**# Import the libraries and load the dataset**  
import tensorflow as tf  
from tensorflow import keras  
from tensorflow.keras import layers  
from keras.datasets import mnist  
  
**# the data, split between train and test sets**  
(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()  
print(x\_train.shape, y\_train.shape)  
  
**# Defining parameters**  
batch\_size = 128  
num\_classes = 10  
epochs = 15  
  
**# Preprocess the data**  
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)  
  
**# Convert class vectors to binary class matrices**  
y\_train = keras.utils.to\_categorical(y\_train, num\_classes)  
y\_test = keras.utils.to\_categorical(y\_test, num\_classes)  
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')  
  
**# Create the model**  
model=tf.keras.models.Sequential(  
[  
 keras.Input(shape=input\_shape),  
 layers.Conv2D(32, kernel\_size=(3, 3), activation="relu"),  
 layers.MaxPooling2D(pool\_size=(2, 2)),  
 layers.Conv2D(64, kernel\_size=(3, 3), activation="relu"),  
 layers.MaxPooling2D(pool\_size=(2, 2)),  
 layers.Flatten(),  
 layers.Dropout(0.5),  
 layers.Dense(num\_classes, activation="softmax"),  
 ])  
model.summary()  
model.compile(loss=keras.losses.categorical\_crossentropy,optimizer=keras.optimizers.Adam(),metrics=['accuracy'])  
  
**# Train the model**  
hist = model.fit(x\_train, y\_train,batch\_size=batch\_size,epochs=epochs,verbose=1,validation\_data=(x\_test, y\_test))  
print("The model has successfully trained")  
model.save('mnist.h5')  
print("Saving the model as mnist.h5")  
  
**# Evaluate the model**  
score = model.evaluate(x\_test, y\_test, verbose=0)  
print('Test loss:', score[0])  
print('Test accuracy:', score[1])