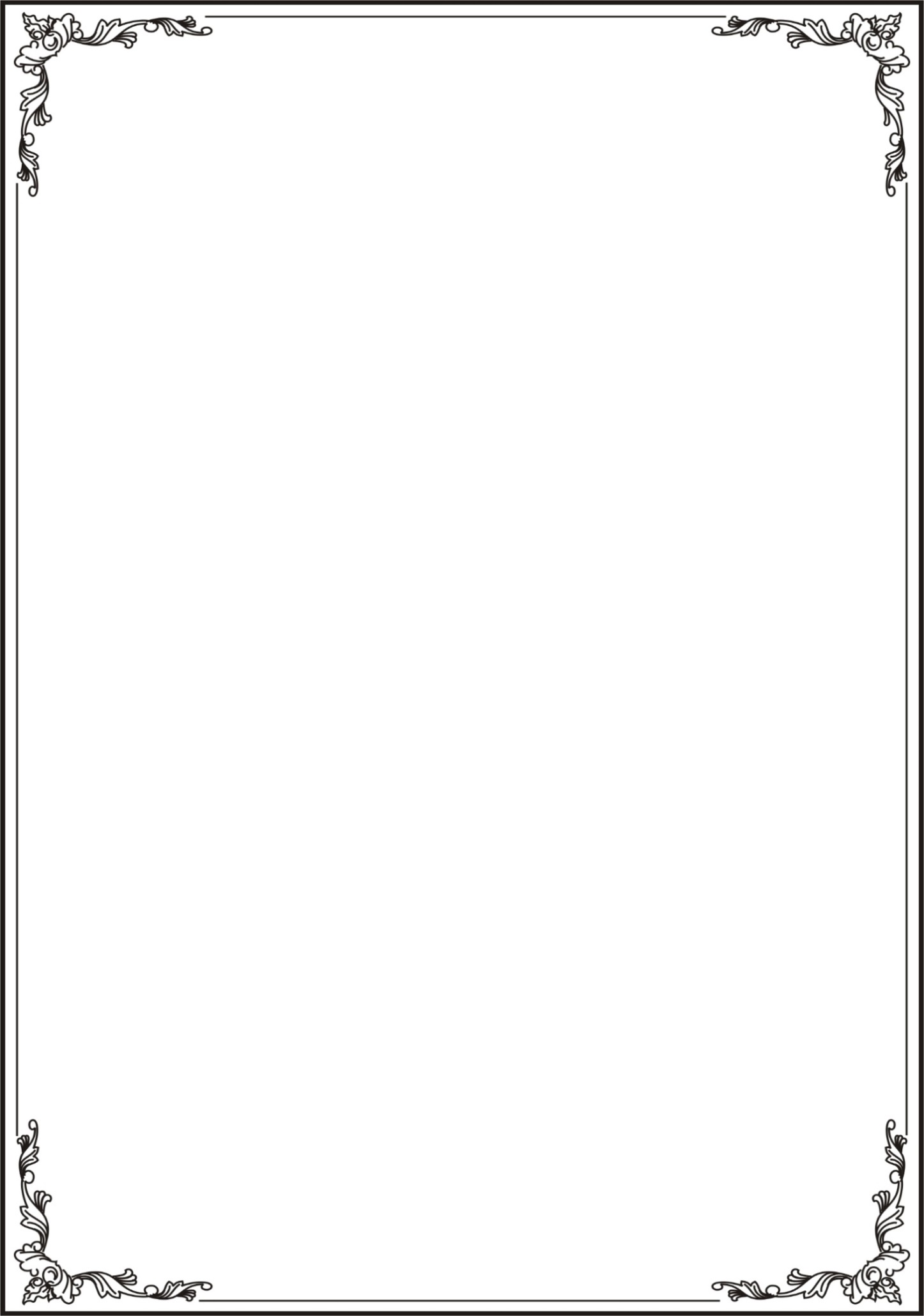
**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA CƠ KHÍ CHẾ TẠO MÁY**

**BỘ MÔN CƠ ĐIỆN TỬ**

****

**HOMEWORK AI: WEEK 14**

**GVHD:** TS. Nguyễn Trường Thịnh

**SVTH:** Phạm Phúc Chương

**MSSV:** 19146311

TP.Hồ Chí Minh, tháng năm 2022

**Mục Lục**

[**1.** **Phân loại trái cây** 3](#_Toc103989420)

[**2.** **CNN\_Nhận dạng 11 loại tiền Việt Nam** 6](#_Toc103989421)

[**3.** **Dự đoán món ăn Việt Nam** 9](#_Toc103989422)

[**4.** **Training CNN MNIST** 12](#_Toc103989423)

[**5.** **Training FASION\_MNIST** 15](#_Toc103989424)

[**6.** **Training Cifar-100** 18](#_Toc103989425)

1. **Phân loại trái cây**

**Code:**

**#import thu vien**

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

#dung file data up san tren google

from google.colab import drive

drive.mount('/content/drive')

#Ham ve do thi

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

with open('/content/drive/MyDrive/Colab Notebooks/Data/data\_fruit.pickle', 'rb') as f:

    (x\_train, y\_train) = pickle.load(f)

# Reshape Data

x\_pre\_1 = x\_train[1]

x\_pre\_2 = x\_train[13]

x\_pre\_3 = x\_train[26]

x\_pre\_4 = x\_train[38]

x\_pre\_5 = x\_train[56]

 Preprocessing Data

x\_train = x\_train.astype('float32')

x\_train /= 255

# Encoding Y

y\_train = np\_utils.to\_categorical(y\_train, 10)

# Shuffe Data

x\_train, y\_train = shuffle(x\_train, y\_train)

# Tao model

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()

# Training

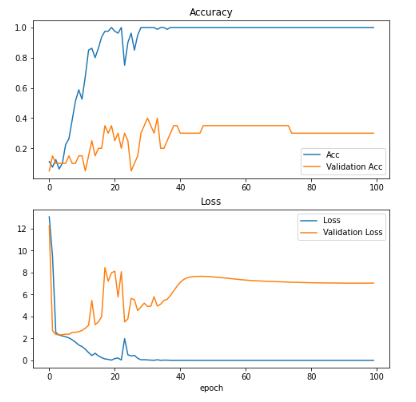
opt = Adam(lr = 0.001)

model.compile(optimizer = opt, loss = 'categorical\_crossentropy', metrics = ['acc'])

his = model.fit(x\_train, y\_train, epochs = 100, batch\_size = 64, validation\_split = 0.2)

#Bieu do ket qua training

plot\_history(his)



Hình 1 Kết quả training

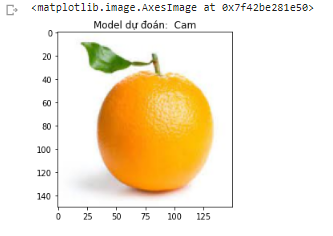
# add label cho model

label = ['Cam','Dao','Dua hau','Du du','Khe','Le','Oi','Man','Sapoche','Xoai']

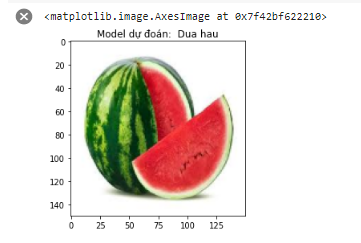
# Xet lai

plt.title("Model dự đoán:  " + 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'))



Hình 2 Kết quả test model trên quả cam



Hình 3 Kết quả test model trên dưa hấu

1. **CNN\_Nhận dạng 11 loại tiền Việt Nam**

**#Import lib**

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

#chuan bi ham ve bieu do

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()

#add data in drive

from google.colab import drive

drive.mount('/content/drive')

# Load Data

with open('/content/drive/MyDrive/Colab Notebooks/Data/data\_MoneyVN.pickle', 'rb') as f:

    (x\_train, y\_train) = pickle.load(f)

# Reshape Data

x\_pre\_1 = x\_train[1]

x\_pre\_2 = x\_train[12]

x\_pre\_3 = x\_train[25]

x\_pre\_4 = x\_train[38]

x\_pre\_5 = x\_train[56]

x\_pre\_6 = x\_train[86]

# Preprocessing Data

x\_train = x\_train.astype('float32')

x\_train /= 255

# Encoding Y

y\_train = np\_utils.to\_categorical(y\_train, 11)

# Shuffe Data

x\_train, y\_train = shuffle(x\_train, y\_train)

#tao model CNN

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()

#Training

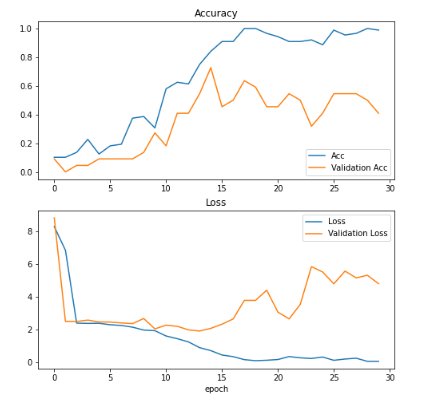
opt = Adam(lr = 0.001)

model.compile(optimizer = opt, loss = 'categorical\_crossentropy', metrics = ['acc'])

his = model.fit(x\_train, y\_train, epochs = 30, batch\_size = 64, validation\_split = 0.2)

#ve bieu do xem ket qua training

plot\_history(his)



Hình 4 Kết quả training

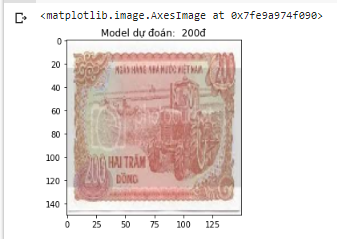
# Add label

label = ['200đ','500đ','1000đ','2000đ','5000đ','10000đ','20000đ','50000đ','100000đ','200000đ','500000']

# model du doan

plt.title("Model dự đoán:  " + 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'))



Hình 5 Kết quả model dự đoán

1. **Dự đoán món ăn Việt Nam**

**#Add lib**

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

from google.colab import drive

drive.mount('/content/drive')

#chuan bi ham ve bieu do

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

with open('/content/drive/MyDrive/Colab Notebooks/Data/data\_food.pickle', 'rb') as f:

    (x\_train, y\_train) = pickle.load(f)

# Reshape Data

x\_pre\_1 = x\_train[48]

x\_pre\_2 = x\_train[22]

x\_pre\_3 = x\_train[64]

x\_pre\_4 = x\_train[75]

x\_pre\_5 = x\_train[92]

# Preprocessing Data

x\_train = x\_train.astype('float32')

x\_train /= 255

# Encoding Y

y\_train = np\_utils.to\_categorical(y\_train, 10)

# Shuffe Data

x\_train, y\_train = shuffle(x\_train, y\_train)

#chuan bi model

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()

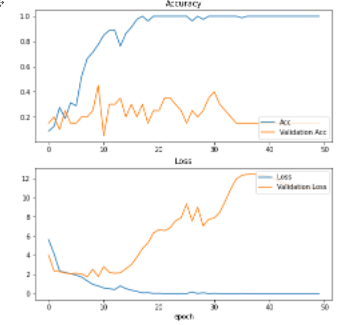
#training

opt = Adam(lr = 0.001)

model.compile(optimizer = opt, loss = 'categorical\_crossentropy', metrics = ['acc'])

his = model.fit(x\_train, y\_train, epochs = 50, batch\_size = 64, validation\_split = 0.2)

#Draw bieu do

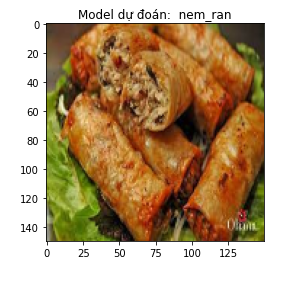
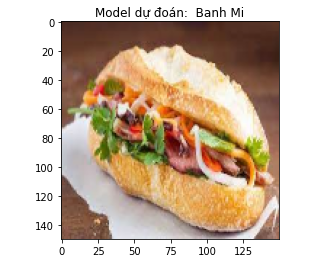


Hình 6 Biểu đồ training 10 món ăn

#add label

label = ['Banh\_chung','Banh\_day','Banh Mi','Bun dau mam tom','Che\_buoi','com\_tam','nem\_ran','Pho','Thit kho hot vit','Vit lon']

# test model



Hình 7 Kết quả training model

1. **Training CNN MNIST**

#add lib

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

#chuan bi ham ve bieu do

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])

print(x\_train.shape, y\_train.shape)

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)

y\_test = np\_utils.to\_categorical(y\_test)

y\_train.shape

#tao model

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()

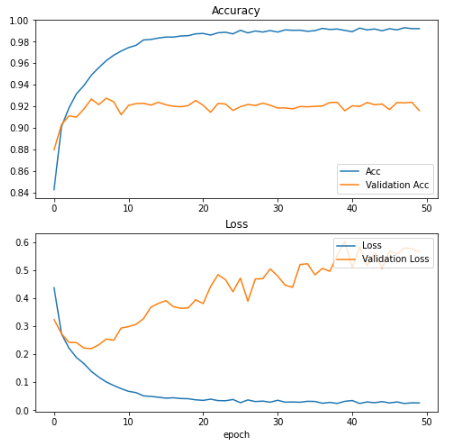
#training

opt = Adam(lr = 0.001)

model.compile(optimizer = opt, loss = 'categorical\_crossentropy', metrics = ['acc'])

his = model.fit(x\_train, y\_train, epochs = 50, batch\_size = 64, validation\_split = 0.2)

#ket qua training



Hình 8 Ket qua training model

#save model

model.save('CNN\_fashion\_mnist.h5')

# chuan bi data test

x\_test\_2 = x\_test.reshape(10000, 28,28)

# label

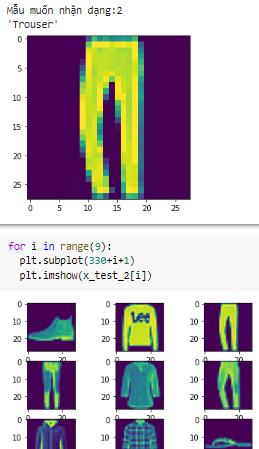
label = ['T-shirt/top','Trouser','Pullover','Dress','Coat','Sandal','Shirt','Sneaker','Bag','Ankle boot']

# ket qua

b = int(input('Mẫu muốn nhận dạng:'))

plt.imshow(x\_test\_2[b])

label[np.argmax(model.predict(x\_test\_2)[b])]



Hình 9 Kết quả nhận diện

1. **Training FASION\_MNIST**

#import thư viện

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

# chuan bi ham ve bieu do

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])

print(x\_train.shape, y\_train.shape)

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)

y\_test = np\_utils.to\_categorical(y\_test)

y\_train.shape

#chuan bi model

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()

#training

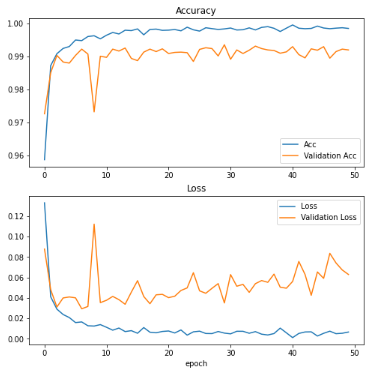
opt = Adam(lr = 0.001)

model.compile(optimizer = opt, loss = 'categorical\_crossentropy', metrics = ['acc'])

his = model.fit(x\_train, y\_train, epochs = 50, batch\_size = 64, validation\_split = 0.2)

#ve bieu do

plot\_history(his)



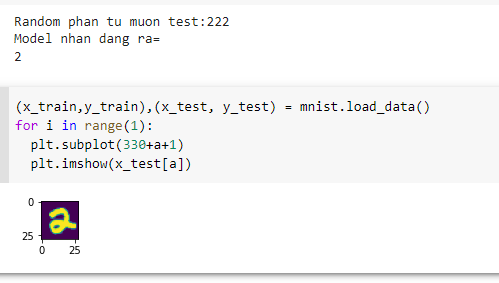
Hình 10 Kết quả training model

# test

a = int(input('Random phan tu muon test:'))

print('Model nhan dang ra= ')

print(np.argmax(model.predict(x\_test),axis=1)[a])

#ket qua  


Hình 11 Kết quả test model

1. **Training Cifar-100**

# add lib  
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  cifar100

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

#chuan bi ham ve bieu do

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) = cifar100.load\_data()

for i in range(9):

  plt.subplot(330+i+1)

  plt.imshow(x\_test[i])

print(x\_train.shape, y\_train.shape)

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,100)

y\_test = np\_utils.to\_categorical(y\_test,100)

#chuan bi model

model = Sequential()

model.add(Conv2D(32,(3,3),activation = 'relu',kernel\_initializer='he\_uniform',padding ='same',input\_shape = (32,32,3)))

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(512, activation ='relu',kernel\_initializer='he\_uniform'))

model.add(Dense(100,activation ='softmax'))

model.summary()

#training

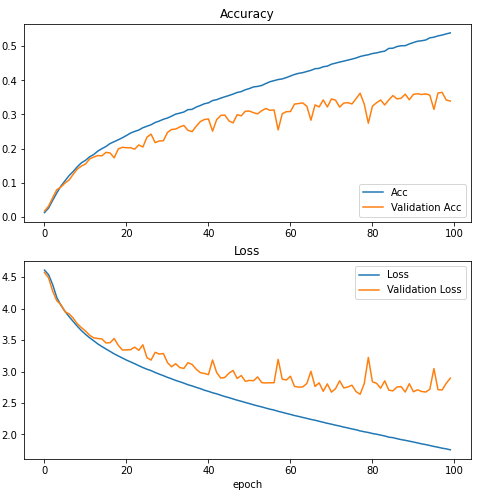
opt = SGD(lr = 0.001)

model.compile(optimizer = opt, loss ='categorical\_crossentropy', metrics = ['acc'])

his = model.fit(x\_train, y\_train, batch\_size = 64, epochs = 100, validation\_data = (x\_test,y\_test))

#bieu do model training duoc

plot\_history(his)



Hình 12 Kết quả training model

#lưu model

model.save('CNN\_cifar100.h5')

# creat label<99>

dict = {0: 'apple',1: 'aquarium\_fish',2: 'baby',3: 'bear',4: 'beaver',5: 'bed',6: 'bee',7: 'beetle',8: 'bicycle',9: 'bottle',10:'bowl',11: 'boy',12: 'bridge',13: 'bus',14: 'butterfly',15: 'camel',16: 'can',17: 'castle',18: 'caterpillar',19: 'cattle',20: 'chair',21: 'chimpanzee',22: 'clock',23: 'cloud',24: 'cockroach',25:'couch',26: 'cra',27: 'crocodile',28:'cup',29: 'dinosaur',30: 'dolphin',31: 'elephant',32: 'flatfish',33: 'forest',34: 'fox',35: 'girl',36: 'hamster',37: 'house',38: 'kangaroo',39: 'keyboard',40: 'lamp',41: 'lawn\_mower',42: 'leopard',43: 'lion',44: 'lizard',45: 'lobster',46: 'man',47: 'maple\_tree',48: 'motorcycle',49: 'mountain',50: 'mouse',51: 'mushroom',52: 'oak\_tree',53: 'orange',54: 'orchid',55: 'otter',56: 'palm\_tree',57: 'pear',58: 'pickup\_truck',59: 'pine\_tree',60: 'plain',61: 'plate',62: 'poppy',63: 'porcupine',64: 'possum',65: 'rabbit',66: 'raccoon',

67: 'ray',68: 'road',69: 'rocket',70: 'rose',71: 'sea',72: 'seal',3: 'shark',74: 'shrew',75: 'skunk',76: 'skyscraper',77: 'snail',

78: 'snake',79: 'spider',80: 'squirrel',81: 'streetcar',82: 'sunflower',

83: 'sweet\_pepper',84: 'table',85: 'tank',86: 'telephone',87: 'television',88: 'tiger',89: 'tractor',90: 'train',91: 'trout',92: 'tulip',93: 'turtle',94: 'wardrobe',95: 'whale',96: 'willow\_tree',97: 'wolf',98: 'woman',99: 'worm'}

# Test

img = tf.keras.utils.load\_img("testmt.jpg", target\_size = (32,32))

plt.imshow(img)

plt.show()

img = tf.keras.utils.img\_to\_array(img)

img = img.reshape(1,32,32,3)

img = img.astype('float32')

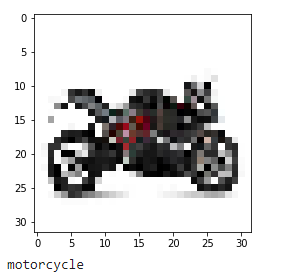
img /=255

print(dict[np.argmax(model.predict(img),axis = 1)[0]])

#import ảnh ngoài



Hình 13 Anh motor dung de test



Hình 14 Ket qua training