Đỗ Minh Triều_19146283_FoodsCNN

```
import tensorflow as tf
import matplotlib.pyplot as plt
import cv2
import os
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
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.optimizers import Adam,SGD
from tensorflow.keras.utils import to_categorical
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
import pandas as pd
```

```
load_train_data='../input/foodcnn/monan'
```

```
train=ImageDataGenerator(rescale=1/255)
validation=ImageDataGenerator(rescale=1/255,validation_split=0.2,rotation_range=10,
                              zoom_range = 0.1,
                               width_shift_range=0.1,
                               height_shift_range=0.1,
                               brightness_range=(0.9, 1.1))
```

```
train_data=train.flow_from_directory(
    load_train_data,
    target_size=(150,150),
    batch_size=100,
    class_mode='categorical',
    subset = 'training'
   )
validation_set=validation.flow_from_directory(
    load_train_data,
    target_size=(150,150),
    batch_size=100,
```

```
    class_mode='categorical',
    subset = 'validation'
    )
```

```
Found 5706 images belonging to 10 classes.
Found 1137 images belonging to 10 classes.
```

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

```
{'Banh chung': 0, 'Banh khot': 1, 'Banh mi': 2, 'Banh trang nuong': 3, 'Banh xeo': 4, 'B
{'Banh chung': 0, 'Banh khot': 1, 'Banh mi': 2, 'Banh trang nuong': 3, 'Banh xeo': 4, 'B
```

```
model = Sequential()
model.add(Conv2D(64,(3,3), activation = 'relu', kernel_initializer = 'he_uniform', padding =
model.add(Conv2D(64,(3,3), activation = 'relu', kernel_initializer = 'he_uniform', padding =
model.add(Dropout(0.3))
model.add(MaxPooling2D((2,2)))
model.add(Conv2D(128,(3,3), activation = 'relu', kernel_initializer = 'he_uniform', padding =
model.add(Conv2D(128,(3,3), activation = 'relu', kernel_initializer = 'he_uniform', padding =
model.add(Dropout(0.3))
model.add(MaxPooling2D((2,2)))
model.add(Flatten())
model.add(Dense(256, activation='relu',kernel_initializer='he_uniform'))
model.add(Dropout(0.3))
model.add(Dense(10,activation='softmax'))
model.compile(loss='categorical_crossentropy',optimizer='SGD',metrics=['accuracy'])
history=model.fit(train_data,batch_size=100,epochs=50,verbose=1,validation_data=validation_se
```

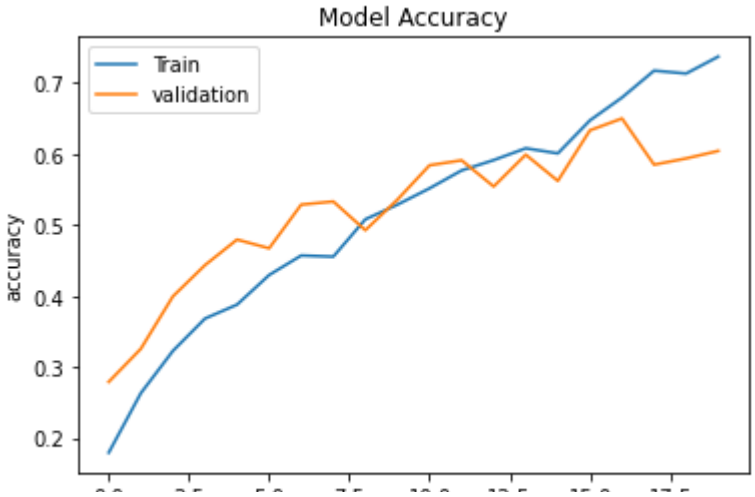
```
Epoch 22/50
58/58 [==============================] - 127s 2s/step - loss: 0.7928 - accuracy: 0.74
Epoch 23/50
58/58 [==============================] - 123s 2s/step - loss: 0.6117 - accuracy: 0.79
Epoch 24/50
58/58 [==============================] - 122s 2s/step - loss: 0.3968 - accuracy: 0.87
Epoch 25/50
58/58 [==============================] - 122s 2s/step - loss: 0.2545 - accuracy: 0.92
Epoch 26/50
58/58 [==============================] - 122s 2s/step - loss: 0.2937 - accuracy: 0.91
Epoch 27/50
58/58 [==============================] - 123s 2s/step - loss: 0.1567 - accuracy: 0.95
Epoch 28/50
58/58 [==============================] - 127s 2s/step - loss: 0.4968 - accuracy: 0.86
Epoch 29/50
58/58 [==============================] - 123s 2s/step - loss: 0.2330 - accuracy: 0.92
Epoch 30/50
58/58 [==============================] - 122s 2s/step - loss: 0.6491 - accuracy: 0.83
Epoch 31/50
58/58 [==============================] - 123s 2s/step - loss: 0.4219 - accuracy: 0.88
Epoch 32/50
58/58 [==============================] - 123s 2s/step - loss: 0.1717 - accuracy: 0.94
```

```
Epoch 33/50
58/58 [==============================] - 123s 2s/step - loss: 0.0971 - accuracy: 0.97
Epoch 34/50
58/58 [==============================] - 126s 2s/step - loss: 0.0729 - accuracy: 0.98
Epoch 35/50
58/58 [==============================] - 122s 2s/step - loss: 0.0632 - accuracy: 0.98
Epoch 36/50
58/58 [==============================] - 122s 2s/step - loss: 0.0555 - accuracy: 0.98
Epoch 37/50
58/58 [==============================] - 123s 2s/step - loss: 0.0521 - accuracy: 0.98
Epoch 38/50
58/58 [==============================] - 122s 2s/step - loss: 0.0837 - accuracy: 0.97
Epoch 39/50
58/58 [==============================] - 123s 2s/step - loss: 0.0431 - accuracy: 0.98
Epoch 40/50
58/58 [==============================] - 123s 2s/step - loss: 0.0352 - accuracy: 0.99
Epoch 41/50
58/58 [==============================] - 124s 2s/step - loss: 0.0402 - accuracy: 0.98
Epoch 42/50
58/58 [==============================] - 123s 2s/step - loss: 0.0269 - accuracy: 0.99
Epoch 43/50
58/58 [==============================] - 123s 2s/step - loss: 0.0235 - accuracy: 0.99
Epoch 44/50
58/58 [==============================] - 123s 2s/step - loss: 0.0315 - accuracy: 0.99
Epoch 45/50
58/58 [==============================] - 122s 2s/step - loss: 0.0272 - accuracy: 0.99
Epoch 46/50
58/58 [==============================] - 123s 2s/step - loss: 0.0294 - accuracy: 0.99
Epoch 47/50
58/58 [==============================] - 123s 2s/step - loss: 0.0226 - accuracy: 0.99
Epoch 48/50
58/58 [==============================] - 122s 2s/step - loss: 1.2770 - accuracy: 0.71
Epoch 49/50
58/58 [==============================] - 124s 2s/step - loss: 0.5924 - accuracy: 0.82
Epoch 50/50
```

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

```
Sai số kiểm tra của mô hình là: 1.0596123933792114
Độ chính xác kiểm tra của môn hình là: 0.6569920778274536
```

```
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()
```




```
food={0:'Banh chung',
       1:'Banh khot',
       2:'Banh mi',
       3:'Banh trang nuong',
       4:'Banh xeo',
       5:'Bun dau mam tom',
       6:'Ca kho to',
       7:'Goi cuon',
       8:'Nem chua',
       9:'Xoi xeo'}
import os
filenames=os.listdir("../input/foodtest/foodtest")
df=pd.DataFrame({'filename':filenames})
url='../input/foodtest/foodtest/'+df['filename']

plt.figure(figsize=(20,20))
for i in range(df.shape[0]):
  plt.subplot(10,10,i+1)
  plt.grid(False)
  plt.xticks([])
  plt.yticks([])
  img=load_img(url[i],target_size=(150,150))
  plt.imshow(img)
  img=img_to_array(img)
  img=img.reshape(1,150,150,3)
  img=img/255.0
  img_pred = model.predict(img)
  plt.xlabel(food[np.argmax(img_pred)])
plt.show()
```

```
model.save('./foodcnn2.h5')
```