Trần Ngọc Đoàn - 19146175 - Lớp chiều T5 - tiết 12-16

https://github.com/DoanAI/Food.git

```
import tensorflow as tf
from tensorflow import keras
from keras.models import Sequential
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.layers import Flatten, Dense, Dropout, Activation
from google.colab import drive
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
```

```
drive.mount('/content/drive',force_remount=True)
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
training_set=train_datagen.flow_from_directory('/content/drive/MyDrive/CNN/FO/training_set
                                               target_size=(256,256),
                                               batch_size=32,
                                               class_mode ='categorical')
test_set=train_datagen.flow_from_directory('/content/drive/MyDrive/CNN/FO/test_set',
                                               target_size=(256,256),
                                               batch_size=32,
                                               class_mode ='categorical')
```

```
Mounted at /content/drive
Found 100 images belonging to 10 classes.
Found 10 images belonging to 10 classes.
```

```
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
```

```
model=Sequential()
model.add(Conv2D(128,(3,3),activation='relu',kernel_initializer='he_uniform',padding='same
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(64,(3,3),activation='relu',kernel_initializer='he_uniform',padding='same'
model.add(MaxPooling2D((2,2)))
model.add(Conv2D(32,(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'))
```

```
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.callbacks import EarlyStopping
model.compile(optimizer = 'adam', loss ='categorical_crossentropy',metrics = ['accuracy'])
callbacks=[EarlyStopping(monitor='val_loss',patience=100)]
history=model.fit(training_set,
                  steps_per_epoch=len(training_set),
                  batch_size = 64,
                  epochs=100,
                  validation_data=test_set,
                  validation_steps=len(test_set),
                  callbacks=callbacks,
                  verbose = 1)
```

```
4/4 [==============================] - 2s 451ms/step - loss: 0.1525 - accuracy: 0.9
Epoch 73/100
4/4 [==============================] - 2s 613ms/step - loss: 0.0583 - accuracy: 0.9
Epoch 74/100
4/4 [==============================] - 2s 446ms/step - loss: 0.0871 - accuracy: 0.9
Epoch 75/100
4/4 [==============================] - 2s 452ms/step - loss: 0.1022 - accuracy: 0.9
Epoch 76/100
4/4 [==============================] - 2s 453ms/step - loss: 0.0289 - accuracy: 0.9
Epoch 77/100
4/4 [==============================] - 2s 623ms/step - loss: 0.0132 - accuracy: 1.0
Epoch 78/100
4/4 [==============================] - 2s 460ms/step - loss: 0.0135 - accuracy: 1.0
Epoch 79/100
4/4 [==============================] - 2s 423ms/step - loss: 0.1977 - accuracy: 0.9
Epoch 80/100
4/4 [==============================] - 2s 436ms/step - loss: 0.2426 - accuracy: 0.9
Epoch 81/100
4/4 [==============================] - 2s 449ms/step - loss: 0.0614 - accuracy: 0.9
Epoch 82/100
4/4 [==============================] - 2s 470ms/step - loss: 0.1350 - accuracy: 0.9
Epoch 83/100
4/4 [==============================] - 2s 458ms/step - loss: 0.1034 - accuracy: 0.9
Epoch 84/100
4/4 [==============================] - 2s 465ms/step - loss: 0.2400 - accuracy: 0.9
Epoch 85/100
4/4 [==============================] - 2s 457ms/step - loss: 0.3151 - accuracy: 0.9
Epoch 86/100
4/4 [==============================] - 2s 423ms/step - loss: 0.1787 - accuracy: 0.9
Epoch 87/100
4/4 [==============================] - 2s 451ms/step - loss: 0.0759 - accuracy: 1.0
Epoch 88/100
4/4 [==============================] - 2s 428ms/step - loss: 0.2590 - accuracy: 0.9
Epoch 89/100
4/4 [==============================] - 2s 631ms/step - loss: 0.1492 - accuracy: 0.9
Epoch 90/100
4/4 [==============================] - 2s 455ms/step - loss: 0.0499 - accuracy: 0.9
Epoch 91/100
4/4 [==============================] - 2s 452ms/step - loss: 0.4279 - accuracy: 0.9
Epoch 92/100
4/4 [==============================] - 2s 455ms/step - loss: 0.2876 - accuracy: 0.9
Epoch 93/100
4/4 [==============================] - 2s 622ms/step - loss: 0.2762 - accuracy: 0.9
Epoch 94/100
4/4 [==============================] - 2s 459ms/step - loss: 0.1785 - accuracy: 0.9
Epoch 95/100
4/4 [==============================] - 2s 456ms/step - loss: 0.1494 - accuracy: 0.
```

```
4/4 [==============================] - 2s 456ms/step - loss: 0.1494 - accuracy: 0.
Epoch 96/100
4/4 [==============================] - 2s 644ms/step - loss: 0.1588 - accuracy: 0.
Epoch 97/100
4/4 [==============================] - 2s 632ms/step - loss: 0.1438 - accuracy: 0.
Epoch 98/100
4/4 [==============================] - 2s 647ms/step - loss: 0.1292 - accuracy: 0.
Epoch 99/100
4/4 [==============================] - 2s 462ms/step - loss: 0.0615 - accuracy: 0.
Epoch 100/100
4/4 [==============================] - 2s 428ms/step - loss: 0.1172 - accuracy: 0.
```

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

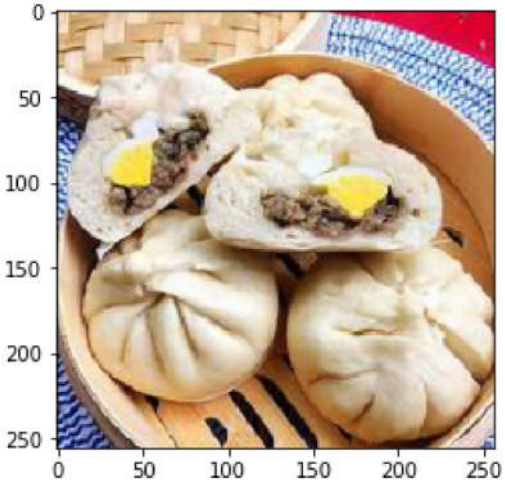
```
Sai số kiểm tra là:  0.37496986985206604
Độ chính xác kiểm tra là:  0.8999999761581421
```

```
model.save('model_fruit.h5')
from tensorflow.keras.models import load_model
model=load_model('model_fruit.h5')
```

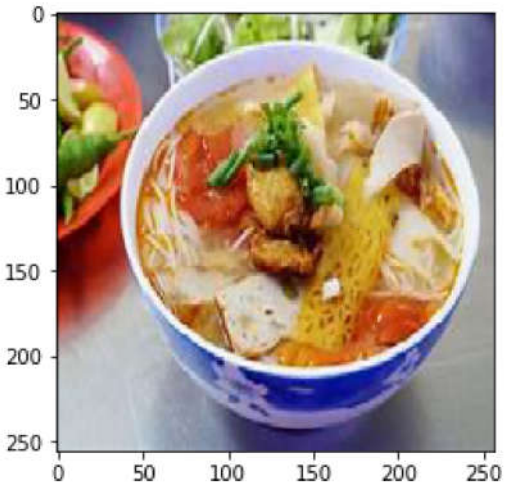
```
from tensorflow.keras.utils import load_img
from tensorflow.keras.utils import img_to_array
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import math

img_0 = load_img('/content/drive/MyDrive/CNN/FO/prediction/Banh-xeo.jpg', target_size=(256
img_1 = load_img('/content/drive/MyDrive/CNN/FO/prediction/BanhBao.jpg', target_size=(256,
img_2 = load_img('/content/drive/MyDrive/CNN/FO/prediction/Bun-ca.jpg', target_size=(256,2
img_3 = load_img('/content/drive/MyDrive/CNN/FO/prediction/Cha.jpg', target_size=(256,256)
img_4 = load_img('/content/drive/MyDrive/CNN/FO/prediction/Com.jpg', target_size=(256,256)
img_5 = load_img('/content/drive/MyDrive/CNN/FO/prediction/Hamburger.jpg', target_size=(25
img_6 = load_img('/content/drive/MyDrive/CNN/FO/prediction/Hotdog.jpg', target_size=(256,2
img_7 = load_img('/content/drive/MyDrive/CNN/FO/prediction/Mi.jpg', target_size=(256,256))
img_8 = load_img('/content/drive/MyDrive/CNN/FO/prediction/Pho.jpg', target_size=(256,256)
img_9 = load_img('/content/drive/MyDrive/CNN/FO/prediction/Pizza.jpg', target_size=(256,25

img = [img_0,img_1,img_2,img_3,img_4,img_5,img_6,img_7,img_8,img_9]
food = ['Bánh xèo','Bánh bao','Bún','Chả','Cơm','Hamburger','Hot dog','Mì','Phở','Pizza']

for i in range(10):
  plt.imshow(img[i])
  imga = img_to_array(img[i])
  imga = imga/255
  imga = np.expand_dims(imga,axis=0)
  result = model.predict(imga)

  if round(result[0][i])==1: prediction = food[i]
  print(prediction)
  plt.show()
```

Bánh xèo

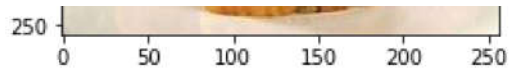
Bánh bao



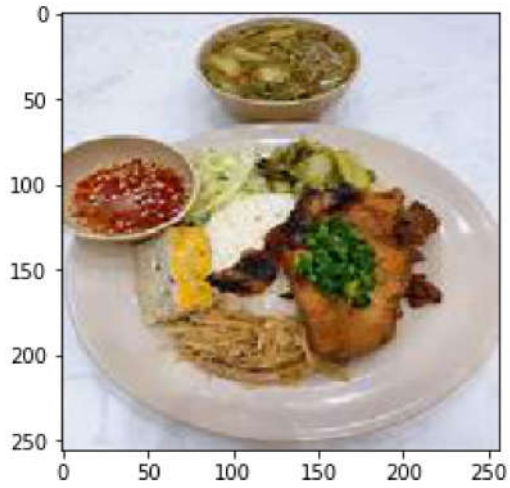
Bún



Chả

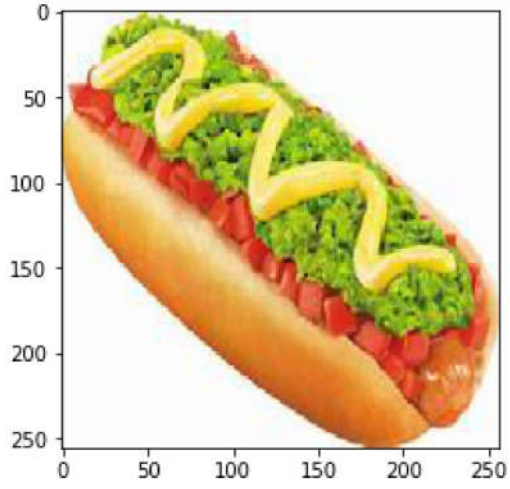


Cơm



Hamburger

Hot dog



Mì