10LoaiQua

```
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/')
```

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

```
import os
train_image_files_path = "/content/drive/MyDrive/AI/Training /Training_1"
valid_image_files_path = "/content/drive/MyDrive/AI/Training /Test_1"
```

```
label=['QuaBo','QuaBuoi','QuaChuoi','QuaLe','QuaMangCut','QuaNa','QuaNho','QuaOi','QuaTao','QuaTha
```

```
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 147 images belonging to 10 classes.
Found 48 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_6"
```

```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_21 (Conv2D)          (None, 198, 198, 32)      896

 max_pooling2d_18 (MaxPoolin  (None, 99, 99, 32)       0
 g2D)

 conv2d_22 (Conv2D)          (None, 97, 97, 64)        18496

 max_pooling2d_19 (MaxPoolin  (None, 48, 48, 64)       0
 g2D)

 conv2d_23 (Conv2D)          (None, 46, 46, 128)       73856

 max_pooling2d_20 (MaxPoolin  (None, 23, 23, 128)      0
 g2D)

 flatten_6 (Flatten)         (None, 67712)             0

 dense_12 (Dense)            (None, 512)               34669056

 dense_13 (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=100
history=model.fit(train_generator, steps_per_epoch=3, epochs=EPOCHS, verbose=1, validation_data =
```

```
3/3 [==============================] - 1s 285ms/step - loss: 0.1864 - acc: 0.9639
Epoch 19/100
3/3 [==============================] - 1s 294ms/step - loss: 0.1350 - acc: 0.9759
Epoch 20/100
3/3 [==============================] - 1s 362ms/step - loss: 0.0526 - acc: 0.9880
Epoch 21/100
3/3 [==============================] - 2s 522ms/step - loss: 0.0773 - acc: 0.9896
Epoch 22/100
3/3 [==============================] - 2s 606ms/step - loss: 0.0547 - acc: 0.9792
Epoch 23/100
3/3 [==============================] - 2s 491ms/step - loss: 0.0161 - acc: 1.0000
Epoch 24/100
3/3 [==============================] - 1s 449ms/step - loss: 0.0159 - acc: 1.0000
Epoch 25/100
3/3 [==============================] - 1s 533ms/step - loss: 0.0157 - acc: 1.0000
Epoch 26/100
3/3 [==============================] - 1s 485ms/step - loss: 0.0047 - acc: 1.0000
Epoch 27/100
3/3 [==============================] - 2s 538ms/step - loss: 0.0031 - acc: 1.0000
Epoch 28/100
3/3 [==============================] - 1s 515ms/step - loss: 0.0028 - acc: 1.0000
Epoch 29/100
3/3 [==============================] - 1s 446ms/step - loss: 0.0021 - acc: 1.0000
Epoch 30/100
```

```
Epoch 30/100
3/3 [==============================] - 1s 377ms/step - loss: 0.0011 - acc: 1.0000
Epoch 31/100
3/3 [==============================] - 2s 450ms/step - loss: 9.9086e-04 - acc: 1.0000
Epoch 32/100
3/3 [==============================] - 2s 543ms/step - loss: 7.7719e-04 - acc: 1.0000
Epoch 33/100
3/3 [==============================] - 2s 394ms/step - loss: 8.9986e-04 - acc: 1.0000
Epoch 34/100
3/3 [==============================] - 1s 432ms/step - loss: 4.5780e-04 - acc: 1.0000
Epoch 35/100
3/3 [==============================] - 1s 452ms/step - loss: 5.5866e-04 - acc: 1.0000
Epoch 36/100
3/3 [==============================] - 1s 343ms/step - loss: 2.9522e-04 - acc: 1.0000
Epoch 37/100
3/3 [==============================] - 1s 467ms/step - loss: 3.3596e-04 - acc: 1.0000
Epoch 38/100
3/3 [==============================] - 2s 516ms/step - loss: 2.0372e-04 - acc: 1.0000
Epoch 39/100
3/3 [==============================] - 1s 455ms/step - loss: 1.7123e-04 - acc: 1.0000
Epoch 40/100
3/3 [==============================] - 2s 570ms/step - loss: 2.2996e-04 - acc: 1.0000
Epoch 41/100
3/3 [==============================] - 1s 418ms/step - loss: 1.7998e-04 - acc: 1.0000
Epoch 42/100
3/3 [==============================] - 2s 617ms/step - loss: 1.7082e-04 - acc: 1.0000
Epoch 43/100
3/3 [==============================] - 2s 506ms/step - loss: 1.5778e-04 - acc: 1.0000
Epoch 44/100
3/3 [==============================] - 2s 552ms/step - loss: 1.9378e-04 - acc: 1.0000
Epoch 45/100
3/3 [==============================] - 1s 452ms/step - loss: 1.7126e-04 - acc: 1.0000
Epoch 46/100
3/3 [==============================] - 2s 466ms/step - loss: 1.5301e-04 - 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 ch.jpg

- **ch.jpg**(image/jpeg) - 13479 bytes, last modified: 19/5/2022 - 100% done

```
Saving ch.jpg to ch (1).jpg
[[0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]]
Giá trị dự đoán:  QuaChuoi
```