Hand Gesture Recognition

Project Supervisor: Dr. Sanchita Paul

Presented by:

Akshay Desai	BE/10357/17
Kumari Nivedita	BE/10245/17
Snigdha Ansu	BE/10124/17

Objective

The aim is to build a hand gesture recognition model with deep learning. This model will classify images of different hand gestures, such as a fist, palm, thumb and others. The basic idea is to use data to produce a model which is capable of returning correct output.

Introduction

- Machine Learning is very useful for a variety of real-life problems.
- It is commonly used for tasks such as classification, recognition, detection and predictions.
- The machine learning algorithm which is used for Image Classification here is Convolutional Neural Network(or CNN).
- CNN is an algorithm for machines to understand the features of the image and remember the features to guess the label of the new image fed to the machine.

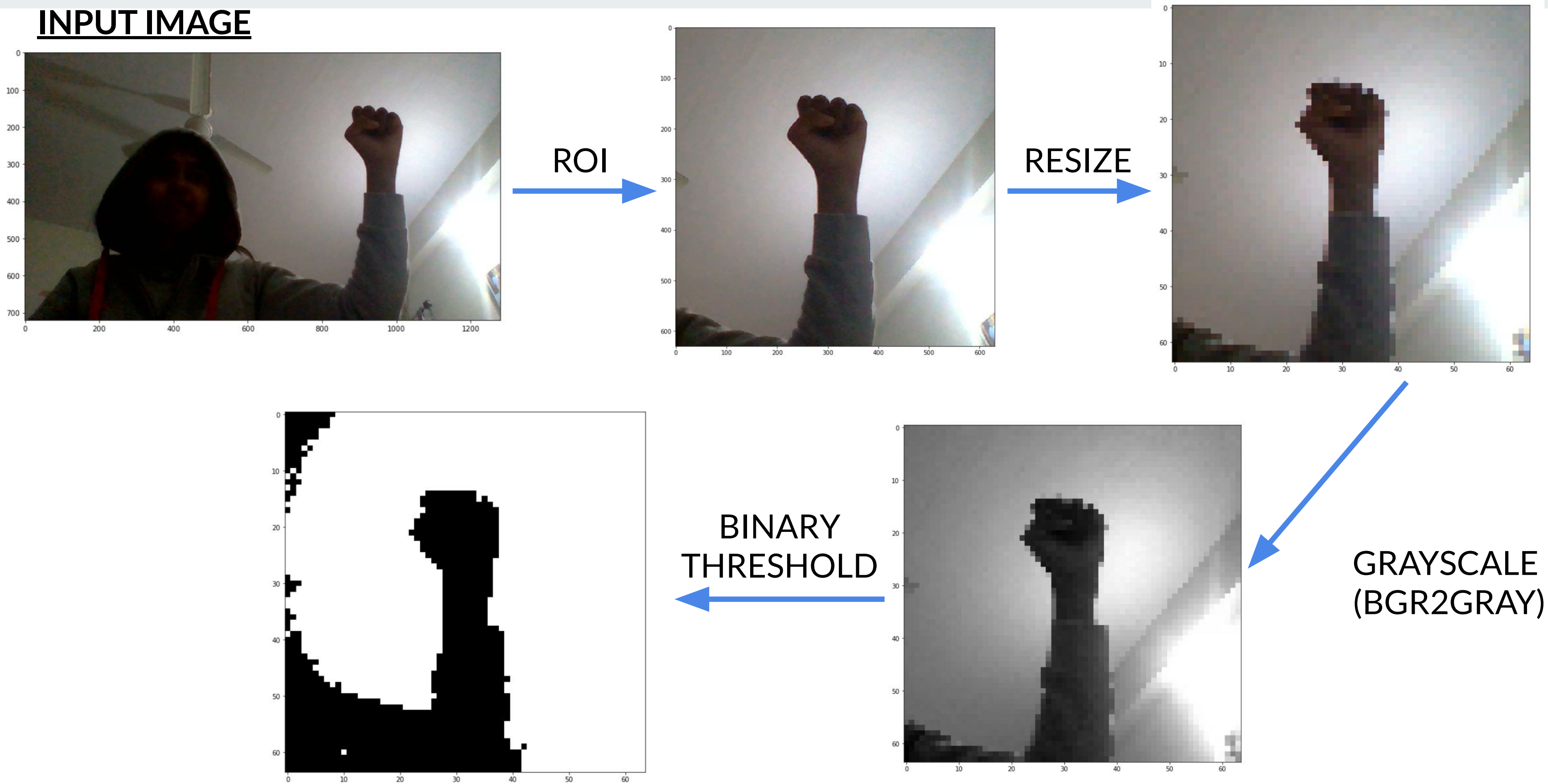
Proposed method

- Building a dataset using OpenCV :
  - Taking images for different gestures manually and putting them into test and train datasets.
  - The training dataset contains 6000 images of 6 different classes of hand gestures as follows:

    1.INDEX 2.PALM 3.C 4.OK 5.THUMBSUP 6.FIST
- Preprocessing the data :
  - Extracting the region of interest
  - The images have been resized to 64px x 64px uniform size
  - Converted images to grayscale
  - Binary Thresholding applied for training the model

PREPROCESSING

INPUT IMAGE



ROI

RESIZE

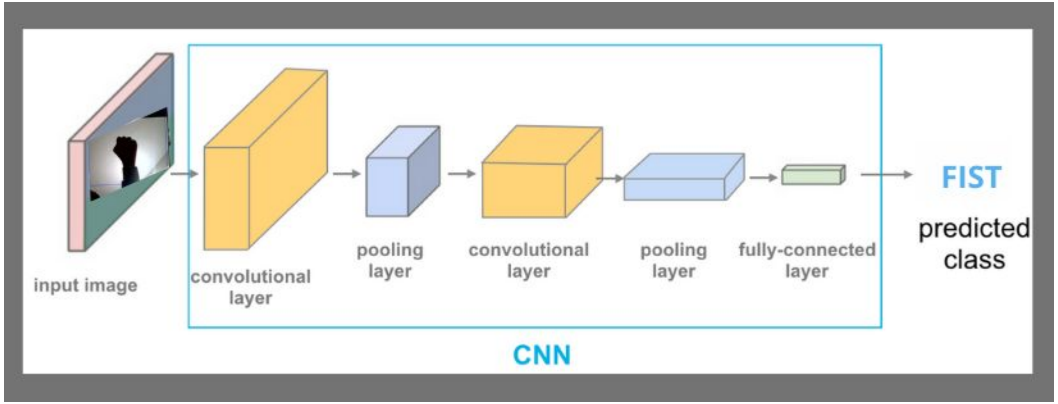
GRAYSCALE (BGR2GRAY)

BINARY THRESHOLD

PREPROCESSED IMAGE

- Creating a CNN and train the model :
    - Using training dataset to train the CNN and test the accuracy of the model using the testing dataset.
    - CNN has following components :
        - **Convolutional** layers - apply a specified number of convolution filters to the image.
          Ex - model.add(Convolution2D(32, (5, 5), input_shape=(64, 64, 1), activation='relu'))
        - **Pooling** layers - downsample the image data extracted by the convolutional layers to reduce the dimensionality of the feature map in order to decrease processing time.
          Ex - model.add(MaxPooling2D(pool_size=(2, 2)))
        - **Flattening** layer : converts the data into a 1-D array for input to the next layer.
          Ex - model.add(Flatten())
        - **Dense** (fully connected) layers - perform classification on the features extracted by the convolutional layers and downsampled by the pooling layers.
          Ex - model.add(Dense(units=128, activation='relu'))

Example :



- After 5 epochs, we received test accuracy of **97.50%**

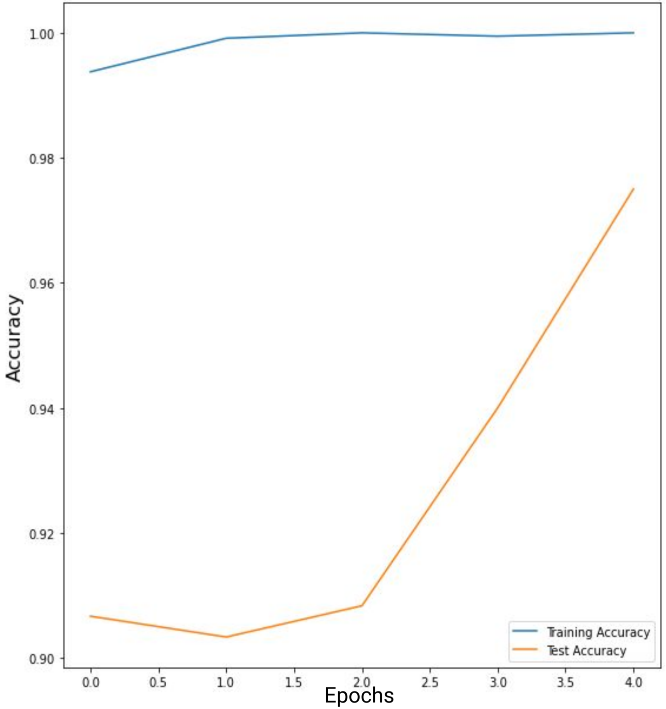
- Predicting real time results :
    - Using OpenCV, real-time frames are taken, preprocessed and are classified using the generated model.

Analysis

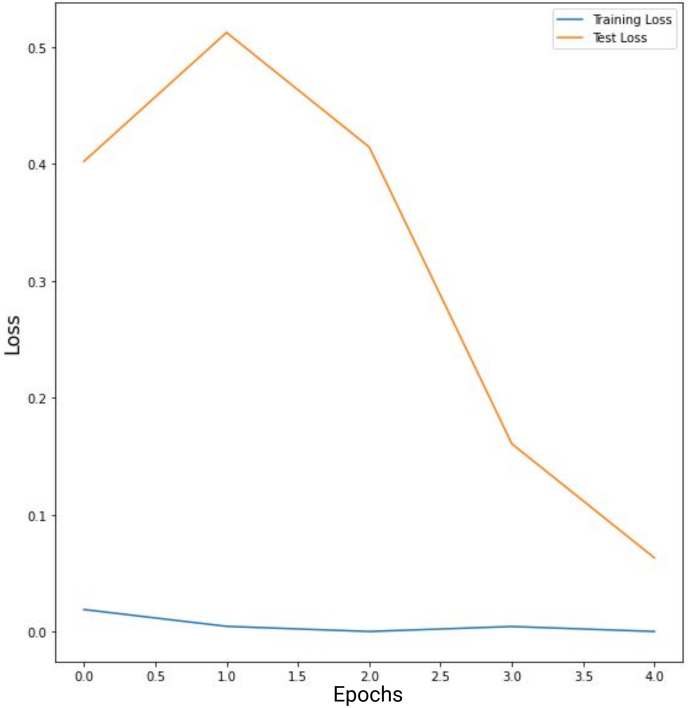
- Training accuracy, test accuracy, training loss and test loss were the four parameters we used to judge our model
- Test accuracy and loss are important as they indicate how our model performs when it gets a new data
- After 5 epochs, our model gave decent results :

```
Test Accuracy : 0.9750
Test Loss : 0.0629
```

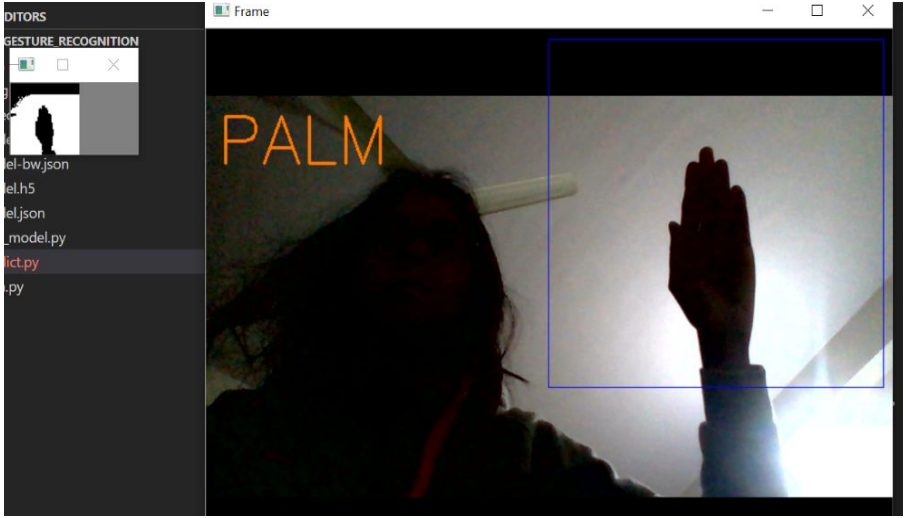
Plotting Accuracy

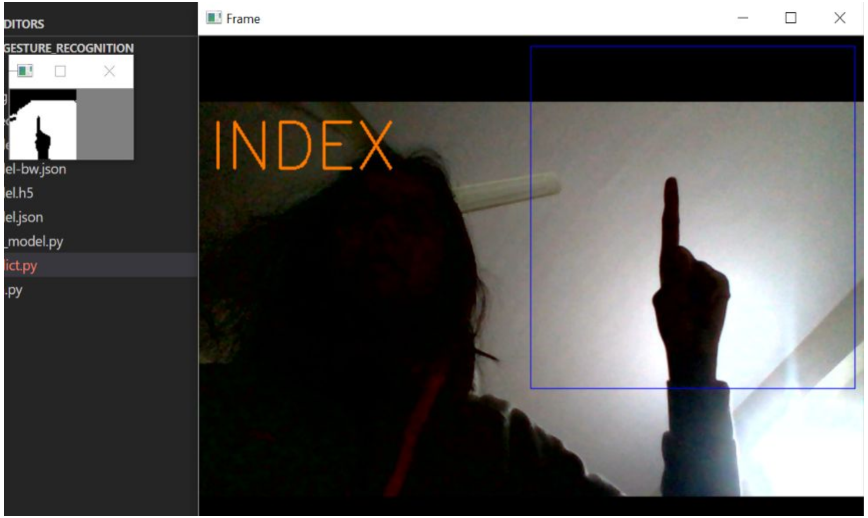


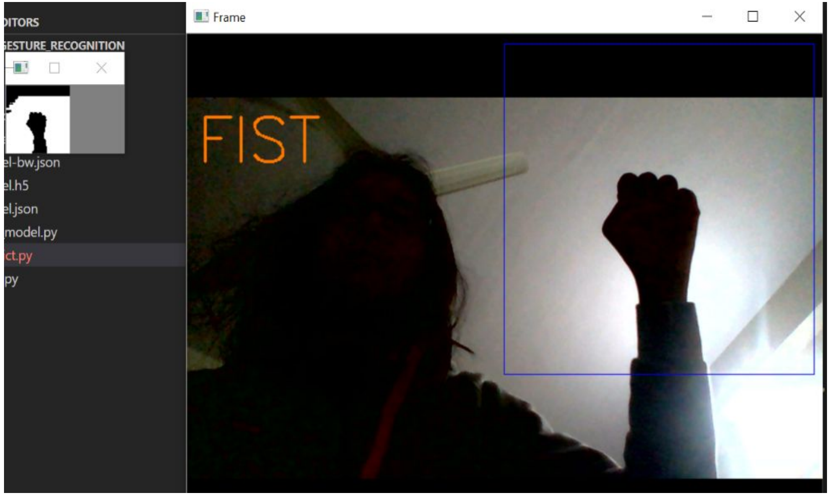
Plotting Loss



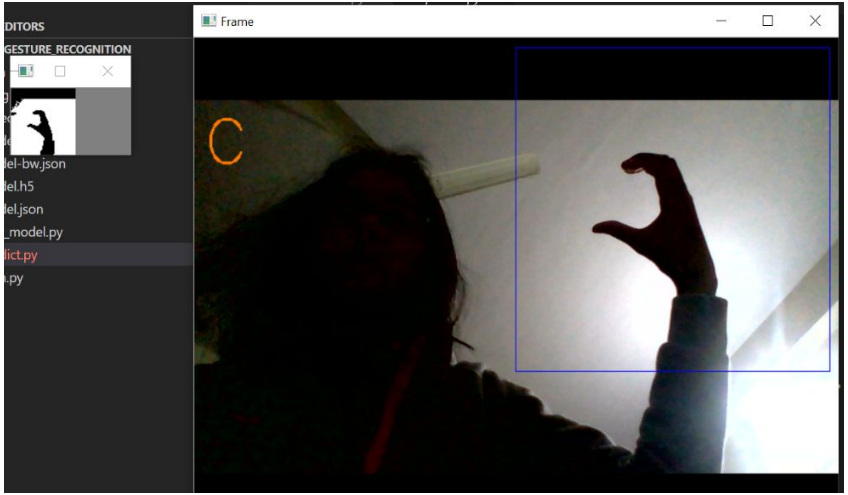
Result


DITORS
GESTURE_RECOGNITION
lel-bw.json
lel.h5
lel.json
_model.py
lict.py
.py
Frame
PALM

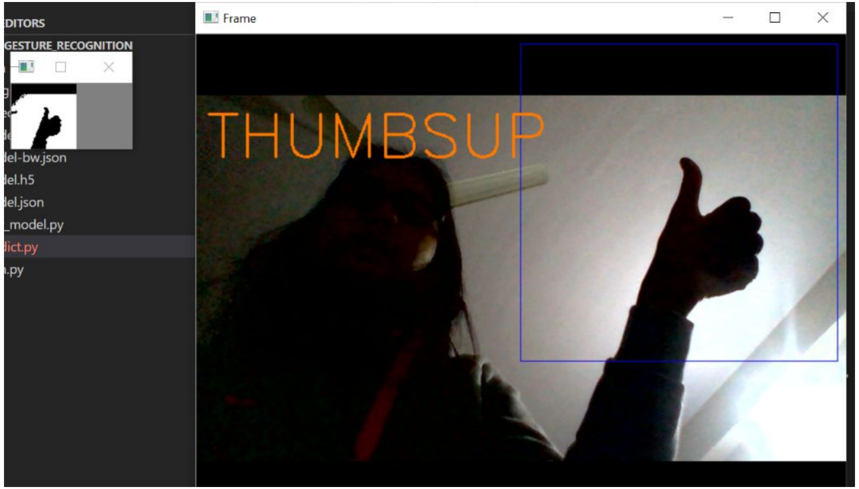

DITORS
GESTURE_RECOGNITION
lel-bw.json
lel.h5
lel.json
_model.py
lict.py
.py
Frame
INDEX

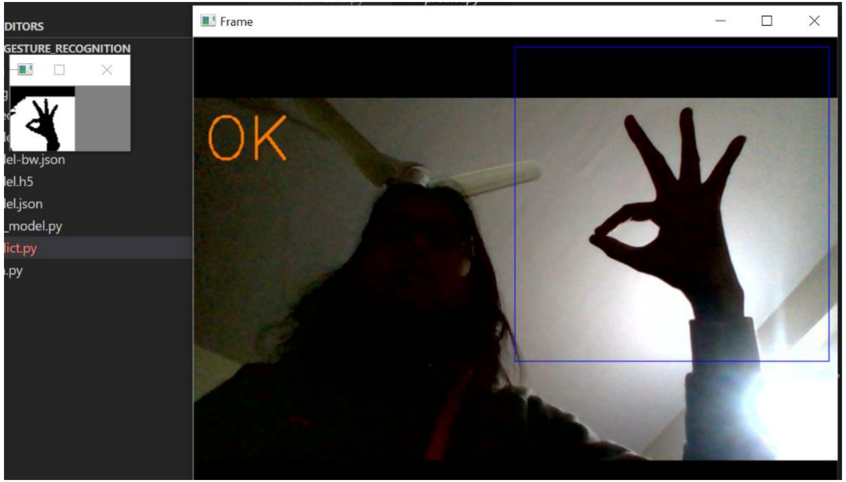

ITORS
ESTURE_RECOGNITION
l-bw.json
l.h5
l.json
model.py
ct.py
py
Frame

FIST


DITORS
GESTURE_RECOGNITION
el-bw.json
el.h5
el.json
_model.py
ict.py
.py
Frame

C


DITORS
GESTURE_RECOGNITION
Frame
THUMBSUP
lel-bw.json
lel.h5
lel.json
_model.py
lict.py
.py


DITORS
GESTURE_RECOGNITION
Frame
OK
lel-bw.json
lel.h5
lel.json
_model.py
lict.py
.py

Future scope

- This model could be used for communicating with the deaf and dumb using these various gestures.
- This will help to make the communication between the deaf/dumb and the people who don’t understand the sign language a lot easier.
- It can also be used in applications involving gesture navigation.

Conclusion

- Based on the results presented in the previous section, we can conclude that our algorithm successfully classifies different hand gesture images with enough confidence (>95%) based on a Deep Learning model.
- The accuracy of our model is directly influenced by a few aspects of our problem. The gestures presented are reasonably distinct, the images are clear and without background.

References

- Using Deep Learning and CNNs to make a Hand Gesture recognition model
  - https://towardsdatascience.com/tutorial-using-deep-learning-and-cnns-to-make-a-hand-gesture-recognition-model-371770b63a51
- From raw images to real-time predictions with Deep Learning
  - https://towardsdatascience.com/from-raw-images-to-real-time-predictions-with-deep-learning-ddbbda1be0e4

Thank You