CS 512 Final Project Report

American Sign Language (ASL) recognition using Convoluted Neural Networks

BY

Mayank Reddy Saddi

A20423747

Sai Manukar Reddy Bapathi

A20427621

November 2018

Problem Statement

Human beings are the most social organisms ever to walk the earth. One of the main aspects that made humans such social animals is the ability to communicate with one another. This ability to express and share knowledge, feelings, thoughts and ideas help create and strengthen bonds between individuals. Early communication was solely based on auditory modes which led to the invention and development of languages. Later as humans communicated and shared knowledge they began thinking and came up with more ways of communication such as writing, gestures etc.

Everyday technology is taking new leaps, exploring new realms making life much easier and simpler. With advancement of technology, communication has now become an unconscious action. Everyday there are new organizations big and small trying to bring people closer by introducing new ways of communication. Online Video calling is one of the most significant development in communication. It brings people irrespective of where they are in the world right in front of each other allowing them to talk more naturally. However often all these new advancements are inaccessible by the challenged individuals. Especially for the mute and hearing impaired, the only way they can communicate is using the Sign Language.

Sign Language uses movement of one or more hands accompanied with facial expressions which correspond to specific meaning. Although the hard of hearing can communicate with no problems among themselves, it is a serious challenge for them trying to blend in with the rest of the population in work or social environments. This is because most of us do not know and cannot interpret the Sign Language.

This project aims at bridging this gap between the hearing impaired and the common population by providing a method to translate the American Sign Language (ASL) to English. By observing the hands and gestures used in sign language the project identifies each sign and displays the translation in English text.

Proposed Solution

The proposed solution for the project comprises of 3 main phases:

- **Skin Pixel Segmentation**
  Create a Region Of Interest (ROI) from the entire camera view to obtain just the region with the hand in the frame. Threshold the ROI to segment the image in order to obtain just the skin pixels in white and the rest of the image in black.

- **Data Generation**
  Once the segmented ROI is obtained data required for training the model is generated. Data is divided into train, validation and test portions which can be used by the model to train and validate its learning.

- **Model Generation**
  A Convoluted Neural Network (CNN) model is generated with network architecture as specified in "Using Deep Convolutional Networks for Gesture Recognition in American Sign Language" by Vivek Bheda and N. Dianna Radpour. State University of New York at Buffalo. [1]

- **Train the model using the Data generated**
  Once the model is generated the model is trained with the generated data with categorical cross entropy as the loss function and softmax function to obtain final class labels. Once the model is trained the model and the weights of the trained model are saved as .hdf5 files.

- **Predict and Display results**
  In the final phase of the project live video feed is taken from the computer's web camera and each frame is passed to the model to predict and return the result. To account for the processing instead of the entire frame just hand portion similar to the first phase of the project is passed to the model. The model returns the class label with the highest probability and the result is displayed on the live video feed.

Implementation Details

Segmentation

To eliminate the unnecessary portions on the frame and obtain only the hand region of the frame, first the portion of the image is extracted by drawing a rectangle on the frame to indicate the region where the user must place his hand. This region of the frame is saved as a Region Of Interest (ROI).

Once the ROI is obtained, it is converted to Hue Saturation Value (HSV) color space to segment the skin pixels from the rest of the elements in the image. The segmentation is done as described in "Skin Detection using HSV color space" by V. A. OLIVEIRA, A. CONCI, Computation Institute – Universidade Federal Fluminense – UFF – NiterÅi, Brazil [2].

The segmentation is subjected to the ambient surrounding light and the user's skin tone and several other aspects. Thus to make up for that we have provided trackbars which user can use to control the lower and upper hsv threshold values. Once the user obtains desired amount of segmentation he presses enter. To elimninate noise the hsv image is subjected to gaussian blur and then a median blur is applied to remove salt and pepper noise. Then the hsv values are used in the OpenCV inRange function to filter only skin elements and mark them white and the rest elements black. The obtained images are resized to 200x200.

Data Generation

The previously obtained segmentation is used to extract hand regions. As the user makes gestures of different signs of the ASL alphabet as shown in Figure 1. User can press 'c' key to proceed capturing the frames. The frames are captured as the user presses the key and this enables the user to orient his hand in different directions and enable the model to be trained on different postures of the same sign. The captured frames are divided as 800 train images, 100 validation images and 100 test images for each alphabet. The images for different alphabets are saved in different directories.

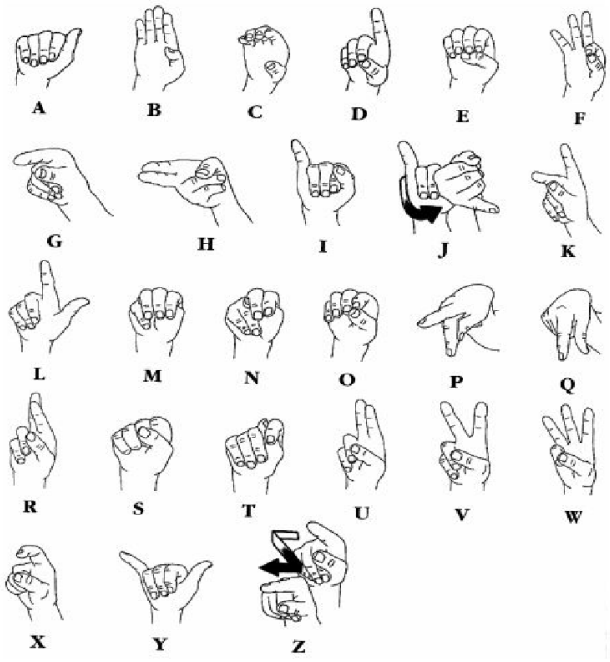

Figure 1. American Sign Language Alphabet

Model Generation

The CNN model is generated using keras. The architecture of the model is generated as in "Using Deep Convolutional Networks for Gesture Recognition in American Sign Language" by Vivek Bheda and N. Dianna Radpour State University of New York at Buffalo [1].

The model consists of 6 Convolutional Layers of filter sizes 3x3 using ReLu activation function, 3 maxpooling layers, and 3 dropout layers of 0.25. The output of these layers is flattened and passed to 2 dense layers the last of which return 25 labels outputs each corresponding to letters from A to Z.

The Network Architecture is shown in Figure 2 below.

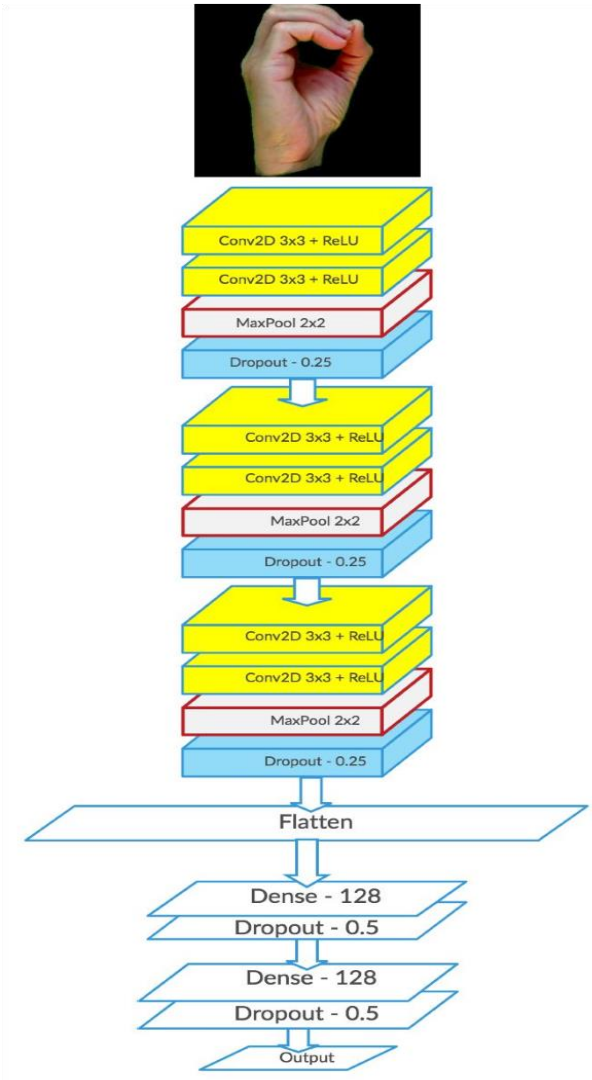

Figure 2. Network Architecture

Training the model with generated data

The model is trained using the Data generated which is saved in different directories by making use of the keras.ImageDataGenerator.flow_from_directory() [3]. The flow_from_directory() takes advantage of the Data arranged in different directories and trains the model as the data belonging to different directories as different classes. The training is done while monitoring categorical cross entropy loss as the classes are not binary categorical cross entropy is best suited for non-binary classification. The output classes are obtained with softmax function which returns the class label with the highest probability. Training is done for 5 epochs with a batch size of 500. Once the training is done the training and validation accuracies, losses and mean-squared error loss are plotted for analysis. The mean squared error (MSE) for train and validation data is used to verify that there is no overfitting of the data by analyzing the MSE curve of training and validation [4]. The different scenarios of MSE and quality of training is shown in Figure 3.

Once the model has been trained the model and its weights are saved as. hdf5 files which can be later used for prediction.

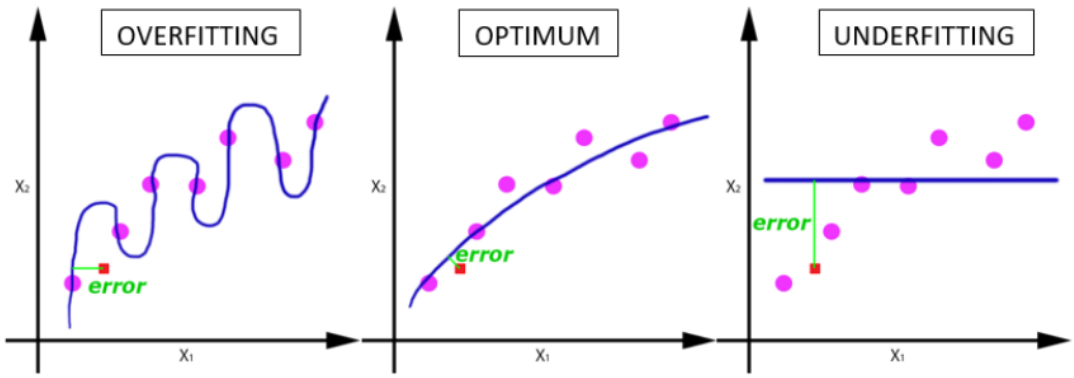

Figure 3 MSE curves for overfitting, optimal, underfitting

Prediction

For predicting the results of gestures, we take the live video feed from the system's web camera and just as in the segmentation step obtain ROI which is segmented. The previously generated model and the weights are loaded using keras' load_model and model.load_weights(). This segmented ROI is then passed to the loaded model and model.predict is used to obtain the class label for the prediction. This result is displayed on the frame using the cv2.putText().

Program Design Issues

- Gestures for some alphabets such as ‘A’, ‘S’, ‘T’ ‘M’ and ‘N’ have very similar looking gestures which model tends to misinterpret.

- ASL sign for ‘Z’ is drawing the Z in space which doesn’t seem to be picked up by the model and only letters from A to Y are proved to work well.

- Segmentation depends on multitude of factors such as the ambient surrounding light, skin tone of the user, resolution of the user’s camera etc., so some noise tends to creep in which disturbs the model.

Program Usage Instructions

- Run ‘predict.py’ using python3 with keras, OpenCV, numpy packages.

- The program displays clear instructions for how to proceed with the program.

- If the data has been generated and model has already been trained. Press y when programs prompts.

- If not press n and menu is displayed for different actions
    - ‘g’ to generate data.
    - ‘t’ to train model.
    - ‘r’ to adjust segmentation parameters.
    - ‘p’ to predict the output.

- At any point press ‘h’ to display the menu which gives instructions.

- Press ‘ESC’ Key to quit the program.

- While adjusting the segmentation parameters once the desired amount of segmentation is obtained press enter. This should save the parameters and to readjust press ‘h’ and then press ‘r’.

Results and Discussions

Segmentation

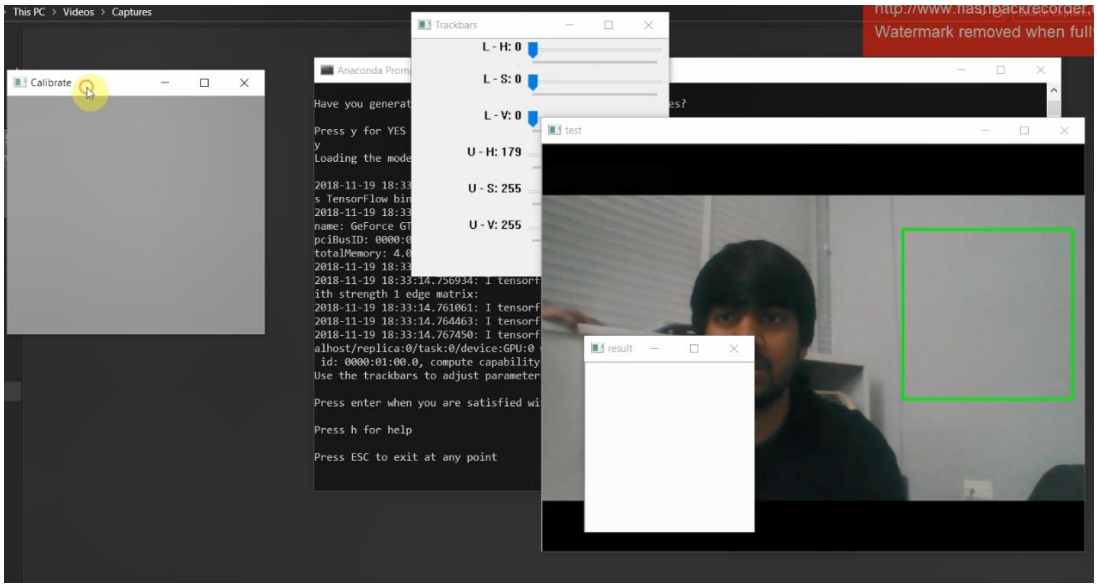

Figure 4 Trackbars to control Segmentation parameters

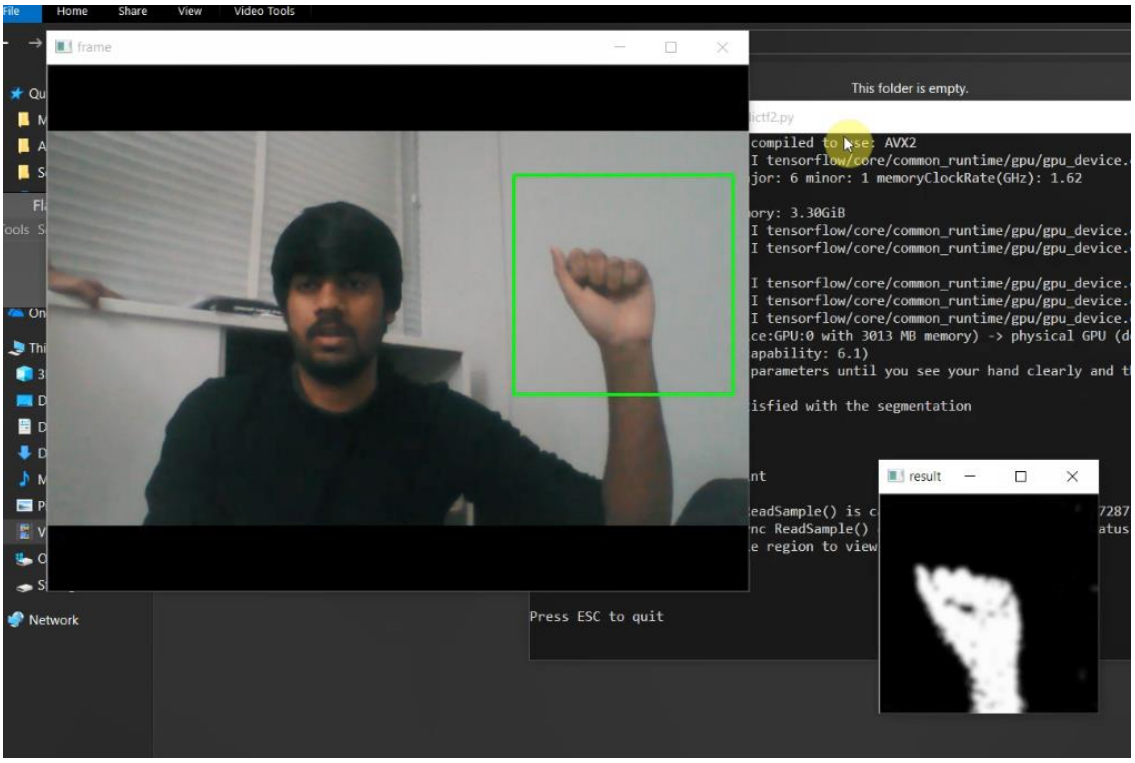

Figure 5 Segmented Result

Training

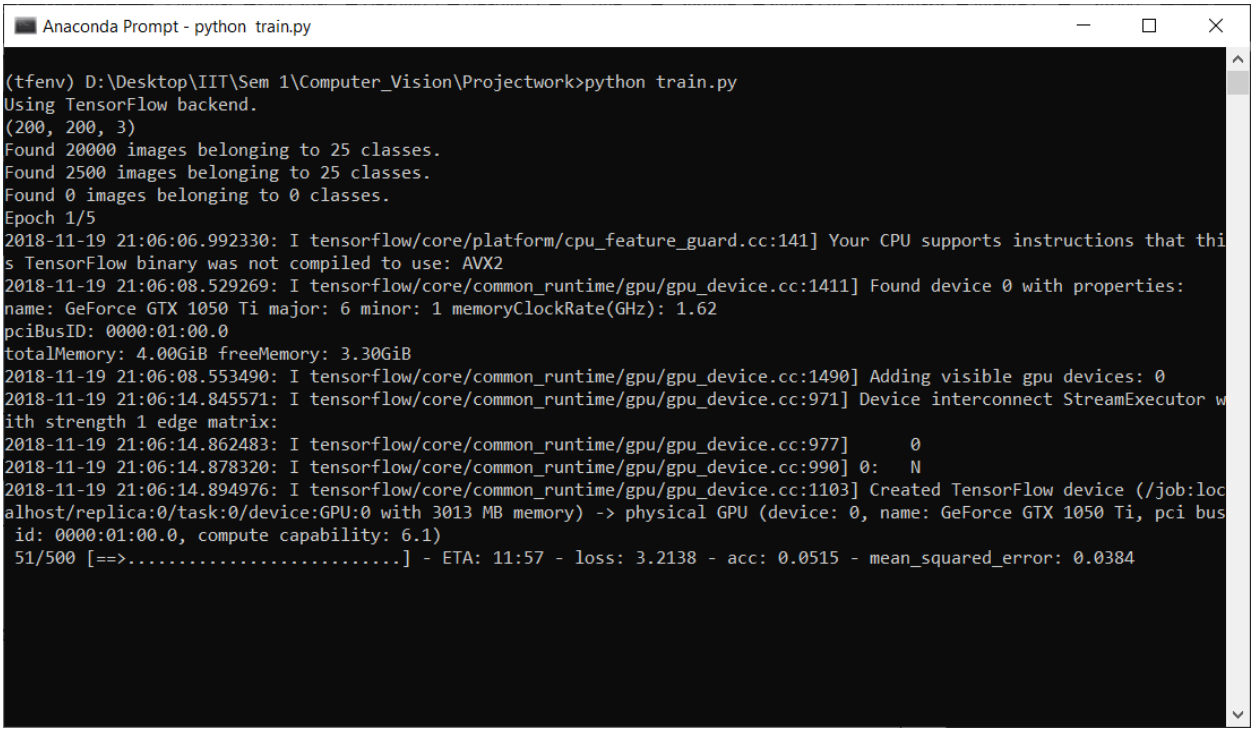

Figure 6 Training the model

After 5 epochs accuracy of about 78% is achieved.

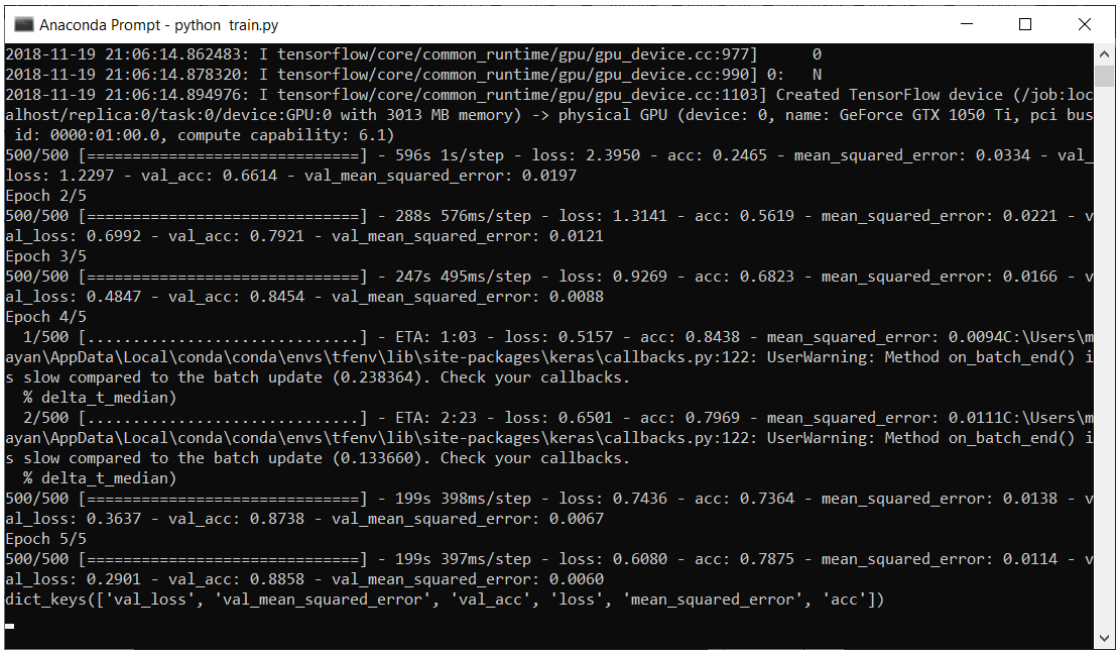

Figure 7 Training status after 5 epochs

Accuracy plots

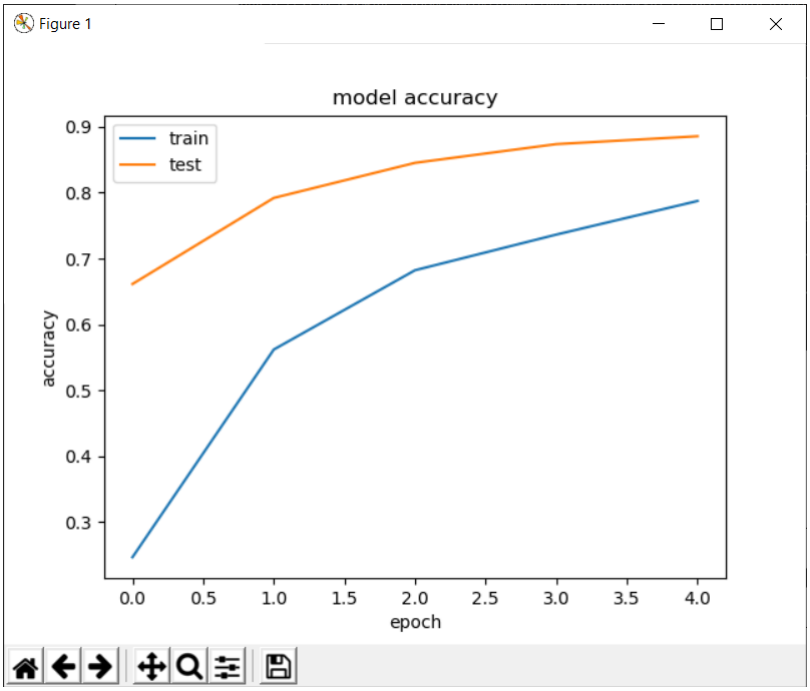

Figure 8 Test and Train accuracy plots

Loss plots

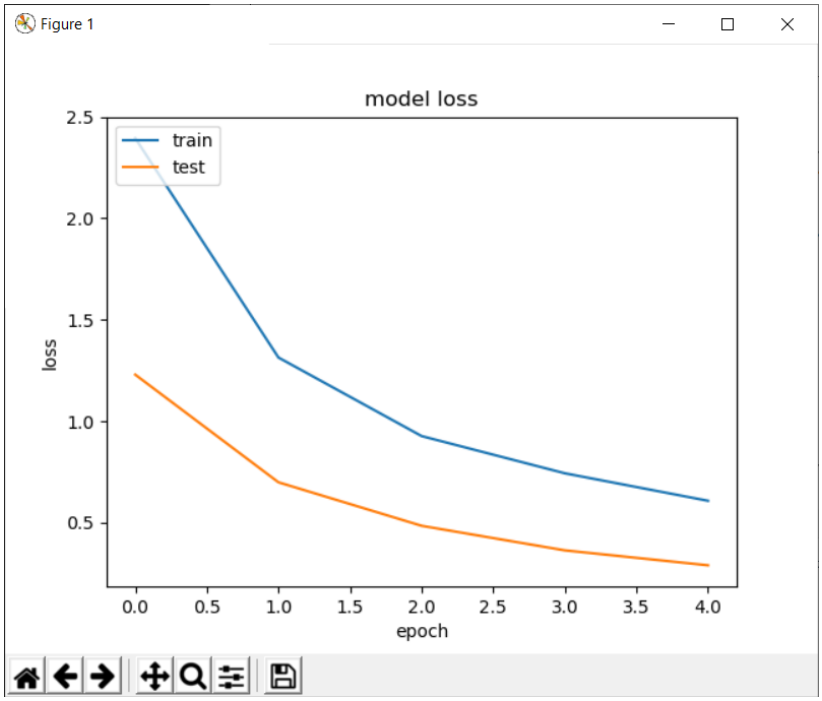

Figure 9 Test and Train Loss plots

Mean Squared Error

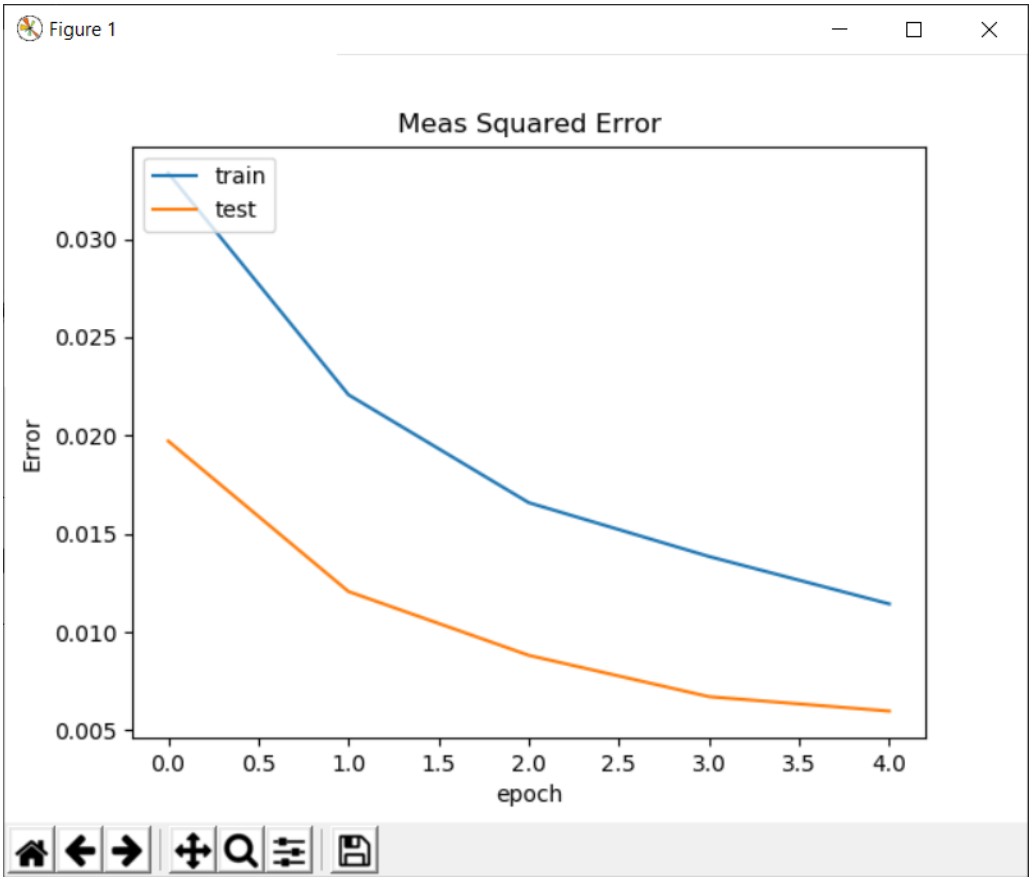

Figure 10 Mean Squared Error plot

Model Saved

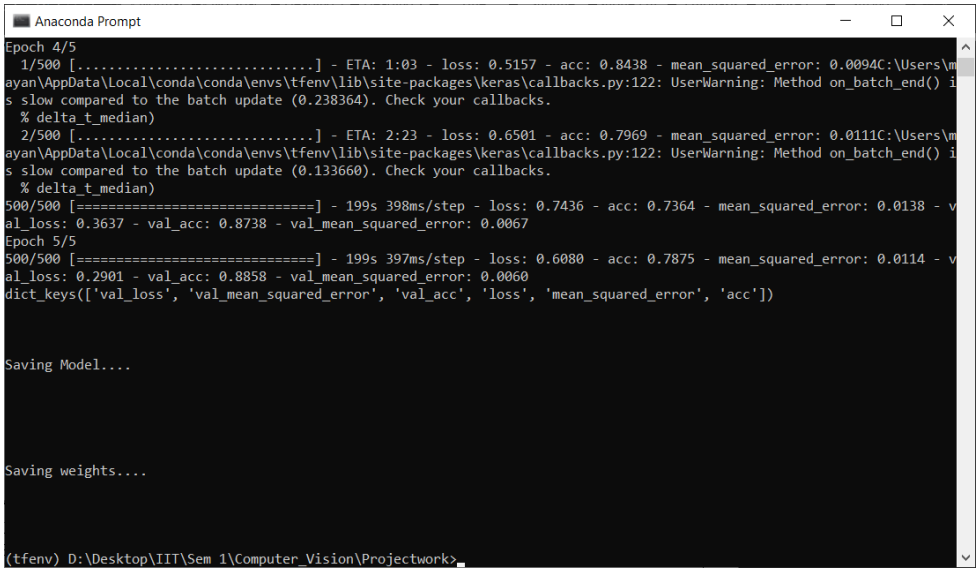

Figure 11 Saving Model and Weights

Predictions

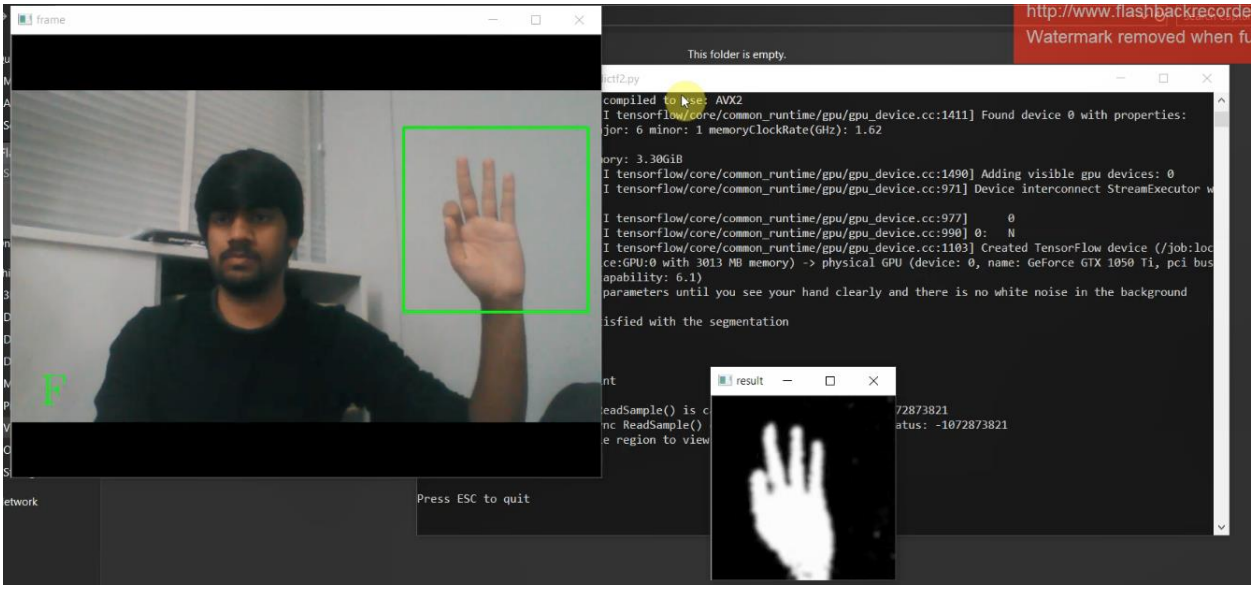

Figure 12 Prediction example 1

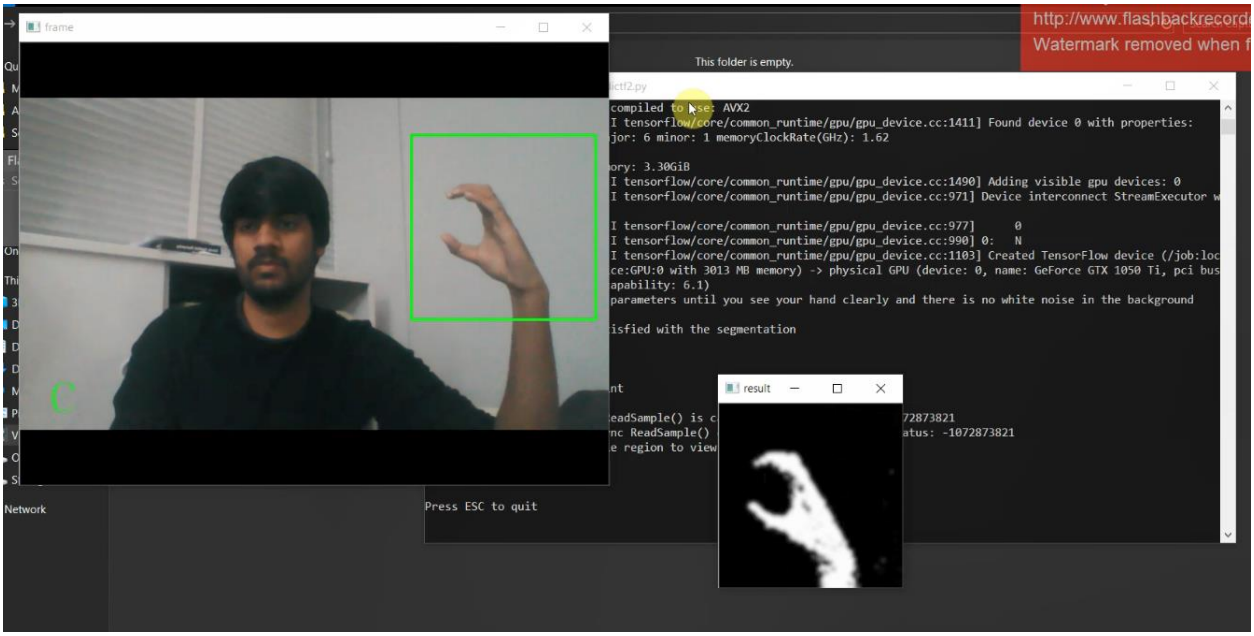

Figure 13 Prediction Example 2

References

[1] "Using Deep Convolutional Networks for Gesture Recognition in American Sign Language" by Vivek Bheda and N. Dianna Radpour State University of New York at Buffalo.

[2] "Skin Detection using HSV color space" by V. A. OLIVEIRA, A. CONCI, Computation Institute – Universidade Federal Fluminense – UFF – NiterÅi, Brazil.

[3] https://medium.com/@vijayabhaskar96/tutorial-image-classification-with-keras-flow-from-directory-and-generators-95f75ebe5720

[4] https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9