```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.utils import load_img,img_to_array
from tensorflow.keras.models import load_model
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras.preprocessing.image import array_to_img
from tensorflow.keras.utils import load_img,img_to_array,array_to_img
import matplotlib.pyplot as plt
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.optimizers import RMSprop
import numpy as np
```

```
from keras.datasets import fashion_mnist
(x_train, y_train),(x_test,y_test) = fashion_mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-
32768/29515 [=================================] - 0s 0us/step
40960/29515 [=========================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-
26427392/26421880 [==============================] - 0s 0us/step
26435584/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-
16384/5148 [===========================================================================
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-
4423680/4422102 [==============================] - 0s 0us/step
4431872/4422102 [==============================] - 0s 0us/step
```

```
x_train.shape
```

```
(60000, 28, 28)
```

```
y_train.shape
```

```
(60000,)
```

```
from numpy import subtract
plt.figure(figsize=(10,10))
for i in range (25):
  plt.subplot(5,5,i+1)
  plt.imshow(x_train[i])
  plt.imshow(x_train[i])
plt.show()
```

```
x_train = x_train.reshape(60000,784)
x_test = x_test.reshape(10000,784)
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /=255
x_test /=255
y_train = to_categorical(y_train,10)
y_test = to_categorical(y_test,10)
```

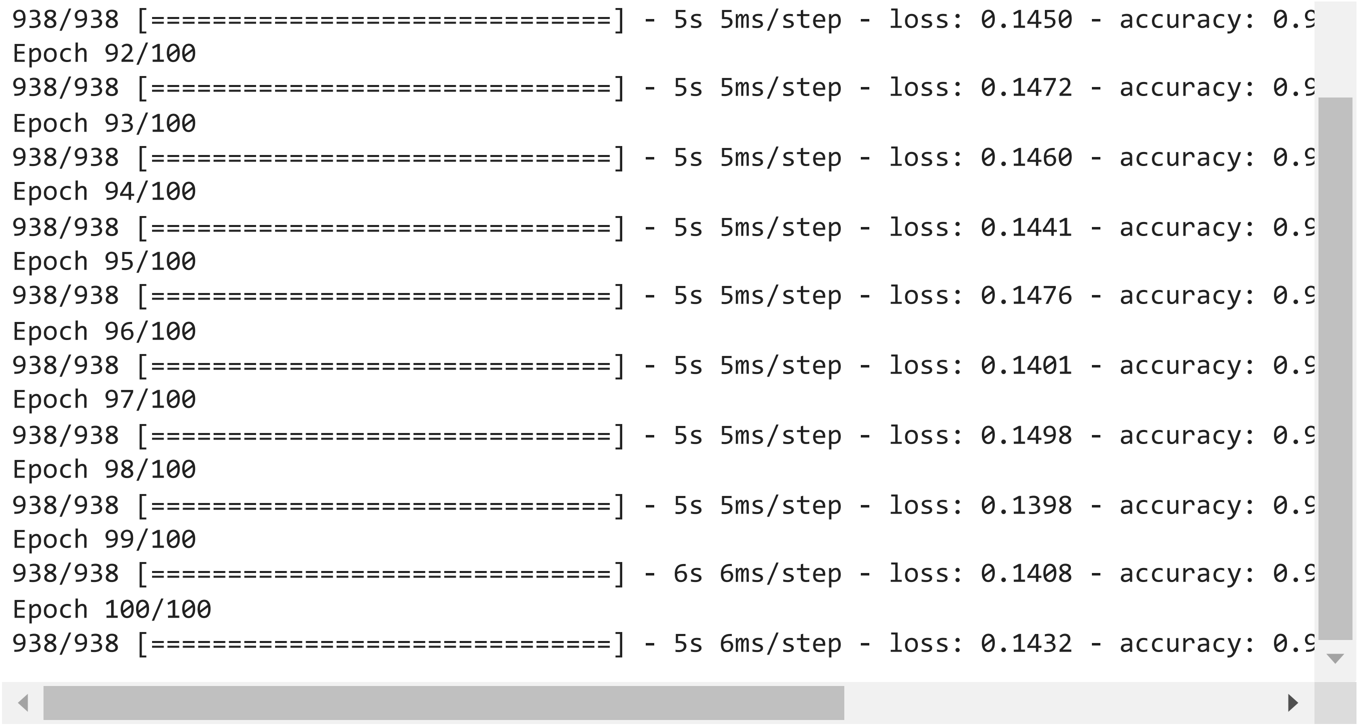
```
model = Sequential()
model.add(Dense(128,activation='relu',input_shape=(784,)))
model.add(Dense(128,activation='relu'))
model.add(Dense(10,activation='softmax'))
model.summary()
```

```
     Model: "sequential"
```

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 dense (Dense)               (None, 128)               100480    
                                                                 
 dense_1 (Dense)             (None, 128)               16512     
                                                                 
 dense_2 (Dense)             (None, 10)                1290      
                                                                 
=================================================================
Total params: 118,282
Trainable params: 118,282
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(loss='categorical_crossentropy',optimizer=RMSprop(), metrics=['accuracy'])
history = model.fit(x_train,y_train,epochs=100,batch_size=64,validation_data=(x_test,y_test),
```

```
Epoch 73/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1517 - accuracy: 0.9
Epoch 74/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1554 - accuracy: 0.9
Epoch 75/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1583 - accuracy: 0.9
Epoch 76/100
938/938 [==============================] - 6s 6ms/step - loss: 0.1522 - accuracy: 0.9
Epoch 77/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1496 - accuracy: 0.9
Epoch 78/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1595 - accuracy: 0.9
Epoch 79/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1526 - accuracy: 0.9
Epoch 80/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1502 - accuracy: 0.9
Epoch 81/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1558 - accuracy: 0.9
Epoch 82/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1554 - accuracy: 0.9
Epoch 83/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1542 - accuracy: 0.9
Epoch 84/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1467 - accuracy: 0.9
Epoch 85/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1510 - accuracy: 0.9
Epoch 86/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1429 - accuracy: 0.9
Epoch 87/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1466 - accuracy: 0.9
Epoch 88/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1459 - accuracy: 0.9
Epoch 89/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1446 - accuracy: 0.9
Epoch 90/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1468 - accuracy: 0.9
Epoch 91/100
```

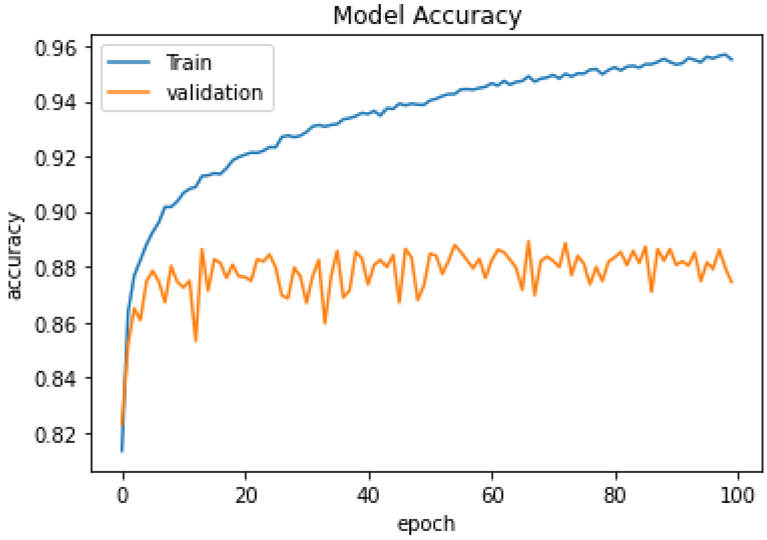


```
938/938 [==============================] - 5s 5ms/step - loss: 0.1450 - accuracy: 0.9
Epoch 92/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1472 - accuracy: 0.9
Epoch 93/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1460 - accuracy: 0.9
Epoch 94/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1441 - accuracy: 0.9
Epoch 95/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1476 - accuracy: 0.9
Epoch 96/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1401 - accuracy: 0.9
Epoch 97/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1498 - accuracy: 0.9
Epoch 98/100
938/938 [==============================] - 5s 5ms/step - loss: 0.1398 - accuracy: 0.9
Epoch 99/100
938/938 [==============================] - 6s 6ms/step - loss: 0.1408 - accuracy: 0.9
Epoch 100/100
938/938 [==============================] - 5s 6ms/step - loss: 0.1432 - accuracy: 0.9
```

```
score = model.evaluate(x_test,y_test,verbose=0)
print('Sai số kiểm tra là: ',score[0])
print('Độ chính xác kiểm tra là: ',score[1])

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['Train','validation'],loc='upper left')
plt.show()
```

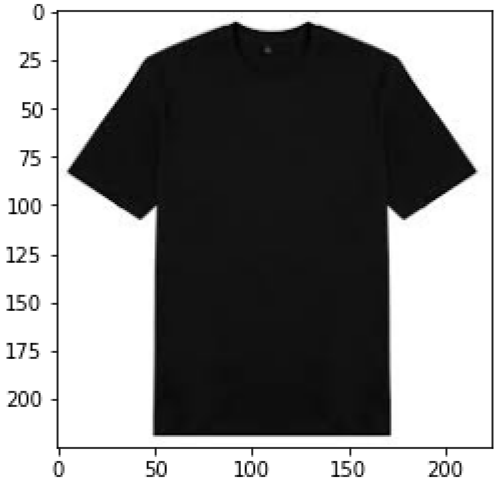
```
Sai số kiểm tra là:  1.6637251377105713
Độ chính xác kiểm tra là:  0.8745999932289124
```



```
model.save('/content/drive/MyDrive/BT AI/fashion.h5')
```

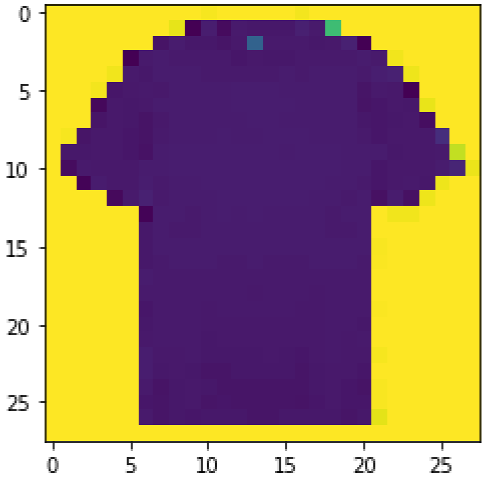
```
load_model('/content/drive/MyDrive/BT AI/fashion.h5')
url='/content/drive/MyDrive/Anh test/ao1.jpeg'
plt.imshow(load_img(url))
```

```
<matplotlib.image.AxesImage at 0x7f60a50b86d0>
```



```
img=load_img('/content/drive/MyDrive/Anh test/ao1.jpeg',target_size=(28,28),color_mode="grays
plt.imshow(img)
img=img_to_array(img)
img=img.reshape(1,784)
img = img.astype('float32')
img = img/255
np.argmax(model.predict(img),axis=-1)
```

```
array([5])
```



✓ 0 giây hoàn thành lúc 10:41 ● ✕