PROJECT DEVELOPMENT PHASE

SPRINT -2 - MODEL BUILDING

DATE	07 NOVEMBER 2022
TEAM ID	PNT2022TMID16587
PROJECT TITLE	A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

MODEL BUILDING

Adding CNN layers

```
model=Sequential()
model.add(Conv2D(64,(3,3),input_shape=(28,28, 1) activation='relu' ) )
model.add(Conv2D(32,(3,3),activation='relu'))
model.add(Flatten())
model.add(Dense(number_of_classes,activation='softmax'))
```

Compiling the model

```
model.compile(loss='categorical_crossentropy ' optimizer='Adanm 'metrics=['accuracy'])
```

Train the model

```
model.fit(X_train,y_train,validation_data=(X_test,y_test), epochs=5,batch_size=32)
Epoch 1/5
1875/1875 [==============================] 175s 93ms/step - loss: 0.2502 -
Epoch 2/5
1875/1875 [==============================] 172s 92ms/step - loss: 0.0696 -
Epoch 3/5
1875/1875 [==============================] 171s 91ms/step - loss: 0.0489 -
Epoch 4/5
1875/1875 [==============================] 170s 91ms/step - loss: 0.0364 -
Epoch 5/5
```

1875/1875 [==============================] 170s 91ms/step _ loss: 0.0289 _

<keras.callbacks.History at Ox7f782c88b350>

Observing the metrics

```
metrics=model.evaluate(X_test,y_test,ver
bose=0) print("Metrics(Test loss & Test
Accuracy):")
print(nmetrics)
```

Metrics(Test loss & Test Accuracy):

[0.09209173172712326, 0.9804999828338623]

Test the model

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

1/1 [==============================] - Os 84ms/step

[(2.7120884e-11 5.1209537e-19 1.6880208e-13 3.8784922e-10 9.2105196e-19
1.6229773e-19 3.1850319e-23 1.OOOOOOOe+OO 4.3093339e-18 8.2710392e-12]
[2.5271413e-07 7.8410173e-10 9.9999595e-01 2.7106433e-12 7.8350200e-18
2.3051807e-15 3.8526869e-06 5.7907921e-19 9.8714995e-11 3.0523931e-20]
[6.1399518e-13 9.9996412e-01 9.4697121e-08 4.7779276e-12 3.5437788e-05
2.0282629e-07 5.2412124e-09 2.0024801e-10 5.9887626e-08 6.6694358e-12]
[I.OOOOOOOe+OO 1.7255879e-20 9.9291560e-14 3.0031913e-20 1.5109380e-16
3.4246311e-14 9.9780113e-09 4.0162242e-16 1.1263576e-13 6.4703303e-12]]

```
import numpy as np
print(np.argmax(prediction,axis=1))
print(y_test[:4])
```

[7 2 1 0]
[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
[0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
[0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]

Observing the metrics

```
metrics=model.evaluate(X_test,y_test,verbose=O) print("Metrics(Test loss & Test Accuracy):") print(metrics)
```

```
Metrics(Test loss & Test Accuracy):
[0.09209173172712326, 0.9804999828338623]
```

Test the model

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

```
1/1 [==============================] - Os 21ms/step
[[2.7120884e-11 5.1209537e-19 1.6880208e-13 3.8784922e-10 9.2105196e-19
1.6229773e-19 3.1850319e-23 1.OOOOOOOe+OO 4.3093339e-18 8.2710392e-12]
[2.5271413e-07 7.8410173e-10 9.9999595e-01 2.7106433e-12 7.8350200e-18
2.3051807e-15 3.8526869e-06 5.7907921e-19 9.8714995e-11 3.0523931e-20]
[6.1399518e-13 9.9996412e-01 9.4697121e-08 4.7779276e-12 3.5437788e-05
2.0282629e-07 5.2412124e-09 2.0024801e-10 5.9887626e-08 6.6694358e-12]
[I.OOOOOOOe+OO 1.7255879e-20 9.9291560e-14 3.0031913e-20 1.5109380e-16
3.4246311e-14 9.9780113e-09 4.0162242e-16 1.1263576e-13 6.4703303e-12]]
```

```
import numpy as np
print(np.argmax(prediction,axis=l))
print(y_test[:4])
```

```
[7 2 1 0]
[O O. o. o. O. o. o. 1. o. o.]
O. O.
O. 1. 1. o. O. o. o. o. o. o.]
1. O.
o. o. O. O. O. o. o. 0.1
o. o. O. o. o. o. o. O .
```

Save the model

```
model.save('models/mnistCNN.h5')
```

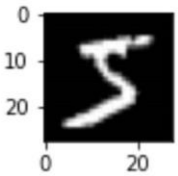
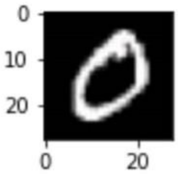
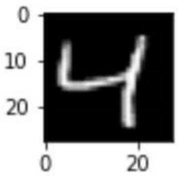
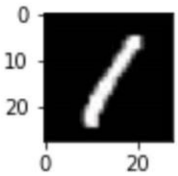
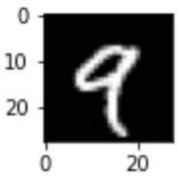
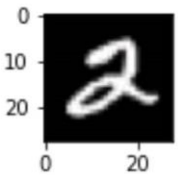
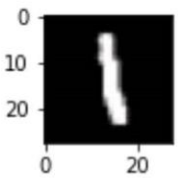
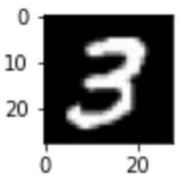
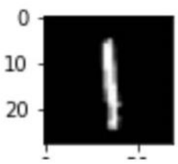
Test with saved model

```
from tensorflow.keras.models import load_model
model=load_model(r'/content/models/mnistCNN.h5') from PIL import Image
import numpy as np for index in range(4):
img=Image.open('/content/models/sample.png').convert("L") img=img.resize((28,28))
im2arr=np.array(img) im2arr=im2arr.reshape(1,28,28,1)
y_pred=model.predict(im2arr) print(y_pred)
```

```
1/1 [==============================] - Os 61ms/step
    [(5.59648324e-05 8.66386131e-07 2.32047445e-04 4.33623005e-04 1.88246977e05
    1.16871546e-04 6.67498807e-06 8.87498800e-07
    1.15397806e-05 9.99122679e-01]]
1/1 [==============================]- Os 21ms/step
    [(5.59648324e-05 8.66386131e-07 2.32047445e-04 4.33623005e-04 1.88246977e05
    1.16871546e-04 6.67498807e-06 8.87498800e-07
    1.15397806e-05 9.99122679e-01]]
1/1 [==============================]- Os 19ms/step
    [15.59648324e-05 8.66386131e-07 2.32047445e-04 4.33623005e-04 1.88246977e05
    1.16871546e-04 6.67498807e-06 8.87498800e-07
    1.15397806e-05 9.99122679e-01]]
1/1 [==============================] - Os 20ms/step
    [(5.59648324e-05 8.66386131e-07 2.32047445e-04 4.33623005e-04 1.88246977e05
    1.16871546e-04 6.67498807e-06 8.87498800e-07
    1.15397806e-05 9.99122679e-01]]
```

```
from keras.datasets import mnist from matplotlib import pyplot
(X_train,y_train),(X_test,y_test)=mnist.load_data() print('X_train: '
+str(X_train.shape)) print('y_train:' +str(y_train.shape)) print('X_test:'
+str(X_test.shape)) print('y_test:' +str(y_test.shape)) from matplotlib import pyplot for
i in range(9):
pyplot.subplot(330+1+i)
pyplot.imshow(X_train[i],cmap=pyplot.get_cmap('gray'))  pyplot.show()
```

```
X_train:(60000,28, 28) y_train: (60000,)
X_test:(10000,28, 28)
y_test: (10000,)
```


0
10
20
0
20

0
10
20
0
20

0
10
20
0
20

0
10
20
0
20

0
10
20
0
20

0
10
20
0
20

0
10
20
0
20

0
10
20
0
20

0
10
20