

IN LAB

LAB TASK 1 :

```
#Import necessary libraries
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten,
Dense
from keras.datasets import mnist
from keras.utils import to_categorical
```

LAB TASK 2:

```
# Load and preprocess the MNIST dataset
(train_images, train_labels), (test_images, test_labels) =
mnist.load_data()
# Reshape and normalize the images
train_images = train_images.reshape((60000, 28, 28,
1)).astype('float32') / 255
test_images = test_images.reshape((10000, 28, 28,
1)).astype('float32') / 255
```

```
[3] # Load and preprocess the MNIST dataset
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
# Reshape and normalize the images
train_images = train_images.reshape((60000, 28, 28, 1)).astype('float32') / 255
test_images = test_images.reshape((10000, 28, 28, 1)).astype('float32') / 255

Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [=====] - 0s 0us/step
```

LAB TASK 3:

```
# One-hot encode the labels
train_labels = to_categorical (train_labels)
test_labels = to_categorical(test_labels)
```

LAB TASK 4:

```
# Build the CNN model
model = Sequential()

# Step 1: Convolutional Layer with ReLU activation
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))

# Step 2: Max Pooling Layer
model.add(MaxPooling2D((2, 2)))

# Step 3: Convolutional Layer with ReLU activation
model.add(Conv2D(64, (3, 3), activation='relu'))

# Step 4: Max Pooling Layer
model.add(MaxPooling2D((2, 2)))

# Step 5: Flatten Layer
model.add(Flatten())

# Step 6: Dense (Fully Connected) Layer with ReLU activation
model.add(Dense (64, activation='relu'))

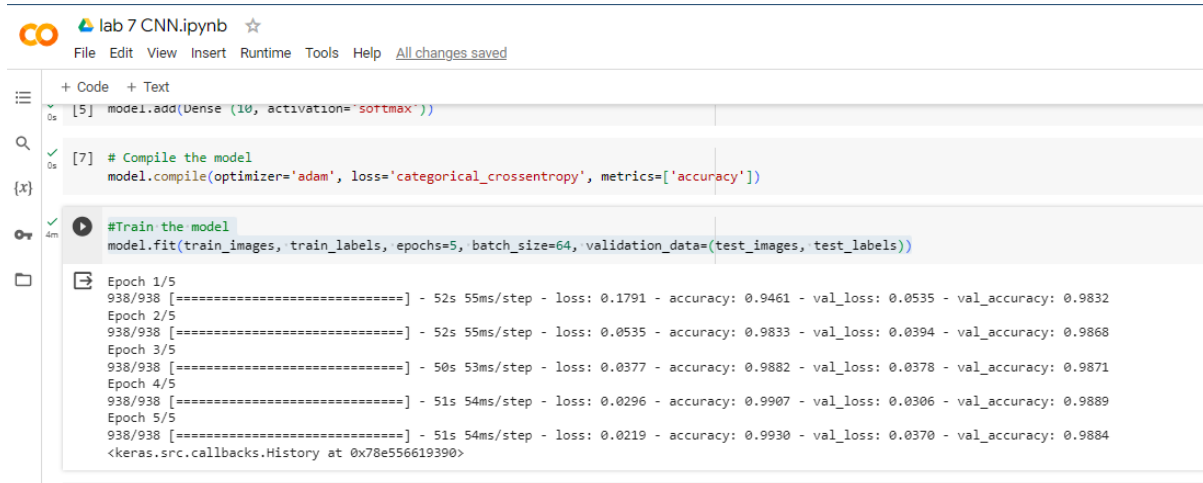
# Step 7: Output Layer with Softmax activation (for multiclass classification)
model.add(Dense (10, activation='softmax'))
```

LAB TASK 5:

```
# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])
```

LAB TASK 6

```
#Train the model
model.fit(train_images, train_labels, epochs=5, batch_size=64,
validation_data=(test_images, test_labels))
```



lab 7 CNN.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

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

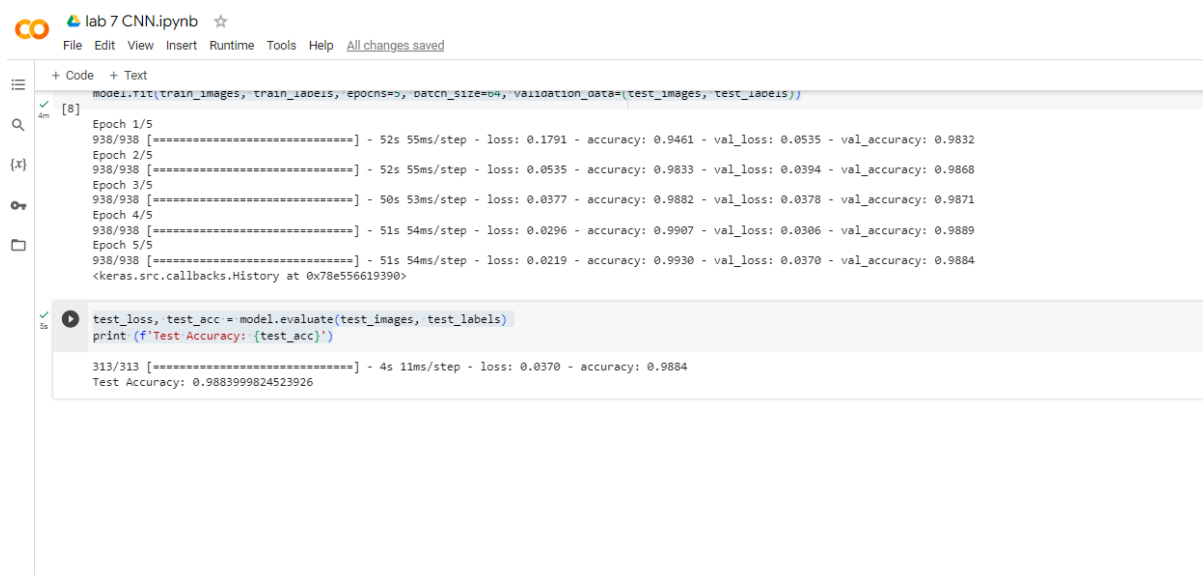
```
[7] # Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

```
#Train the model
model.fit(train_images, train_labels, epochs=5, batch_size=64, validation_data=(test_images, test_labels))
```

```
Epoch 1/5
938/938 [=====] - 52s 55ms/step - loss: 0.1791 - accuracy: 0.9461 - val_loss: 0.0535 - val_accuracy: 0.9832
Epoch 2/5
938/938 [=====] - 52s 55ms/step - loss: 0.0535 - accuracy: 0.9833 - val_loss: 0.0394 - val_accuracy: 0.9868
Epoch 3/5
938/938 [=====] - 50s 53ms/step - loss: 0.0377 - accuracy: 0.9882 - val_loss: 0.0378 - val_accuracy: 0.9871
Epoch 4/5
938/938 [=====] - 51s 54ms/step - loss: 0.0296 - accuracy: 0.9907 - val_loss: 0.0306 - val_accuracy: 0.9889
Epoch 5/5
938/938 [=====] - 51s 54ms/step - loss: 0.0219 - accuracy: 0.9930 - val_loss: 0.0370 - val_accuracy: 0.9884
<keras.src.callbacks.History at 0x78e556619390>
```

LAB TASK 7

```
test_loss, test_acc = model.evaluate(test_images, test_labels)
print (f'Test Accuracy: {test_acc}')
```



lab 7 CNN.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
model.fit(train_images, train_labels, epochs=5, batch_size=64, validation_data=(test_images, test_labels))
```

```
[8] Epoch 1/5
938/938 [=====] - 52s 55ms/step - loss: 0.1791 - accuracy: 0.9461 - val_loss: 0.0535 - val_accuracy: 0.9832
Epoch 2/5
938/938 [=====] - 52s 55ms/step - loss: 0.0535 - accuracy: 0.9833 - val_loss: 0.0394 - val_accuracy: 0.9868
Epoch 3/5
938/938 [=====] - 50s 53ms/step - loss: 0.0377 - accuracy: 0.9882 - val_loss: 0.0378 - val_accuracy: 0.9871
Epoch 4/5
938/938 [=====] - 51s 54ms/step - loss: 0.0296 - accuracy: 0.9907 - val_loss: 0.0306 - val_accuracy: 0.9889
Epoch 5/5
938/938 [=====] - 51s 54ms/step - loss: 0.0219 - accuracy: 0.9930 - val_loss: 0.0370 - val_accuracy: 0.9884
<keras.src.callbacks.History at 0x78e556619390>
```

```
test_loss, test_acc = model.evaluate(test_images, test_labels)
print (f'Test Accuracy: {test_acc}')
```

```
313/313 [=====] - 4s 11ms/step - loss: 0.0370 - accuracy: 0.9884
Test Accuracy: 0.9883999824523926
```