vadivu (July 2020). Convolutional Neural Network-Based Indian Sign Language Communicator .

[2]A Sunitha Nandhini, Shiva Roopan D, Shiyaam S., and Yogesh S. (2021). Convolutional Neural Network for Sign Language Recognition.

[3] Md. Mehedi Hasan,Azmain Yakin Srizon,Abu Sayeed,Md. Al Mehedi Hasan (November 2020). Classification of Sign Language Characters by Applying a Deep Convolutional Neural Network.

[4] Diksha Hatibaruah, Anjan Kumar Talukdar, Kandarpa Kumar Sarma (2020). A Static Hand Gesture Based Sign Language Recognition System using Convolutional Neural Networks.

[5] Prangon Das,Tanvir Ahmed,Md. Firoj Ali (June 2020). Static Hand Gesture Recognition for American Sign Language using Deep Convolutional Neural Network.

[6] Aditya Das, ShantanuGawde,KhyatiSuratwala,Dr.Dhananjay Kalbande (2020).Sign Language Recognition Using Deep Learning on Custom Processed Static Gesture Images.

[7] Cristian Amaya,Victor Murray (2020). Real-Time Sign Language Recognition.