Ex: 3 a

BUILD A CONVOLUTIONAL NEURAL NETWORK

Aim:

To build a simple convolutional neural network using Keras/TensorFlow.

Algorithm:

1. Load and Preprocess Data: Load MNIST dataset, reshape images, and normalize pixel values.

2. One-Hot Encode Labels: Convert labels to one-hot encoded format for classification.

3. Build Model Structure: Use `Sequential` to stack layers for the CNN model.

4. Add Layers:

   - Add a `Conv2D` layer for feature extraction.

   - Add `MaxPooling2D` for dimensionality reduction.

   - Flatten, then add `Dense` layers for classification.

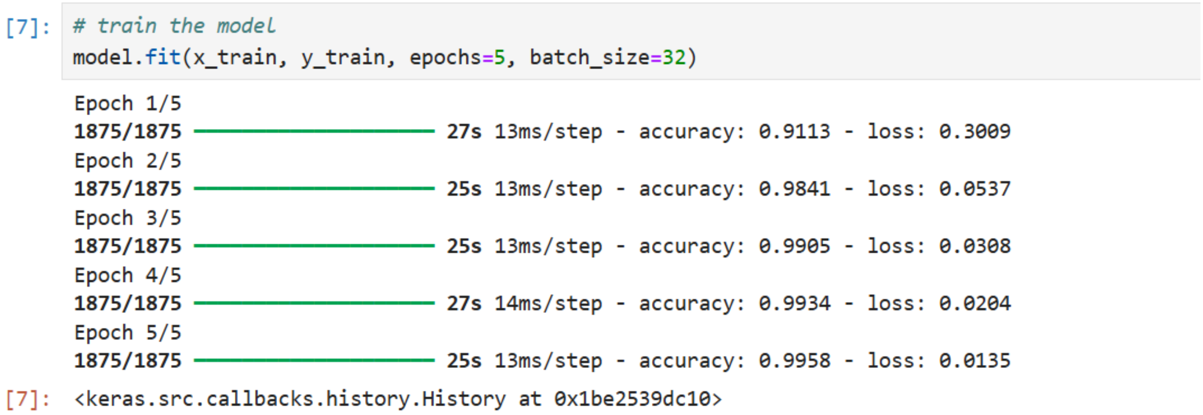
5. Compile Model: Set the optimizer (Adam) and loss function (categorical cross-entropy).

6. Train and Evaluate: Train on training data, then evaluate accuracy and loss on the test set.

Program:

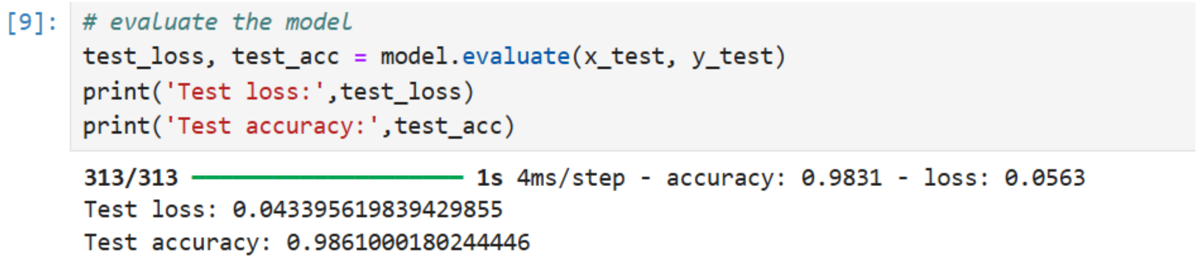
```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()
x_train = x_train.reshape(-1,28,28,1).astype('float32')/255
x_test = x_test.reshape(-1,28,28,1).astype('float32')/255
y_train = tf.keras.utils.to_categorical(y_train, 10)
y_test = tf.keras.utils.to_categorical(y_test, 10)
model = Sequential([
    Conv2D(32,(3,3),activation='relu', input_shape=(28,28,1)),
    MaxPooling2D((2,2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(10, activation='softmax')
])
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train, epochs=5, batch_size=32)
test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test loss:',test_loss)
print('Test accuracy:',test_acc)
```

Output:



```
[7]: # train the model
     model.fit(x_train, y_train, epochs=5, batch_size=32)

     Epoch 1/5
     1875/1875 ━━━━━━━━━━━━━━━━━━━━ 27s 13ms/step - accuracy: 0.9113 - loss: 0.3009
     Epoch 2/5
     1875/1875 ━━━━━━━━━━━━━━━━━━━━ 25s 13ms/step - accuracy: 0.9841 - loss: 0.0537
     Epoch 3/5
     1875/1875 ━━━━━━━━━━━━━━━━━━━━ 25s 13ms/step - accuracy: 0.9905 - loss: 0.0308
     Epoch 4/5
     1875/1875 ━━━━━━━━━━━━━━━━━━━━ 27s 14ms/step - accuracy: 0.9934 - loss: 0.0204
     Epoch 5/5
     1875/1875 ━━━━━━━━━━━━━━━━━━━━ 25s 13ms/step - accuracy: 0.9958 - loss: 0.0135
[7]: <keras.src.callbacks.history.History at 0x1be2539dc10>
```



```
[9]: # evaluate the model
     test_loss, test_acc = model.evaluate(x_test, y_test)
     print('Test loss:',test_loss)
     print('Test accuracy:',test_acc)

     313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.9831 - loss: 0.0563
     Test loss: 0.043395619839429855
     Test accuracy: 0.9861000180244446
```

Result:

Thus the program to build a simple convolutional neural network using Keras/TensorFlow has been executed successfully.