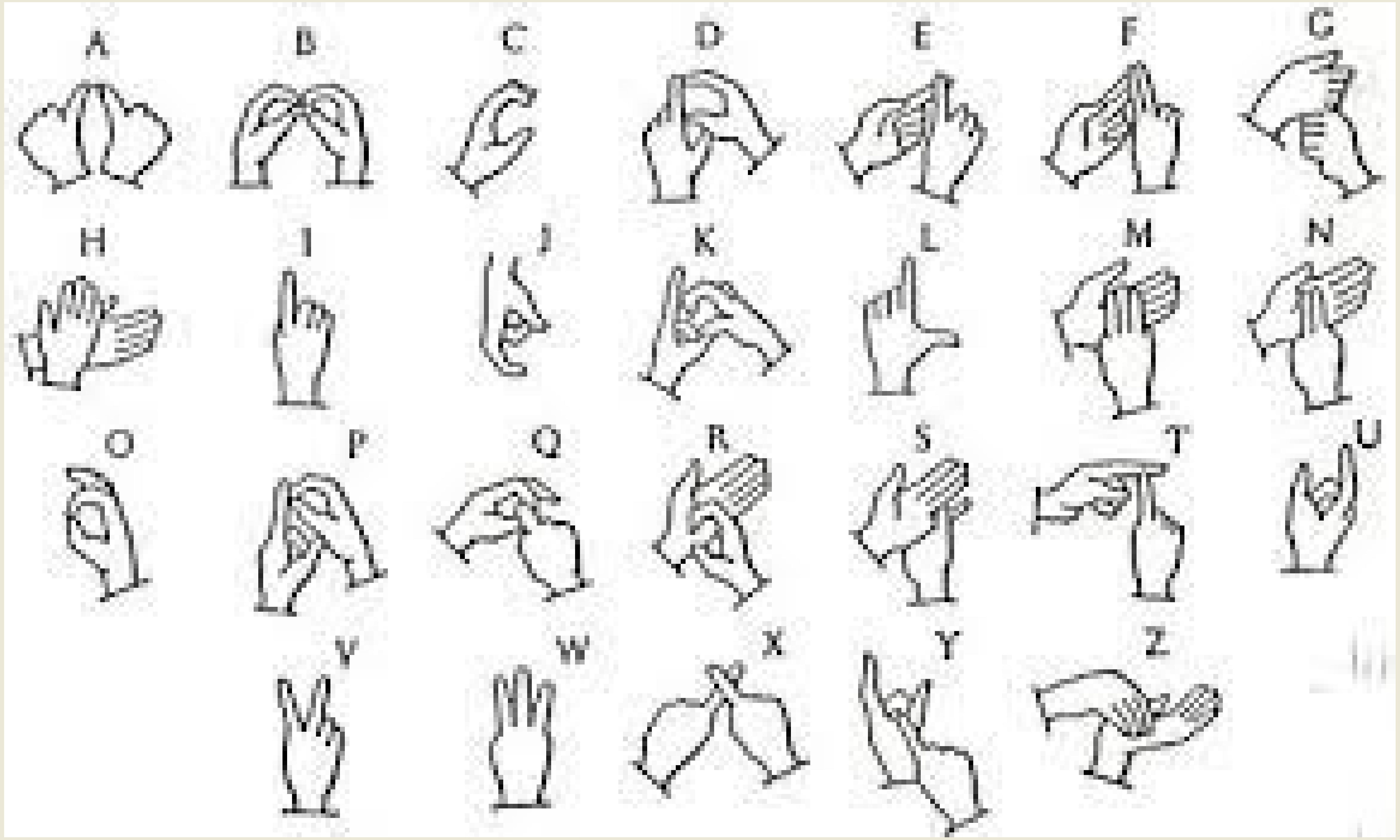
Under the guidance of Dr. Umesh Deshpande

GROUP MEMBERS

BT18CSE015 - Sanskruti
BT18CSE035 - Niraj
BT18CSE036 - Tanmay
BT18CSE040 - Nisarg
BT18CSE044 - Ketan

Indian Sign Language



RECAP

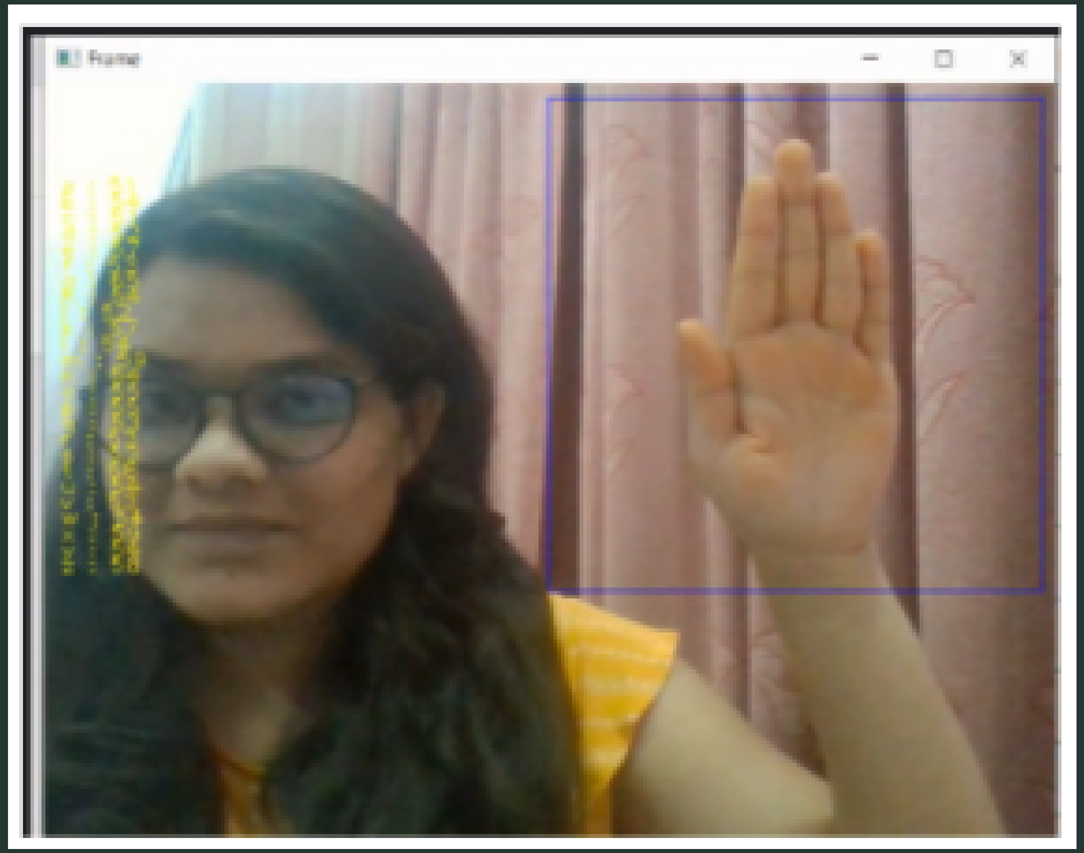
- Trained, tested and made a GUI for real time testing of **ASL** (American Sign Language Recognition).
    - Accuracy : 0.973
    - Step-Loss : 0.105
    - Val-Loss : 0.003
- **ISL** (Indian Sign Language Recognition)
    - Chose a dataset with 1200 images in each of the 36 classes (0-9 & A-Z).
    - We tested our previous model with new ISL images we got 89.66% accuracy.
    - Used preprocessing of canny-edge.
    - Implemented SIFT Feature detection for BOVW Model.
    - Used K-means for clustering

FEEDBACK RECEIVED

- K-means is not good for large datasets
- Canny-edge and SIFT

Preprocessing

Capturing Raw Image



Gray Scale Image

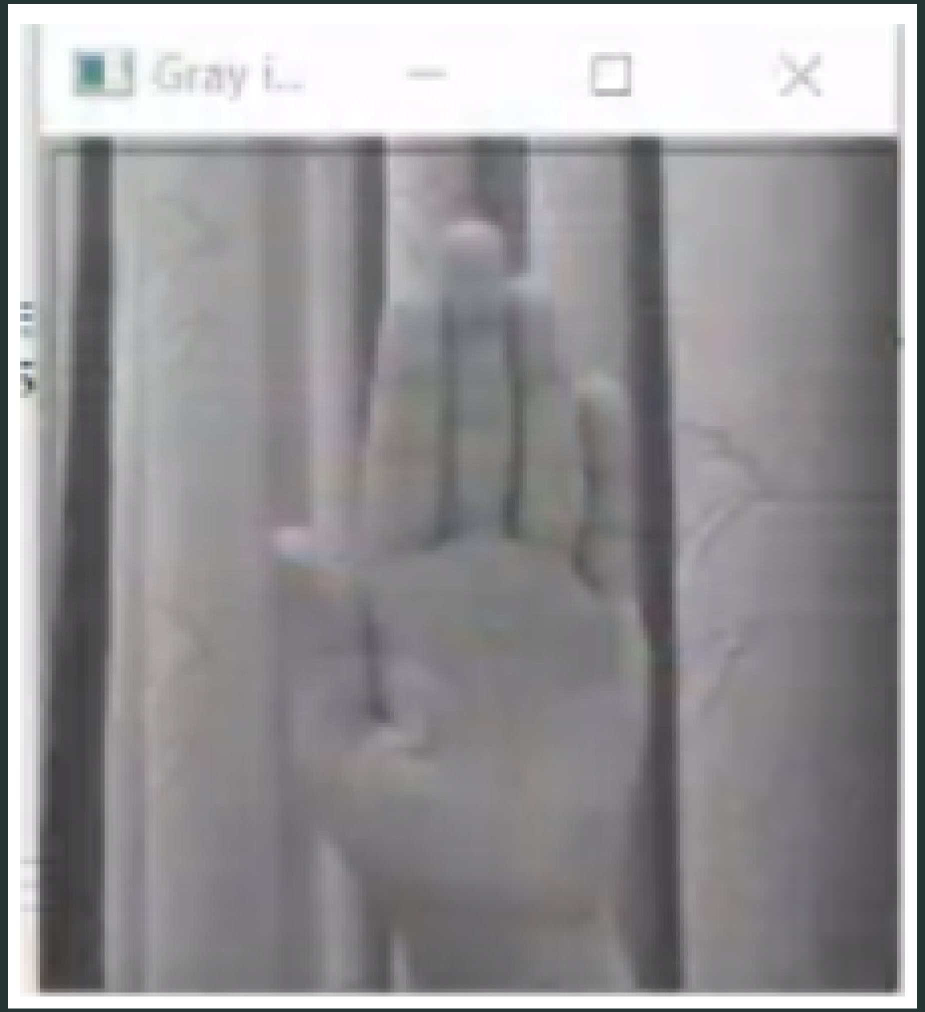
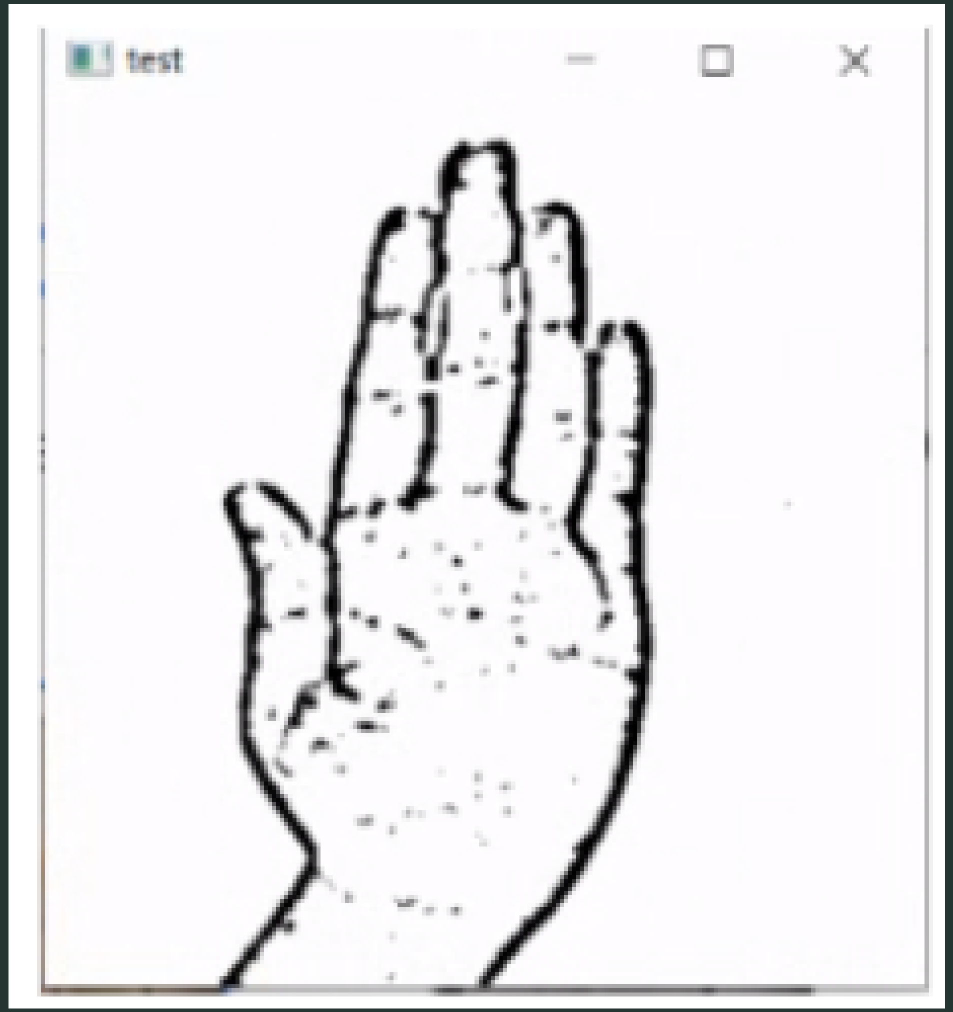


Image Post Gaussian Blur and thresholding



ORB (ORIENTED FAST AND ROTATED BRIEF)

ORB builds on the well-known FAST keypoint detector and the BRIEF descriptor.

1) Fast(Features from Accelerated and Segments Test)

- Given a pixel p in an array fast compares the brightness of p to surrounding 16 pixels that are in a small circle around p. Pixels in the circle is then sorted into three classes (lighter than p, darker than p or similar to p).
- If more than 8 pixels are darker or brighter than p than it is selected as a keypoint. So keypoints found by fast gives us information on the location of determining edges in an image.
- ORB algorithm uses a multiscale image pyramid. In this way, ORB is partial scale invariant.
- After locating keypoints orb now assign an orientation to each keypoint like left or right facing depending on how the levels of intensity change around that keypoint. For detecting intensity change orb uses intensity centroid. The intensity centroid assumes that a corner's intensity is offset from its center, and this vector may be used to impute an orientation.
- Once we've calculated the orientation of the patch, we can rotate it to a canonical rotation and then compute the descriptor, thus obtaining some rotation invariance.

2) Brief (Binary Robust Independent Elementary Feature)

- Brief takes all keypoints found by the fast algorithm and convert it into a binary feature vector so that together they can represent an object
- Binary features vector also know as binary feature descriptor is a feature vector that only contains 1 and 0.
- In brief, each keypoint is described by a feature vector which is 128–512 bits string.

ORB performs as well as SIFT on the task of feature detection (and is better than SURF) while being almost two orders of magnitude faster.

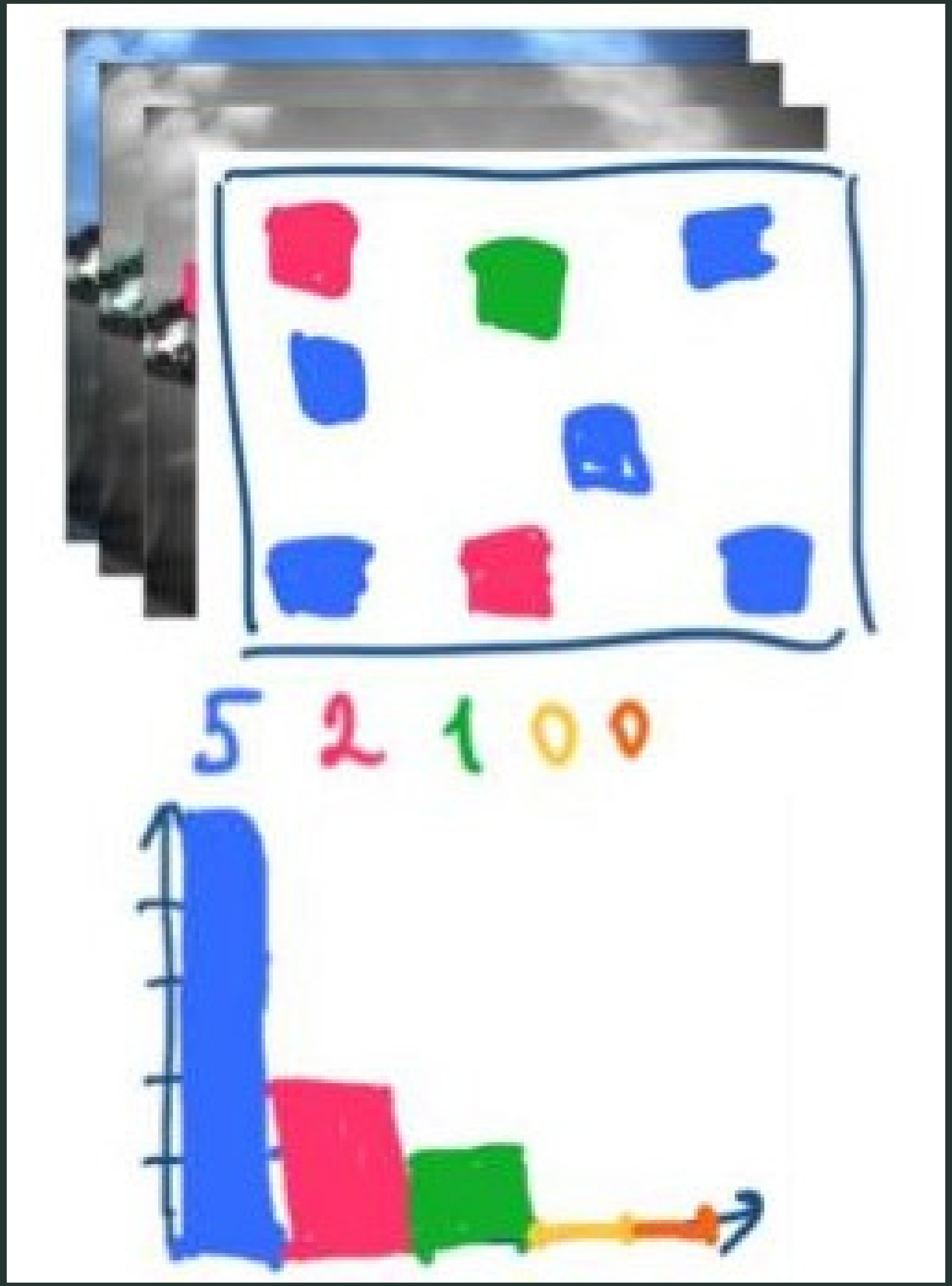
CLUSTERING

Mini Batch K-means Clustering

- The Mini-batch K-means clustering algorithm is a version of the standard **K-means** algorithm in machine learning.
- It uses small, random, fixed-size batches of data to store in memory, and then with each iteration, a random sample of the data is collected and used to update the clusters.
- Sometimes it performs better than the standard K-means algorithm while working on huge datasets because it doesn’t iterate over the entire dataset.
- It creates random batches of data to be stored in memory, then a random batch of data is collected on each iteration to update the clusters.
- The main advantage of using the Mini-batch K-means algorithm is that it reduces the computational cost of finding a cluster.

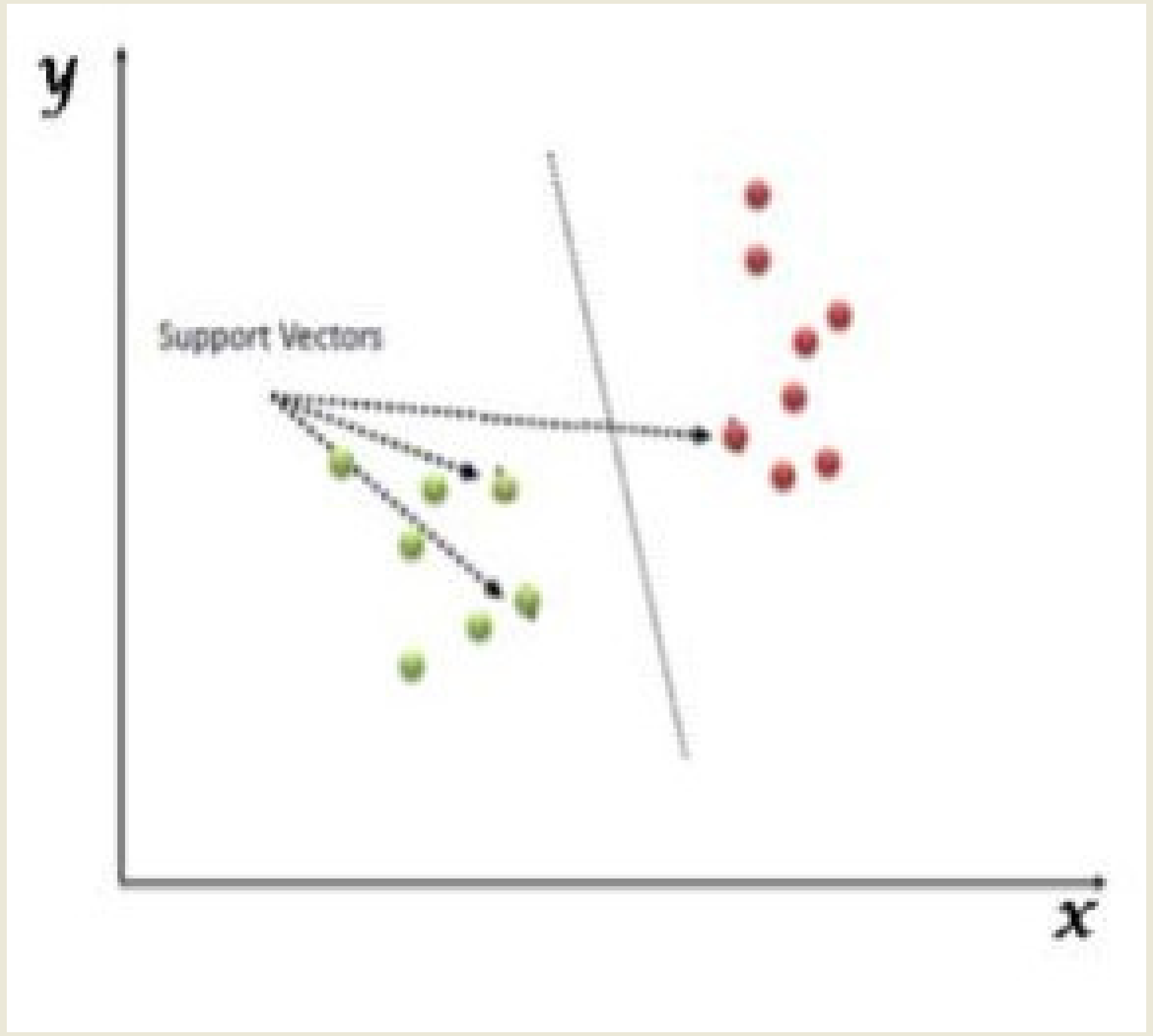
Creating Histogram

- Now we will create a 2D-array of zeros of shape (N,K) , Then we iterate through our images again and look for words in the image present in our dictionary.
- Once we detect a word present in both the dictionary and image, we increase the count of that particular word(i.e array[i][w]+=1 where i is the current image and w is the word).
- This is how we create the histograms for the images. Similarly,all the images will be converted to histograms.



CLASSIFICATION

- Having generated a BOVW histogram for each of our training images, the final step in setting up our image classifier is to train a SVM (Support Vector Machine) or any other classifier on the BOVW histograms.
- Once trained, the intention is that the SVM is then able to take the BOVW histogram for a new image that was not part of the training set and correctly classify it.
- Support Vector Machine" (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems.
- So In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the respective classes very well.



ACCURACY

The accuracy of Bag of Visual Words model is **92.38%**.

CUSHION

Transfer Learning is used in the field of deep learning which facilitates the use of already trained models for the related applications.

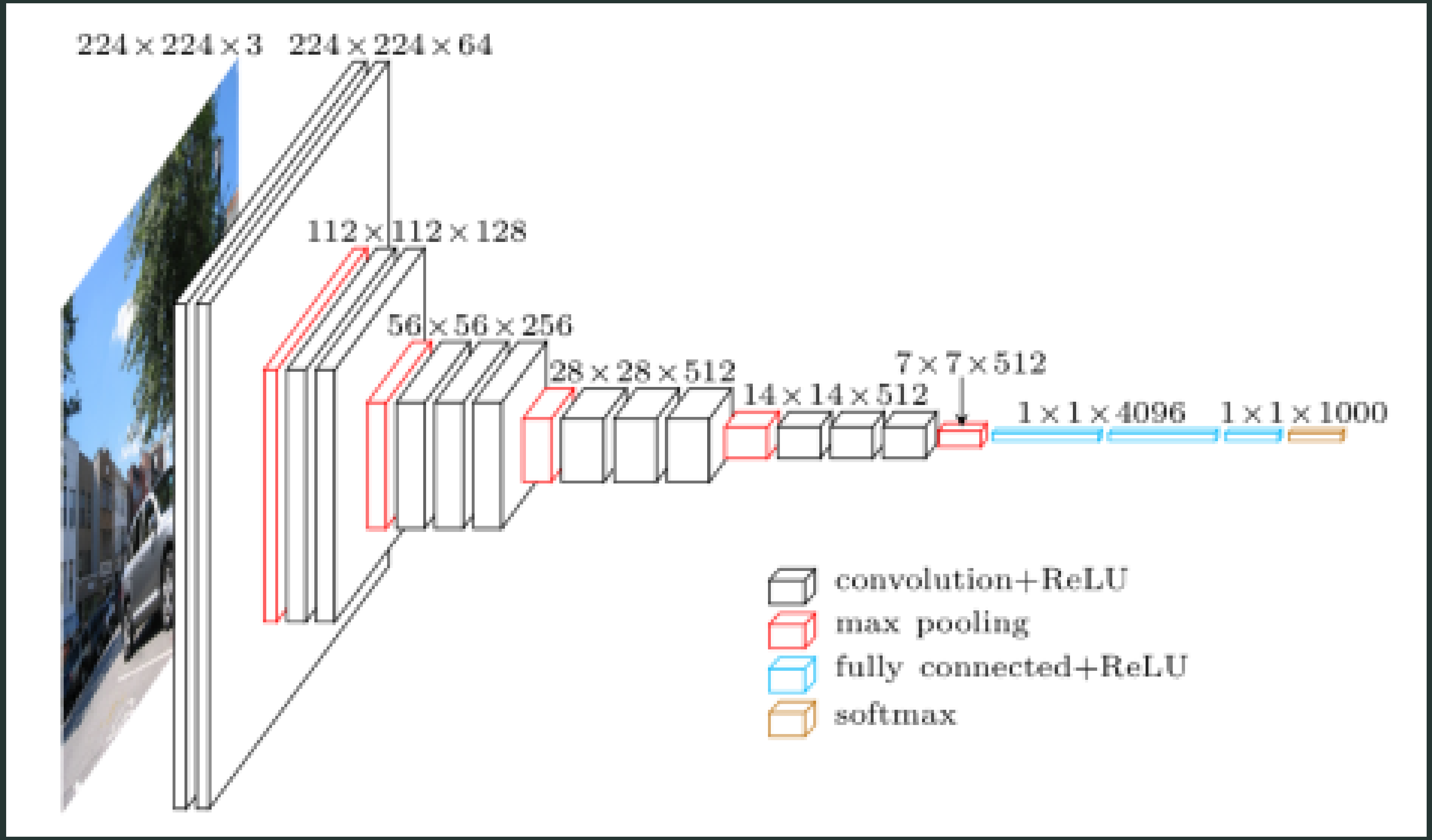
We basically remove the top layer and make all these layers non-trainable. Then we add our own layers which use these already identified features to train.

This helps to lower the number of training parameters. Helps to converge faster.

Used transfer learning with VGG16 and Resnet50 on ISL images.

VGG16

Visual Geometric Group



- It is known for its simplicity with less number of hyper-parameters.
- Convolution layers of 3x3 filter with a stride 1 and always used same padding and maxpool layer of 2x2 filter of stride 2.
- These layers are stacked on top of each other increasing depth and volume size handled by max pool layer.

Interesting problem: VGG accepts only RGB ie. 3 channels, But we had Greyscale images so we had to use same channel 3 times during training

RESNET50

Residual Networks

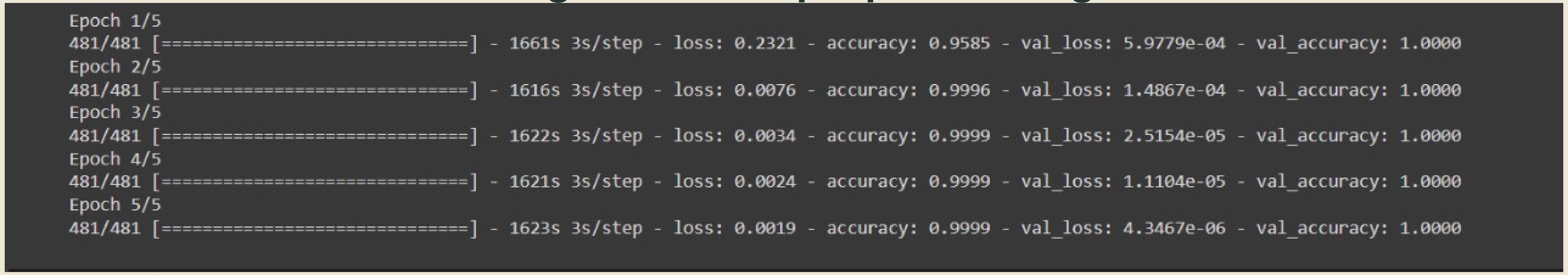
ResNet50 is a variant of ResNet model which has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer.

Here we basically create shortcuts so that we will not loose the information from earlier layers.

- Known for giving good accuracy in classification problems.
- Skip connection helps to solve the vanishing gradient problem.

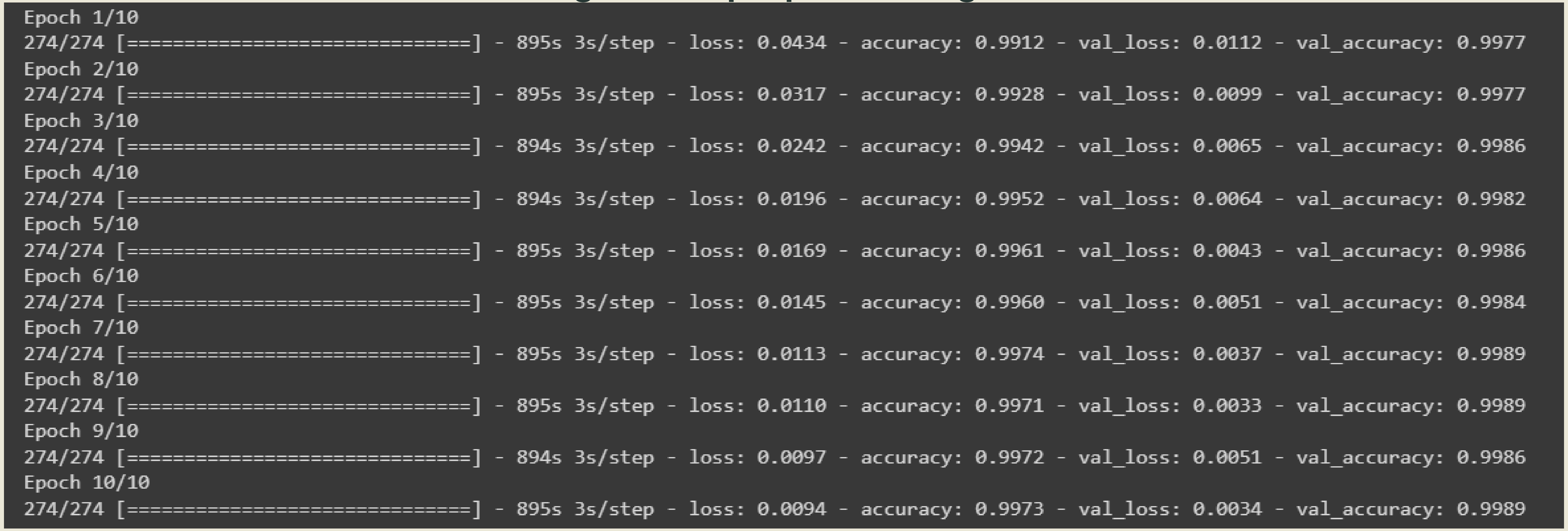
ACCURACY (VGG16)

Images without preprocessing



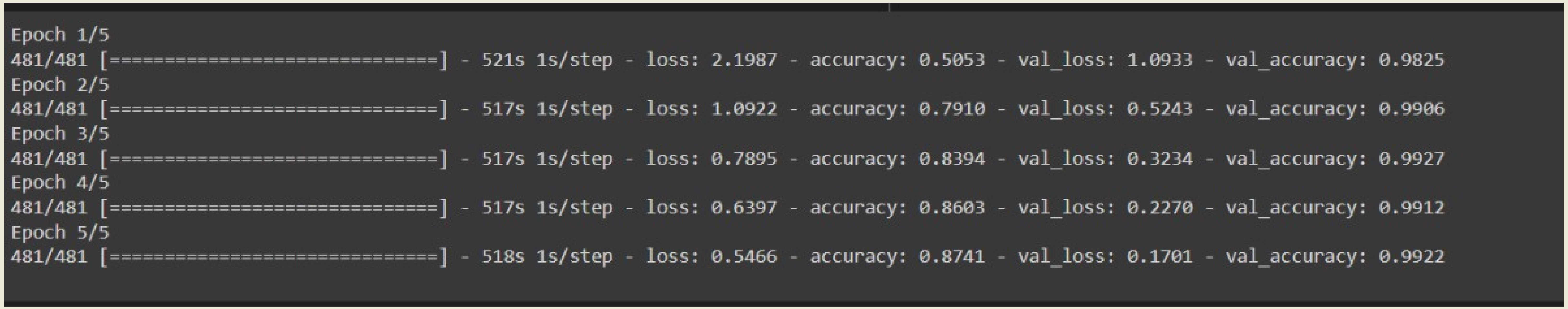
```
Epoch 1/5
481/481 [==============================] - 1661s 3s/step - loss: 0.2321 - accuracy: 0.9585 - val_loss: 5.9779e-04 - val_accuracy: 1.0000
Epoch 2/5
481/481 [==============================] - 1616s 3s/step - loss: 0.0076 - accuracy: 0.9996 - val_loss: 1.4867e-04 - val_accuracy: 1.0000
Epoch 3/5
481/481 [==============================] - 1622s 3s/step - loss: 0.0034 - accuracy: 0.9999 - val_loss: 2.5154e-05 - val_accuracy: 1.0000
Epoch 4/5
481/481 [==============================] - 1621s 3s/step - loss: 0.0024 - accuracy: 0.9999 - val_loss: 1.1104e-05 - val_accuracy: 1.0000
Epoch 5/5
481/481 [==============================] - 1623s 3s/step - loss: 0.0019 - accuracy: 0.9999 - val_loss: 4.3467e-06 - val_accuracy: 1.0000
```

Images with preprocessing



```
Epoch 1/10
274/274 [==============================] - 895s 3s/step - loss: 0.0434 - accuracy: 0.9912 - val_loss: 0.0112 - val_accuracy: 0.9977
Epoch 2/10
274/274 [==============================] - 895s 3s/step - loss: 0.0317 - accuracy: 0.9928 - val_loss: 0.0099 - val_accuracy: 0.9977
Epoch 3/10
274/274 [==============================] - 894s 3s/step - loss: 0.0242 - accuracy: 0.9942 - val_loss: 0.0065 - val_accuracy: 0.9986
Epoch 4/10
274/274 [==============================] - 894s 3s/step - loss: 0.0196 - accuracy: 0.9952 - val_loss: 0.0064 - val_accuracy: 0.9982
Epoch 5/10
274/274 [==============================] - 895s 3s/step - loss: 0.0169 - accuracy: 0.9961 - val_loss: 0.0043 - val_accuracy: 0.9986
Epoch 6/10
274/274 [==============================] - 895s 3s/step - loss: 0.0145 - accuracy: 0.9960 - val_loss: 0.0051 - val_accuracy: 0.9984
Epoch 7/10
274/274 [==============================] - 895s 3s/step - loss: 0.0113 - accuracy: 0.9974 - val_loss: 0.0037 - val_accuracy: 0.9989
Epoch 8/10
274/274 [==============================] - 895s 3s/step - loss: 0.0110 - accuracy: 0.9971 - val_loss: 0.0033 - val_accuracy: 0.9989
Epoch 9/10
274/274 [==============================] - 894s 3s/step - loss: 0.0097 - accuracy: 0.9972 - val_loss: 0.0051 - val_accuracy: 0.9986
Epoch 10/10
274/274 [==============================] - 895s 3s/step - loss: 0.0094 - accuracy: 0.9973 - val_loss: 0.0034 - val_accuracy: 0.9989
```

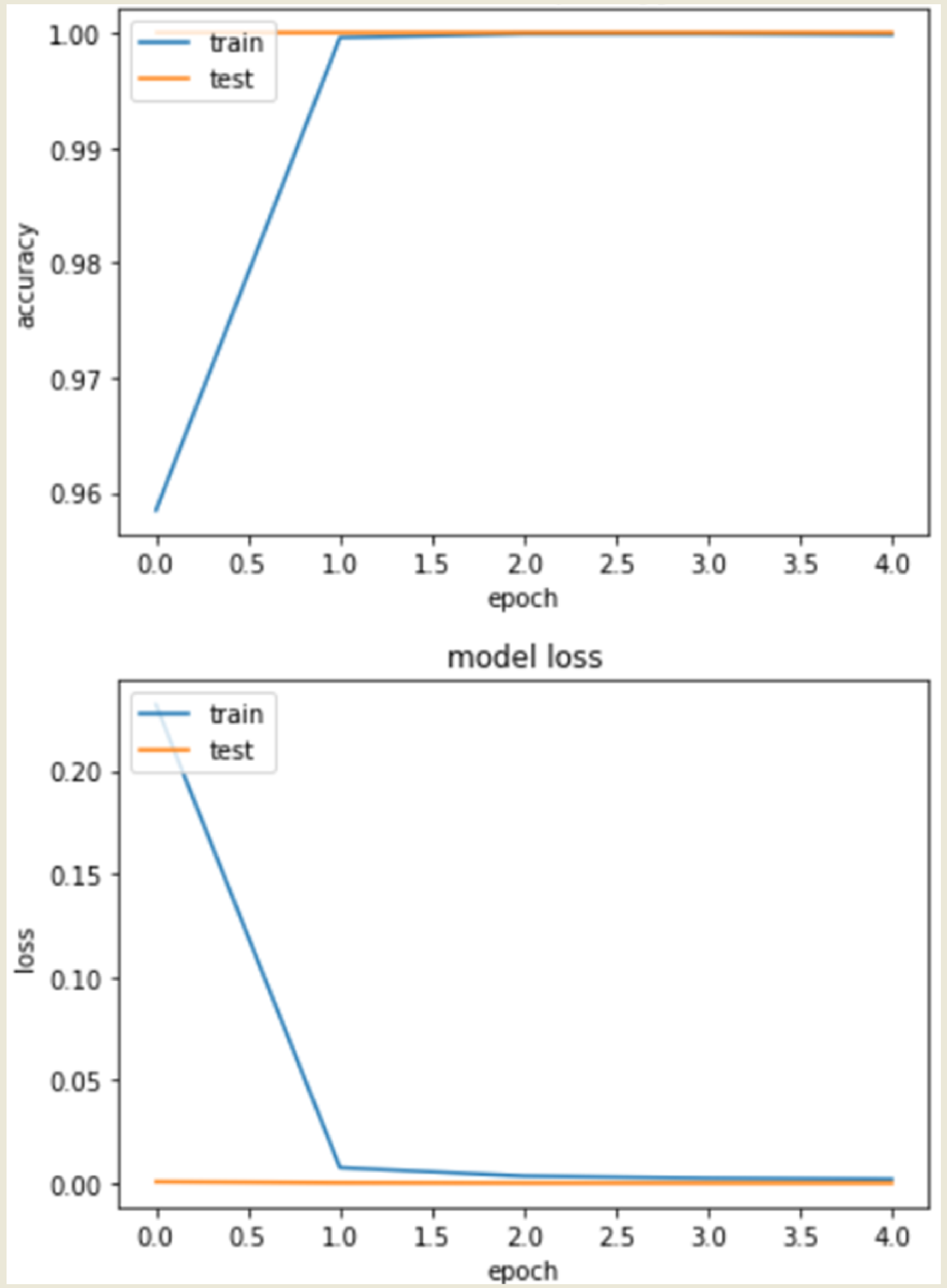
ACCURACY (RESNET50)

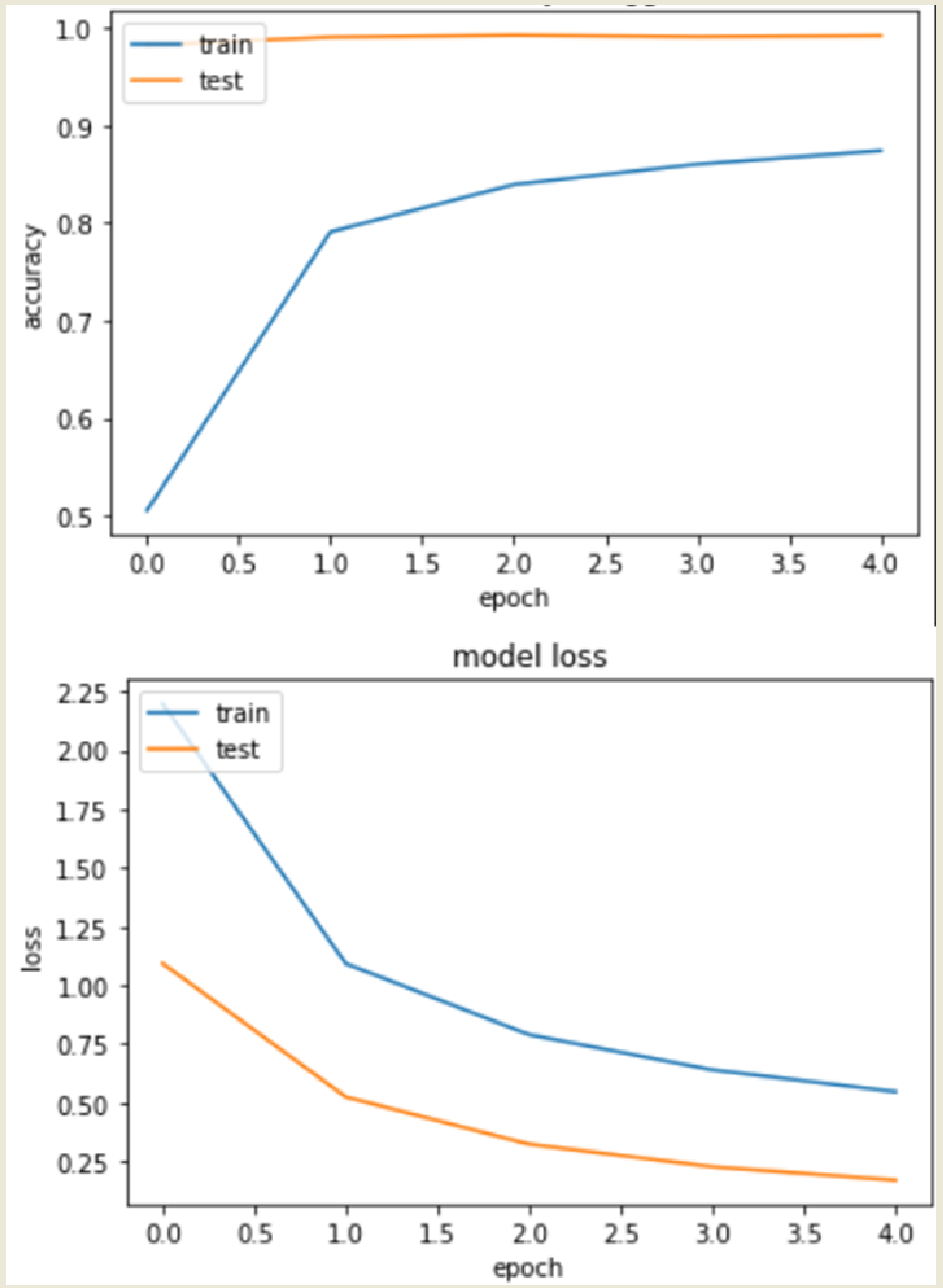


```
Epoch 1/5
481/481 [==============================] - 521s 1s/step - loss: 2.1987 - accuracy: 0.5053 - val_loss: 1.0933 - val_accuracy: 0.9825
Epoch 2/5
481/481 [==============================] - 517s 1s/step - loss: 1.0922 - accuracy: 0.7910 - val_loss: 0.5243 - val_accuracy: 0.9906
Epoch 3/5
481/481 [==============================] - 517s 1s/step - loss: 0.7895 - accuracy: 0.8394 - val_loss: 0.3234 - val_accuracy: 0.9927
Epoch 4/5
481/481 [==============================] - 517s 1s/step - loss: 0.6397 - accuracy: 0.8603 - val_loss: 0.2270 - val_accuracy: 0.9912
Epoch 5/5
481/481 [==============================] - 518s 1s/step - loss: 0.5466 - accuracy: 0.8741 - val_loss: 0.1701 - val_accuracy: 0.9922
```

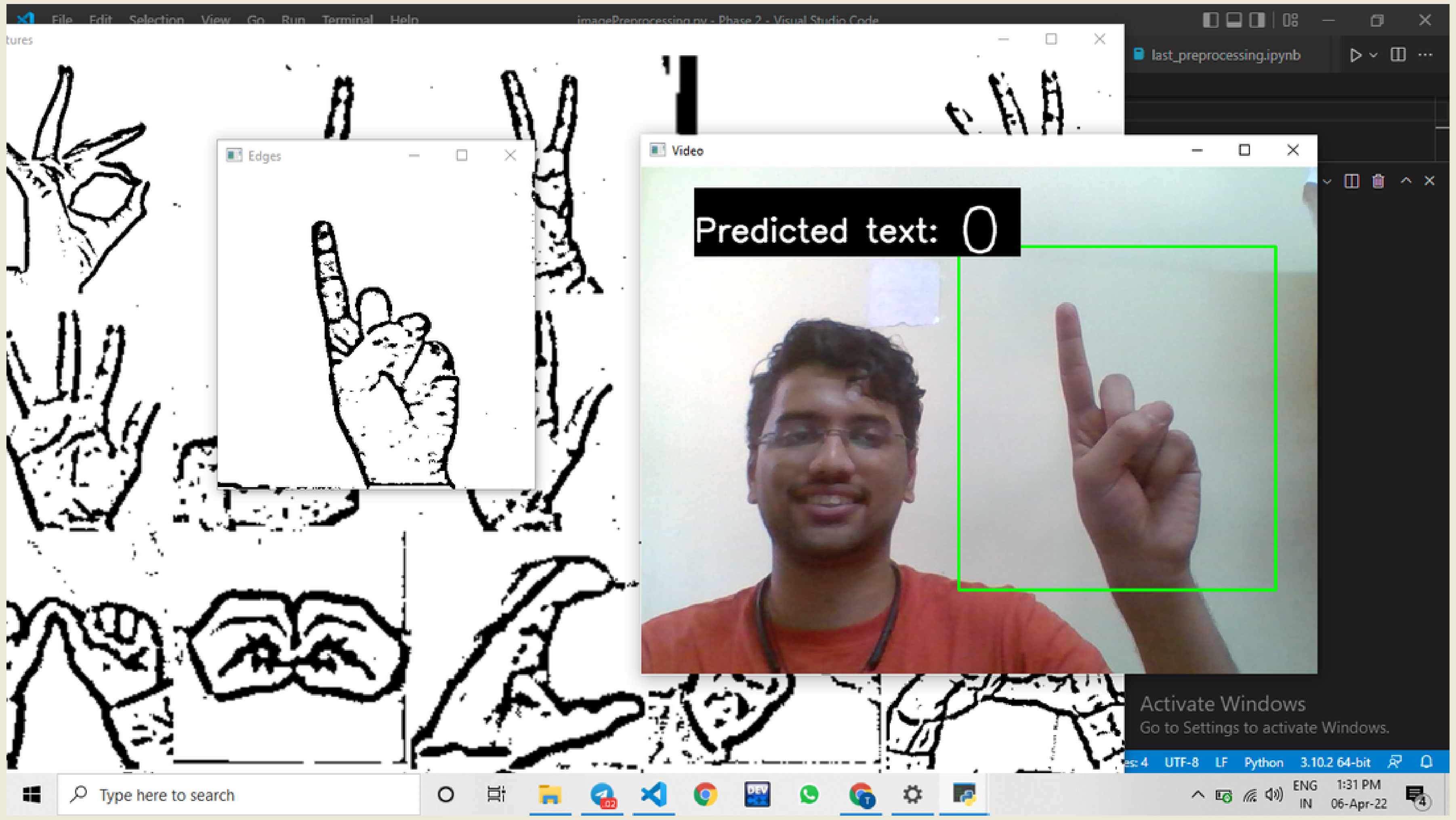
Model Accuracy plot for VGG16

Model Accuracy plot for ResNet50





PROBLEMS WE FACED



- The Preprocessing we chose last time didn't give us the results as expected.
- We had to find a suitable clustering for huge datasets. So we used minibatch k-means.
- No way to test real-time accuracy.

REAL TIME ACCURACY

Test 5 times for each alphanumeric character and calculate the total accuracy for each model.

$$\text{Accuracy} = \frac{\Sigma\,(\text{True Positives})[i]}{36*5}$$

where, i : 1 --> 36

Is there any suggestion to do this better?

CONCLUSION

We have learnt that the accuracy that we get while training the model is not a correct indicator of how good the model is for real-life implementation.

TODOS

- Reorganise all codes for pre-processing, training and live GUI using tkinter on github.
- Complete fine tuning of GUI and check live accuracy.
- Complete thesis work.

FACE MASK
DETOXIFY & BRIGHTEN
No Parabens, No MIT,
No Artificial Colours,
No Artificial Fragrances.
Vegan Friendly. Not tested on animals.

THANK YOU