Ex No: 4 **HANDWRITTEN DIGITS RECOGNITION WITH MNIST**

Aim:

To build a handwritten digit's recognition with MNIST dataset.

Procedure:

1. Download and load MNIST the dataset.
2. Perform analysis and preprocessing of the dataset.
3. Build a convolutional neural network model using Keras/TensorFlow.
4. Compile and fit the model.
5. Perform prediction with the test dataset.
6. Calculate performance metrics.

Program:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import tensorflow as tf

import keras

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense,Dropout, Flatten, Conv2D, MaxPooling2D,
BatchNormalization, LeakyReLU, Activation

from tensorflow.keras.optimizers import SGD,Adam

import warnings

warnings.filterwarnings('ignore')

from tensorflow.keras.datasets import mnist

(X_train,y_train),(X_test,y_test)=mnist.load_data()
```

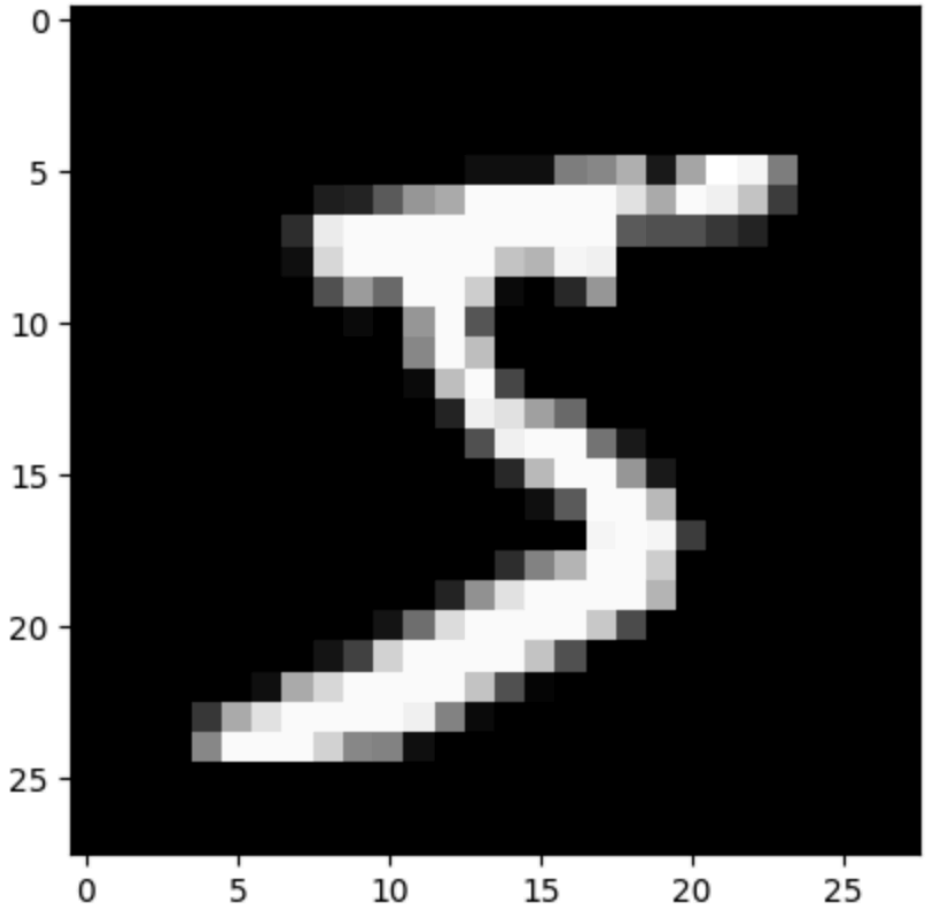
```
plt.imshow(X_train[0],cmap=plt.get_cmap('gray'))
plt.figure(figsize=(8,8))
j=1
for i in range(0,9):
  plt.subplot(3,3,j)
  plt.imshow(X_train[i],cmap=plt.get_cmap('gray'))
  plt.title(y_train[i],color='red')
  j=j+1
X_train=X_train.reshape((X_train.shape[0],28,28,1))
X_test=X_test.reshape((X_test.shape[0],28,28,1))
X_train_norm=X_train.astype('float32')/255.0
X_test_norm=X_test.astype('float32')/255.0
y_train_enc=tf.keras.utils.to_categorical(y_train)
y_test_enc=tf.keras.utils.to_categorical(y_test)
model1=Sequential()
model1.add(Conv2D(64,(3,3),activation='relu',padding='same',input_shape=(28,28,1)))
model1.add(MaxPooling2D((2,2),padding='same'))

model1.add(Flatten())
model1.add(Dense(100,activation='relu'))
model1.add(Dense(10,activation='softmax'))
model1.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
model1.summary()
```

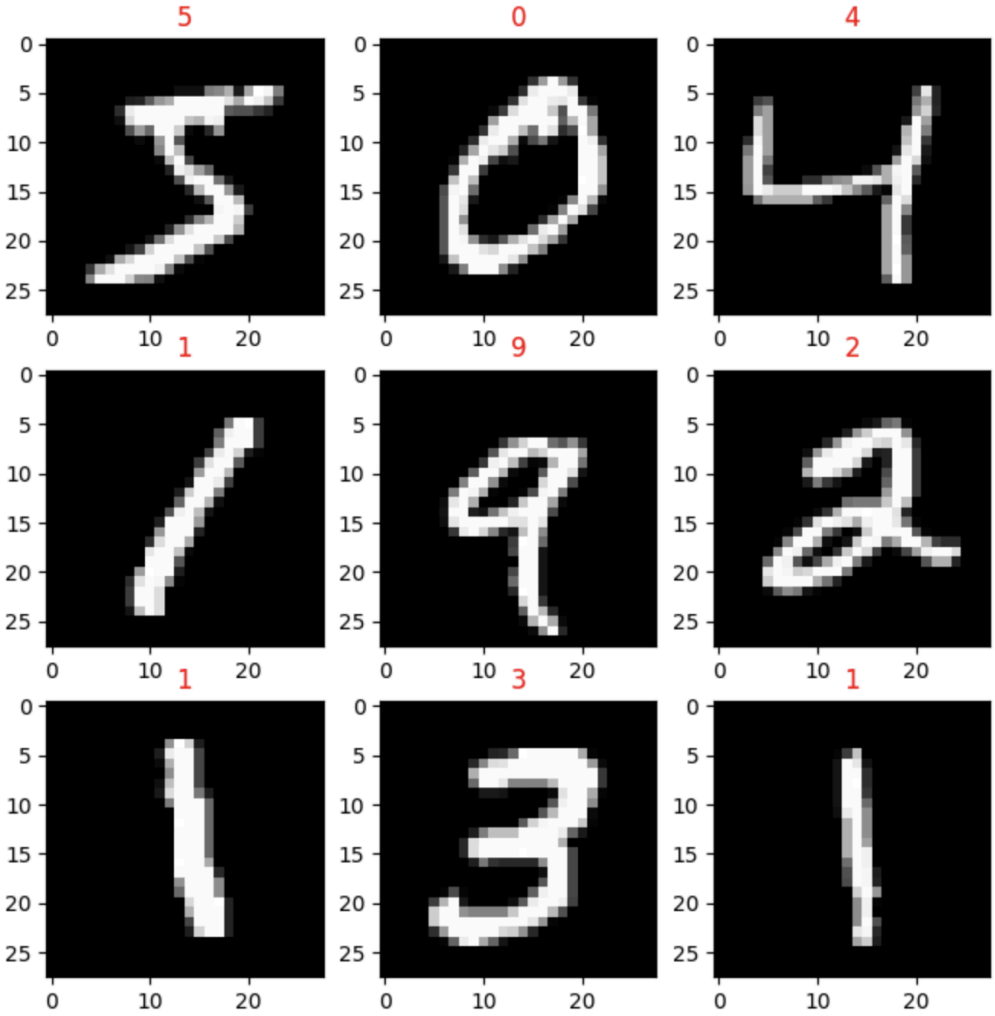
```
history=model1.fit(X_train_norm,y_train_enc,epochs=3,validation_split=0.2,verbose=2)

pred=model1.predict(X_test_norm)

pred=np.argmax(pred,axis=1)

y_test=np.argmax(y_test_enc,axis=1)

cm=tf.math.confusion_matrix(y_test,pred)

sns.heatmap(cm,annot=True,fmt='d')
```

Output:

```
plt.imshow(X_train[0],cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7d92b0915780>
```





Model: "sequential"

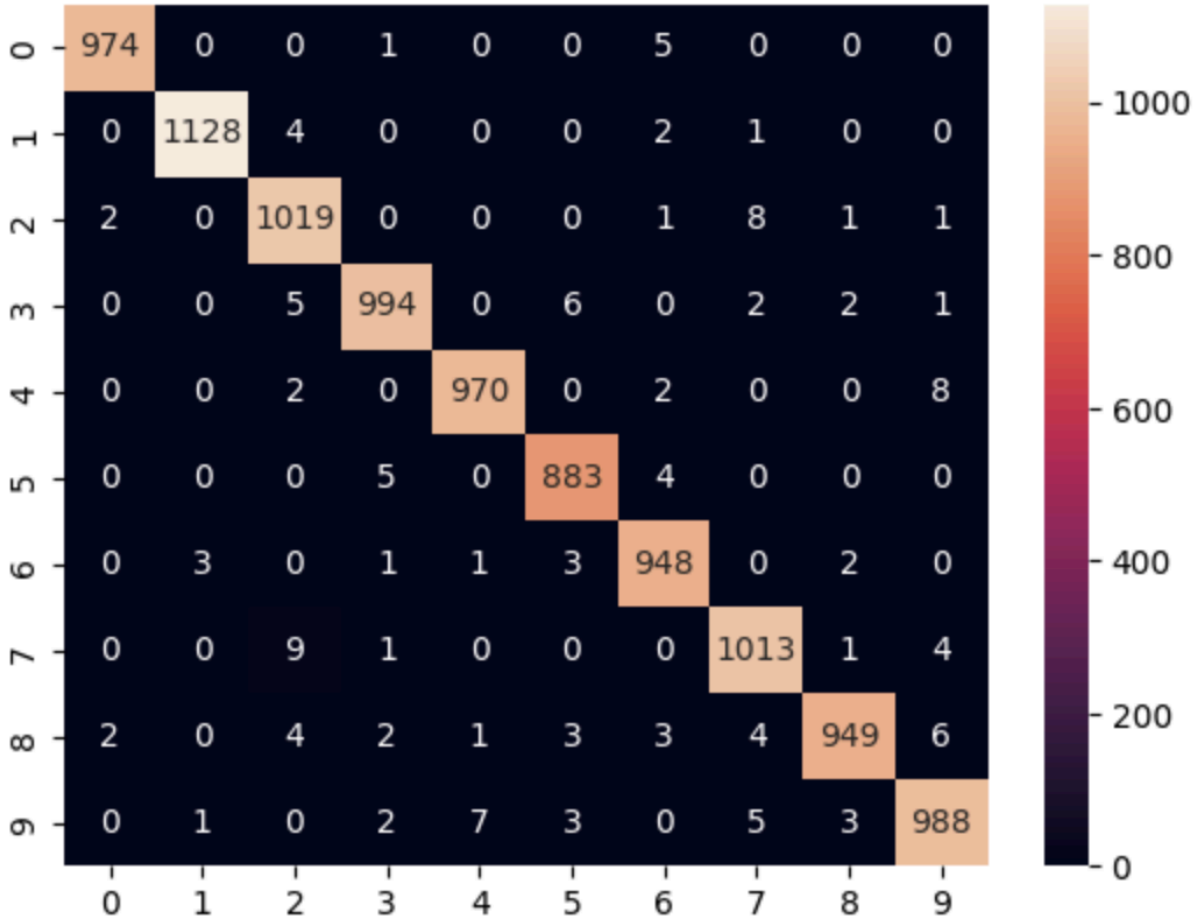
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 28, 28, 64)	640
max_pooling2d (MaxPooling2D)	(None, 14, 14, 64)	0
flatten (Flatten)	(None, 12544)	0
dense (Dense)	(None, 100)	1,254,500
dense_1 (Dense)	(None, 10)	1,010

Total params: 1,256,150 (4.79 MB)
Trainable params: 1,256,150 (4.79 MB)
Non-trainable params: 0 (0.00 B)

```
Epoch 1/3
1500/1500 - 61s - 40ms/step - accuracy: 0.9566 - loss: 0.1474 - val_accuracy: 0.9762 - val_loss: 0.0787
Epoch 2/3
1500/1500 - 57s - 38ms/step - accuracy: 0.9834 - loss: 0.0516 - val_accuracy: 0.9811 - val_loss: 0.0628
Epoch 3/3
1500/1500 - 77s - 51ms/step - accuracy: 0.9895 - loss: 0.0328 - val_accuracy: 0.9853 - val_loss: 0.0514
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 3s 9ms/step
```

```
<Axes: >
```



Result:

Thus the handwritten digits are recognised with MNIST data