****

**- Faculty of Computer and Information Sciences –**

**- AIN Shams University -**

**Computer Vision - SC**

**-----------------------------------------------------------**

**Handwritten Signature Identification and**

**Verification**

|  |  |
| --- | --- |
| **رقــــم الجلوس** | **الأســـــــم** |
| **20191700221** | **خالد احمد عبدالظاهر محمد** |
| **20191700685** | **ندى مجدى عبد الكريم بابا ادم** |
| **20191700040** | **احمد طارق فتحى محمد** |
| **20191700034** | **احمد سعيد نوح ابراهيم محمد** |
| **20191700029** | **احمد خالد احمد عبداللاه** |

**PART I – Object Classification**

**Stage 1**

**BOW**

# Data preparation process:

First, we read the data from the dataset folder and put the paths of images in train\_images and

test \_images lists

for folder in os.listdir(path):  
 for innerfolder in os.listdir(os.path.join(path, folder)):  
 for file in os.listdir(os.path.join(path, folder,innerfolder)):  
 if (file.\_\_contains\_\_(".png")):  
 if(innerfolder.\_\_contains\_\_("Train")):  
 train\_images.append(os.path.join(path, os.path.join(folder,innerfolder, file)))  
 else:  
 test\_images.append(os.path.join(path, os.path.join(folder,innerfolder, file)))

We create a BOW.py class in each of the train\_bow and test\_bow bow functions

def train\_bow(train\_images,encoded,NO\_OF\_CLUSTERS):

def test\_bow(test\_images,encoded,kmeans,scale,svm,NO\_OF\_CLUSTERS):

1. Read images
2. Get a descriptor list for each image
3. Gea t label for each image

# Brief Description of the technique:

## In stage 1 we apply BOW using SIFT

**In the training process:**

1. we flatten the descriptor lists for whole images “ using np.vstack()”
2. apply kmeans for clustering the descriptors ”10 clusters”
3. the computed histogram for each image
4. then apply a standard scaler for those histograms
5. Finally apply the classifier “SVM”

**In the test process:**

1. we flatten the descriptor lists for whole images “ using np.vstack()”
2. apply kmeans for predict the descriptors centers that belong to
3. then computed histogram for each image
4. then apply a standard scaler for those histograms
5. predict the labels of the test data ”using SVM classifier”
6. Finally calculate the accuracy of the testing process.

# Training and Testing time of BOW:

**Text

Description automatically generated**

# Training and Testing accuracy of BOW:

**A picture containing text, orange

Description automatically generated**

# Screenshots of the test sets classification with visualization:

**Text

Description automatically generated**

**Stage 1**

**CNN**

# Data preparation process:

**Text

Description automatically generated**

we create model helper class have a function get data that return x\_train, y\_train, x\_test, y\_test

then , we read the data from the dataset folder and put the paths of images in train\_images and

test \_images lists and csv files that contains images type “real or forged”

**Text

Description automatically generated**

# Brief Description of the technique:

## In stage 2 we apply CNN model with that architecture to predict either the signature is real or forged

**Text

Description automatically generated**

**Model A**

**Training and testing time and accuracy**

**Text

Description automatically generated**

**Model B**

**Training and testing time and accuracy**

**Text

Description automatically generated**

**Model C**

**Training and testing time and accuracy**

Text

Description automatically generated

**Model D**

**Training and testing time and accuracy**

Text

Description automatically generated

**Model E**

**Training and testing time and accuracy**

Text

Description automatically generated

**PART II – Object Detection  
Faster R-CNN**

**Read Data from Images Files**

Text

Description automatically generated

Iteration on Files in **TrainGroundTruth** ,  
read Line by line in each file and make it in Special Form

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1.tif | sign | 509 | 847 | 938 | 903 |

And all lines saved in **output.csv** that is ready to pass to Faster R-CNN Model

Table

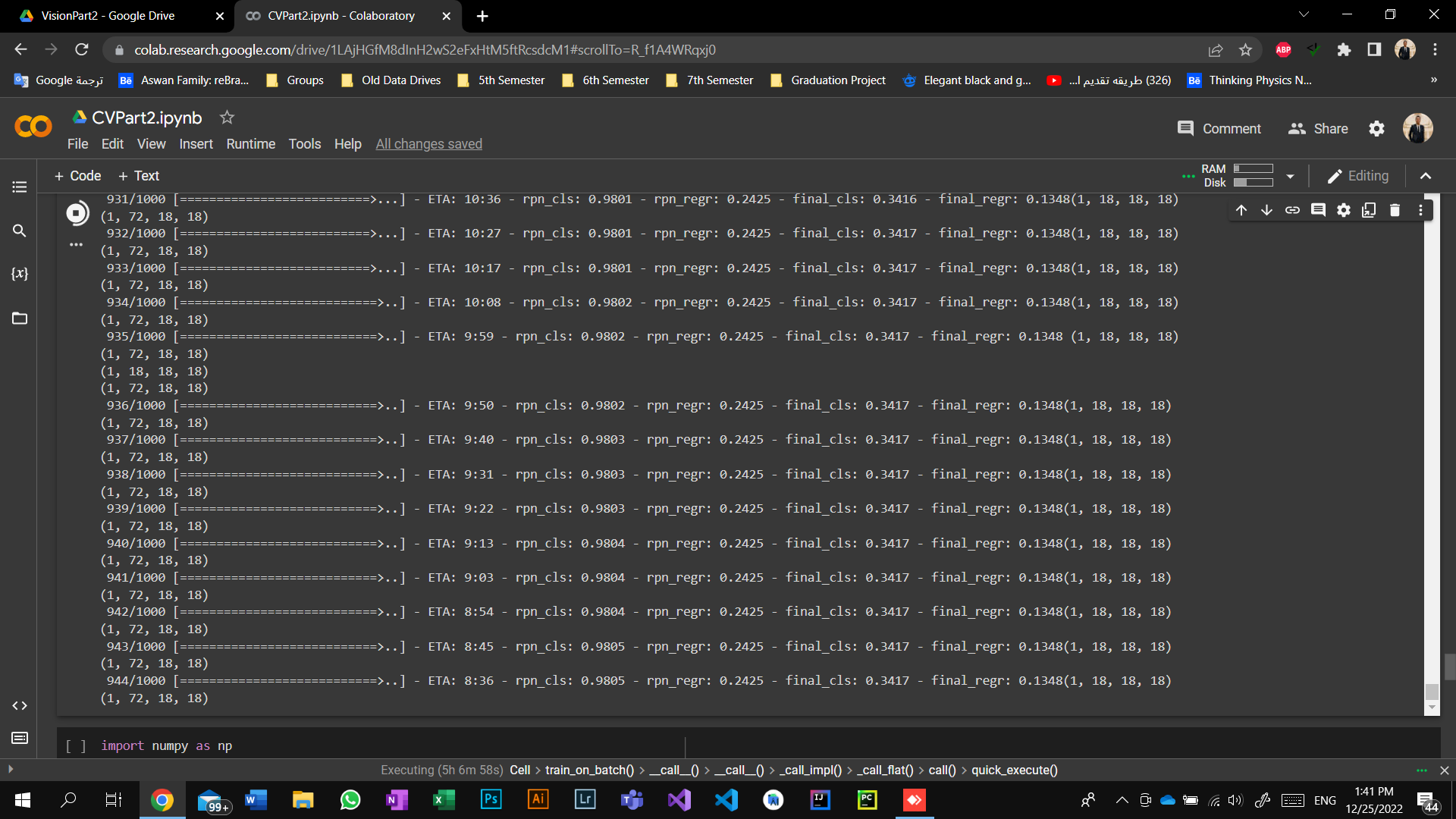
Description automatically generated

There may be More than one Signature in one Paper , Script can Record it as Highlighted Lines Above

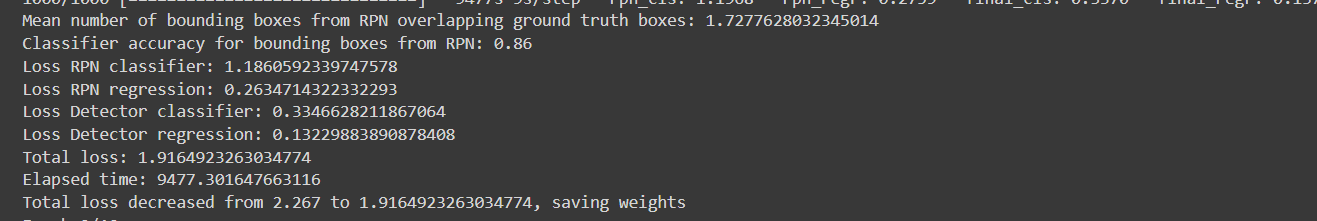
**Epochs**

We train Model on Google Colab, we train 10 Epochs and each epoch Length 1000

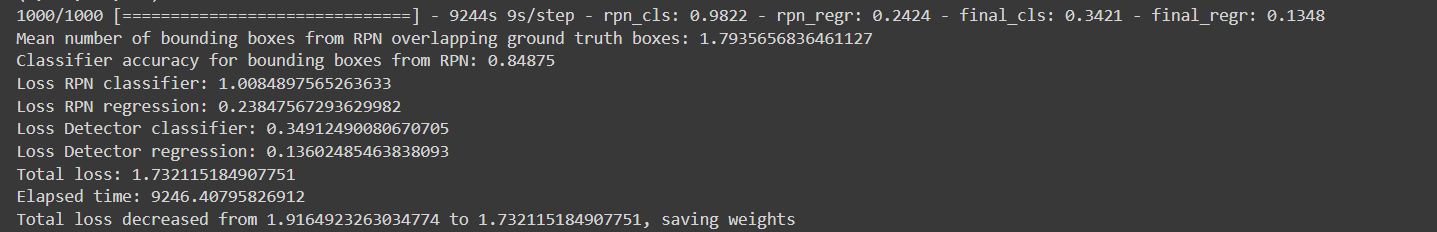
Changing all paths to new dataset path and all epochs take about 2.5 Hours to train



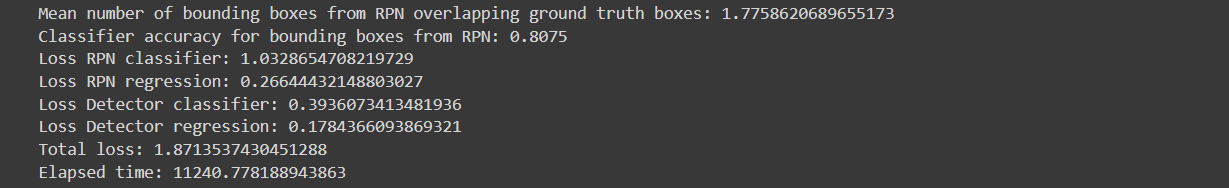
**Best Accuracy** was in epoch no. 8 was **0.86**



Epoch no. 9 .. Accuracy was **0.848**



Epoch no. 10 .. Accuracy was **0.8075**



All Epochs recorded in **record.csv** and **model\_frcnn\_vgg.hdf5** ,  
the model continue from you finished

**Performance metrics** can be consider as record.csv that save all Measures for each epoch

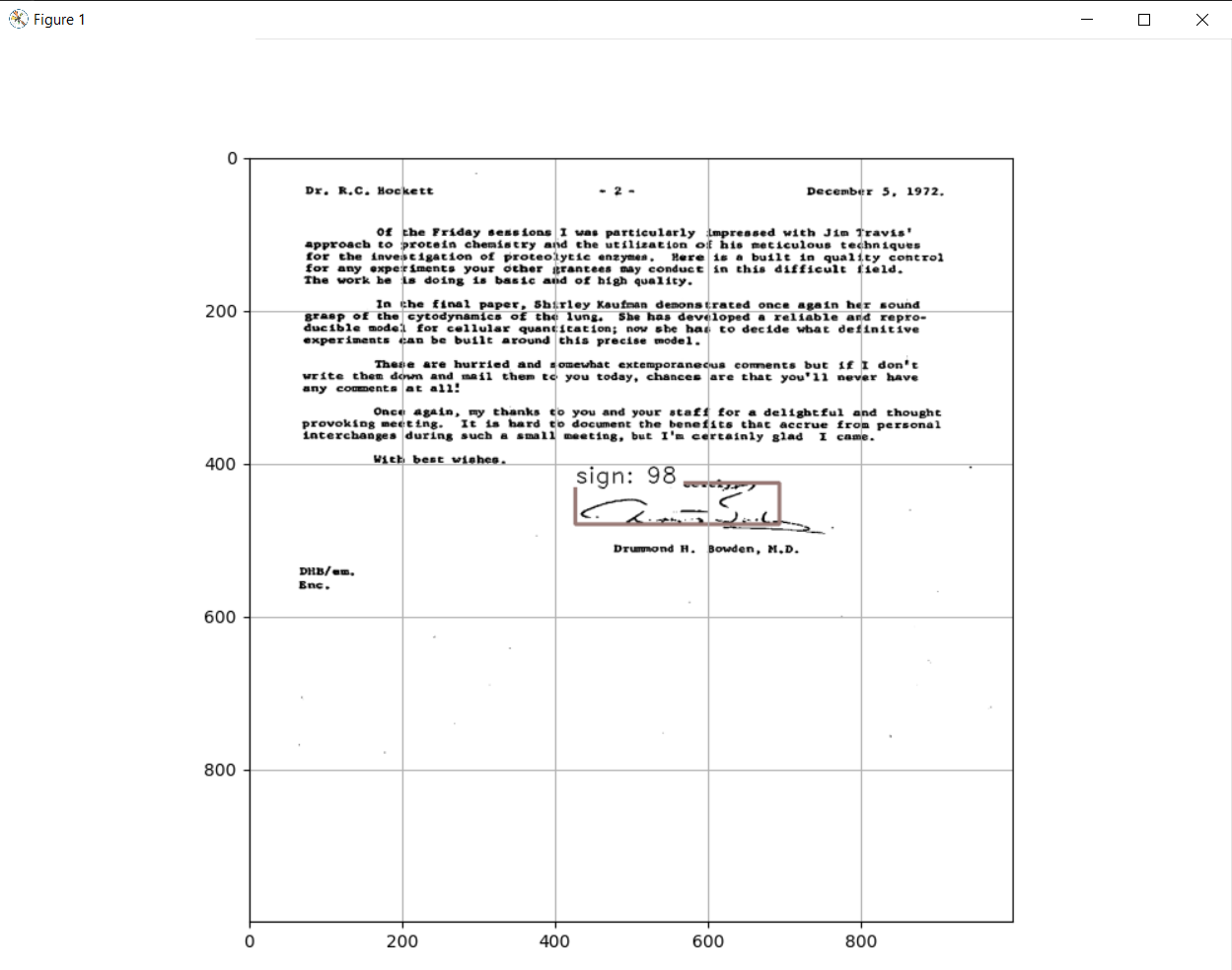
Table

Description automatically generated with medium confidence

In our detection Model we use iteration no.6 with accuracy **0.837** that real accuracy which appears Boundary box in detected signuature in test

**TESTING Phase**

We using box threshold = 0.9 to ignore all boxes with probability less than threshold



A picture containing diagram

Description automatically generated

And not all tested images detected Correct signature 100% .. may be cannot detect , there was best result with Faster R-CNN Model after 6 epochs trained

Diagram

Description automatically generated