import tensorflow as tf

import matplotlib as plt

import numpy as np

import keras

#Loading Data

mnistDB=tf.keras.datasets.mnist

#Splitting The Data

(X\_train,Y\_train),(X\_test,Y\_test)=mnistDB.load\_data()

X\_train=X\_train.reshape(60000,28,28,1)

X\_test=X\_test.reshape(10000,28,28,1)

#Data Normalisation

X\_train=X\_train.astype('float32')/255

X\_test=X\_test.astype('float32')/255

#Defining the model

ML=keras.models.Sequential()

ML.add(keras.layers.Conv2D(32,(3,3),activation="relu",input\_shape=X\_train.shape[1:]))

ML.add(keras.layers.Conv2D(64,(3,3),activation="relu"))

ML.add((keras.layers.BatchNormalization()))

ML.add(keras.layers.MaxPooling2D((2,2)))

ML.add(keras.layers.Dropout(0.25))

ML.add(keras.layers.Flatten())

ML.add(keras.layers.Dense(128,activation='relu'))

ML.add(keras.layers.Dropout(0.25))

ML.add(keras.layers.Dense(units=10,activation="softmax"))

ML.compile(loss="sparse\_categorical\_crossentropy",optimizer="adam",metrics=['Accuracy'])

es=keras.callbacks.EarlyStopping(monitor='loss',patience=3,restore\_best\_weights=True)

cp=keras.callbacks.ModelCheckpoint("modelname1.h5",monitor="val\_loss")

ML.summary()

#Training the data

history=ML.fit(X\_train,Y\_train,epochs=25,batch\_size=16,callbacks=[es,cp])

#Evaluation

testloss,testaccuracy=ML.evaluate(X\_test,Y\_test)

print("Test loss:", testloss)

print("Test accuracy:",testaccuracy)

Output

