**Identifying the digits**

**Problem Statement:**

Automatic digit recognition is of popular interest today. Deep Learning techniques makes it possible for object recognition in image data.

**Background:**

Ordinarily a CNN is comprised of three different layers namely convolutional layer, pooling layer

and fully-connected layer. When these layers are stacked together, a complete CNN architecture is

formed. The convolution layer is used to produce the activation map for all the features by

convolving or sliding a kernel or filter across every location of the pixel matrix of the input image.

This layer is essential to find out which feature exists especially in which part of the image. The

convolutional layer of the CNN determines the output of neurons connected to local regions of input

through the calculation of the scalar product between their weights and the regions connected to the

input volume. The rectified linear unit (commonly shortened to ReLU) is used to apply an

’elementwise’ activation function. The ReLU function is more advantageous than the sigmoid

activation since it decreases the possibility of vanishing gradient where the weight updating process

slows down significantly.

**Datasets:**

The data set used for this problem is from the popular MNIST data set. Developed by Yann LeCun,

Corina Cortes and Christopher Burger for evaluating machine learning model on the handwritten digit

classification problem. It is a widely used data set in the machine learning community.

**Python Libraries:**

* Tensorflow
* openCV
* Keras
* Pandas
* Matplotlib
* numpy

**CODE:**

%pylab inline

import os

import numpy as np

import pandas as pd

from scipy.misc import imread

from sklearn.metrics import accuracy\_score

import tensorflow as tf

import keras

*# To stop potential randomness*

seed = 128

rng = np.random.RandomState(seed)

root\_dir = os.path.abspath('../..') data\_dir = os.path.join(root\_dir, 'Deep Learning') sub\_dir = os.path.join(root\_dir, 'sub') *# check for existence* os.path.exists(root\_dir) os.path.exists(data\_dir) os.path.exists(sub\_dir)

train = pd.read\_csv(os.path.join(data\_dir,'Identify\_The\_Digits' ,'train.csv'))

test = pd.read\_csv(os.path.join(data\_dir, 'Identify\_The\_Digits','test.csv'))

sample\_submission = pd.read\_csv(os.path.join(data\_dir,'Identify\_The\_Digits', 'Sample\_Submission.csv'))

train.head()

test.head()

img\_name = rng.choice(train.filename)

filepath = os.path.join(data\_dir, 'Identify\_The\_Digits', 'images', 'train', img\_name)

img = imread(filepath, flatten=True)

pylab.imshow(img, cmap='gray')

pylab.axis('off')

pylab.show()

temp = []

for img\_name in train.filename:

image\_path = os.path.join(data\_dir, 'Identify\_The\_Digits', 'images', 'train', img\_name)

img = imread(image\_path, flatten=True)

img = img.astype('float32')

temp.append(img)

train\_x = np.stack(temp)

train\_x /= 255.0

train\_x = train\_x.reshape(-1, 784).astype('float32')

temp = []

for img\_name in test.filename:

image\_path = os.path.join(data\_dir, 'Identify\_The\_Digits', 'images' ,'test', img\_name)

img = imread(image\_path, flatten=True)

img = img.astype('float32')

temp.append(img)

test\_x = np.stack(temp)

test\_x /= 255.0

test\_x = test\_x.reshape(-1, 784).astype('float32')

train\_y = keras.utils.np\_utils.to\_categorical(train.label.values)

split\_size = int(train\_x.shape[0]\*0.7) train\_x, val\_x = train\_x[:split\_size], train\_x[split\_size:] train\_y, val\_y = train\_y[:split\_size], train\_y[split\_size:]

train\_x.shape

*# define vars*

input\_num\_units = 784

hidden\_num\_units = 50

output\_num\_units = 10

epochs = 5

batch\_size = 128

from keras.models import Sequential

from keras.layers import Dense

model = Sequential

model.add(Conv2D(28,kernel\_size = (3,3),input\_shape = (28,28,1)))

model.add(MaxPooling2D(pool\_size = (2,2)))

model.add(Conv2D(28,kernel\_size = (3,3),input\_shape = (28,28,1)))

model.add(MaxPooling2D(pool\_size = (2,2)))

model.add(Conv2D(28,kernel\_size = (3,3),input\_shape = (28,28,1)))

model.add(MaxPooling2D(pool\_size = (2,2)))

model.add(Flatten())

model.add(Dense(128,activation = tf.nn.relu))

model.add(Dropout(0.2))

model.add(Dense(10, activation = tf.nn.softmax))

model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])

model.summary()

trained\_model = model.fit(train\_x, train\_y, nb\_epoch=epochs, batch\_size=batch\_size, validation\_data=(val\_x, val\_y))

pred = model.predict\_classes(test\_x)

img\_name = rng.choice(test.filename)

filepath = os.path.join(data\_dir, 'Identify\_The\_Digits', 'images', 'test', img\_name)

img = imread(filepath, flatten=True)

test\_index = int(img\_name.split('.')[0]) - train.shape[0]

print("Prediction is: ", pred[test\_index])

pylab.imshow(img, cmap='gray')

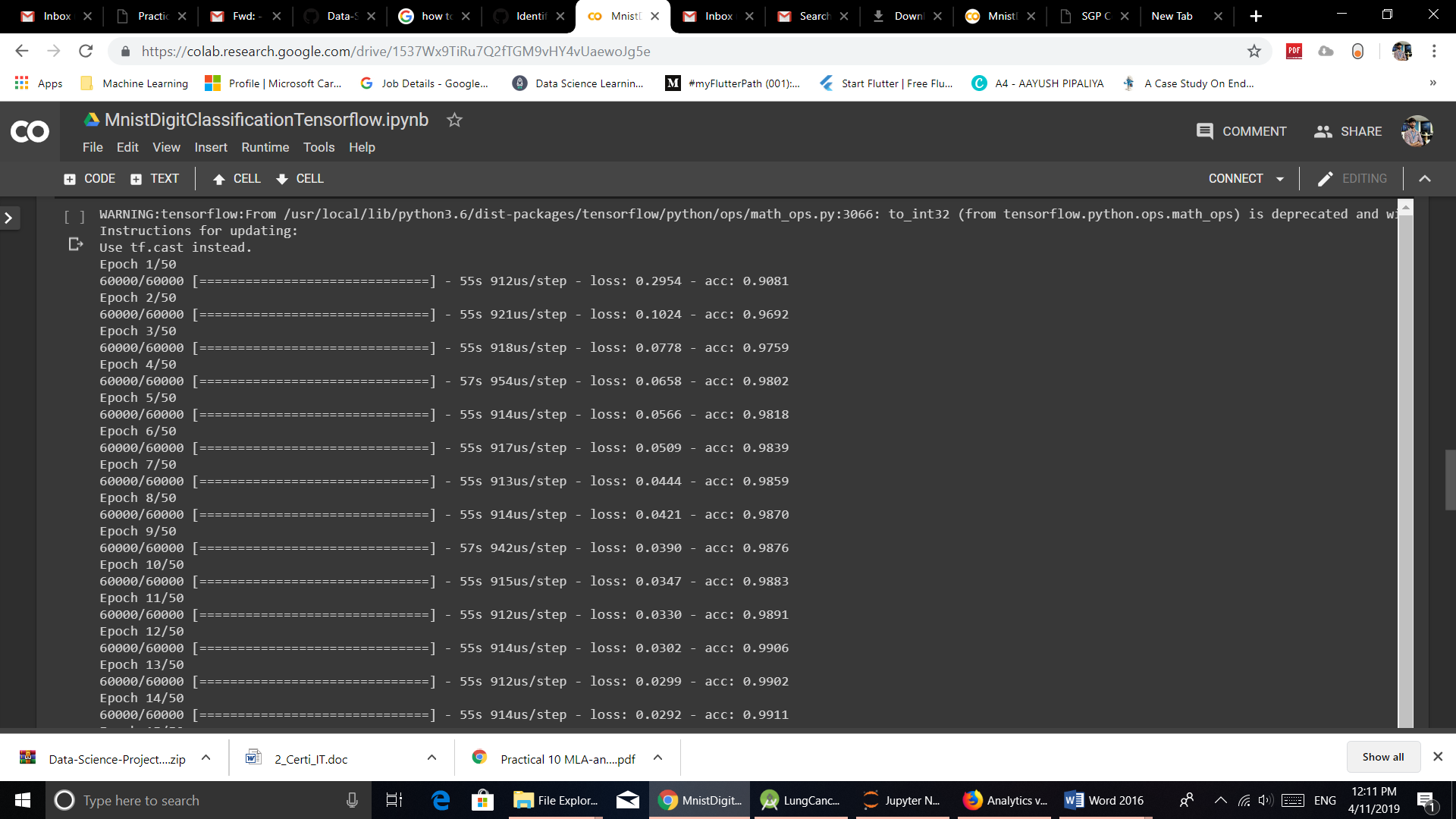
pylab.axis('off')

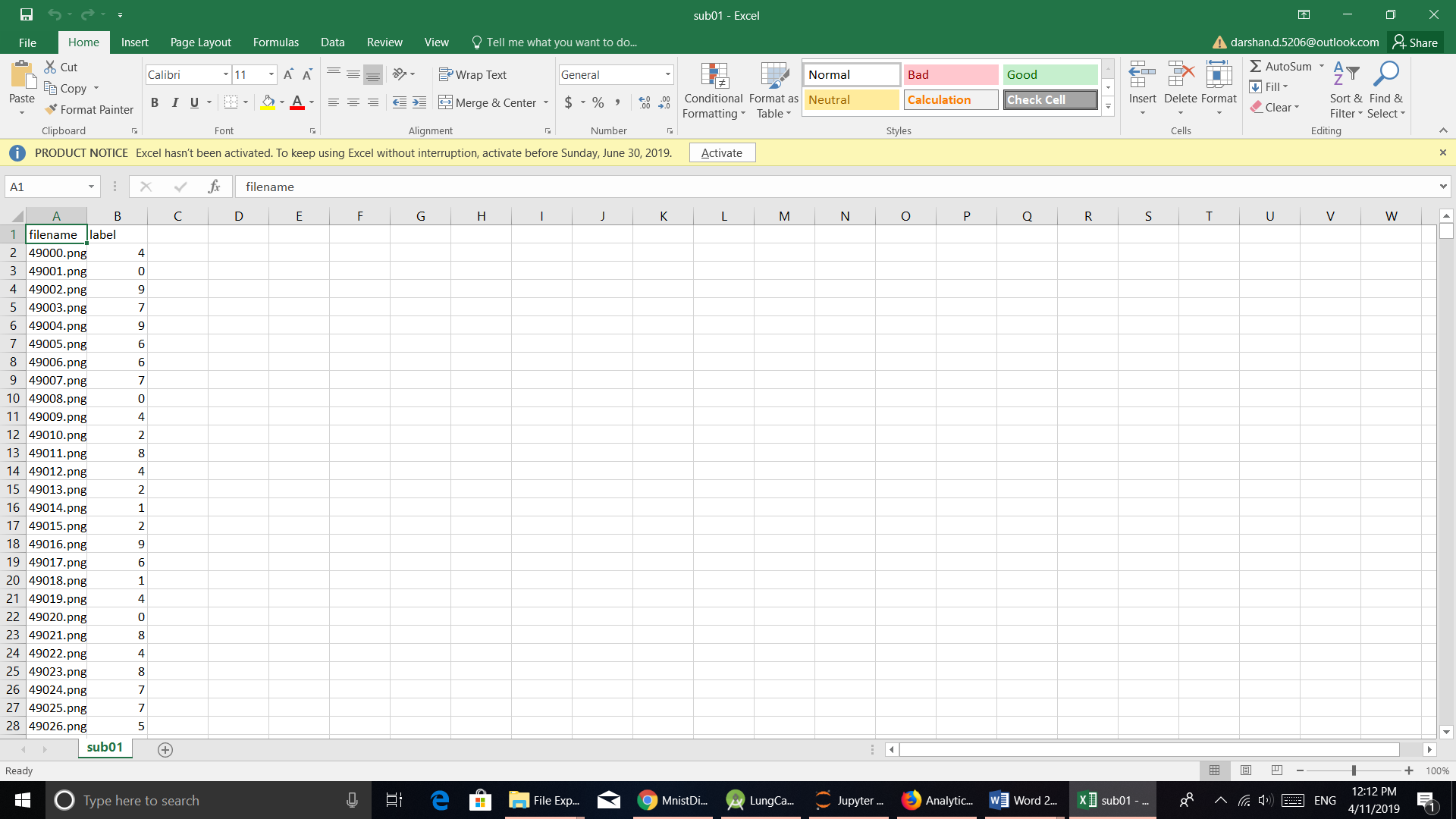
pylab.show()

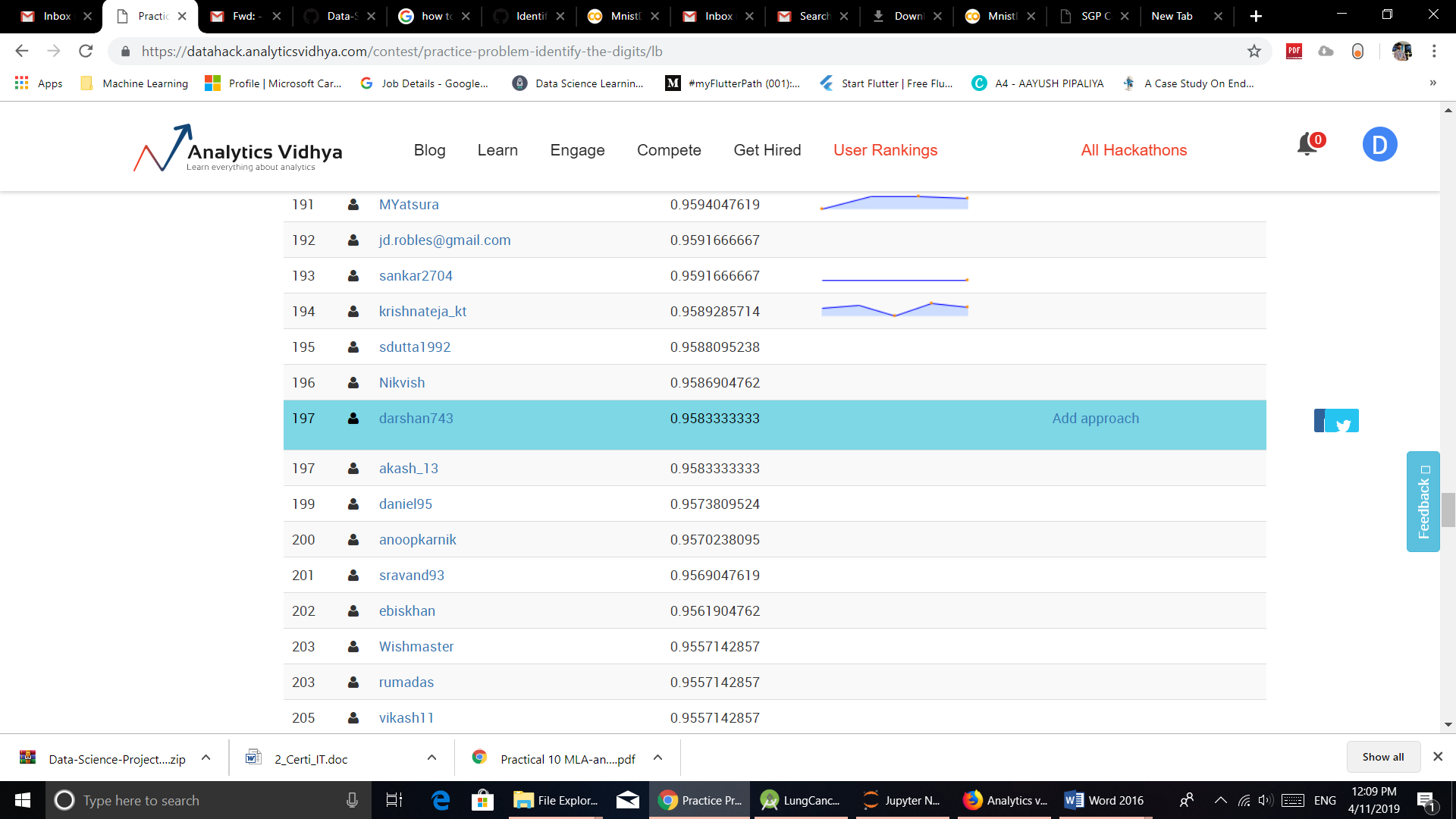
sample\_submission.filename = test.filename; sample\_submission.label = pred

sample\_submission.to\_csv(os.path.join(data\_dir, 'sub02.csv'), index=False)

**OUTPUT:**







**CONCLUSION:** We have successfully created an application which detects digits from paper.