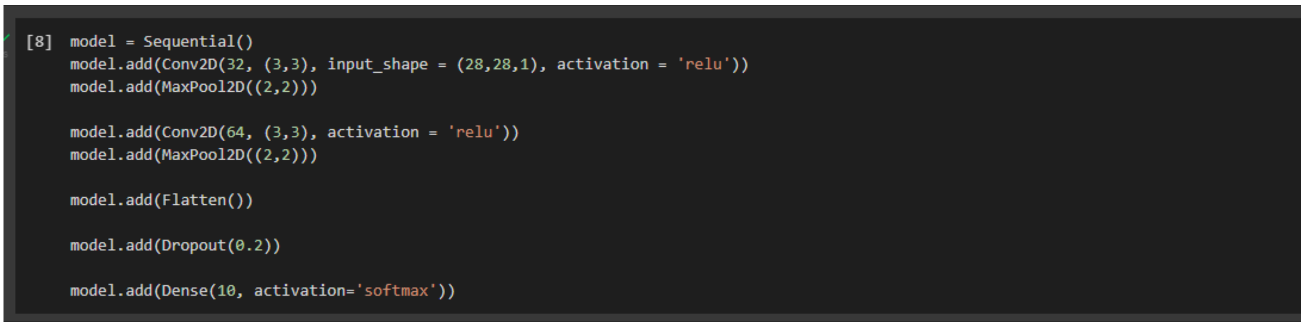
Project Development Phase
Delivery of Sprint-2

Date	**05.11.2022**
Team ID	PNT2022TMID53601
Project Name	Project - A Novel Method for Handwritten Digit Recognition System

Add CNN Layers

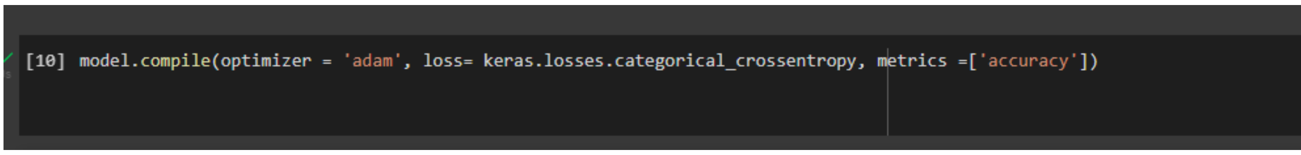
1. Create a new variable with the Sequential model and add the required CNN layers to it.



```
[8]  model = Sequential()
     model.add(Conv2D(32, (3,3), input_shape = (28,28,1), activation = 'relu'))
     model.add(MaxPool2D((2,2)))

     model.add(Conv2D(64, (3,3), activation = 'relu'))
     model.add(MaxPool2D((2,2)))

     model.add(Flatten())

     model.add(Dropout(0.2))

     model.add(Dense(10, activation='softmax'))
```

Compiling the model

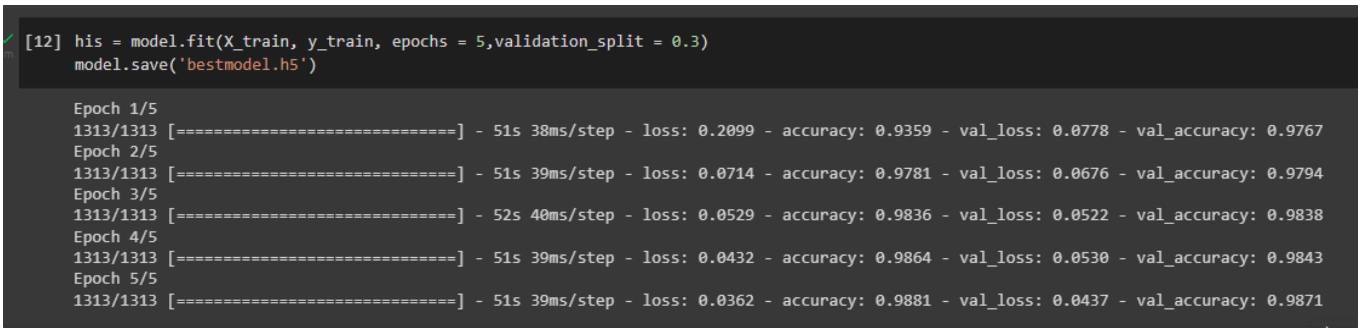
2. Compile the model using Adam optimizer.



```
[10]  model.compile(optimizer = 'adam', loss= keras.losses.categorical_crossentropy, metrics =['accuracy'])
```

Train the model

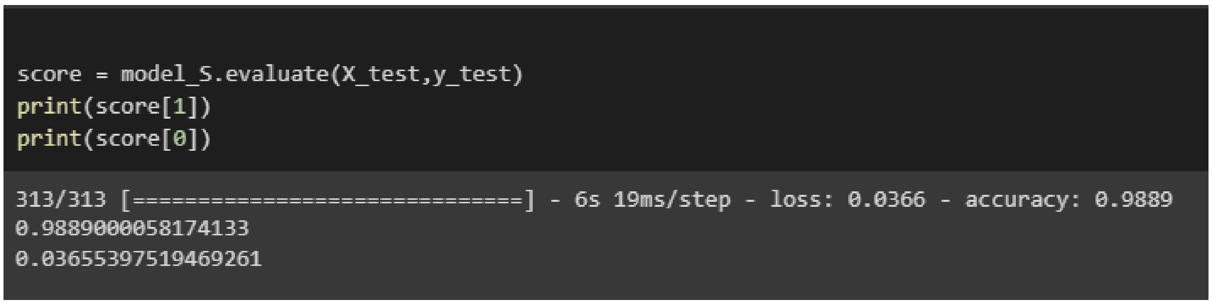
3. Train the model for at least 5 epochs.



```
[12]  his = model.fit(X_train, y_train, epochs = 5,validation_split = 0.3)
      model.save('bestmodel.h5')

      Epoch 1/5
      1313/1313 [==============================] - 51s 38ms/step - loss: 0.2099 - accuracy: 0.9359 - val_loss: 0.0778 - val_accuracy: 0.9767
      Epoch 2/5
      1313/1313 [==============================] - 51s 39ms/step - loss: 0.0714 - accuracy: 0.9781 - val_loss: 0.0676 - val_accuracy: 0.9794
      Epoch 3/5
      1313/1313 [==============================] - 52s 40ms/step - loss: 0.0529 - accuracy: 0.9836 - val_loss: 0.0522 - val_accuracy: 0.9838
      Epoch 4/5
      1313/1313 [==============================] - 51s 39ms/step - loss: 0.0432 - accuracy: 0.9864 - val_loss: 0.0530 - val_accuracy: 0.9843
      Epoch 5/5
      1313/1313 [==============================] - 51s 39ms/step - loss: 0.0362 - accuracy: 0.9881 - val_loss: 0.0437 - val_accuracy: 0.9871
```

Observing the metrics

4. Observe the accuracy and losses incurred in the model

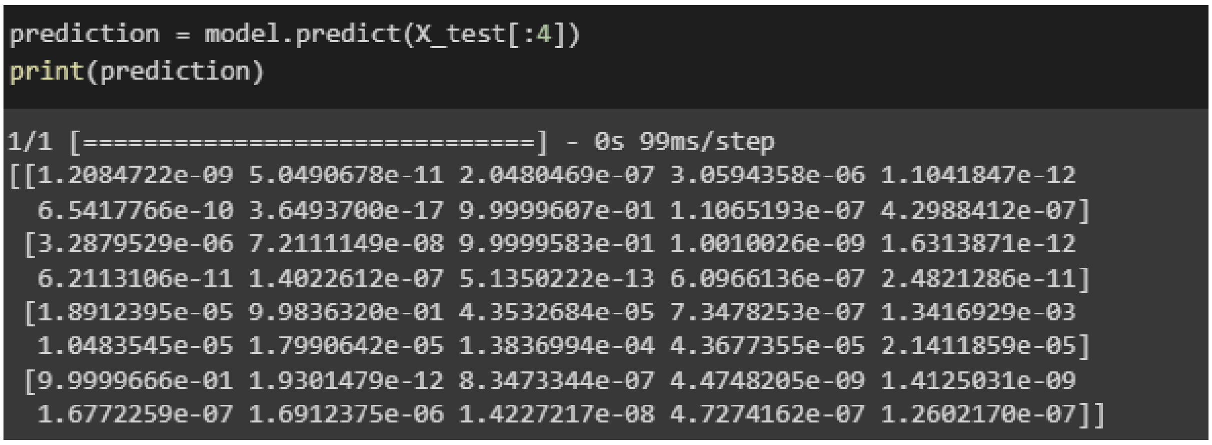


```
score = model_S.evaluate(X_test,y_test)
print(score[1])
print(score[0])
```

```
313/313 [==============================] - 6s 19ms/step - loss: 0.0366 - accuracy: 0.9889
0.9889000058174133
0.03655397519469261
```

Test the model

5. Test the model using X_test dataset images

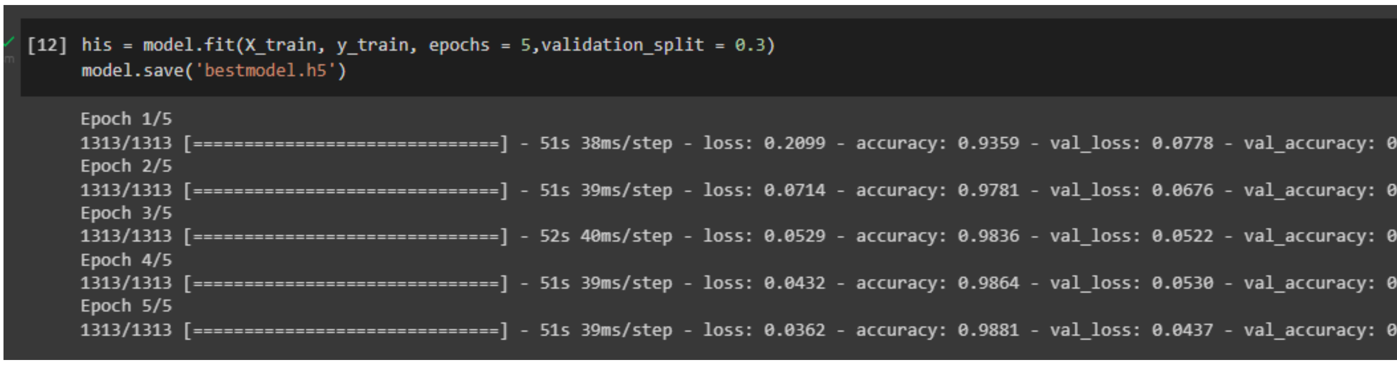


```
prediction = model.predict(X_test[:4])
print(prediction)
```

```
1/1 [==============================] - 0s 99ms/step
[[1.2084722e-09 5.0490678e-11 2.0480469e-07 3.0594358e-06 1.1041847e-12
  6.5417766e-10 3.6493700e-17 9.9999607e-01 1.1065193e-07 4.2988412e-07]
 [3.2879529e-06 7.2111149e-08 9.9999583e-01 1.0010026e-09 1.6313871e-12
  6.2113106e-11 1.4022612e-07 5.1350222e-13 6.0966136e-07 2.4821286e-11]
 [1.8912395e-05 9.9836320e-01 4.3532684e-05 7.3478253e-07 1.3416929e-03
  1.0483545e-05 1.7990642e-05 1.3836994e-04 4.3677355e-05 2.1411859e-05]
 [9.9999666e-01 1.9301479e-12 8.3473344e-07 4.4748205e-09 1.4125031e-09
  1.6772259e-07 1.6912375e-06 1.4227217e-08 4.7274162e-07 1.2602170e-07]]
```

Save the model

6. Save the model in the name of “bestmodel.h5” file.

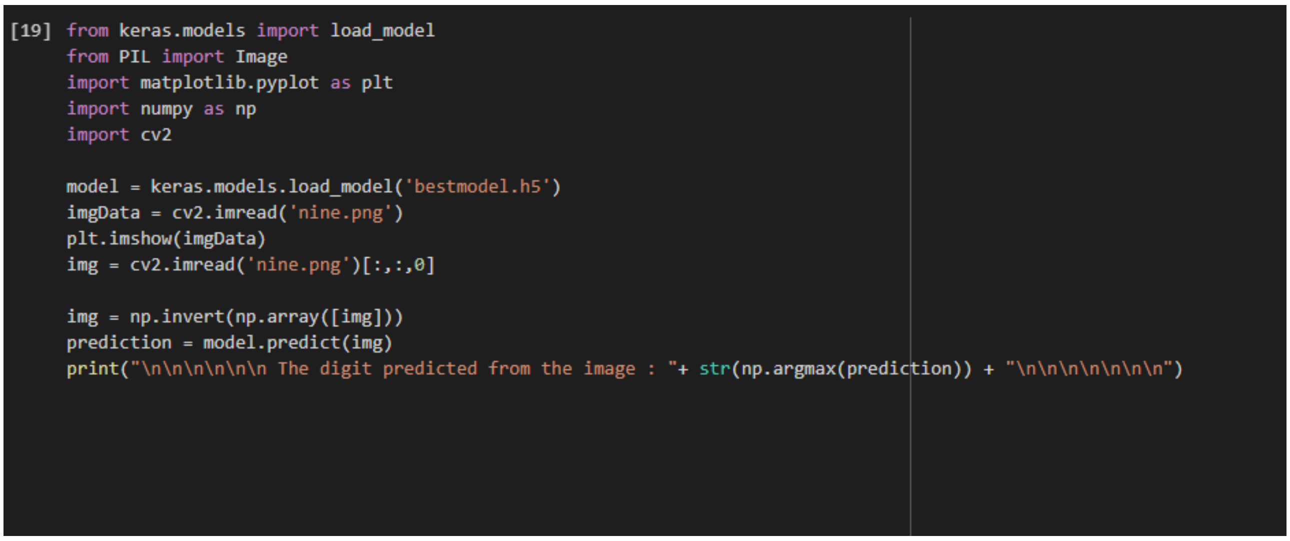


```
[12] his = model.fit(X_train, y_train, epochs = 5,validation_split = 0.3)
     model.save('bestmodel.h5')
```

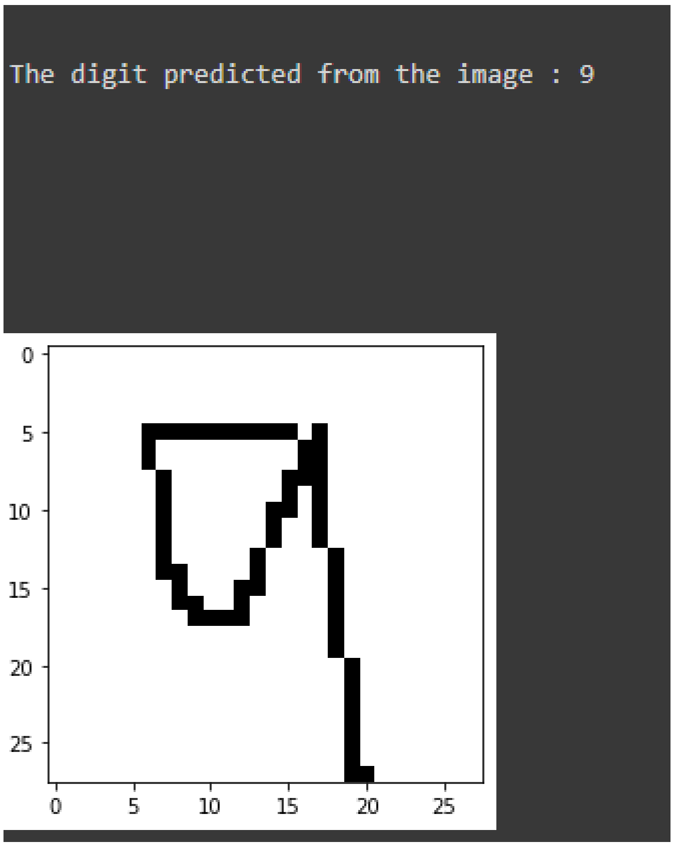
```
Epoch 1/5
1313/1313 [==============================] - 51s 38ms/step - loss: 0.2099 - accuracy: 0.9359 - val_loss: 0.0778 - val_accuracy: 0
Epoch 2/5
1313/1313 [==============================] - 51s 39ms/step - loss: 0.0714 - accuracy: 0.9781 - val_loss: 0.0676 - val_accuracy: 0
Epoch 3/5
1313/1313 [==============================] - 52s 40ms/step - loss: 0.0529 - accuracy: 0.9836 - val_loss: 0.0522 - val_accuracy: 0
Epoch 4/5
1313/1313 [==============================] - 51s 39ms/step - loss: 0.0432 - accuracy: 0.9864 - val_loss: 0.0530 - val_accuracy: 0
Epoch 5/5
1313/1313 [==============================] - 51s 39ms/step - loss: 0.0362 - accuracy: 0.9881 - val_loss: 0.0437 - val_accuracy: 0
```

Test with the saved model

7. Test the saved model by adding an image from google drive or from your local drive and predict the result.



```
[19] from keras.models import load_model
     from PIL import Image
     import matplotlib.pyplot as plt
     import numpy as np
     import cv2

     model = keras.models.load_model('bestmodel.h5')
     imgData = cv2.imread('nine.png')
     plt.imshow(imgData)
     img = cv2.imread('nine.png')[:,:,0]

     img = np.invert(np.array([img]))
     prediction = model.predict(img)
     print("\n\n\n\n\n\n The digit predicted from the image : "+ str(np.argmax(prediction)) + "\n\n\n\n\n\n\n")
```



```
The digit predicted from the image : 9
```