10MonAn

```
import tensorflow as tf
from tensorflow import keras
import matplotlib.pyplot as plt
import numpy as np
```

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

```
Mounted at /content/drive/
```

```
import os
train_image_files_path = "/content/drive/MyDrive/AI/Training /Training_2"
valid_image_files_path = "/content/drive/MyDrive/AI/Training /Test_2"
```

```
label=['BanhBao','BanhCuon','BanhXeo','BunBoHue','Cha','ComTam','Hamburger','HotDog','Mi','Pizza'
```

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_data_gen = ImageDataGenerator(rescale=1/255)
validation_data_gen = ImageDataGenerator(rescale=1/255)
```

```
train_generator = train_data_gen.flow_from_directory(train_image_files_path, target_size=(200, 200
validation_generator = validation_data_gen.flow_from_directory(valid_image_files_path, target_size
```

```
Found 145 images belonging to 10 classes.
Found 50 images belonging to 10 classes.
```

```
#MoHinh
from keras.models import Sequential
from keras.layers import Dense, Dropout, Conv2D, MaxPooling2D, Flatten
```

```
model=tf.keras.models.Sequential()
```

```
model.add(Conv2D(32,(3,3), activation='relu',input_shape=(200,200,3)))
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(512, activation=tf.nn.relu))
model.add(Dense(10, activation=tf.nn.softmax))
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
```

```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 198, 198, 32)      896

 max_pooling2d (MaxPooling2D  (None, 99, 99, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 97, 97, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 48, 48, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 46, 46, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 23, 23, 128)      0
 2D)

 flatten (Flatten)           (None, 67712)             0

 dense (Dense)               (None, 512)               34669056

 dense_1 (Dense)             (None, 10)                5130

=================================================================
Total params: 34,767,434
Trainable params: 34,767,434
Non-trainable params: 0
_________________________________________________________________
```

```
from tensorflow.keras.optimizers import Adam
model.compile(optimizer=Adam(learning_rate=0.001), loss='categorical_crossentropy', metrics=['acc
```

```
EPOCHS=50
history=model.fit(train_generator, steps_per_epoch=3, epochs=EPOCHS, verbose=1, validation_data =
```

```
Epoch 1/50
3/3 [==============================] - ETA: 0s - loss: 5.9433 - acc: 0.1146WARNING:tensor
3/3 [==============================] - 99s 34s/step - loss: 5.9433 - acc: 0.1146 - val_lo
Epoch 2/50
3/3 [==============================] - 7s 2s/step - loss: 2.4480 - acc: 0.1458
Epoch 3/50
3/3 [==============================] - 2s 947ms/step - loss: 2.2859 - acc: 0.1771
Epoch 4/50
3/3 [==============================] - 2s 548ms/step - loss: 2.2191 - acc: 0.2716
Epoch 5/50
3/3 [==============================] - 1s 394ms/step - loss: 2.1405 - acc: 0.2083
Epoch 6/50
3/3 [==============================] - 1s 309ms/step - loss: 2.0259 - acc: 0.3086
Epoch 7/50
3/3 [==============================] - 1s 416ms/step - loss: 1.9176 - acc: 0.2840
Epoch 8/50
3/3 [==============================] - 1s 407ms/step - loss: 1.6927 - acc: 0.3580
Epoch 9/50
3/3 [==============================] - 1s 392ms/step - loss: 1.5644 - acc: 0.4815
Epoch 10/50
3/3 [==============================] - 1s 266ms/step - loss: 1.2453 - acc: 0.5926
Epoch 11/50
3/3 [==============================] - 1s 381ms/step - loss: 1.0377 - acc: 0.7083
Epoch 12/50
```

```
3/3 [==============================] - 1s 380ms/step - loss: 0.8227 - acc: 0.7284
Epoch 13/50
3/3 [==============================] - 1s 281ms/step - loss: 0.6555 - acc: 0.8272
Epoch 14/50
3/3 [==============================] - 1s 414ms/step - loss: 0.4306 - acc: 0.8642
Epoch 15/50
3/3 [==============================] - 1s 468ms/step - loss: 0.4306 - acc: 0.8889
Epoch 16/50
3/3 [==============================] - 1s 332ms/step - loss: 0.1945 - acc: 0.9877
Epoch 17/50
3/3 [==============================] - 1s 420ms/step - loss: 0.2165 - acc: 0.9375
Epoch 18/50
3/3 [==============================] - 1s 352ms/step - loss: 0.1113 - acc: 0.9877
Epoch 19/50
3/3 [==============================] - 1s 332ms/step - loss: 0.0975 - acc: 0.9896
Epoch 20/50
3/3 [==============================] - 1s 270ms/step - loss: 0.1229 - acc: 0.9630
Epoch 21/50
3/3 [==============================] - 1s 369ms/step - loss: 0.0776 - acc: 0.9896
Epoch 22/50
3/3 [==============================] - 1s 435ms/step - loss: 0.0538 - acc: 0.9896
Epoch 23/50
3/3 [==============================] - 1s 308ms/step - loss: 0.0406 - acc: 1.0000
Epoch 24/50
3/3 [==============================] - 1s 400ms/step - loss: 0.0253 - acc: 1.0000
Epoch 25/50
3/3 [==============================] - 1s 415ms/step - loss: 0.0128 - acc: 1.0000
Epoch 26/50
3/3 [==============================] - 1s 423ms/step - loss: 0.0128 - acc: 1.0000
Epoch 27/50
3/3 [==============================] - 1s 327ms/step - loss: 0.0036 - acc: 1.0000
Epoch 28/50
3/3 [==============================] - 1s 358ms/step - loss: 0.0020 - acc: 1.0000
```

```
from google.colab import files
from keras.preprocessing import image
%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib.image as mpimg


uploaded=files.upload()
for fn in uploaded.keys():
  #predicting images
  path='/content/' + fn
  #In ảnh đọc được
  plt.imshow(mpimg.imread(path))
  img=image.load_img(path,target_size=(200,200))
  x=image.img_to_array(img)
  x=np.expand_dims(x,axis=0)
  images=np.vstack([x])
  y_predict = model.predict(images,batch_size=10)
  print(y_predict)
  print('Giá trị dự đoán: ', label[np.argmax(y_predict)])
```

Chọn tệp asassasasa.jpg

- **asassasasa.jpg**(image/jpeg) - 189951 bytes, last modified: 19/5/2022 - 100% done

```
Saving asassasasa.jpg to asassasasa.jpg
[[0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]]
Giá trị dự đoán:  BanhCuon
```