TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP HỒ CHÍ MINH

KHOA CƠ KHÍ CHẾ TẠO MÁY

BỘ MÔN CƠ ĐIỆN TỬ

HCMUTE

BÀI TẬP VỀ NHÀ

MÔN AI TUẦN 13

GVHD:	**PGS.TS Nguyễn Trường Thịnh**
LỚP:	**Thứ 7**
TIẾT:	**3_6**
SVTH:	**Nguyễn Lê Vũ**
MSSV:	**19146428**

Tp. Hồ Chí Minh, tháng 05 năm 2022

1. NHẬN DIỆN TRÁI CÂY

[93]
```
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout, Conv2D, MaxPooling2D, Flatten
from tensorflow.keras.optimizers import Adam
from sklearn.preprocessing import StandardScaler
from keras.utils import np_utils
from sklearn.utils import shuffle
import cv2
import matplotlib.pyplot as plt
import numpy as np
import pickle
import tensorflow as tf
import math as m
```

```
with open('data.pickle', 'rb') as f:
    (x_train, y_train) = pickle.load(f)
x_pre_0 = x_train[3]
x_pre_1 = x_train[10]
x_pre_2 = x_train[22]
x_pre_3 = x_train[39]
x_pre_4 = x_train[47]
x_pre_5 = x_train[57]
x_pre_6 = x_train[61]
x_pre_7 = x_train[79]
x_pre_8 = x_train[82]
x_pre_9 = x_train[93]
x_train = x_train.astype('float32')
x_train /= 255
y_train = np_utils.to_categorical(y_train, 10)
x_train, y_train = shuffle(x_train, y_train)
```

```
def plot_history(history_fine):
  f1 = history_fine.history['acc']
  val_f1 = history_fine.history['val_acc']
  loss = history_fine.history['loss']
  val_loss = history_fine.history['val_loss']
  plt.figure(figsize=(8, 8))
  plt.subplot(2, 1, 1)
  plt.plot(f1, label='Acc')
  plt.plot(val_f1, label='Validation Acc')
  plt.legend(loc='lower right')
  plt.title('Accuracy')
  plt.subplot(2, 1, 2)
  plt.plot(loss, label='Loss')
  plt.plot(val_loss, label='Validation Loss')
  plt.legend(loc='upper right')
  plt.title('Loss')
  plt.xlabel('epoch')
  plt.show()
```

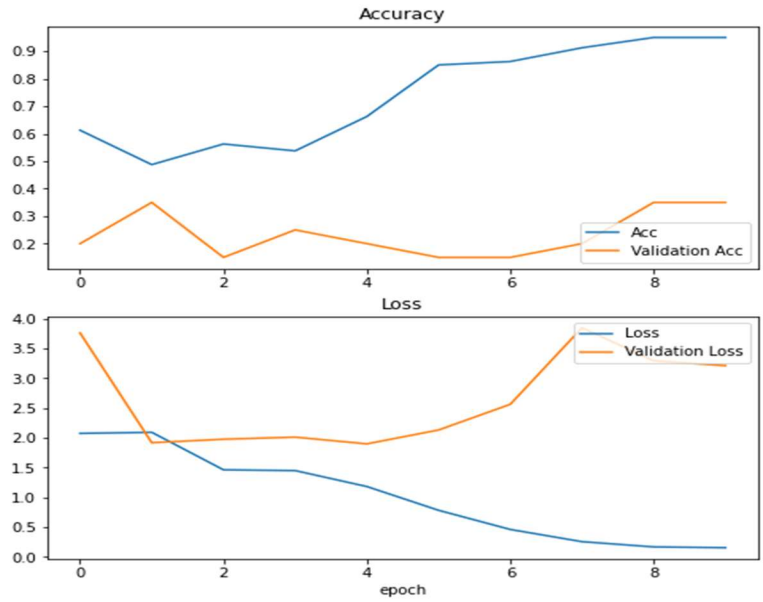
+ Code + Text

[96]
```
model = Sequential()
model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same', input_shape = (150,150,3)))
model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Flatten())
model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
model.add(Dense(10, activation='softmax'))
model.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_48 (Conv2D)          (None, 150, 150, 32)      896

 conv2d_49 (Conv2D)          (None, 150, 150, 32)      9248

 max_pooling2d_24 (MaxPoolin  (None, 75, 75, 32)       0
 g2D)

 conv2d_50 (Conv2D)          (None, 75, 75, 64)        18496

 conv2d_51 (Conv2D)          (None, 75, 75, 64)        36928

 max_pooling2d_25 (MaxPoolin  (None, 37, 37, 64)       0
 g2D)

 conv2d_52 (Conv2D)          (None, 37, 37, 128)       73856

 conv2d_53 (Conv2D)          (None, 37, 37, 128)       147584

 max_pooling2d_26 (MaxPoolin  (None, 18, 18, 128)      0
 g2D)

 flatten_8 (Flatten)         (None, 41472)             0

 dense_22 (Dense)            (None, 128)               5308544

 dense_23 (Dense)            (None, 10)                1290

=================================================================
Total params: 5,596,842
Trainable params: 5,596,842
Non-trainable params: 0
```

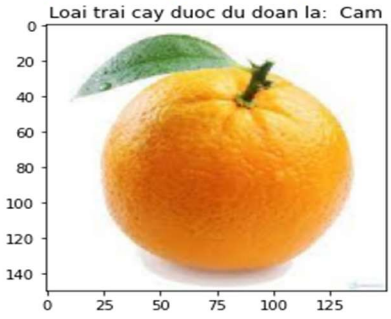
```
[106] opt = Adam(lr = 0.001)
model.compile(optimizer = opt, loss = 'categorical_crossentropy', metrics = ['acc'])
his = model.fit(x_train, y_train, epochs = 10, batch_size = 64, validation_split = 0.2)
plot_history(his)
```

```
Epoch 1/10
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105: UserWarning: The `lr` argument is deprecated, use `learning
  super(Adam, self).__init__(name, **kwargs)
2/2 [==============================] - 1s 239ms/step - loss: 2.0731 - acc: 0.6125 - val_loss: 3.7603 - val_acc: 0.2000
Epoch 2/10
2/2 [==============================] - 0s 68ms/step - loss: 2.0915 - acc: 0.4875 - val_loss: 1.9166 - val_acc: 0.3500
Epoch 3/10
2/2 [==============================] - 0s 71ms/step - loss: 1.4616 - acc: 0.5625 - val_loss: 1.9751 - val_acc: 0.1500
Epoch 4/10
2/2 [==============================] - 0s 70ms/step - loss: 1.4486 - acc: 0.5375 - val_loss: 2.0099 - val_acc: 0.2500
Epoch 5/10
2/2 [==============================] - 0s 71ms/step - loss: 1.1809 - acc: 0.6625 - val_loss: 1.8974 - val_acc: 0.2000
Epoch 6/10
2/2 [==============================] - 0s 73ms/step - loss: 0.7818 - acc: 0.8500 - val_loss: 2.1302 - val_acc: 0.1500
Epoch 7/10
2/2 [==============================] - 0s 68ms/step - loss: 0.4601 - acc: 0.8625 - val_loss: 2.5611 - val_acc: 0.1500
Epoch 8/10
2/2 [==============================] - 0s 71ms/step - loss: 0.2550 - acc: 0.9125 - val_loss: 3.8465 - val_acc: 0.2000
Epoch 9/10
2/2 [==============================] - 0s 68ms/step - loss: 0.1672 - acc: 0.9500 - val_loss: 3.2921 - val_acc: 0.3500
Epoch 10/10
2/2 [==============================] - 0s 68ms/step - loss: 0.1538 - acc: 0.9500 - val_loss: 3.2097 - val_acc: 0.3500
```




```
label = ['Cam','Dao','Dua Hau','Du Du','Khe','Le','Oi','Man','Sau Rieng','Xoai']
plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_0.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_0, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

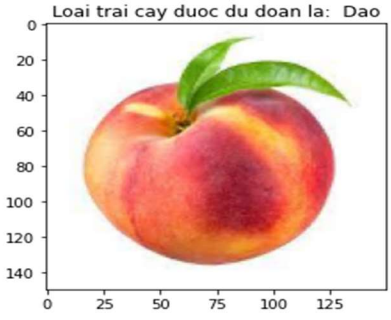
```
<matplotlib.image.AxesImage at 0x7f118e36f250>
```



+ Code + Text

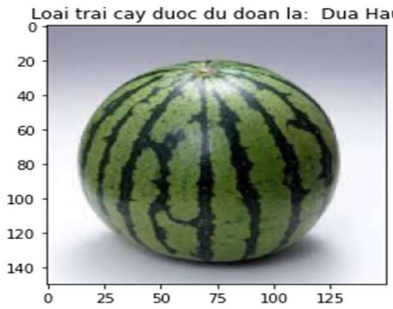
```
[119] plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_1.reshape(1,150,150,3)))])
      plt.imshow(cv2.cvtColor(x_pre_1, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7f0f19a87110>
```



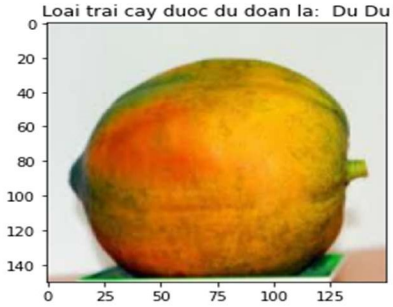
```
[109] plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_2.reshape(1,150,150,3)))])
      plt.imshow(cv2.cvtColor(x_pre_2, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7f0f19ad0ad0>
```




```
plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_3.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_3, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

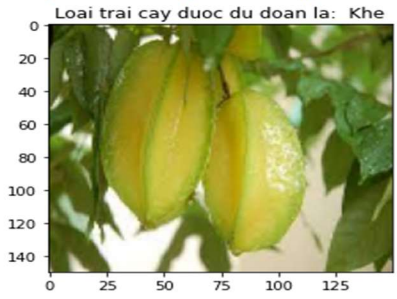
```
<matplotlib.image.AxesImage at 0x7f0f19937210>
```



+ Code + Text

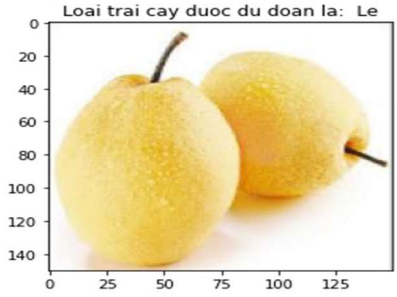
```
plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_4.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_4, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7f0f19954910>
```



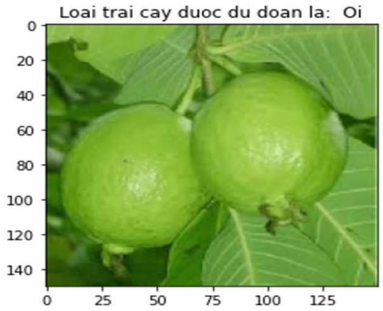

```
[122] plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_5.reshape(1,150,150,3)))])
      plt.imshow(cv2.cvtColor(x_pre_5, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7f1178e92e90>
```



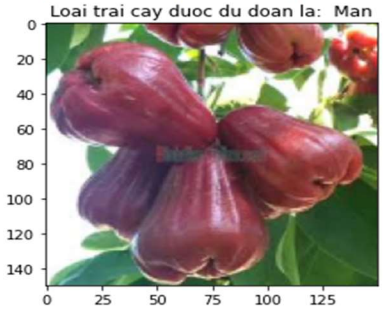
```
[123] plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_6.reshape(1,150,150,3)))])
      plt.imshow(cv2.cvtColor(x_pre_6, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7f117808b390>
```



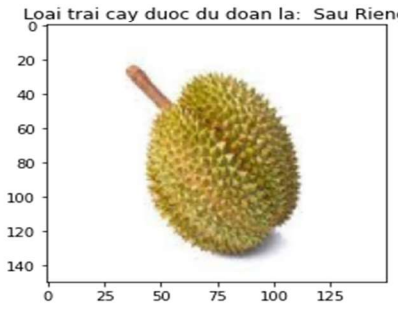

```
[124] plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_7.reshape(1,150,150,3)))])
      plt.imshow(cv2.cvtColor(x_pre_7, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7f0f1be39bd0>
```



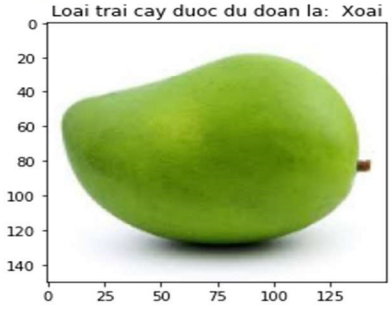

```
[ ] plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_8.reshape(1,150,150,3)))])
    plt.imshow(cv2.cvtColor(x_pre_8, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7f0f1bf7acd0>
```




```
plt.title("Loai trai cay duoc du doan la:  " + label[np.argmax(model.predict(x_pre_9.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_9, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7f10bc06fbd0>
```



2. NHẬN DIỆN MÓN ĂN

[73]
```
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout, Conv2D, MaxPooling2D, Flatten
from tensorflow.keras.optimizers import Adam
from sklearn.preprocessing import StandardScaler
from keras.utils import np_utils
from sklearn.utils import shuffle
import cv2
import matplotlib.pyplot as plt
import numpy as np
import pickle
import tensorflow as tf
import math as m
```

[74]
```
with open('data.pickle', 'rb') as f:
    (x_train, y_train) = pickle.load(f)
x_pre_0 = x_train[2]
x_pre_1 = x_train[13]
x_pre_2 = x_train[27]
x_pre_3 = x_train[37]
x_pre_4 = x_train[45]
x_pre_5 = x_train[52]
x_pre_6 = x_train[63]
x_pre_7 = x_train[76]
x_pre_8 = x_train[83]
x_pre_9 = x_train[94]
x_train = x_train.astype('float32')
x_train /= 255
y_train = np_utils.to_categorical(y_train, 10)
x_train, y_train = shuffle(x_train, y_train)
```

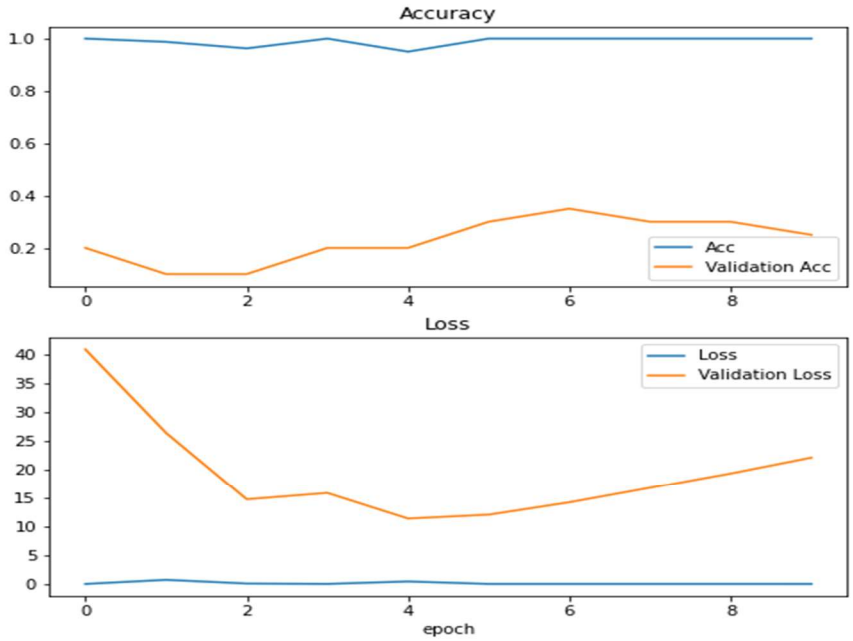
[75]
```
def plot_history(history_fine):
  f1 = history_fine.history['acc']
  val_f1 = history_fine.history['val_acc']
  loss = history_fine.history['loss']
  val_loss = history_fine.history['val_loss']
  plt.figure(figsize=(8, 8))
  plt.subplot(2, 1, 1)
  plt.plot(f1, label='Acc')
  plt.plot(val_f1, label='Validation Acc')
  plt.legend(loc='lower right')
  plt.title('Accuracy')
  plt.subplot(2, 1, 2)
  plt.plot(loss, label='Loss')
  plt.plot(val_loss, label='Validation Loss')
  plt.legend(loc='upper right')
  plt.title('Loss')
  plt.xlabel('epoch')
  plt.show()
```

```
model = Sequential()
model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same', input_shape = (150,150,3)))
model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Flatten())
model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
model.add(Dense(10, activation='softmax'))
model.summary()
```

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_24 (Conv2D)          (None, 150, 150, 32)      896

 conv2d_25 (Conv2D)          (None, 150, 150, 32)      9248

 max_pooling2d_12 (MaxPoolin  (None, 75, 75, 32)       0
 g2D)

 conv2d_26 (Conv2D)          (None, 75, 75, 64)        18496

 conv2d_27 (Conv2D)          (None, 75, 75, 64)        36928

 max_pooling2d_13 (MaxPoolin  (None, 37, 37, 64)       0
 g2D)

 conv2d_28 (Conv2D)          (None, 37, 37, 128)       73856

 conv2d_29 (Conv2D)          (None, 37, 37, 128)       147584

 max_pooling2d_14 (MaxPoolin  (None, 18, 18, 128)      0
 g2D)

 flatten_4 (Flatten)         (None, 41472)             0

 dense_8 (Dense)             (None, 128)               5308544

 dense_9 (Dense)             (None, 10)                1290

=================================================================
Total params: 5,596,842
Trainable params: 5,596,842
Non-trainable params: 0
_________________________________________________________________
```

```
opt = Adam(lr = 0.001)
model.compile(optimizer = opt, loss = 'categorical_crossentropy', metrics = ['acc'])
his = model.fit(x_train, y_train, epochs = 10, batch_size = 64, validation_split = 0.2)
plot_history(his)
```

```
Epoch 1/10
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105: UserWarning: The `lr` argument is deprecated, use `learning_rate` instead.
  super(Adam, self).__init__(name, **kwargs)
2/2 [==============================] - 1s 379ms/step - loss: 9.1678e-04 - acc: 1.0000 - val_loss: 40.9086 - val_acc: 0.2000
Epoch 2/10
2/2 [==============================] - 0s 168ms/step - loss: 0.7103 - acc: 0.9875 - val_loss: 26.4022 - val_acc: 0.1000
Epoch 3/10
2/2 [==============================] - 0s 134ms/step - loss: 0.0720 - acc: 0.9625 - val_loss: 14.7351 - val_acc: 0.1000
Epoch 4/10
2/2 [==============================] - 0s 133ms/step - loss: 0.0066 - acc: 1.0000 - val_loss: 15.8414 - val_acc: 0.2000
Epoch 5/10
2/2 [==============================] - 0s 135ms/step - loss: 0.4279 - acc: 0.9500 - val_loss: 11.3776 - val_acc: 0.2000
Epoch 6/10
2/2 [==============================] - 0s 137ms/step - loss: 0.0041 - acc: 1.0000 - val_loss: 12.0511 - val_acc: 0.3000
Epoch 7/10
2/2 [==============================] - 0s 135ms/step - loss: 0.0013 - acc: 1.0000 - val_loss: 14.2106 - val_acc: 0.3500
Epoch 8/10
2/2 [==============================] - 0s 134ms/step - loss: 0.0028 - acc: 1.0000 - val_loss: 16.7573 - val_acc: 0.3000
Epoch 9/10
2/2 [==============================] - 0s 135ms/step - loss: 0.0029 - acc: 1.0000 - val_loss: 19.3121 - val_acc: 0.3000
Epoch 10/10
2/2 [==============================] - 0s 133ms/step - loss: 5.2805e-04 - acc: 1.0000 - val_loss: 22.0896 - val_acc: 0.2500
```

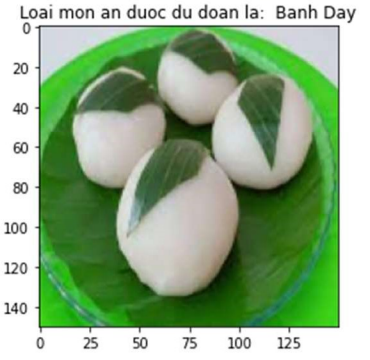



```
label = ['Banh Chung','Banh Day','Banh Mi','Bun Dau Mam Tom','Che Buoi','Nem','Com Tam','Pho','Thit Kho Trung','Trung Vit Lon']
plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_0.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_0, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d17147d0>
```

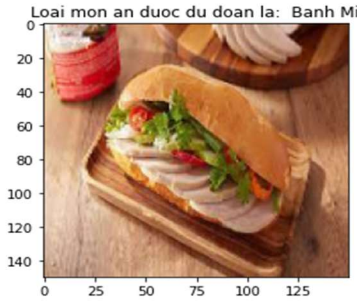

```
[115] plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_1.reshape(1,150,150,3)))])
      plt.imshow(cv2.cvtColor(x_pre_1, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d2c62550>
```



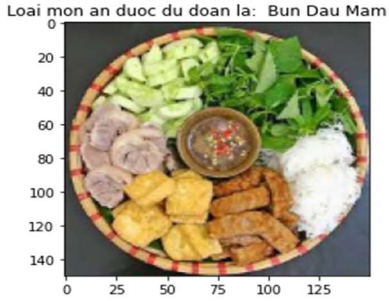
```
plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_2.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_2, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d285bcd0>
```



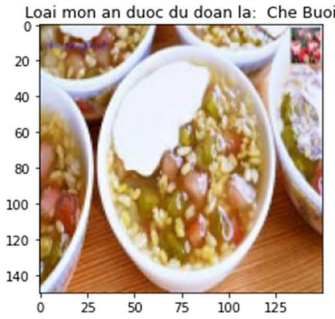

```
[117] plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_3.reshape(1,150,150,3)))])
      plt.imshow(cv2.cvtColor(x_pre_3, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d2826d90>
```



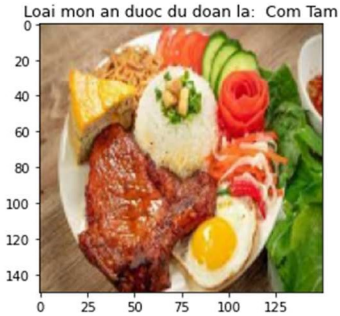

```
plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_4.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_4, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d27b1590>
```



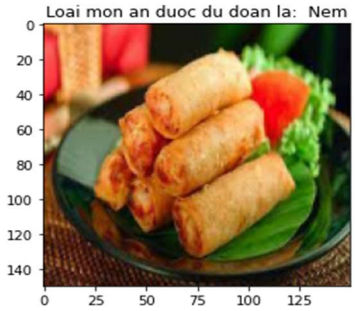

```
[128] plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_5.reshape(1,150,150,3)))])
      plt.imshow(cv2.cvtColor(x_pre_5, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d16a6150>
```



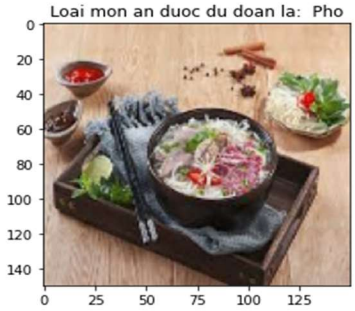
```
plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_6.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_6, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d161ab50>
```



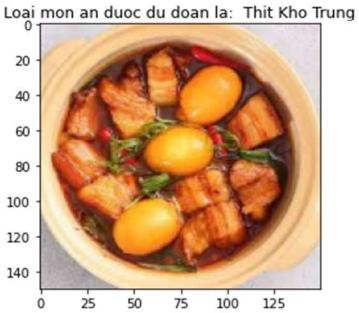

```
plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_7.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_7, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d266cfd0>
```



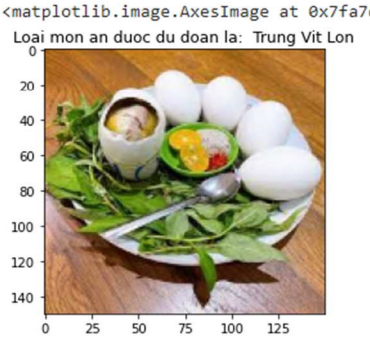

```
[122] plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_8.reshape(1,150,150,3)))])
      plt.imshow(cv2.cvtColor(x_pre_8, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d2640e50>
```




```
plt.title("Loai mon an duoc du doan la:  " + label[np.argmax(model.predict(x_pre_9.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_9, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fa7d255be50>
```



3. NHẬN DIỆN TIỀN

```
[42] from keras.models import Sequential
     from keras.layers import Dense, Activation, Dropout, Conv2D, MaxPooling2D, Flatten
     from tensorflow.keras.optimizers import Adam
     from sklearn.preprocessing import StandardScaler
     from keras.utils import np_utils
     from sklearn.utils import shuffle
     import cv2
     import matplotlib.pyplot as plt
     import numpy as np
     import pickle
     import tensorflow as tf
     import math as m
```

```
     with open('data.pickle', 'rb') as f:
         (x_train, y_train) = pickle.load(f)
     x_pre_0 = x_train[1]
     x_pre_1 = x_train[12]
     x_pre_2 = x_train[23]
     x_pre_3 = x_train[31]
     x_pre_4 = x_train[44]
     x_pre_5 = x_train[51]
     x_pre_6 = x_train[65]
     x_pre_7 = x_train[74]
     x_pre_8 = x_train[86]
     x_pre_9 = x_train[92]
     x_pre_10 = x_train[104]
     x_train = x_train.astype('float32')
     x_train /= 255
     y_train = np_utils.to_categorical(y_train, 11)
     x_train, y_train = shuffle(x_train, y_train)
```

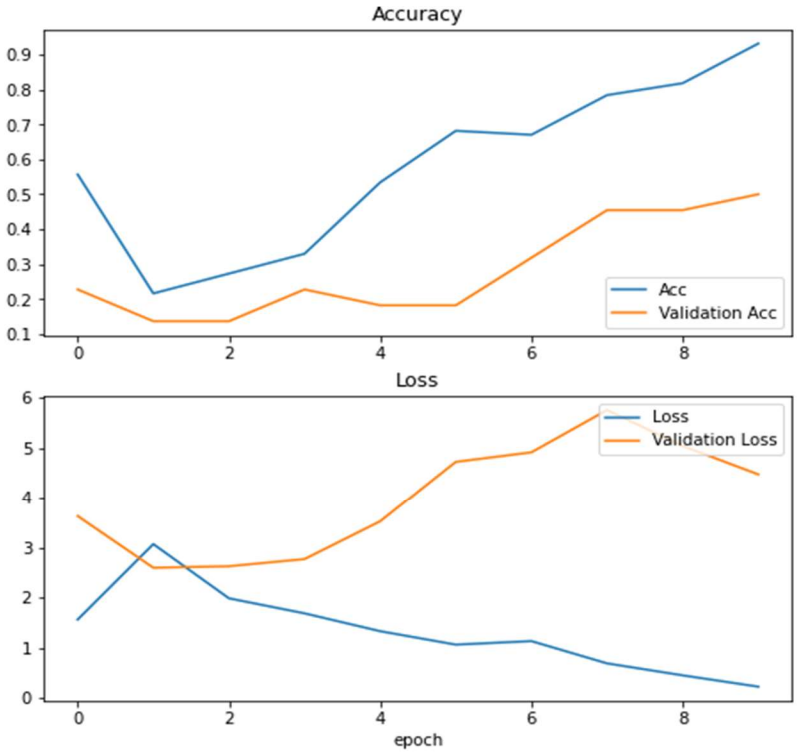
```
[46] def plot_history(history_fine):
       f1 = history_fine.history['acc']
       val_f1 = history_fine.history['val_acc']
       loss = history_fine.history['loss']
       val_loss = history_fine.history['val_loss']
       plt.figure(figsize=(8, 8))
       plt.subplot(2, 1, 1)
       plt.plot(f1, label='Acc')
       plt.plot(val_f1, label='Validation Acc')
       plt.legend(loc='lower right')
       plt.title('Accuracy')
       plt.subplot(2, 1, 2)
       plt.plot(loss, label='Loss')
       plt.plot(val_loss, label='Validation Loss')
       plt.legend(loc='upper right')
       plt.title('Loss')
       plt.xlabel('epoch')
       plt.show()
```

```
[47] model = Sequential()
     model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same', input_shape = (150,150,3)))
     model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
     model.add(MaxPooling2D(2,2))
     model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
     model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
     model.add(MaxPooling2D(2,2))
     model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
     model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
     model.add(MaxPooling2D(2,2))
     model.add(Flatten())
     model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
     model.add(Dense(11, activation='softmax'))
     model.summary()
```

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_30 (Conv2D)          (None, 150, 150, 32)      896

 conv2d_31 (Conv2D)          (None, 150, 150, 32)      9248

 max_pooling2d_15 (MaxPoolin  (None, 75, 75, 32)       0
 g2D)

 conv2d_32 (Conv2D)          (None, 75, 75, 64)        18496

 conv2d_33 (Conv2D)          (None, 75, 75, 64)        36928

 max_pooling2d_16 (MaxPoolin  (None, 37, 37, 64)       0
 g2D)

 conv2d_34 (Conv2D)          (None, 37, 37, 128)       73856

 conv2d_35 (Conv2D)          (None, 37, 37, 128)       147584

 max_pooling2d_17 (MaxPoolin  (None, 18, 18, 128)      0
 g2D)

 flatten_5 (Flatten)         (None, 41472)             0

 dense_10 (Dense)            (None, 128)               5308544

 dense_11 (Dense)            (None, 11)                1419

=================================================================
Total params: 5,596,971
Trainable params: 5,596,971
Non-trainable params: 0
_________________________________________________________________
```

```
opt = Adam(lr = 0.001)
model.compile(optimizer = opt, loss = 'categorical_crossentropy', metrics = ['acc'])
his = model.fit(x_train, y_train, epochs = 10, batch_size = 64, validation_split = 0.2)
plot_history(his)
```

```
Epoch 1/10
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105: UserWarning: The `lr` argument is deprecated, use `learni
  super(Adam, self).__init__(name, **kwargs)
2/2 [==============================] - 1s 245ms/step - loss: 1.5588 - acc: 0.5568 - val_loss: 3.6226 - val_acc: 0.2273
Epoch 2/10
2/2 [==============================] - 0s 84ms/step - loss: 3.0618 - acc: 0.2159 - val_loss: 2.5902 - val_acc: 0.1364
Epoch 3/10
2/2 [==============================] - 0s 90ms/step - loss: 1.9810 - acc: 0.2727 - val_loss: 2.6203 - val_acc: 0.1364
Epoch 4/10
2/2 [==============================] - 0s 87ms/step - loss: 1.6816 - acc: 0.3295 - val_loss: 2.7633 - val_acc: 0.2273
Epoch 5/10
2/2 [==============================] - 0s 88ms/step - loss: 1.3263 - acc: 0.5341 - val_loss: 3.5182 - val_acc: 0.1818
Epoch 6/10
2/2 [==============================] - 0s 91ms/step - loss: 1.0583 - acc: 0.6818 - val_loss: 4.7176 - val_acc: 0.1818
Epoch 7/10
2/2 [==============================] - 0s 88ms/step - loss: 1.1287 - acc: 0.6705 - val_loss: 4.9080 - val_acc: 0.3182
Epoch 8/10
2/2 [==============================] - 0s 88ms/step - loss: 0.6852 - acc: 0.7841 - val_loss: 5.7511 - val_acc: 0.4545
Epoch 9/10
2/2 [==============================] - 0s 90ms/step - loss: 0.4468 - acc: 0.8182 - val_loss: 5.0350 - val_acc: 0.4545
Epoch 10/10
2/2 [==============================] - 0s 88ms/step - loss: 0.2212 - acc: 0.9318 - val_loss: 4.4689 - val_acc: 0.5000
```

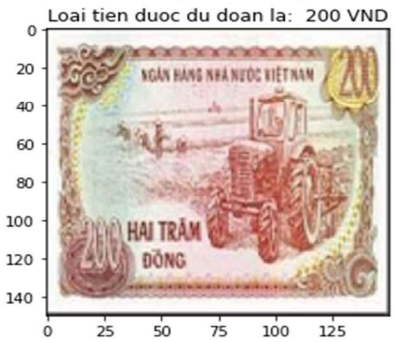



```
label = ['100 VND','200 VND','500 VND','1000 VND','2000 VND','5000 VND','10.000 VND','20.000 VND','50.000 VND','100.000 VND','20
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_0.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_0, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc0e429c5d0>
```


```
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_1.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_1, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc0da388490>
```



```
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_2.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_2, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc0e437ecd0>
```


```
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_3.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_3, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc0fa21b190>
```


```
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_4.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_4, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc0d87db690>
```


```
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_5.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_5, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc0e40fb610>
```

```
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_6.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_6, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc0da23e110>
```


```
[58] plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_7.reshape(1,150,150,3)))])
     plt.imshow(cv2.cvtColor(x_pre_7, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc0da3a6090>
```


```
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_8.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_8, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc0e431c2d0>
```


```
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_9.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_9, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

```
<matplotlib.image.AxesImage at 0x7fc02859a650>
```

```
plt.title("Loai tien duoc du doan la:  " + label[np.argmax(model.predict(x_pre_10.reshape(1,150,150,3)))])
plt.imshow(cv2.cvtColor(x_pre_10, cv2.COLOR_BGR2RGB), cmap=plt.get_cmap('gray'))
```

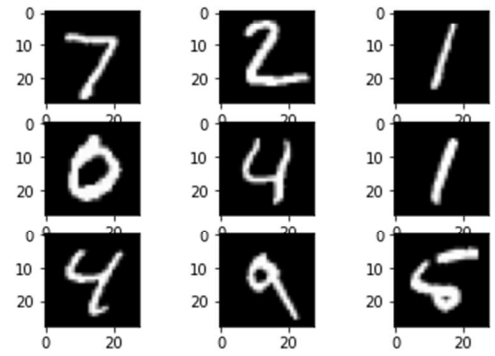
```
<matplotlib.image.AxesImage at 0x7fc16030fdd0>
```

4. MNIST_CNN

```
from keras.models import Sequential, load_model
from keras.layers import Flatten,Dense
from tensorflow.keras.optimizers import Adam,SGD
from keras.callbacks import EarlyStopping
from keras.utils import np_utils
from keras.datasets import mnist
import matplotlib.pyplot as plt
import numpy as np
import tensorflow as tf
from keras.layers.convolutional import Conv2D
from keras.layers.pooling import MaxPooling2D
```

```
def plot_history(history_fine):
  f1 = history_fine.history['acc']
  val_f1 = history_fine.history['val_acc']
  loss = history_fine.history['loss']
  val_loss = history_fine.history['val_loss']
  plt.figure(figsize=(8, 8))
  plt.subplot(2, 1, 1)
  plt.plot(f1, label='Acc')
  plt.plot(val_f1, label='Validation Acc')
  plt.legend(loc='lower right')
  plt.title('Accuracy')
  plt.subplot(2, 1, 2)
  plt.plot(loss, label='Loss')
  plt.plot(val_loss, label='Validation Loss')
  plt.legend(loc='upper right')
  plt.title('Loss')
  plt.xlabel('epoch')
  plt.show()
```

```
# Load Data
(x_train,y_train),(x_test, y_test) = mnist.load_data()
for i in range(9):
  plt.subplot(330+i+1)
  plt.imshow(x_test[i], cmap='gray')
```



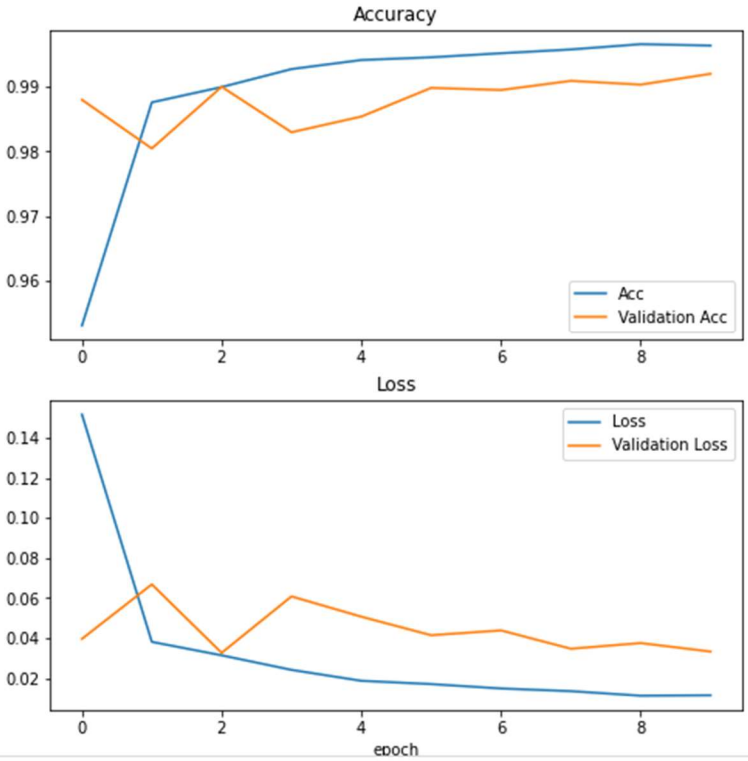
```
x_train=x_train.reshape(60000,28,28,1)
x_test=x_test.reshape(10000,28,28,1)
x_train=x_train.astype('float32')
x_test=x_test.astype('float32')
x_train/=255
x_test/=255
y_train=np_utils.to_categorical(y_train,10)
y_test=np_utils.to_categorical(y_test,10)
```

```
model = Sequential()
model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same', input_shape = (28,28,1)))
model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Flatten())
model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
model.add(Dense(10, activation='softmax'))
model.summary()
```

```
opt = Adam(lr = 0.001)
model.compile(optimizer = opt, loss = 'categorical_crossentropy', metrics = ['acc'])
his = model.fit(x_train, y_train, epochs = 10, batch_size = 64, validation_split = 0.2)
```

```
Epoch 1/10
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105: UserWarning: The `lr` argument is deprecated, use `learning_rat
  super(Adam, self).__init__(name, **kwargs)
750/750 [==============================] - 15s 19ms/step - loss: 0.1516 - acc: 0.9531 - val_loss: 0.0397 - val_acc: 0.9879
Epoch 2/10
750/750 [==============================] - 14s 19ms/step - loss: 0.0382 - acc: 0.9875 - val_loss: 0.0669 - val_acc: 0.9804
Epoch 3/10
750/750 [==============================] - 14s 18ms/step - loss: 0.0315 - acc: 0.9899 - val_loss: 0.0327 - val_acc: 0.9899
Epoch 4/10
750/750 [==============================] - 14s 18ms/step - loss: 0.0242 - acc: 0.9926 - val_loss: 0.0609 - val_acc: 0.9829
Epoch 5/10
750/750 [==============================] - 14s 18ms/step - loss: 0.0188 - acc: 0.9940 - val_loss: 0.0508 - val_acc: 0.9853
Epoch 6/10
750/750 [==============================] - 14s 18ms/step - loss: 0.0171 - acc: 0.9945 - val_loss: 0.0415 - val_acc: 0.9898
Epoch 7/10
750/750 [==============================] - 14s 18ms/step - loss: 0.0149 - acc: 0.9951 - val_loss: 0.0439 - val_acc: 0.9894
Epoch 8/10
750/750 [==============================] - 14s 18ms/step - loss: 0.0136 - acc: 0.9957 - val_loss: 0.0348 - val_acc: 0.9908
Epoch 9/10
750/750 [==============================] - 14s 18ms/step - loss: 0.0113 - acc: 0.9965 - val_loss: 0.0376 - val_acc: 0.9902
Epoch 10/10
750/750 [==============================] - 14s 18ms/step - loss: 0.0116 - acc: 0.9963 - val_loss: 0.0334 - val_acc: 0.9919
```

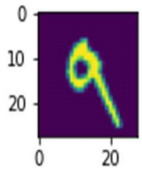
```
plot_history(his)
```



```
a = int(input('Random phan tu muon test:'))
print('Model nhan dang ra= ')
print(np.argmax(model.predict(x_test),axis=1)[a])
```

```
Random phan tu muon test:9
Model nhan dang ra=
9
```

```
(x_train,y_train),(x_test, y_test) = mnist.load_data()
for i in range(1):
  plt.subplot(330+a+1)
  plt.imshow(x_test[a])
```

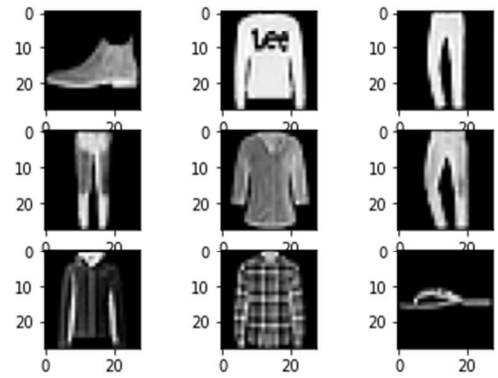


5. FASHION_MNIST

```
from keras.models import Sequential, load_model
from keras.layers import Flatten,Dense
from tensorflow.keras.optimizers import Adam,SGD
from keras.callbacks import EarlyStopping
from keras.utils import np_utils
from keras.datasets import fashion_mnist
import matplotlib.pyplot as plt
import numpy as np
import tensorflow as tf
from keras.layers.convolutional import Conv2D
from keras.layers.pooling import MaxPooling2D
```

```
def plot_history(history_fine):
  f1 = history_fine.history['acc']
  val_f1 = history_fine.history['val_acc']
  loss = history_fine.history['loss']
  val_loss = history_fine.history['val_loss']
  plt.figure(figsize=(8, 8))
  plt.subplot(2, 1, 1)
  plt.plot(f1, label='Acc')
  plt.plot(val_f1, label='Validation Acc')
  plt.legend(loc='lower right')
  plt.title('Accuracy')
  plt.subplot(2, 1, 2)
  plt.plot(loss, label='Loss')
  plt.plot(val_loss, label='Validation Loss')
  plt.legend(loc='upper right')
  plt.title('Loss')
  plt.xlabel('epoch')
  plt.show()
```

```
# Load Data
(x_train,y_train),(x_test, y_test) = fashion_mnist.load_data()
for i in range(9):
  plt.subplot(330+i+1)
  plt.imshow(x_test[i], cmap='gray')
```



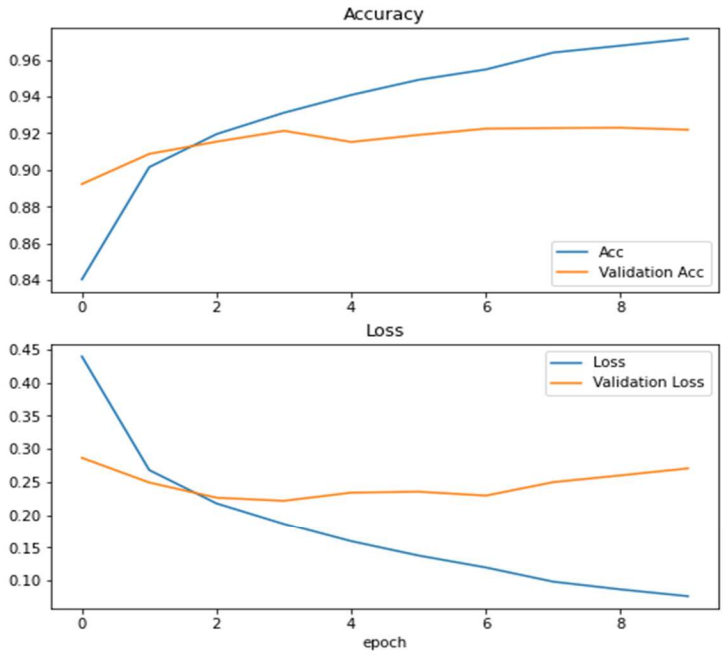
```
x_train=x_train.reshape(60000,28,28,1)
x_test=x_test.reshape(10000,28,28,1)
x_train=x_train.astype('float32')
x_test=x_test.astype('float32')
x_train/=255
x_test/=255
y_train=np_utils.to_categorical(y_train,10)
y_test=np_utils.to_categorical(y_test,10)
```

```
model = Sequential()
model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same', input_shape = (28,28,1)))
model.add(Conv2D(32, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(Conv2D(64, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(Conv2D(128, (3,3), activation='relu',kernel_initializer='he_uniform', padding ='same'))
model.add(MaxPooling2D(2,2))
model.add(Flatten())
model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
model.add(Dense(10, activation='softmax'))
model.summary()
```

```
opt = Adam(lr = 0.001)
model.compile(optimizer = opt, loss = 'categorical_crossentropy', metrics = ['acc'])
his = model.fit(x_train, y_train, epochs = 10, batch_size = 64, validation_split = 0.2)
```

```
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105: UserWarning: The `lr` argument is deprecated, use `learning_rate`
  super(Adam, self).__init__(name, **kwargs)
Epoch 1/10
750/750 [==============================] - 17s 8ms/step - loss: 0.4392 - acc: 0.8404 - val_loss: 0.2863 - val_acc: 0.8923
Epoch 2/10
750/750 [==============================] - 6s 8ms/step - loss: 0.2679 - acc: 0.9016 - val_loss: 0.2491 - val_acc: 0.9087
Epoch 3/10
750/750 [==============================] - 6s 8ms/step - loss: 0.2175 - acc: 0.9195 - val_loss: 0.2262 - val_acc: 0.9154
Epoch 4/10
750/750 [==============================] - 5s 7ms/step - loss: 0.1863 - acc: 0.9312 - val_loss: 0.2214 - val_acc: 0.9213
Epoch 5/10
750/750 [==============================] - 6s 7ms/step - loss: 0.1593 - acc: 0.9409 - val_loss: 0.2338 - val_acc: 0.9153
Epoch 6/10
750/750 [==============================] - 6s 8ms/step - loss: 0.1375 - acc: 0.9491 - val_loss: 0.2352 - val_acc: 0.9191
Epoch 7/10
750/750 [==============================] - 6s 7ms/step - loss: 0.1195 - acc: 0.9548 - val_loss: 0.2293 - val_acc: 0.9225
Epoch 8/10
750/750 [==============================] - 5s 7ms/step - loss: 0.0981 - acc: 0.9640 - val_loss: 0.2496 - val_acc: 0.9228
Epoch 9/10
750/750 [==============================] - 5s 7ms/step - loss: 0.0864 - acc: 0.9677 - val_loss: 0.2598 - val_acc: 0.9230
Epoch 10/10
750/750 [==============================] - 5s 7ms/step - loss: 0.0762 - acc: 0.9715 - val_loss: 0.2704 - val_acc: 0.9219
```

```
plot_history(his)
# Predict
print(x_test.shape)
y_predict = model.predict(x_test[3].reshape(1,28,28,1))
print('Predicted value: ', np.argmax(y_predict))
print('Correct value: ',np.argmax(y_test[3]))
# Evaluate
score = model.evaluate(x_test, y_test, verbose=0)
print('Test accuracy: ',score[1])
```




```
(10000, 28, 28, 1)
Predicted value:  1
Correct value:  1
Test accuracy:  0.9190999865531921
```