Roll Number: 210701120

Exp No: 4 HANDWRITTEN DIGITS RECOGNITION WITH MNIST

Aim:

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

Procedure:

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

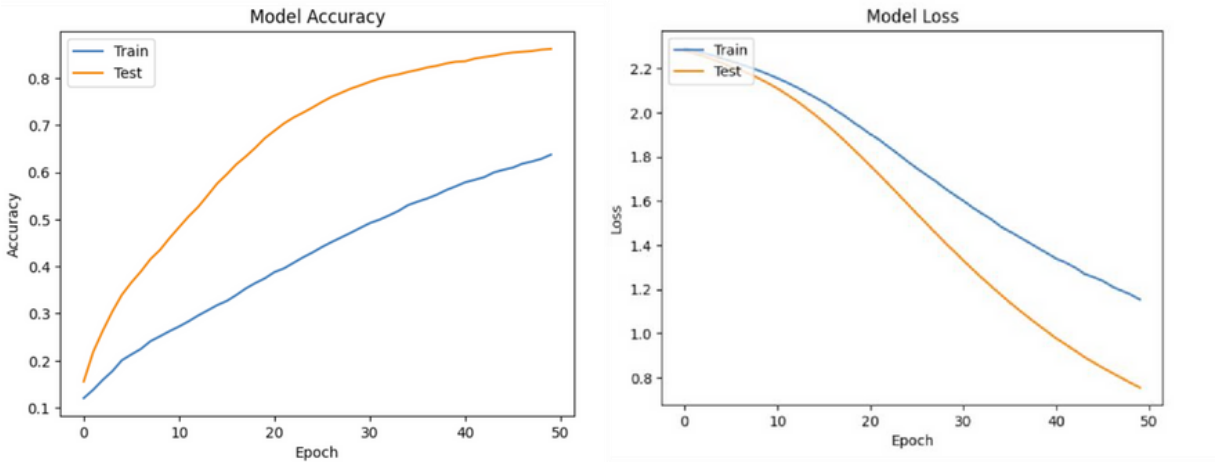
Program:

```
from tensorflow import keras from tensorflow.keras.datasets import mnist from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Dense, Dropout, Flatten from tensorflow.keras.layers import Conv2D, MaxPooling2D from tensorflow.keras import backend as K


(x_train, y_train), (x_test, y_test) = mnist.load_data() print(x_train.shape, y_train.shape) x_train = x_train.reshape(x_train.shape[0], 28, 28, 1) x_test = x_test.reshape(x_test.shape[0], 28, 28, 1) input_shape = (28, 28, 1) y_train = keras.utils.to_categorical(y_train, 10) y_test = keras.utils.to_categorical(y_test, 10) x_train =
```

```
x_train.astype('float32') x_test =
x_test.astype('float32') x_train /= 255
x_test /= 255
print('x_train shape:', x_train.shape) print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples') batch_size = 128 num_classes = 10
epochs = 50 model = Sequential() model.add(Conv2D(32, kernel_size=(5,
5),activation='relu',input_shape=input_shape))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten()) model.add(Dense(128,
activation='relu')) model.add(Dropout(0.3))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes,
activation='softmax'))
model.compile(loss=keras.losses.categorical_crossentropy,optimizer=keras.optimizers.Adadelta(),
metrics
=['accuracy']) hist = model.fit(x_train,
y_train,batch_size=batch_size,epochs=epochs,verbose=1,validation_data=(x_tes
t, y_test)) print("The model has successfully trained") score =
model.evaluate(x_test, y_test, verbose=0) print('Test loss:', score[0]) print('Test
accuracy:', score[1]) import matplotlib.pyplot as plt


plt.plot(hist.history['accuracy'])
plt.plot(hist.history['val_accurac
y']) plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'],
loc='upper left')
plt.show()
```

```
# Plot training & validation loss values
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title('Model  Loss')  plt.ylabel('Loss')
plt.xlabel('Epoch')    plt.legend(['Train',
'Test'], loc='upper left')
plt.show()
```

Output



Result

Handwritten digit recognition with MNIST has been successfully created.