TRƯỜNG ĐẠI HỌC SƯ PHẠM KĨ THUẬT TP.HCM

KHOA ĐÀO TẠO CHẤT LƯỢNG CAO

HCMUTE

BÁO CÁO MÔN HỌC

MÔN HỌC: ARTIFICAL INTELLIGENCE

ASSIGNMENT:

NHẬN DIỆN KHUÔN MẶT CỦA THÀNH VIÊN NHÓM

GVGD: PGS.TS. Nguyễn Trường Thịnh

SVTH: Trần Quang Huy_19146195

Nguyễn Đức Tài_19146255

Phạm Minh Tuấn_19146297

Lớp AI_Nhóm 02CLC_CT6_Tiết 12-15

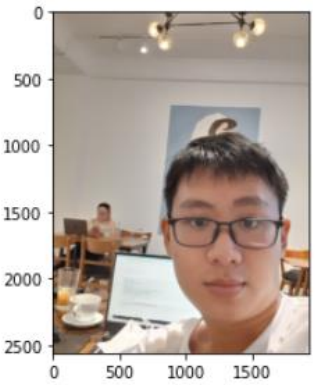
Thành phố Hồ Chí Minh, Tháng 5 năm 2022

Face Member Regconition

```
# Import Libraries
import tensorflow as tf
import matplotlib.pyplot as plt
import cv2 as cv
import os
import numpy as np
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras import layers
# Gọi các thư viện cần thiết
import numpy as np
import pandas as pd # Xu lý bảng
import seaborn as sns # Vẽ biểu đồ thị của dữ liệu
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler # Xử lý chuẩn hóa dữ liệu
from sklearn.model_selection import train_test_split # Chia dữ liệu ra làm 2 phầ
n
from keras.layers import Dense, Activation, Dropout, BatchNormalization, LSTM
 # LSTM  biên dạng ANN, BatchNormalization: cho nhỏ lại
from keras.models import Sequential
from tensorflow.keras.utils import to_categorical # Sử dung để làm nổi đối tượng
 cần phân loại
from keras import callbacks
from sklearn.metrics import precision_score, recall_score, confusion_matrix, cla
ssification_report, accuracy_score, f1_score # Để đo lường

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from keras.utils import np_utils
from tensorflow.keras.preprocessing import image
from tensorflow.keras.optimizers import RMSprop
from keras.models import Sequential
from keras.layers import Dense, Dropout
from keras.callbacks import EarlyStopping
import matplotlib.pyplot as plt
import tensorflow as tf
import numpy as np
import cv2
import os

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from keras.layers import Dense, Activation, Dropout, BatchNormalization, LSTM
from keras.models import Sequential
from tensorflow.keras.utils import to_categorical
from keras import callbacks
from sklearn.metrics import precision_score, recall_score, confusion_matrix, cla
ssification_report, accuracy_score, f1_score
import keras
from keras.models import Sequential
from keras.layers import Dense # fully connected
from keras.datasets import boston_housing
from tensorflow.keras.optimizers import RMSprop # toi uu
from keras.callbacks import EarlyStopping # dung lai ngay lap tuc
from sklearn.preprocessing import scale # xu li du lieu
from sklearn.preprocessing import StandardScaler # xu li du lieu

# Load 1 image
img = image.load_img("../input/traintth1/Tai/1.png")
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x7
```




```
image_generator = ImageDataGenerator(rescale=1/255, validation_split=0.2)

train_dataset = image_generator.flow_from_directory(batch_size=32,
                                                    directory="../input/traintth1/"
,
                                                    shuffle=True,
                                                    target_size=(150, 150),
                                                    subset="training",
                                                    class_mode='categorical')

validation_dataset = image_generator.flow_from_directory(batch_size=32,
                                                    directory="../input/traintth1/"
,
                                                    shuffle=True,
```

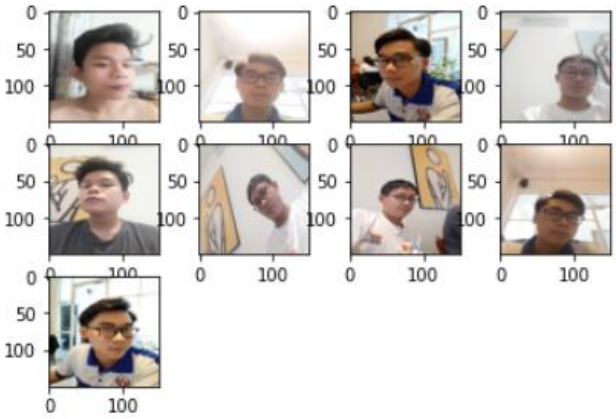
```
                                                target_size=(150, 150),
                                                subset="validation",
                                                class_mode='categorical')
                        Found 143 images belonging to 3 classes.
                        Found 35 images belonging to 3 classes.
train_dataset.class_indices
                              {'Huy': 0, 'Tai': 1, 'Tuan': 2}

# Create model
from keras.layers import Conv2D, MaxPooling2D
model = Sequential()
model.add(Conv2D(32,(3,3), activation='relu',kernel_initializer='he_uniform',pad
ding='same',input_shape=(150,150,3)))
model.add(Conv2D(32,(3,3), activation='relu',kernel_initializer='he_uniform',pad
ding='same'))
model.add(MaxPooling2D(2,2))

model.add(Conv2D(64,(3,3), activation='relu',kernel_initializer='he_uniform',pad
ding='same')) # 64 lan tich chap
model.add(Conv2D(64,(3,3), activation='relu',kernel_initializer='he_uniform',pad
ding='same'))
model.add(MaxPooling2D(2,2))

model.add(Conv2D(128,(3,3), activation='relu',kernel_initializer='he_uniform',pa
dding='same')) # 128 lan tich chap
model.add(Conv2D(128,(3,3), activation='relu',kernel_initializer='he_uniform',pa
dding='same'))
model.add(MaxPooling2D(2,2))

# model.add(Conv2D(256,(3,3), activation='relu',kernel_initializer='he_uniform',
padding='same')) # 128 lan tich chap
# model.add(Conv2D(256,(3,3), activation='relu',kernel_initializer='he_uniform',
padding='same'))
# model.add(MaxPooling2D(2,2))
from keras.layers import Dense, Activation, Flatten
model.add(Flatten())
model.add(Dense(128, activation = 'relu', kernel_initializer='he_uniform'))
model.add(Dense(3))
model.summary()
```

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_115 (Conv2D)          (None, 150, 150, 32)      896
_________________________________________________________________
conv2d_116 (Conv2D)          (None, 150, 150, 32)      9248
_________________________________________________________________
max_pooling2d_57 (MaxPooling (None, 75, 75, 32)        0
_________________________________________________________________
conv2d_117 (Conv2D)          (None, 75, 75, 64)        18496
_________________________________________________________________
conv2d_118 (Conv2D)          (None, 75, 75, 64)        36928
_________________________________________________________________
max_pooling2d_58 (MaxPooling (None, 37, 37, 64)        0
_________________________________________________________________
conv2d_119 (Conv2D)          (None, 37, 37, 128)       73856
_________________________________________________________________
conv2d_120 (Conv2D)          (None, 37, 37, 128)       147584
_________________________________________________________________
max_pooling2d_59 (MaxPooling (None, 18, 18, 128)       0
_________________________________________________________________
flatten_17 (Flatten)         (None, 41472)             0
_________________________________________________________________
dense_34 (Dense)             (None, 128)               5308544
_________________________________________________________________
dense_35 (Dense)             (None, 3)                 387
=================================================================
Total params: 5,595,939
Trainable params: 5,595,939
Non-trainable params: 0
```

```
# Compile
model.compile(loss='mse',optimizer=RMSprop(),metrics=['accuracy'])
# Train model
history=model.fit(train_dataset,batch_size=100,epochs=10,validation_data=validat
ion_dataset)
```

```
Epoch 1/10
5/5 [==============================] - 17s 3s/step - loss: 2634.1877 - accuracy: 0.3217 - val_loss: 0.2621 - val_accuracy: 0.3
Epoch 2/10
5/5 [==============================] - 14s 3s/step - loss: 0.2744 - accuracy: 0.3916 - val_loss: 0.3605 - val_accuracy: 0.3143
Epoch 3/10
5/5 [==============================] - 14s 3s/step - loss: 0.2313 - accuracy: 0.4965 - val_loss: 0.1880 - val_accuracy: 0.3429
Epoch 4/10
5/5 [==============================] - 14s 3s/step - loss: 0.1276 - accuracy: 0.7343 - val_loss: 0.1122 - val_accuracy: 0.8857
Epoch 5/10
5/5 [==============================] - 14s 3s/step - loss: 0.0924 - accuracy: 0.9161 - val_loss: 0.1753 - val_accuracy: 0.6571
Epoch 6/10
5/5 [==============================] - 14s 3s/step - loss: 0.1725 - accuracy: 0.7203 - val_loss: 0.0853 - val_accuracy: 0.8571
Epoch 7/10
5/5 [==============================] - 14s 3s/step - loss: 0.0846 - accuracy: 0.9790 - val_loss: 0.0649 - val_accuracy: 0.9143
Epoch 8/10
5/5 [==============================] - 14s 3s/step - loss: 0.0343 - accuracy: 0.9860 - val_loss: 0.0600 - val_accuracy: 0.9429
Epoch 9/10
5/5 [==============================] - 14s 3s/step - loss: 0.0725 - accuracy: 0.9930 - val_loss: 0.0685 - val_accuracy: 0.9143
Epoch 10/10
5/5 [==============================] - 14s 3s/step - loss: 0.0577 - accuracy: 0.9650 - val_loss: 0.0552 - val_accuracy: 0.9143
```

```
for i in range(10,19):
  plt.subplot(330+i+1)
  plt.imshow(X_train[i])
plt.show()
```

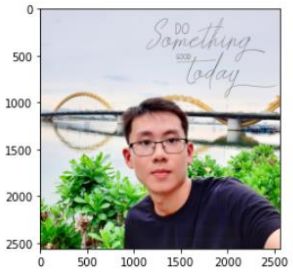


```
# Save model
from tensorflow.keras.models import load_model
model.save('Final.h5')
model_ANN = load_model('Final.h5')

# Check accuracy
from tensorflow.keras.utils import load_img, img_to_array
import numpy as np
filename = "../input/taistest/tai1.png"

predict = ['QuangHuy-19146195','ĐứcTài-19146255','PMinhTuan-19146297']
predict = np.array(predict)

img = load_img(filename,target_size=(150,150))
img = img_to_array(img)
img = img.reshape(1,150,150,3)
img = img.astype('float32')
img = img/255

result = np.argmax(model_ANN.predict(img),axis=-1)
predict[result]
```

```
array(['ĐứcTài-19146255'], dtype='<U18')
```

```
# See input
from tensorflow.keras.utils import load_img, img_to_array
img = image.load_img("../input/taistest/tai1.png")
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x7f455d
```



```
# Draw plot
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epochs')
plt.legend(['train','Validation'])
plt.show()
```