Nguyễn Thái Bình - 19146050 - Nhóm 02CLC - 10_Food

```
import numpy as np
from tensorflow import keras
from tensorflow.keras.models import load_model
from tensorflow.keras.utils import load_img,img_to_array
from tensorflow.keras.preprocessing import image
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt
import os
import matplotlib.pyplot as plt
from skimage import io
from keras.models import Sequential
from keras.utils import np_utils
from keras.layers import Dense,Activation,Dropout,LSTM,BatchNormalization
from keras.layers import Flatten
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras.utils import to_categorical
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
```

```
trainset='/content/drive/MyDrive/10_Food_data/train'
validationset='/content/drive/MyDrive/10_Food_data/validation'
train=ImageDataGenerator(rescale=1/255.0,validation_split=0)
validation=ImageDataGenerator(rescale=1/255.0,validation_split=0.9)
```

```
train_data=train.flow_from_directory(trainset,target_size=(150,150),batch_size=10,class_mode=
validation_set=validation.flow_from_directory(validationset,target_size=(150,150),batch_size=
```

```
Found 127 images belonging to 10 classes.
Found 79 images belonging to 10 classes.
```

```
print(train_data.class_indices)
print(validation_set.class_indices)
```

```
{'banh_mi': 0, 'banh_xeo': 1, 'bun_dau': 2, 'che': 3, 'coffee': 4, 'com_tam': 5, 'goi_cu
{'banh_mi': 0, 'banh_xeo': 1, 'bun_dau': 2, 'che': 3, 'coffee': 4, 'com_tam': 5, 'goi_cu
```

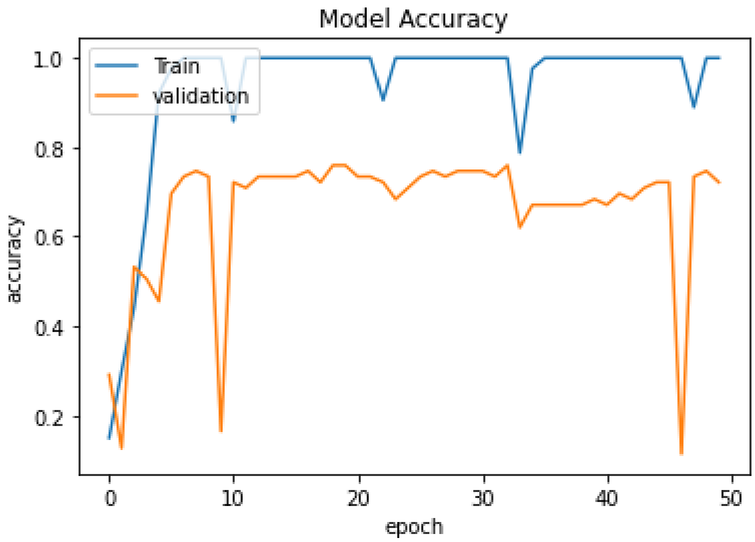
```
model=Sequential()
model.add(Conv2D(32,(3,3),activation='relu',input_shape=(150,150,3)))
model.add(MaxPooling2D((2,2)))
model.add(Conv2D(64,(3,3),activation='relu'))
model.add(MaxPooling2D((2,2)))
model.add(Conv2D(128,(3,3),activation='relu'))
model.add(MaxPooling2D((2,2)))
model.add(Flatten())
model.add(Dense(128,activation='relu'))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


model.compile(loss='categorical_crossentropy',optimizer='rmsprop',metrics=['accuracy'])
history=model.fit(train_data,batch_size=5,epochs=50,verbose=1,validation_data=validation_set)
```

```
13/13 [==============================] - 18s 1s/step - loss: 2.4067e-06 - accuracy: 1
Epoch 23/50
13/13 [==============================] - 18s 1s/step - loss: 4.9297 - accuracy: 0.905
Epoch 24/50
13/13 [==============================] - 18s 1s/step - loss: 0.0280 - accuracy: 1.000
Epoch 25/50
13/13 [==============================] - 18s 1s/step - loss: 0.0016 - accuracy: 1.000
Epoch 26/50
13/13 [==============================] - 18s 1s/step - loss: 3.6078e-04 - accuracy: 1
Epoch 27/50
13/13 [==============================] - 18s 1s/step - loss: 8.1281e-05 - accuracy: 1
Epoch 28/50
13/13 [==============================] - 18s 1s/step - loss: 2.1495e-05 - accuracy: 1
Epoch 29/50
13/13 [==============================] - 18s 1s/step - loss: 1.4842e-05 - accuracy: 1
Epoch 30/50
13/13 [==============================] - 18s 1s/step - loss: 6.3565e-06 - accuracy: 1
Epoch 31/50
13/13 [==============================] - 18s 1s/step - loss: 3.7781e-06 - accuracy: 1
Epoch 32/50
13/13 [==============================] - 18s 1s/step - loss: 2.4302e-06 - accuracy: 1
Epoch 33/50
13/13 [==============================] - 18s 1s/step - loss: 1.4530e-06 - accuracy: 1
Epoch 34/50
13/13 [==============================] - 18s 1s/step - loss: 5.1557 - accuracy: 0.787
Epoch 35/50
13/13 [==============================] - 18s 1s/step - loss: 0.0871 - accuracy: 0.976
Epoch 36/50
13/13 [==============================] - 18s 1s/step - loss: 0.0042 - accuracy: 1.000
Epoch 37/50
13/13 [==============================] - 18s 1s/step - loss: 0.0018 - accuracy: 1.000
Epoch 38/50
13/13 [==============================] - 18s 1s/step - loss: 9.4491e-04 - accuracy: 1
Epoch 39/50
13/13 [==============================] - 18s 1s/step - loss: 3.9565e-04 - accuracy: 1
Epoch 40/50
13/13 [==============================] - 18s 1s/step - loss: 1.4916e-04 - accuracy: 1
Epoch 41/50
13/13 [==============================] - 18s 1s/step - loss: 5.3919e-05 - accuracy: 1
Epoch 42/50
13/13 [==============================] - 18s 1s/step - loss: 1.9419e-05 - accuracy: 1
Epoch 43/50
13/13 [==============================] - 18s 1s/step - loss: 9.5797e-06 - accuracy: 1
Epoch 44/50
13/13 [==============================] - 18s 1s/step - loss: 4.4257e-06 - accuracy: 1
Epoch 45/50
13/13 [==============================] - 18s 1s/step - loss: 2.1908e-06 - accuracy: 1
```

```
Epoch 46/50
13/13 [==============================] - 18s 1s/step - loss: 1.3169e-06 - accuracy: 1
Epoch 47/50
13/13 [==============================] - 18s 1s/step - loss: 5.0963e-04 - accuracy: 1
Epoch 48/50
13/13 [==============================] - 18s 1s/step - loss: 3.1208 - accuracy: 0.889
Epoch 49/50
13/13 [==============================] - 18s 1s/step - loss: 0.0011 - accuracy: 1.000
Epoch 50/50
13/13 [==============================] - 18s 1s/step - loss: 4.8043e-04 - accuracy: 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()
```



```
model.save('/content/drive/MyDrive/BT AI/10_Food.h5')
```

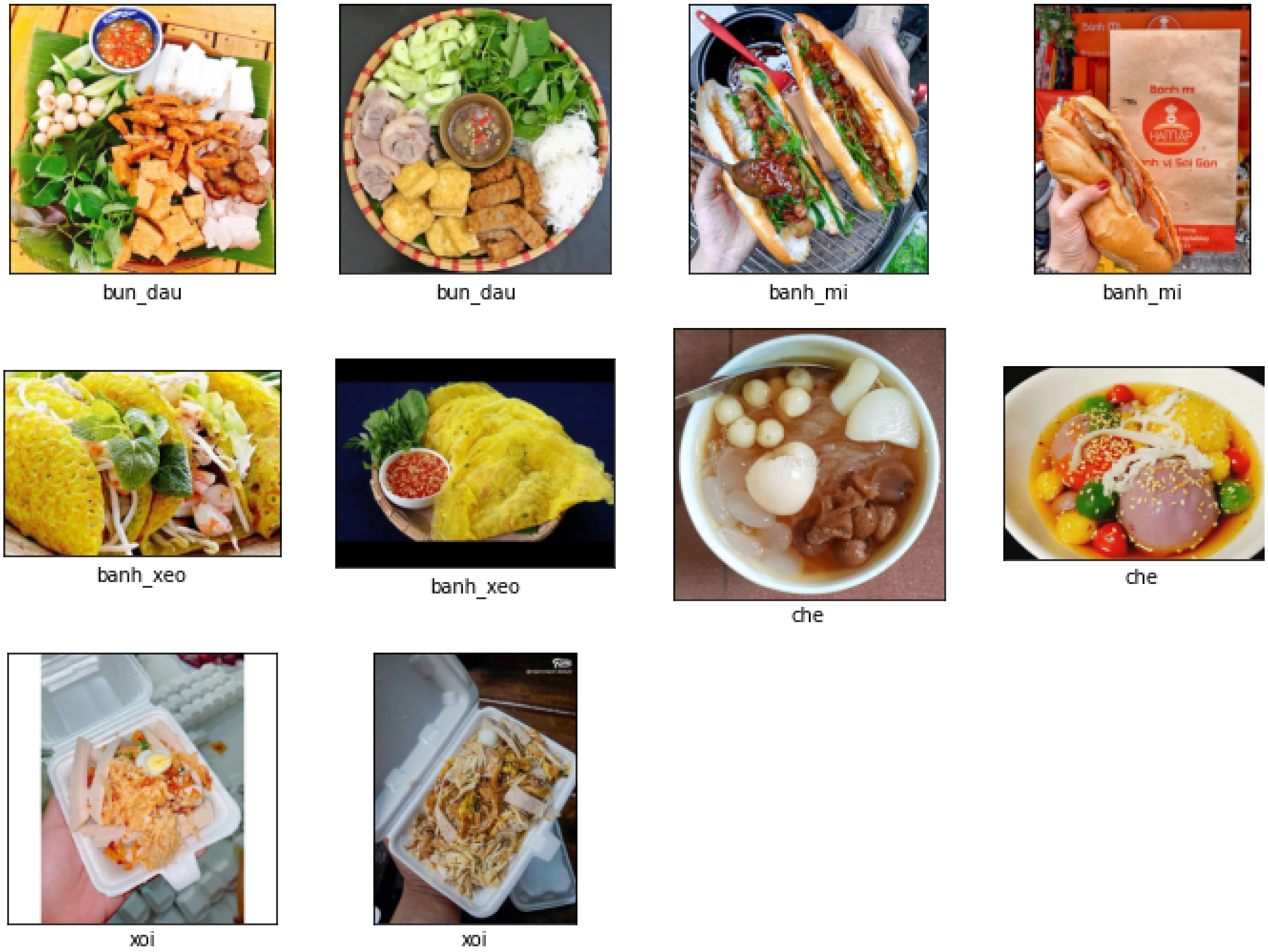
```
load_model('/content/drive/MyDrive/BT AI/10_Food.h5')
```

```
<keras.engine.sequential.Sequential at 0x7fb41e6ff990>
```

```
generator= ImageDataGenerator(rescale=1./255)
generator_data=generator.flow_from_directory('/content/drive/MyDrive/Test food',batch_size=5,

food={0:'banh_mi',
       1:'banh_xeo',
       2:'bun_dau',
       3:'che',
       4:'coffee',
       5:'com_tam',
       6:'goi_cuon',
       7:'hot_vit_lon',
       8:'pho',
       9:'xoi'}
plt.figure(figsize=(12,12))
for i in range(len(generator_data.filenames)):
    plt.subplot(4,4,i+1)
    plt.imshow(io.imread(os.path.join(generator_data.directory,generator_data.filenames[i])))
    plt.xticks([])
    plt.yticks([])
    img=load_img('/content/drive/MyDrive/Test food/'+generator_data.filenames[i],target_size=
    img=img_to_array(img)
    img=img.reshape(1,150,150,3)
    img=img.astype('float')
    img=img/255
    plt.xlabel(food[np.argmax(model.predict(img))])
plt.show()
```

```
Found 10 images belonging to 1 classes.
```



✓ 10 giây hoàn thành lúc 14:20 ● ✕