Hand Gesture Recognition

Peddemoni Dhanush Yadav (B19BB032)
Prince Gupta (B19BB033)
Riddhi Sera (B19BB036)

INTRODUCTION:

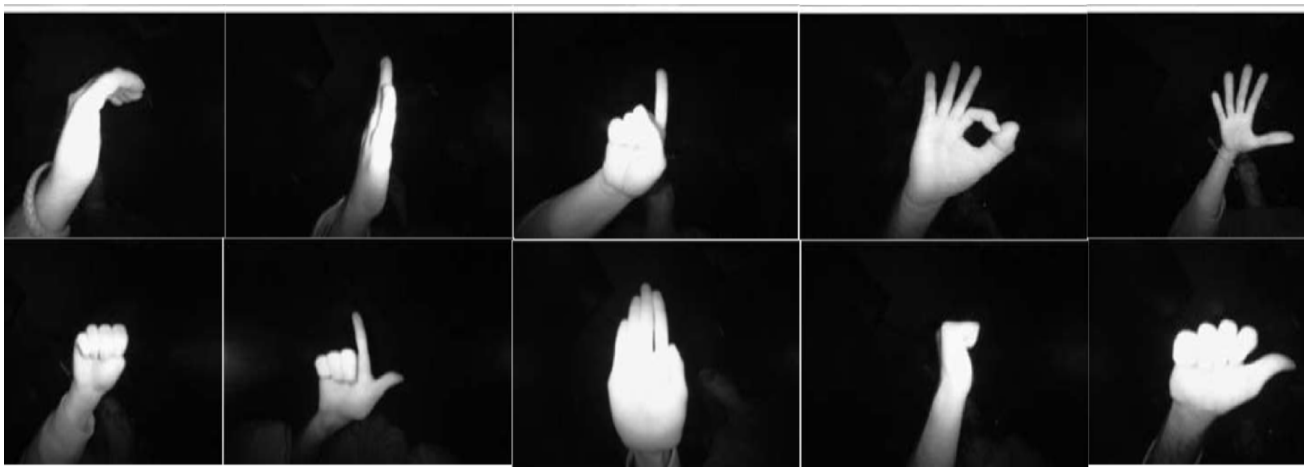
We humans can easily recognize signs and body languages easily. We do this with the help of vision and neuronal interactions that were formed when our brain was evolving with us. However, to replicate this process in computers, we need to solve some problems like how to separate objects of interest in images and which image capturing technology to use and which classification techniques to use. Our project proposes a gesture recognition method

using Convolution Neural Network (CNN) to recognize different kinds of static hand gestures.

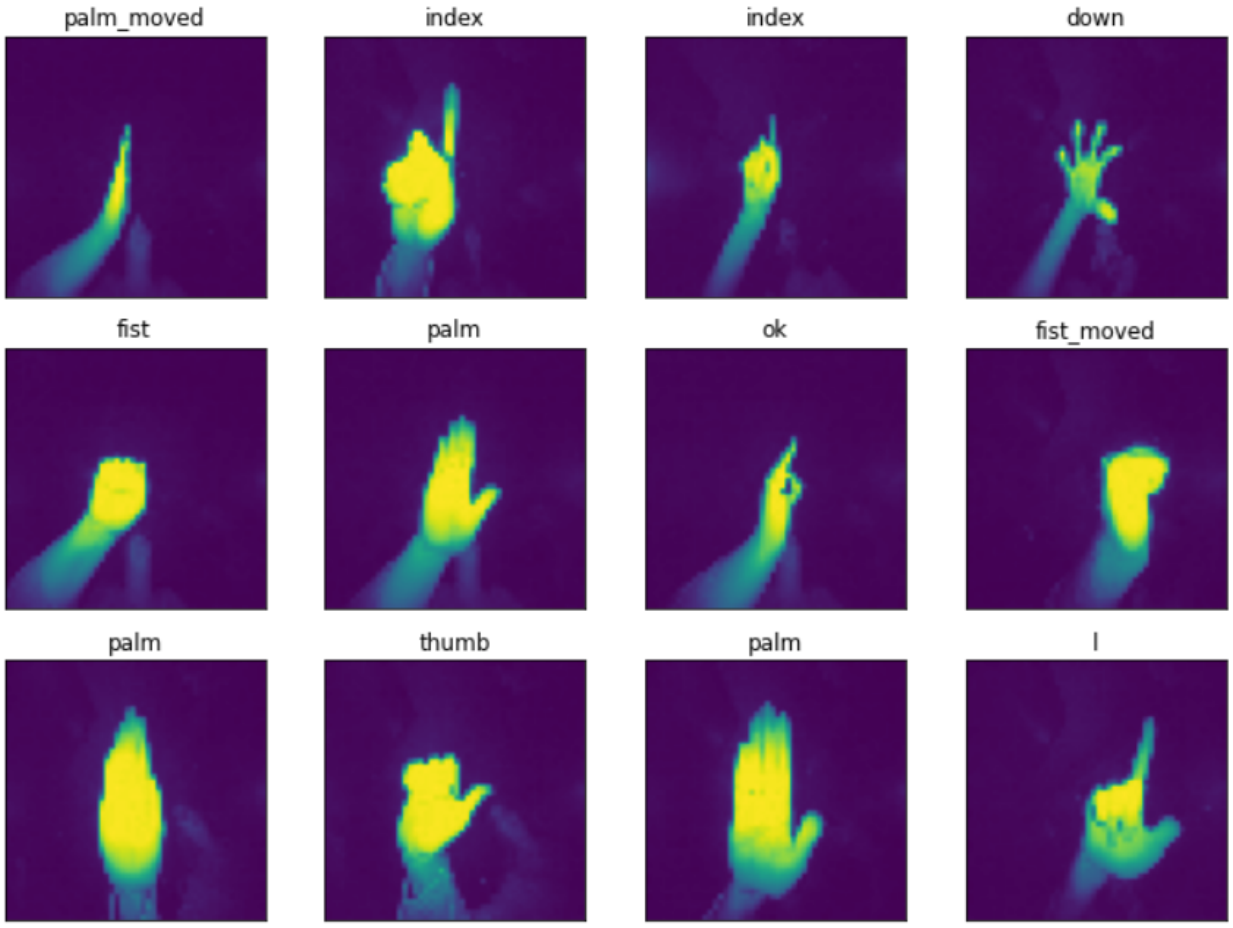
Data Visualization:

Name : Hand Gesture Recognition Database

- The Hand gesture recognition database that we have used is made up of a set of near infrared images acquired by the Leap Motion sensor.
- The database is composed of 10 different hand-gestures that were performed by 10 different subjects (5 men and 5 women).
- There are 10 categories in total, palm, I gesture, fist, fist_moved, thumb, index, ok, palm_moved, c and down
- This dataset contains around 20,000 images made up of 10 different hand gestures.
- The dataset which is over 2 GB contains 10 different folder named as 00, 01..., 09.
- Each of these folders is made up of 10 subfolders named over a respective category that we want to classify and contain images corresponding to that specific category.



The following image shows some sample training images:



- We first divided our dataset into a 70:30 ratio for training and testing. As a result, we used 14,000 images for our training dataset and 6,000 images to make the testing dataset.

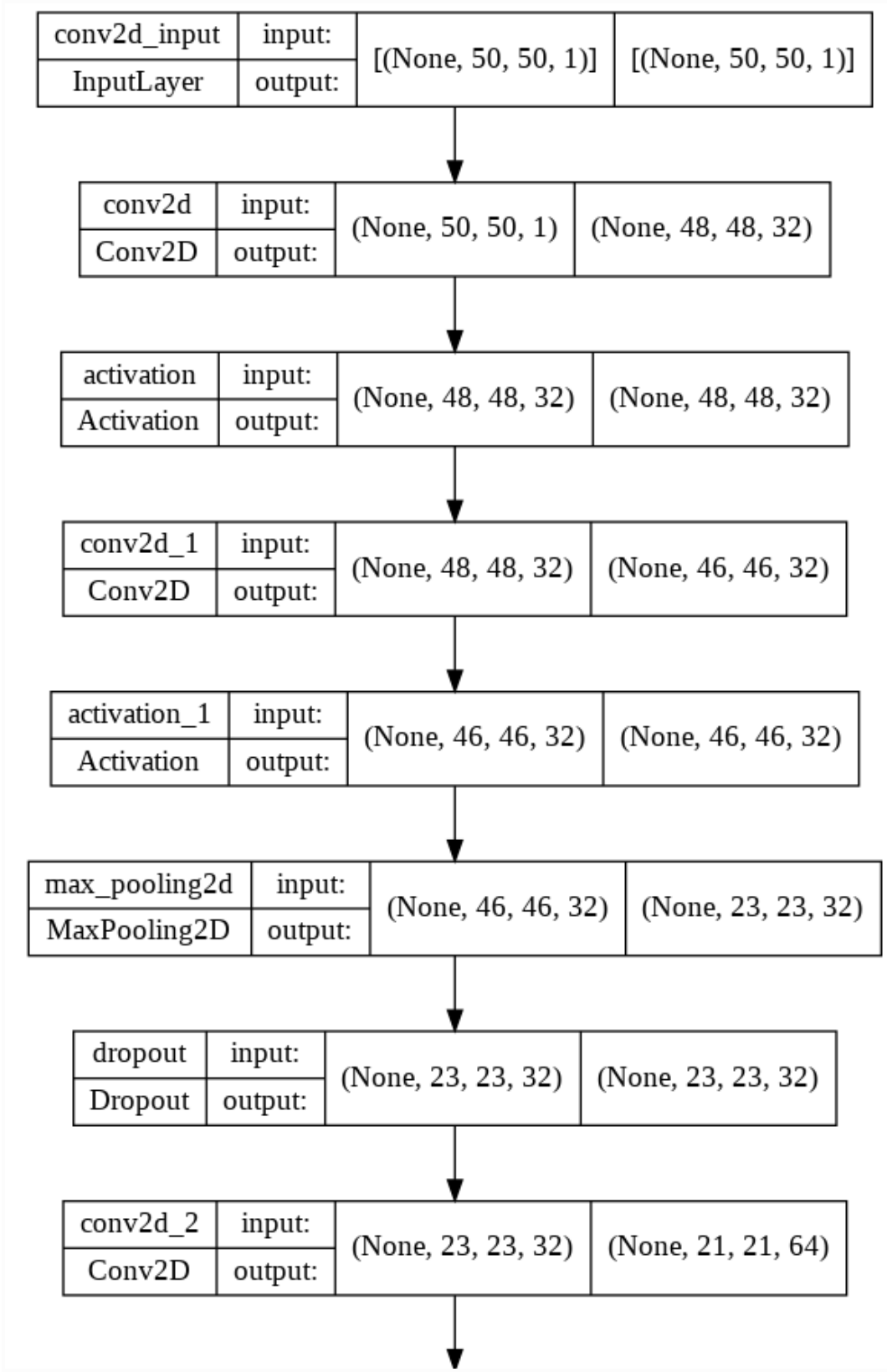
Procedure:

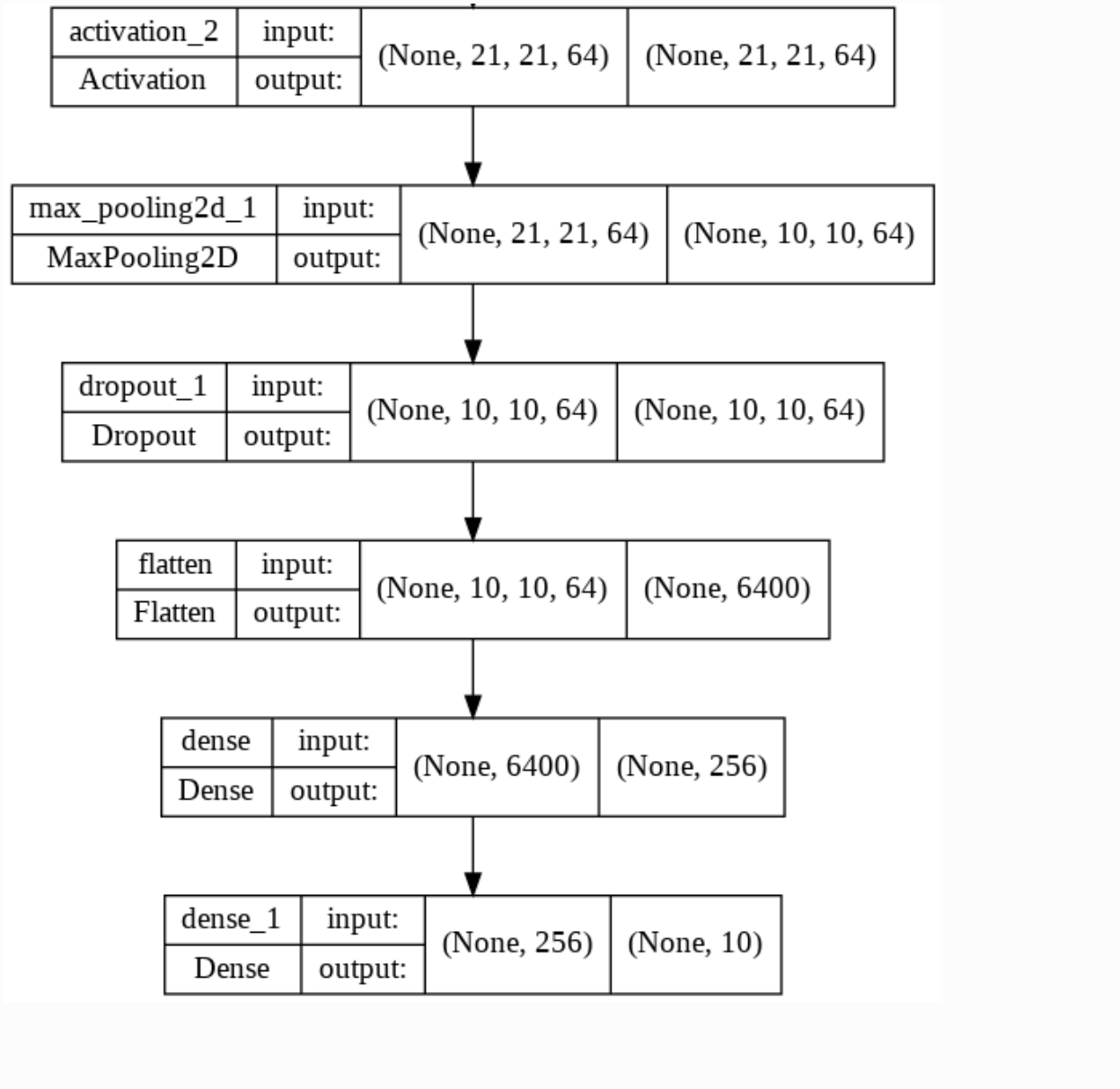
We first imported the data from our google drive and unzipped it. Then we loaded the dataset in an array using the 'os' module.

We preprocess the data in the following way**:**

1. We normalized the entire image dataset by dividing the values with 255.0
2. We used one-hot encoding to encode all the 10 labels
3. Then we reshaped the data into (-1, 50, 50, 1) size. This image size will be fed in our CNN model

Then we designed a CNN model with the following architecture**:**


conv2d_input
InputLayer
input:
output:
[(None, 50, 50, 1)]
[(None, 50, 50, 1)]
conv2d
Conv2D
input:
output:
(None, 50, 50, 1)
(None, 48, 48, 32)
activation
Activation
input:
output:
(None, 48, 48, 32)
(None, 48, 48, 32)
conv2d_1
Conv2D
input:
output:
(None, 48, 48, 32)
(None, 46, 46, 32)
activation_1
Activation
input:
output:
(None, 46, 46, 32)
(None, 46, 46, 32)
max_pooling2d
MaxPooling2D
input:
output:
(None, 46, 46, 32)
(None, 23, 23, 32)
dropout
Dropout
input:
output:
(None, 23, 23, 32)
(None, 23, 23, 32)
conv2d_2
Conv2D
input:
output:
(None, 23, 23, 32)
(None, 21, 21, 64)



The summary of the architecture looks like**:**

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 48, 48, 32)        320

activation (Activation)      (None, 48, 48, 32)        0

conv2d_1 (Conv2D)            (None, 46, 46, 32)        9248

activation_1 (Activation)    (None, 46, 46, 32)        0

max_pooling2d (MaxPooling2D  (None, 23, 23, 32)        0
)

dropout (Dropout)            (None, 23, 23, 32)        0

conv2d_2 (Conv2D)            (None, 21, 21, 64)        18496
```

```
activation_2 (Activation)   (None, 21, 21, 64)        0

max_pooling2d_1 (MaxPooling  (None, 10, 10, 64)       0
2D)

dropout_1 (Dropout)         (None, 10, 10, 64)        0

flatten (Flatten)           (None, 6400)              0

dense (Dense)               (None, 256)               1638656

dense_1 (Dense)             (None, 10)                2570

=================================================================
Total params: 1,669,290
Trainable params: 1,669,290
Non-trainable params: 0
_________________________________________________________________
```

We have a total of 1,669,290 trainable parameter

Hyperparameters:

Epoch	Batch size	Optimizer used	Loss function
7	32	RMSprop	Categorical cross entropy

Training phase:

```
Epoch 1/7
438/438 [==============================] - 112s 252ms/step - loss: 0.3409 - accuracy: 0.8887 - val_loss: 0.0209 - val_accuracy: 0.9938
Epoch 2/7
438/438 [==============================] - 110s 252ms/step - loss: 0.0206 - accuracy: 0.9936 - val_loss: 0.0064 - val_accuracy: 0.9987
Epoch 3/7
438/438 [==============================] - 110s 252ms/step - loss: 0.0103 - accuracy: 0.9974 - val_loss: 0.0046 - val_accuracy: 0.9992
Epoch 4/7
438/438 [==============================] - 112s 256ms/step - loss: 0.0048 - accuracy: 0.9986 - val_loss: 0.0063 - val_accuracy: 0.9993
Epoch 5/7
438/438 [==============================] - 111s 254ms/step - loss: 0.0042 - accuracy: 0.9989 - val_loss: 0.0074 - val_accuracy: 0.9992
Epoch 6/7
438/438 [==============================] - 111s 253ms/step - loss: 0.0048 - accuracy: 0.9988 - val_loss: 0.0039 - val_accuracy: 0.9992
Epoch 7/7
438/438 [==============================] - 111s 253ms/step - loss: 0.0019 - accuracy: 0.9992 - val_loss: 0.0059 - val_accuracy: 0.9993
```

The above image shows the training and validation accuracy and losseses over 7 epochs.

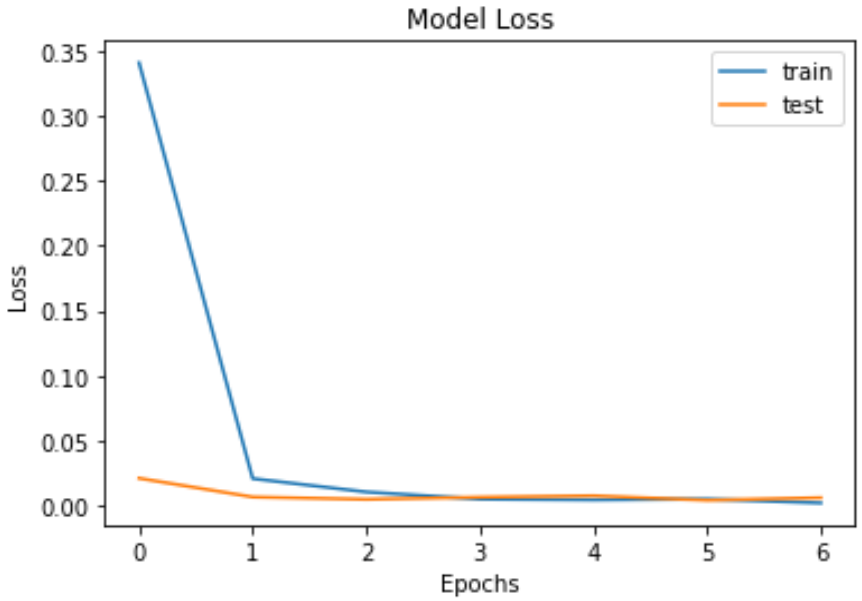
Model Evaluation:

1. Accuracy score on the test set

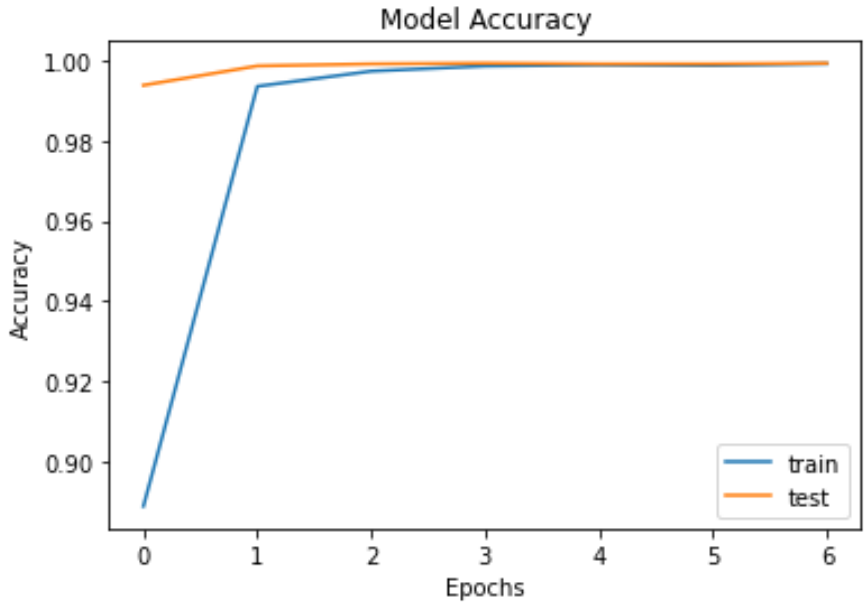
```
188/188 [==============================] - 10s 54ms/step - loss: 0.0059 - accuracy: 0.9993
Test accuracy: 99.93%
```

We got a loss of 0.059 on the test set and a high accuracy of 99.93%.

2. Loss vs. Epochs graph

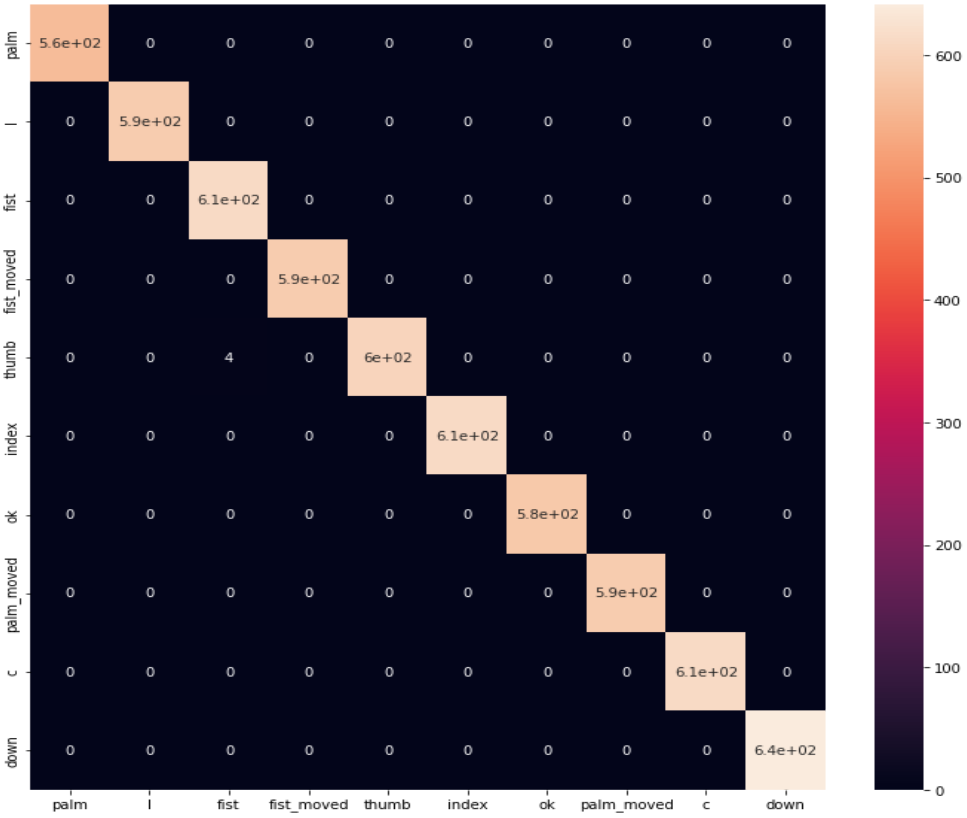


3. Accuracy vs Epochs graph



4. Confusion matrix

Confusion matrix is a table that denotes how well a classification model has performed based on the known actual values.
In our case, it matches the predicted class against the actual labels.



Conclusion and Future works:

We achieved an accuracy of 99% using Convolution Neural Network for this task at hand gestures. Even though this is a good accuracy, we have made a few assumptions while solving this task. We had a database of just 10 gestures, we can add more gestures to this dataset to make the model more robust. We also assumed that the images are less complex, i.e the image is only made up of a hand doing a gesture. Real world images would be much more complex than this. This could be a future work to improve this project. Also, this system cannot recognise a gesture done with both hands. So this could also be an area of work to make this model more useful in the future.