BÁO CÁO ĐỒ ÁN CUỐI KỲ
MÔN MÁY HỌC

XÂY DỰNG ỨNG DỤNG NHẬN DIỆN ẢNH MẶT NGƯỜI ĐEO KHẨU TRANG

Giáo viên phụ trách:
Lê Đình Duy, Phạm Nguyễn Trường An

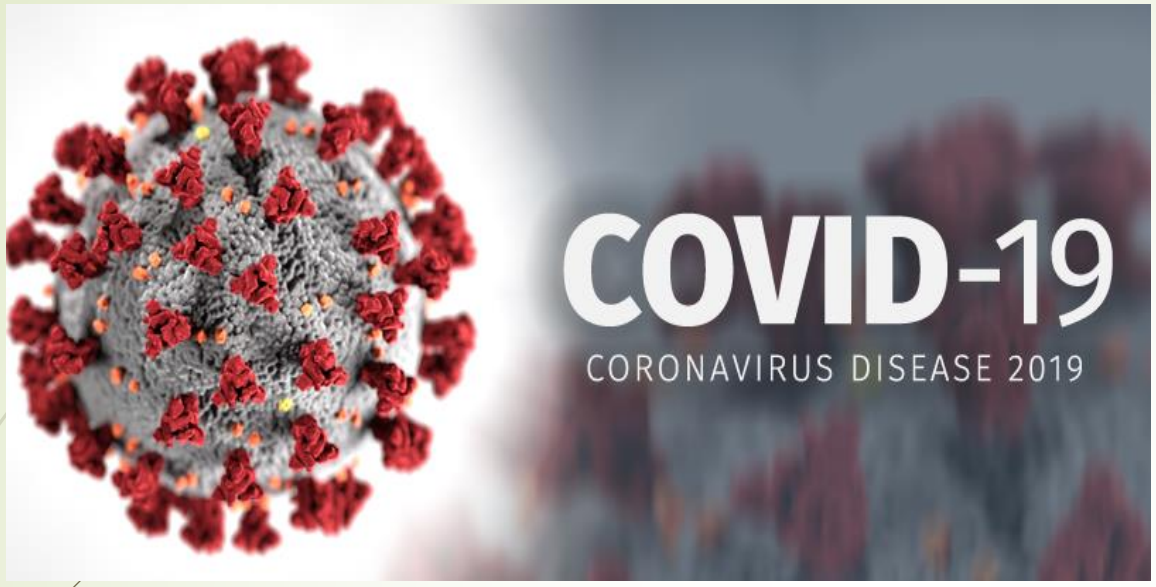
Thành Viên

- Nguyễn Hoàng Thắng - 18521394
- Trần Đỗ Quốc Khiêm - 18520076
- Lê Đại Thành - 18521404

1. MÔ TẢ BÀI TOÁN
2. MÔ TẢ BỘ DỮ LIỆU
3. CÀI ĐẶT MODEL
4. DEMO
5. ĐÁNH GIÁ KẾT QUẢ
6. THAM KHẢO

1

MÔ TẢ BÀI TOÁN



12/2019

215

99 ngày

Sử dụng khẩu trang y tế đúng cách để phòng chống vi rút corona

Mặt xanh có tính chống ẩm quay ra ngoài.

Mặt trắng hút ẩm quay vào trong.

Khẩu trang phải che kín cả mũi lẫn miệng

Khẩu trang chỉ dùng 1 lần mỗi lần không quá 8 tiếng hoặc khi thấy khẩu trang ẩm.

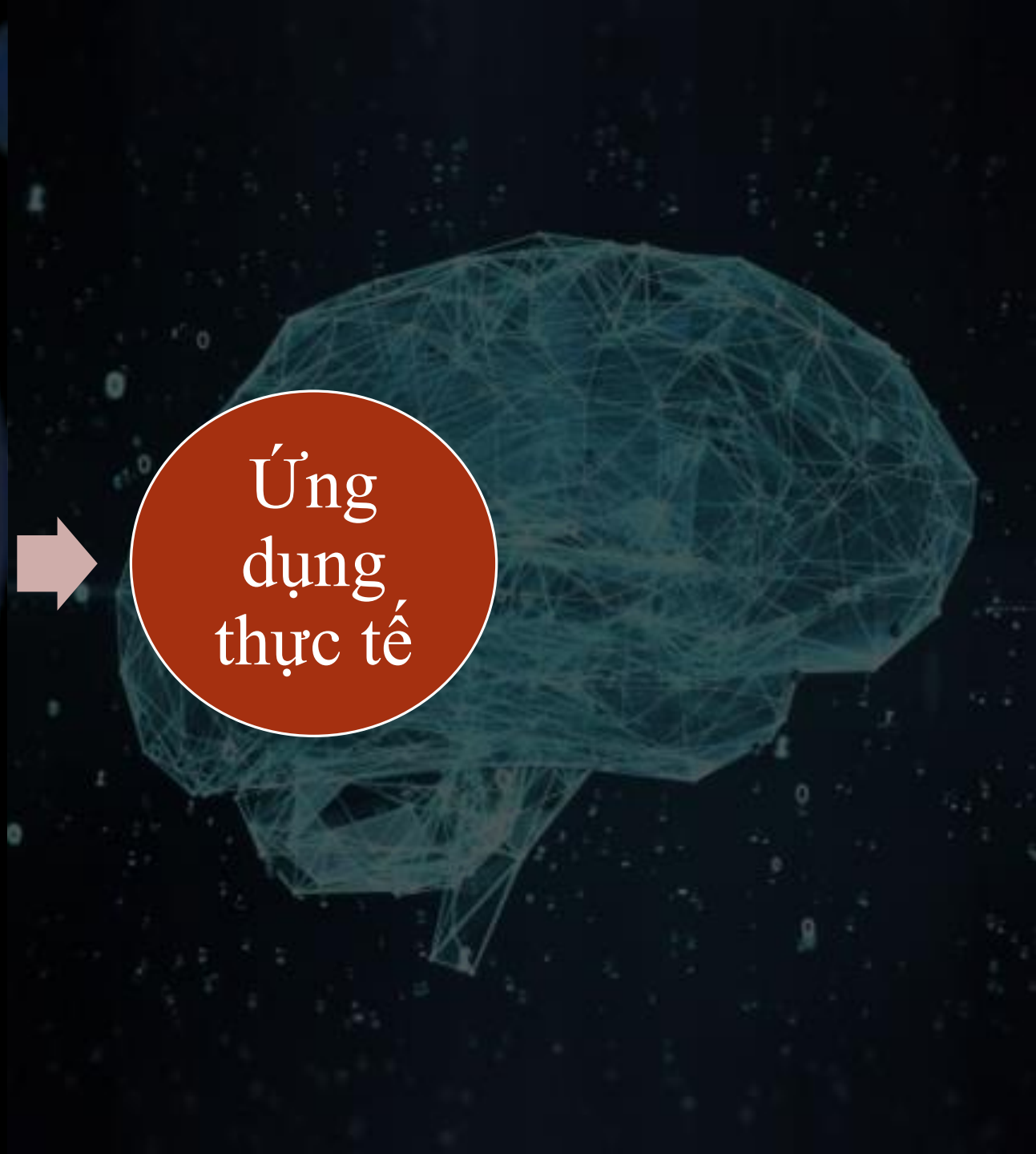
Tuyệt đối không sờ tay vào khẩu trang khi đã đeo vì có thể vi rút sẽ lây nhiễm sang tay và lan truyền sang vật khác

Khi tháo khẩu trang chỉ được cầm vào phần dây, tháo khỏi tai và cho ngay vào thùng rác có nắp đậy.

Đeo khẩu trang ?

+

Nhận diện khuôn mặt người



Ứng dụng thực tế

Mô tả bài toán

Ảnh chân dung

Model

0

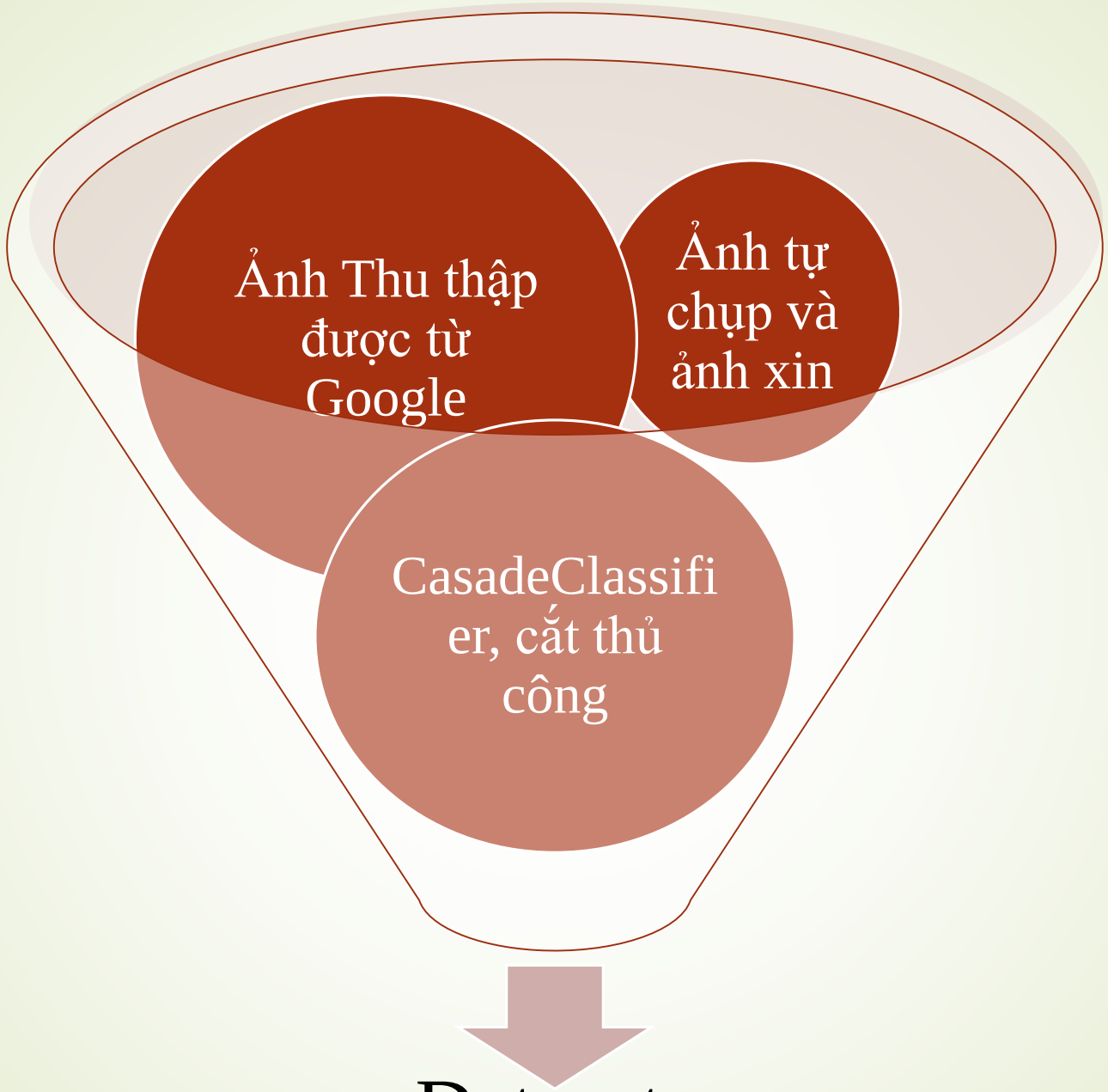
1

INPUT

OUTPUT

2

MÔ TẢ
BỘ DỮ LIỆU

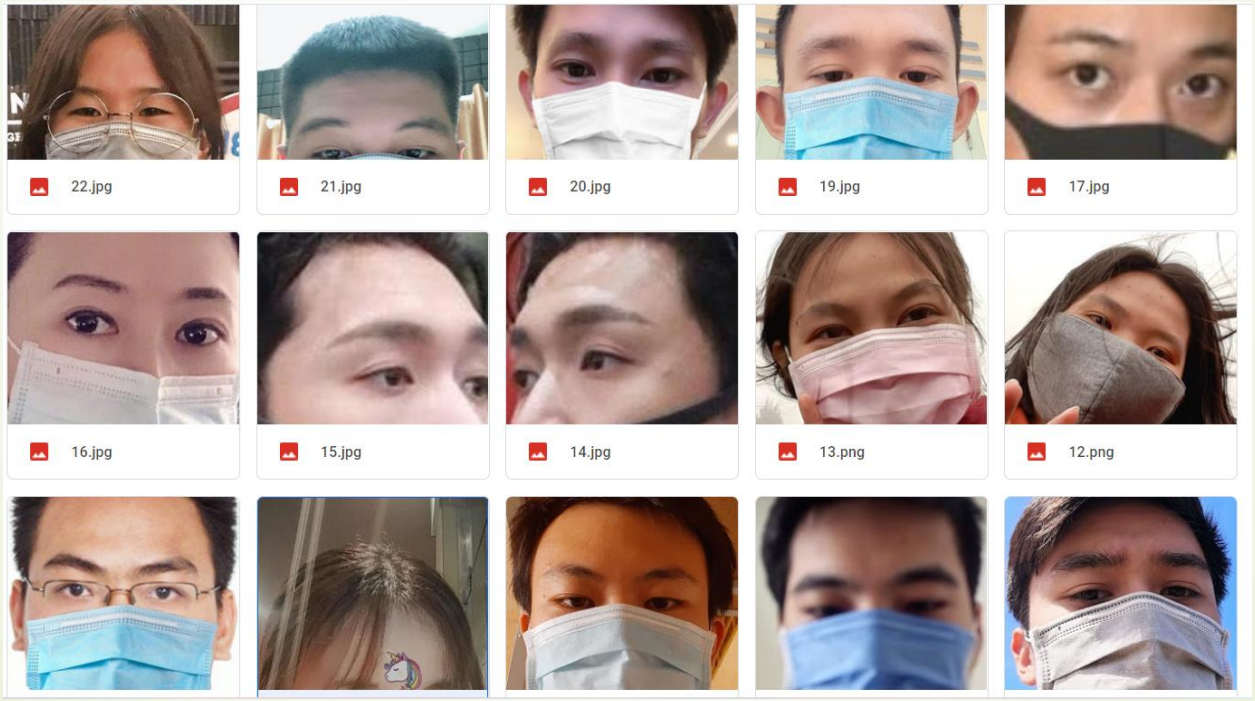

Ảnh Thu thập được từ Google
Ảnh tự chụp và ảnh xin
CasadeClassifier, cắt thủ công

Datasets

Bộ datasets

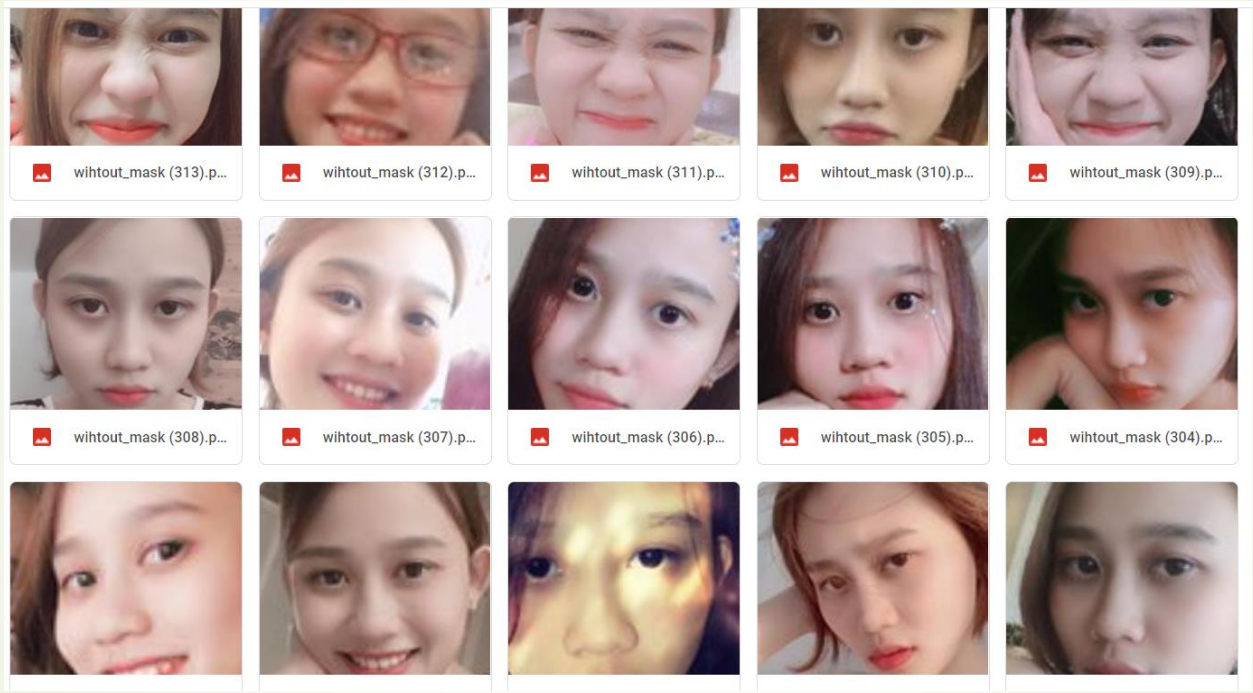
Datasets gồm có 1034 ảnh trong đó có :

Ảnh có khẩu trang: 509 ảnh



Bộ datasets

Ảnh không có khẩu trang: 525 ảnh



Bộ data khá cân bằng và khá đa dạng với các gốc mặt khác nhau, các loại khẩu trang khác nhau. Tuy nhiên, số lượng ảnh trong Dataset vẫn còn khá ít

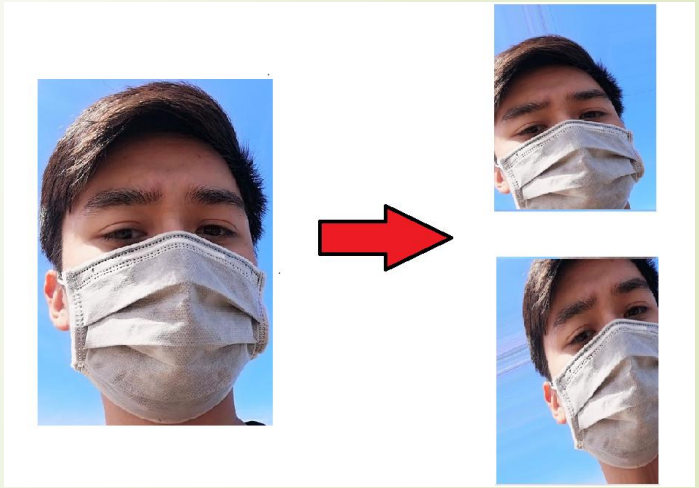
Tiền xử lý dữ liệu

Vì ảnh vẫn còn khá ít nên nhóm sử dụng hàm ImageDataGenerator của thư viện Keras để tăng dữ liệu cho bộ dataset
=> Cụ thể từ 1 ảnh sẽ tăng thành 2 ảnh khác nhau

Fine-turning hàm ImageDataGenerator

- rotation_range : góc quay (0 - 20)
- zoom_range : độ phóng (0 - 0.15)
- width_shift_range, height_shift_range : độ dịch ảnh (0 - 0.2)
- horizontal_flip : lật ảnh theo chiều ngang

In [203]:
```
# Sử dụng kỹ thuật Data Augmentation để tăng kích thước cho bộ train data
aug = ImageDataGenerator(
      rotation_range=20,
      zoom_range=0.15,
      width_shift_range=0.2,
      height_shift_range=0.2,
      horizontal_flip=True )
```



Tiền xử lý dữ liệu

+ Ảnh sau khi được load lên sẽ được resize về kích thước 32x32
+ Sau đó ảnh từ dạng mảng 3 chiều sẽ được làm phẳng thành 1 Vector

```
# load hình ảnh
image = cv2.imread(imagePath)


# preprocess ảnh
image = cv2.resize(image,(32,32)).flatten()
```

Tiền xử lý dữ liệu

+ Cuối cùng các Vector ảnh là label của nó được chuyển sang dạng numpy

```
# chuyển label và data sang dạng mảng
labels = np.array(labels)
data = np.array(data)
```

Tiền xử lý dữ liệu

Tiến hành chia dữ liệu thành 2 phần : 80% dùng để train, Validation và 20% dùng để test

```
# chia dữ liệu thành 80% để train, validation bằng K-Fold và 20% để test
(trainX, testX, trainY, testY) = train_test_split(data, labels, test_size=0.2,stratify=labels,random_state=1)
```

3

MÔ TẢ ĐẶC TRƯNG

Mô tả đặc trưng

- Biến đổi từ 1 bức ảnh có mảng 3 chiều thành 1 vector
- Từ một ảnh có kí thước 32x32 thì sẽ chuyển thành 1 vector có 32x32x3 phần tử vì một ảnh có 3 kênh màu Red , Green , Blue.

```
# preprocess ảnh
image = cv2.resize(image,(32,32)).flatten()
```

- Hàm flatten sẽ lấy hàng ngang của từng cột sau đó ghép nó thành 1 hàng ngang duy nhất

Mô tả đặc trưng

pixel image

imread

Blue

Green

Red

255	134	93	22
255	134	202	22
255	231	42	22
123	94	83	2
34	44	187	92
34	76	232	124
67	83	194	202

reshaped image vector

[255, 231, 42, 22, 123, 94, ... , 94, 142]

4

MÔ TẢ THUẬT TOÁN

Thuật toán

Do label có dạng binary : 0 và 1 vì thế chọn các thuật toán Binary Classification

Các model được sử dụng :

+ Decision Tree
+ Random Forest
+ Logictic Regression

Sử dụng phương pháp Cross-Validation để tiến hành đánh giá các model được chọn. Cụ thể là phương pháp K-Fold Training và Validation cho các model được chọn sau đó chọn ra model có độ chính xác cao nhất
Tiến hành training và đánh giá model được chọn trên tập test

Cài đặt thuật toán

```
# Sử dụng hàm StratifiedKFold của sklearn để train và test model
results = []
names = []

# Vòng lặp để tiến hành train và validation cho mỗi model
for name,model in models:

  # Fine-turning cho hàm StratifiedKFold
  kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)

  # Đánh giá model bằng phương pháp cross-validation
  cv_results = cross_val_score(model, trainX, trainY, cv=kfold, scoring='accuracy')

  # đẩy kết quả vào results
  results.append(cv_results)
  names.append(name)

  # Xuất ra giá trị trung bình của results
  print('%s: %f ' % (name, cv_results.mean()))
```

4

DEMO

5

ĐÁNH GIÁ KẾT QUẢ

Đánh giá kết quả

Sử dụng K-Fold để đánh giá 3 model được chọn

```
# Xuất ra giá trị trung bình của results
print('%s: %f' % (name, cv_results.mean()))
```

```
LR: 0.850628
DT: 0.813782
RF: 0.906261
```

=> Model Random Forest có độ chính xác cao nhất trong 3 model

Đánh giá kết quả

Tiến hành đánh giá model Random Forest trên tập test

```
# Đánh giá model
print(confusion_matrix(testY, predictions))
print(classification_report(testY, predictions))
```

```
[[200  10]
 [ 30 174]]
              precision    recall  f1-score   support

           0       0.87      0.95      0.91       210
           1       0.95      0.85      0.90       204

    accuracy                           0.90       414
   macro avg       0.91      0.90      0.90       414
weighted avg       0.91      0.90      0.90       414
```

Độ chính xác của Random Forest khá cao : 90%

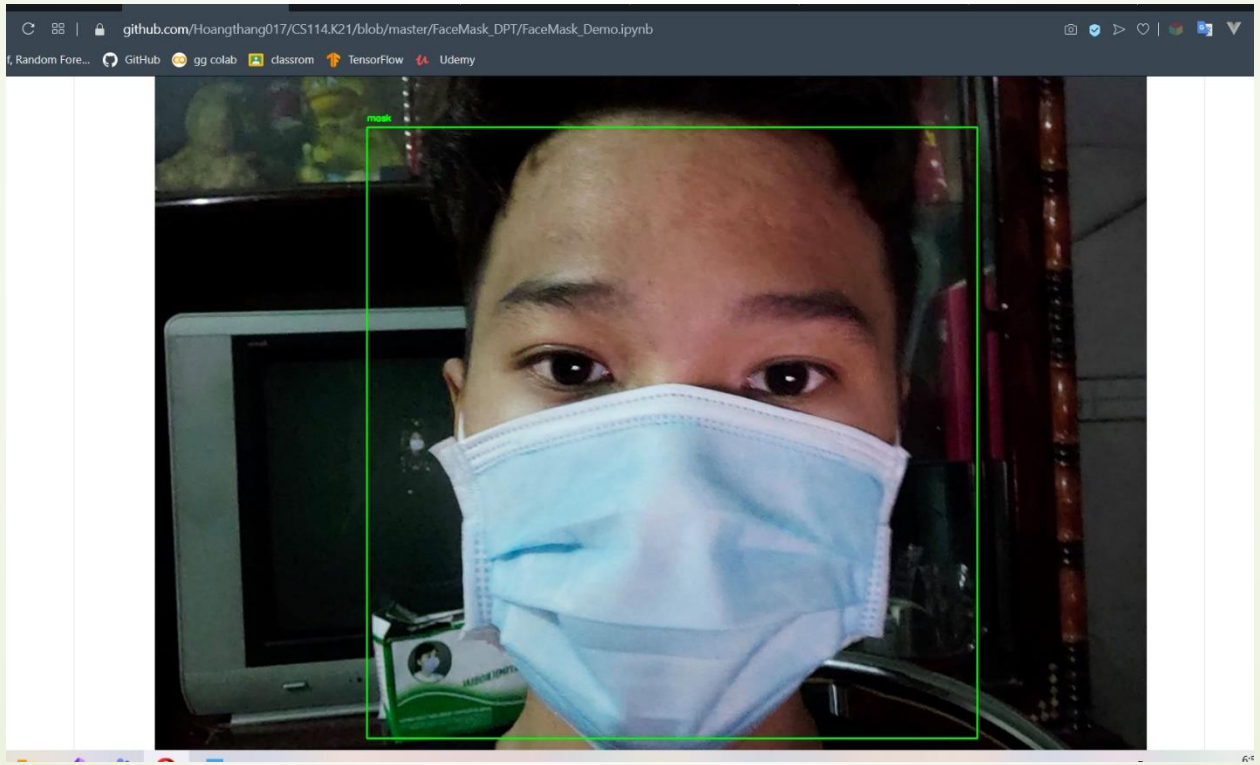
Độ chính xác của tập train và tập test gần như bằng nha nên model có thể sử dụng được với lỗi không quá lớn

Như có thế thấy ở Confusion_matrix :

+ Model đều dự đoán khá chính xác ở trên cả 2 label
+ Giá trị của False Positive cao hơn False Negative , tuy nhiên ở bài toán này ta có thể chấp nhận được lỗi này vì thiệt hại của nó không quá lớn.

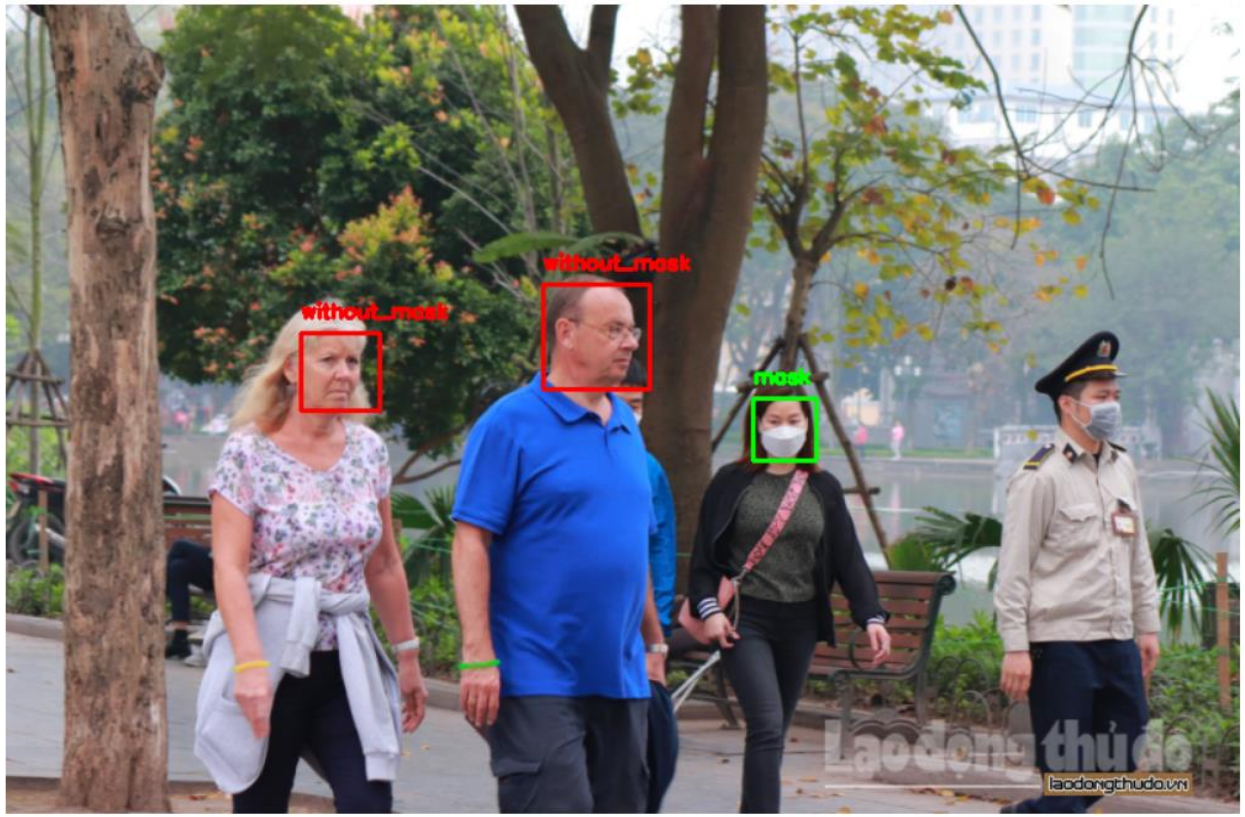
Đánh giá kết quả

Đối với các ảnh thực tế model nhận diện cũng khá tốt



Đánh giá kết quả

Nhận diện trên ảnh có nhiều người



Đánh giá kết quả

Tiến hành dự đoán trên ảnh mà Tool không nhận diện được bằng cách cắt riêng mặt đó ra

```
# Load hình
face = cv2.imread("/content/cut _ img tool no detected.png")

cv2_imshow(cv2.resize(face,(224,224)))

face = cv2.resize(face,(32,32)).flatten()
face = np.array(face).reshape(1,-1)

result = Model_LR.predict(face)

if (result == 0 ):
  print("---------------------------------Wihtout_Mask---------------------------------")
else:
  print("---------------------------------Mask---------------------------------")
```

---------------------------------Mask---------------------------------

Đánh giá kết quả

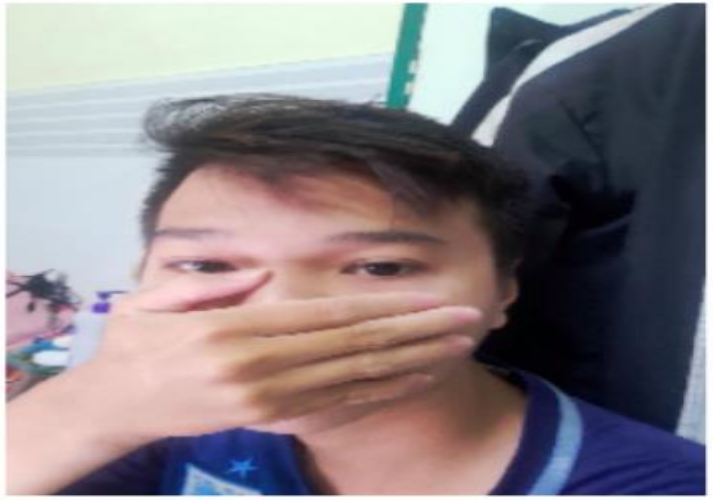
Kết luận :

+ Model nhận diện khá chính xác trên ảnh thực tế

+ Tuy nhiên , để đánh giá trên ảnh nhiều người phải phụ thuộc vào tool nhận diện khuôn mặt

+ Model vẫn còn một số hạn chế như việc nhầm lẫn giữ các vật thể tương tự khẩu trang che vào mặt

```
if (result == 0 ):
  print("----------Wihtout_Mask----------")
else:
  print("----------Mask----------")
```



----------Mask----------

6

THAM KHẢO

https://towardsdatascience.com/classifying-cat-pics-with-a-logistic-regression-model-e35dfb9159bb
https://numpy.org/doc/stable/reference/generated/numpy.ndarray.flatten.html
https://medium.com/@alkeshab/face-detection-using-opencv-in-google-colaboratory-a7529a2bb921
https://scikit-learn.org/stable/modules/cross_validation.html