TRƯỜNG ĐẠI HỌC SƯ PHẠM KĨ THUẬT TP.HCM

KHOA ĐÀO TẠO CHẤT LƯỢNG CAO

TRÍ TUỆ NHÂN TẠO

Đề tài:

NHẬN DIỆN DANH TÍNH MẶT NGƯỜI KHI ĐEO VÀ KHÔNG ĐEO KHẨU TRANG

GVHD: PSG.TS Nguyễn Trường Thịnh

SVTH: Trần Quang Huy

MSSV: 19146195

MÃ HỌC PHẦN: ARIN337629

Thành phố Hồ Chí Minh, Tháng 6 năm 2022

NHẬN XÉT CỦA GIẢNG VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

Ngày 23 tháng 6 năm 2022

Giảng viên chấm điểm

LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành đến thầy Nguyễn Trường Thịnh đã tạo điều kiện thuận lợi cho chúng em học tập và hoàn thành bài báo cáo cuối kì này. Đặc biệt, em bày tỏ lòng biết ơn vô cùng sâu sắc đến thầy đã rất tận tâm truyền đạt những kiến thức bổ ích và hướng dẫn chúng em thực hiện bài báo cáo đồ án này.

Em đã cố gắng vận dụng những kiến thức đã học được trong trong những môn lý thuyết và kết hợp lời góp ý từ thầy để hoàn thành bài báo cáo cuối kì này. Nhưng do kiến thức còn có giới hạn và chưa có nhiều kinh nghiệm thực tế nên khó tránh khỏi một vài sai sót trong quá trình thực hiện và trình bày. Em kính mong có được sự góp ý của thầy để tiểu luận được hoàn thiện hơn.

Một lần nữa, em xin trân trọng cảm ơn sự quan tâm giúp đỡ của thầy Nguyễn Trường Thịnh và anh trợ giảng Nguyễn Minh Triều trong quá trình thực hiện bài báo cáo này.

Xin trân trọng cảm ơn.

MỤC LỤC

MỤC LỤC HÌNH ẢNH

CHƯƠNG 1: GIỚI THIỆU

1.1. Tính cấp thiết của đề tài

Hiện tại, dịch bệnh COVID-19 vẫn có diễn biến phức tạp ở Việt Nam nói riêng và trên toàn thế giới nói chung, đại dịch này qua từng ngày lại cướp đi hàng triệu sinh mạng trên toàn thế giới do tốc độ lây lan của virus rất nhanh và chúng có khả năng biến đổi thành nhiều loại biến thể giúp thích nghi với nhiều môi trường sống khác nhau, nó không chỉ ảnh hưởng đến kinh tế, thương mại, dịch vụ mà còn ảnh hưởng không nhỏ đến tâm lý xã hội của người dân và chưa có vacxin điều trị cho nên cần kiểm soát tốt nguồn lây nhiễm. Chính vì thế có những nghiên cứu chỉ ra rằng giữ khoảng cách xã hội, rửa tay và đeo khẩu trang là các biện pháp có hiệu quả trong việc giảm tỷ lệ mắc COVID-19, trong đó đeo khẩu trang mang lại hiệu quả cao nhất. Nhóm nghiên cứu, bao gồm các chuyên gia y tế công cộng và bệnh truyền nhiễm ở Úc, Trung Quốc và Anh, đã đánh giá 72 nghiên cứu về các biện pháp phòng ngừa COVID-19 trong đại dịch. Sau đó, họ xem xét 8 nghiên cứu tập trung vào rửa tay, đeo khẩu trang và khoảng cách xã hội. Kết quả nghiên cứu cho thấy, đeo khẩu trang giúp tỷ lệ mắc COVID-19 giảm 53%. Trong phân tích rộng hơn đối với các nghiên cứu bổ sung, đeo khẩu trang làm giảm sự lây truyền virus COVID-19, các trường hợp mắc và tử vong. Trong một nghiên cứu trên 200 quốc gia, việc đeo khẩu trang bắt buộc giúp giảm gần 46% tình trạng sức khỏe bất lợi do COVID-19. Một nghiên cứu khác ở Mỹ cho thấy, sự lây truyền virus SARS-CoV-2 đã giảm 29% ở những bang bắt buộc người dân phải đeo khẩu trang [4]. Theo trung tâm kiểm soát bệnh tật CDC (Centers for Disease Control and Prevention), khẩu trang được khuyến nghị là một rào chắn đơn giản để giúp ngăn chặn các giọt bắn từ đường hô hấp bay vào không khí và lên người khác. Khuyến nghị này dựa trên những nghiên cứu về vai trò của những giọt nước bọt bắn lên các bề mặt kim loại, con người từ đó đi vào đường hô hấp, kết hợp với bằng chứng mới xuất hiện từ các nghiên cứu lâm sàng và trong

phòng thí nghiệm cho thấy khẩu trang làm giảm việc phun các giọt bắn khi đeo khẩu trang qua miệng và mũi. Vì vậy việc sử dụng khẩu trang đặc biệt quan trọng ở thời điểm hiện tại nhằm hạn chế việc lây lan bệnh.

1.2. Tính thiết thực và ý nghĩa thực tiễn của đề tài

Với thời đại 4.0 thì trí tuệ nhân tạo đang dần được áp dụng vào hầu hết các vấn đề. Và hiện tại ở Việt Nam, các thiết bị nhận dạng người đeo khẩu trang chưa được phổ biến. Nên việc nghiên cứu về việc tạo ra một mô hình trí tuệ nhân tạo phát hiện người đeo khẩu trang là cần thiết để có thể bắt kịp xu hướng quốc tế trong ứng dụng công nghệ 4.0. Không những thế ứng dụng này còn có thể áp dụng vào trong y tế, các bệnh viện cụ thể hơn là các phòng khoa có thể đưa ra các cảnh báo nếu bệnh nhân không đeo khẩu trang , hay có thể ứng dụng vào các phương tiện công cộng như xe bus để cảnh báo các hành khách khi tham gia không đeo khẩu trang nhằm nâng cao ý thức cũng như đẩy lùi dịch bệnh ở nước ta.

1.3. Mục tiêu nghiên cứu đề tài

Đây là báo cáo cuối kì môn Trí tuệ nhân tạo nên mục tiêu chính của đề tài này là nhận diện danh tính người khi đeo và không đeo khẩu trang, và số người nhận diện cụ thể là 5 người. Bằng cách xây dựng mô hình CNN – Mạng nơ-ron tích chập để có thể nhận dạng người khi đeo và không đeo khẩu trang thông qua các đặc tính của mặt người khi đeo và không đeo khẩu trang, và mô hình sẽ được áp dụng và chạy thời gian thực.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Trí tuệ nhân tạo.

Hiện nay, ngành công nghiệp 4.0 tập trung vào công nghệ kỹ thuật số với một cấp độ hoàn toàn mới, nổi trổi trong số đó là Trí tuệ nhân tạo (Artificial Intelligence hay gọi tắt là AI) thuộc lĩnh vực khoa học máy tính. Công nghệ AI là công nghệ mô phỏng cac quá trình suy nghĩ và học tập của con người cho máy móc, đặc biệt là các hệ thống máy tính . Các quá trình này bao gồm việc học tập, lập luận và tự sửa lỗi. Đây là một thành quả vĩ đại của khoa học hiện đại, các ứng dụng của nó mang lại rất nhiều lợi ích cho người sử dụng, nó có thể phát hiện và hạn chế các rủi ro, tiết kiệm sức lao động của con người, giải phóng sức sáng tạo, cầu nối ngôn ngữ, cá nhân hóa. Có thể thấy rằng ta dễ dàng nhận ra được sự hiện diện của AI trong các sản phẩm công nghệ như các trợ lý ảo trên các thiết bị số của các hãng điện thoại như Apple là Siri, Samsung là Bixby và Alexa của Amazon. Hay sự phát triển của ngành xe điện như xe điện tự hành của Tesla,…

Trí tuệ nhân tạo có rất nhiều lĩnh vực, một trong số đó là Học máy (Machine Learning – hay gọi tắt là ML) là một lĩnh vực con. Nó sử dụng những thuật toán cho phép máy tính có thể học từ dữ liệu để thực hiện các công việc thây vì được lập trình một cách rõ ràng. Machine Learning thường được chia thành học có giám sát, trong đó máy tính học bằng ví dụ từ dữ liệu được gắn nhãn và học không giám sát, trong đó các máy tính nhóm các dữ liệu tương tự và xác định chính xác sự bất thường.

Học sâu (Deep Learning – DL) là một tập hợp con của Machine Learning, cho phép máy tính giải quyết một loạt các vấn đề phức tạp không thể giải quyết được. Một trong những mô hình Deep Learning nổi tiếng là mạng nơ ron tích chập (Convolutional Neural Network hay gọi tắt là CNN). CNN cho phép ta xây dựng các

hệ thống thông minh với độ chính xác vô cùng cao. Hiện nay, CNN được ứng dụng rất nhiều trong những bài toán nhận dạng vật thể trong ảnh.

2.2. Mạng nơron tích chập (Convolutional Neural Network)

Convolutional Neural Network là một kiến trúc Deep Neural Network, nó cũng chính là một dạng của Artificial Neural Network. mô hình mạng neural tích chập (CNN) là 1 trong những mô hình để nhận dạng và phân loại hình ảnh. Trong đó, xác định đối tượng và nhận dạng khuôn mặt là 1 trong số những lĩnh vực mà CNN được sử dụng rộng rãi. [1]

2.2.1. Các lớp cơ bản của mạng nơ-ron tích chập.

Convolutional layer

Đây là lớp quan trọng nhất của CNN. Thông thường các input trong mạng neural network sẽ cho qua các hidden layer rồi ra được output. Với CNN thì convolutional layer cũng là hidden layer. Convolution layer là một tập các feature map và mỗi feature map này là một bản scan của input ban đầu. Những yếu tố của một Convolution layer là: feature map, filter map, stride, padding [6]:

- Feature map sẽ thể hiện kết quả sau mỗi lần filter map quét qua input, sau mỗi lần quét thì sẽ xảy ra quá trình tính toán. Kích thước filter của tầng convolution hầu hết đều là số lẻ, ví dụ như 3x3 hay 5x5. Với kích thước filter lẻ, các giá trị của feature map sẽ xác định một tâm điểm ở tầng phía trước. Nếu chọn filter có kích thước 2x2, 4x4 thì sẽ gặp khó khăn khi muốn tìm vị trí tương ứng của các giá trị feature map trên không gian ảnh.
- CNN sử dụng các filter để áp dụng vào vùng của hình ảnh. Các filter map được gọi là ma trận 3 chiều (kernel), bên trong đó là các con số được gọi là parameter.

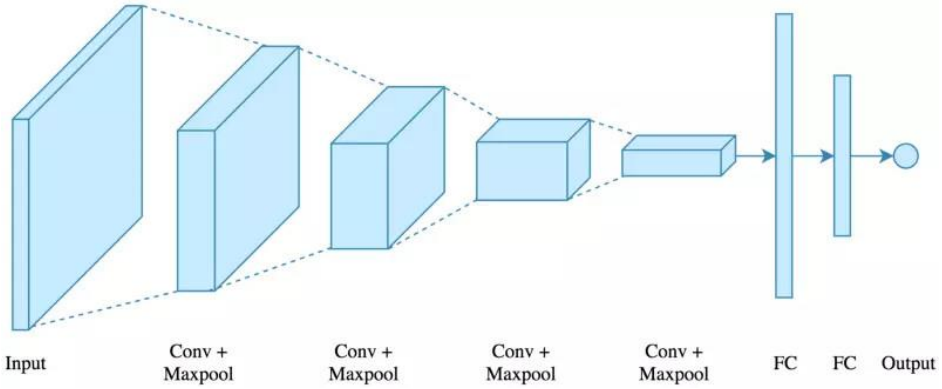
- Khi Stride là dịch chuyển filter map theo pixel dựa vào giá trị từ trái sang phải. Với stride =1, kernel sẽ quét 2 ô ngay cạnh nhau, nhưng với stride =2, kernel sẽ quét ô số 1 và ô số 3. Bỏ qua ô ở giữa. Điều này nhằm tránh việc lặp đi lặp lại các giá trị ở các ô bị quét [7].
- Padding được hiểu đơn giản là các giá trị 0 được thêm vào với lớp input

Pooling layer

Mục đích của lớp Pooling rất đơn giản, nếu số hyperparameter khá lớn nó sẽ giúp giảm chúng giúp cho số lượng tính toán, thời gian tính toán được giảm xuống đồng thời tránh overfitting. Pooling layer có 2 loại chủ yếu là max pooling và average. Thường gặp nhất là max pooling, lấy giá trị lớn nhất trong một pooling window

Fully Connected layer

Lớp cuối cùng của mô hình CNN trong bài toán phân loại ảnh là lớp fully connected layer. Lớp này có chức năng chuyển ma trận đặc trưng ở lớp trước thành vector chứa xác suất của các đối tượng cần được dự đoán.



Hình 2.1. *Cấu trúc của mạng CNN*

2.3. Các thư viện sử dụng

2.3.1. Thư viện numpy

Numpy (Numeric Python): là một thư viện toán học phổ biến và mạnh mẽ của Python. Cho phép làm việc hiệu quả với ma trận và mảng, đặc biệt là dữ liệu ma trận và mảng lớn với tốc độ xử lý nhanh hơn nhiều lần khi chỉ sử dụng "core Python" đơn thuần.

2.3.2. Thư viện Matplotlib

Matplotlib là một thư viện sử dụng để vẽ các đồ thị trong python, để thực hiện các suy luận thống kê cần thiết, cần phải trực quan hóa dữ liệu và Matplotlib là một trong những giải pháp như vậy cho người dùng Python. Module được sử dụng nhiều nhất của Matplotib là Pyplot cung cấp giao diện như Matlab nhưng thay vào đó, nó sử dụng python và nó là nguồn mở.

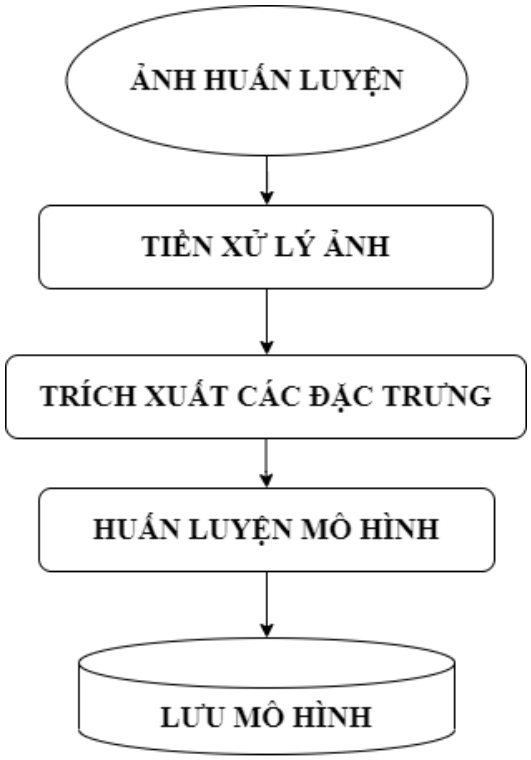
2.3.3. Thư viện Keras

Keras là một mã nguồn mở cho Neural Network được viết bởi ngôn ngữ Python. Nó là một library được phát triển vào năm 205 bởi Francois Chollet, là một kỹ sư nghiên cứu Deep Learning. Keras có thể sử dụng chung với các thư viện nổi tiếng như Tensorflow, CNTK, Theano. Một số ưu điểm của Keras như: dễ sử dụng, xây dựng model nhanh. Chạy được trên cả CPU và GPU. Hỗ trợ xây dựng CNN , RNN hoặc cả hai.

CHƯƠNG 3: NỘI DUNG NGHIÊN CỨU

3.1. Bài toán phát hiện khẩu trang

Để phát hiện đối tượng trong ảnh hay video thì bài toán cần phải xử lý được người trong ảnh hoặc trong video có đeo khẩu trang hay không. Bài toán phân loại này gồm hai bước là xây dựng mô hình và vận hành mô hình. Xây dựng mô hình sẽ dựa trên một tập dữ liệu với dữ liệu là khuôn mặt của 5 người đeo và không đeo khẩu trang được gán nhãn sẵn và đầu ra của mô hình sẽ là 10 với 5 khuôn mặt không đeo khẩu trang và 5 khuôn mặt đó khi đeo khẩu trang. Sau khi xây dựng mô hình phân lớp từ dữ liệu mẫu, sau đó mô hình này sẽ dự đoán những khuôn mặt của 5 người khi chưa biết nhãn.



Hình 3.1. *Lưu đồ thuật toán cho bài toán nhận diện khẩu trang*

3.2. Khởi tạo các thư viện

Các thư viện cần thiết cho huấn luyện mô hình:

```
import keras
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing.image import Image
DataGenerator
from keras.layers import Dense, Dropout,BatchNormaliza
tion, LSTM
from tensorflow.keras.utils import to_categorical
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras.optimizers import RMSprop
from keras.layers import Conv2D, MaxPooling2D
from tensorflow.keras.utils import load_img, img_to_ar
ray
```

3.3. Tiền xử lý dữ liệu

Sau khi thu thập bộ dữ liệu của 5 khuôn mặt khi đeo và không đeo khẩu, sau đó tiến hành tải bộ dữ liệu lên Kaggle để chia sẻ dữ liệu. Bộ dữ liệu gồm 10 thư mục, 5 thư mục được gán nhãn của 5 người khi không đeo khẩu trang, 5 thư mục được gán nhãn của khuôn mặt 5 người khi đeo khẩu trang [5].

Tập dữ liệu đầu tiên là khuôn mặt người đi đeo khẩu trang. Tiến hành chia tập dữ liệu thành 2 phần. Phần thứ nhất là train_dataset, dùng được huấn luyện cho mô hình. Phần thứ hai là validation_dataset, được dùng để kiểm tra độ chính xác của mô hình. Dưới đây là đoạn code chia tập dữ liệu.

```
# WITH MASK
image_generator = ImageDataGenerator(rescale=1./255, validation_split=0.
2,shear_range=0.2, zoom_range = 0.2,)

train_dataset = image_generator.flow_from_directory(batch_size=batch,
                                                    directory='../input/cl3
dataset-
final/DATASET_FINAL_19146195/DATASET_FINAL_19146195/DATASET_10P_FINAL',
                                                    shuffle=True,
                                                    target_size= input_size
,
                                                    subset="training",
                                                    class_mode='categorical
')

validation_dataset = image_generator.flow_from_directory(batch_size=batc
h,
                                                    directory='../input/cl3
dataset-
final/DATASET_FINAL_19146195/DATASET_FINAL_19146195/DATASET_10P_FINAL',
                                                    shuffle=True,
                                                    target_size=input_size,

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

Ở đây, nhằm tăng cường dữ liệu cho bộ dữ liệu sẽ sử dụng hàm ImageDataGenerator để biến đổi đào tạo, giúp cho mô hình đã học có thể mạnh mẽ và chính xác hơn. Các thuộc tính zoom_range và shear_range lần lượt thực hiện zoom ngẫu nhiên, làm méo ảnh. Sử dụng phương thức phân tách dữ liệu validation_split, và chọn giá trị giá trị là 0.2 tương ứng với sẽ lấy 80% của tập dữ liệu để huấn luyện và 20% còn lại sẽ được dùng để kiểm tra. Và chuyển đổi tỷ lệ ảnh từ số nguyên 0-255 và dạng float 0-1. Kích thước của những tấm ảnh đầu vào là 150 x 150. Và đây là kết quả thu được.

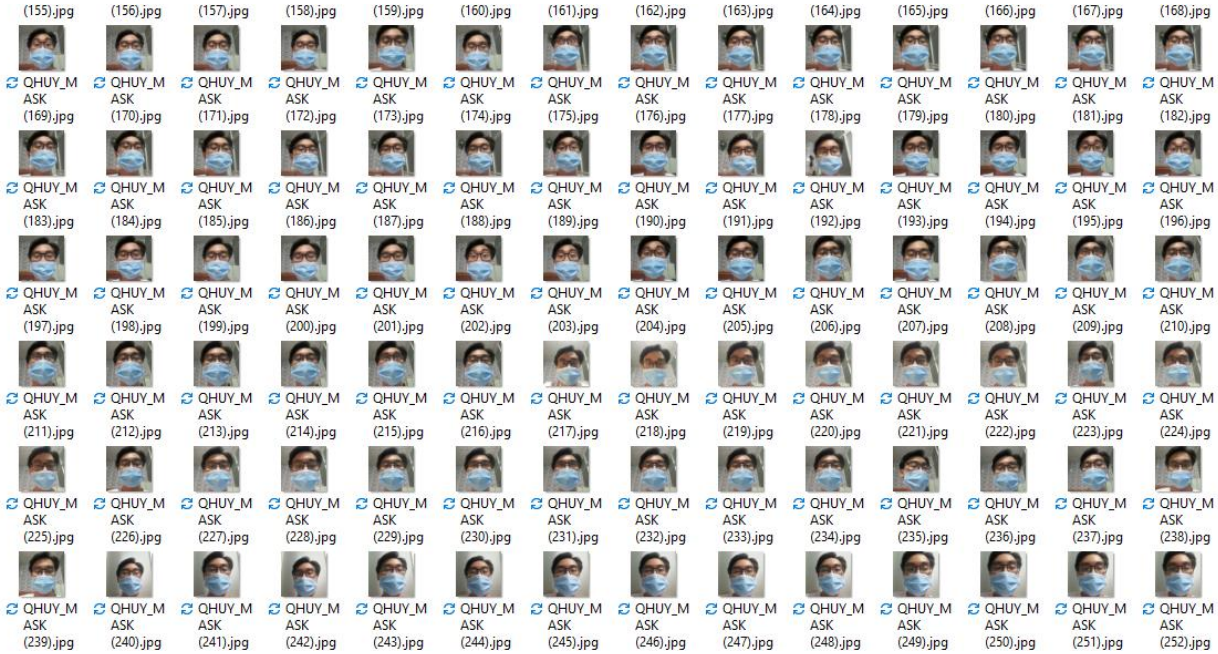
```
Found 3688 images belonging to 10 classes.
Found 918 images belonging to 10 classes.
```

Hình 3.2. *Tập dữ liệu được chia thành 10 classes*

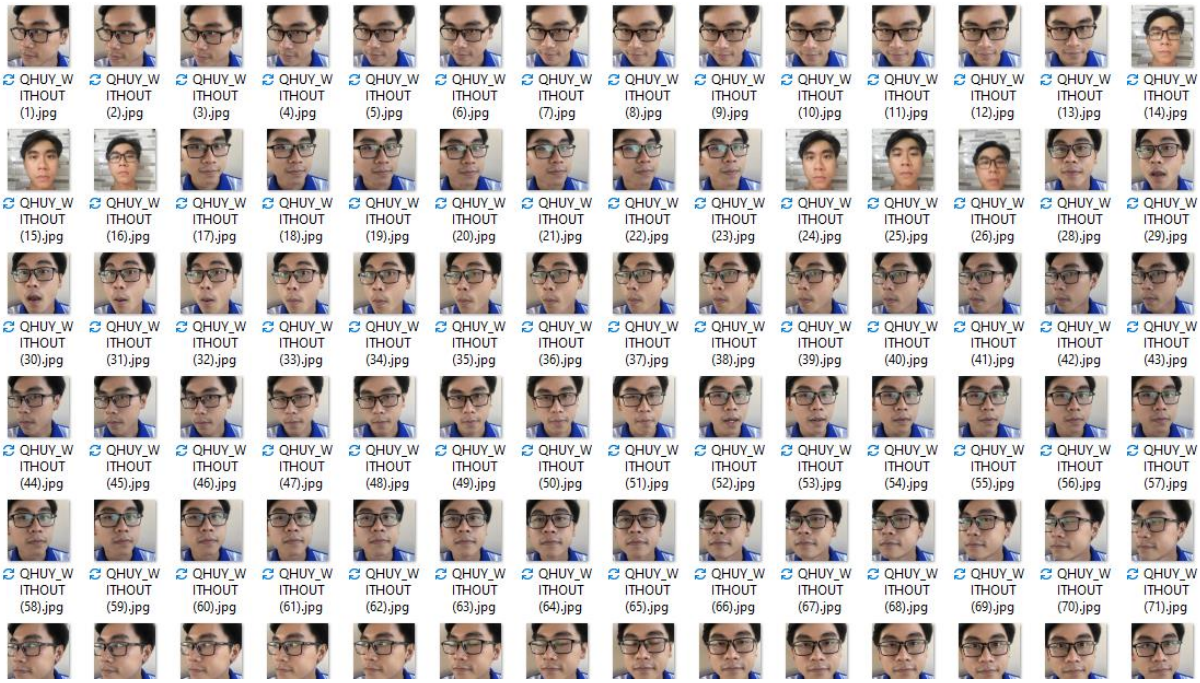
```
Out[4]:
        {'BHUY_MASK': 0,
         'BHUY_WITHOUTMASK': 1,
         'DHUNG_MASK': 2,
         'DHUNG_WITHOUTMASK': 3,
         'DTAI_MASK': 4,
         'DTAI_WITHOUTMASK': 5,
         'MTUAN_MASK': 6,
         'MTUAN_WITHOUTMASK': 7,
         'QHUY_MASK': 8,
         'QHUY_WITHOUTMASK': 9}
```

Hình 3.3. *Các class được gán nhãn*

Tương tự như tập dữ liệu khuôn mặt người khi đeo khẩu trang. Tập dữ liệu khuôn mặt người không đeo khẩu trang sẽ được thực hiện tương tự.



Hình 3.4. *Tập dữ liệu QHUY đeo khẩu trang*



Hình 3.5. *Tập dữ liệu QHUY không đeo khẩu trang*

3.4. Huấn luyện mô hình

3.4.1. Xây dựng mô hình

Định nghĩa mô hình, và xây dựng mô hình.

```
# Định nghĩa model
model = Sequential()
# Thêm Convolutional layer với 128 kernel, kích thước kernel 3*3
# Dùng hàm relu làm activation và chỉ rõ input_shape cho layer đầu tiên
mask_model.add(Conv2D
model.add(Conv2D(128,(3,3), kernel_initializer='he_uniform',padding='same', activation='relu', input_shape=(150,150,3)))
# Thêm Max pooling layer
model.add(MaxPooling2D((2,2)))
```

```
model.add(Conv2D(128,(3,3),kernel_initializer='he_uniform',padding='same', activation='relu'))
model.add(MaxPooling2D((2,2)))

model.add(Conv2D(256,(3,3),kernel_initializer='he_uniform',padding='same', activation='relu'))
model.add(MaxPooling2D((2,2)))

model.add(Conv2D(512, (3,3),kernel_initializer='he_uniform',padding='same', activation='relu'))
model.add(MaxPooling2D((2,2)))
# Flatten layer chuyển từ tensor sang vector
model.add(Flatten())
model.add(Dropout(0.5))
# Thêm Fully Connected layer với 128 nodes và dùng hàm relu
model.add(Dense(128,activation='relu'))
# Thêm Fully Connected layer với 512 nodes và dùng hàm relu
model.add(Dense(512, activation='relu'))
# Output layer với 10 node
model.add(Dense(10,activation='softmax', name ='wearmask'))
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_4 (Conv2D)            (None, 150, 150, 128)     3584
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 75, 75, 128)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 75, 75, 128)       147584
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 37, 37, 128)       0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 37, 37, 256)       295168
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 18, 18, 256)       0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 18, 18, 512)       1180160
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 9, 9, 512)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 41472)             0
_________________________________________________________________
dropout_1 (Dropout)          (None, 41472)             0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               5308544
_________________________________________________________________
dense_3 (Dense)              (None, 512)               66048
_________________________________________________________________
wearmask (Dense)             (None, 10)                5130
=================================================================
Total params: 7,006,218
Trainable params: 7,006,218
Non-trainable params: 0
```

Hình 3.6. *Tổng tham số*

Tổng tham số trong mô hình đầu tiên là 7.006.218, các tham số trong mỗi lớp của mô hình mạng như sau:

- **Ảnh đầu vào:** Đầu vào ảnh với kích thước 150 x 150 x 3 = 67500 (3 tương ứng với 3 màu đỏ, xanh lục, xanh lam trong hệ màu RGB thông thường)

❖ **Lớp đầu tiên** (Tích chập):

- Số bộ lọc: 128
- Kích thước bộ lọc: 3 x 3 x 128
- Padding = same

- Hàm kích hoạt : relu

❖ **Lớp chuyển tiếp sang lớp thứ hai**

- Kích thước đầu ra của dữ liệu giảm 1/2 , từ (150 x 150 x 3) xuống (75 x 75 x 3), và chiều sâu được giữ nguyên

❖ **Lớp thứ hai (**tích chập)

- Đầu vào: 75 x 75 x 3
- Kích thước bộ lọc: 3 x 3 x 128
- Padding = same
- Hàm kích hoạt : relu

❖ **Lớp chuyển tiếp sang lớp thứ ba**

- Kích thước đầu ra của dữ liệu giảm 1/2 , từ (75 x 75 x 3) xuống (37.5 x 37.5 x 3), và chiều sâu được giữ nguyên

❖ **Lớp thứ ba**

- Đầu vào: 37.5 x 37.5 x 3
- Kích thước bộ lọc: 3 x 3 x 256
- Padding = same
- Hàm kích hoạt : relu

❖ **Lớp chuyển tiếp sang lớp thứ tư**

- Kích thước đầu ra của dữ liệu giảm 1/2 , từ (37.5 x 37.5 x 3) xuống (17.75 x 17.75 x 3), và chiều sâu được giữ nguyên

❖ **Lớp thứ tư**

- Đầu vào: 17.75 x 17.75 x 3
- Kích thước bộ lọc: 3 x 3 x 256
- Padding = same
- Hàm kích hoạt : relu

❖ **Lớp chuyển tiếp sang lớp thứ tư**

- Kích thước đầu ra của dữ liệu giảm 1/2 , từ (17.75 x 17.75 x 3) xuống (9.375 x 9.375 x 3), và chiều sâu được giữ nguyên

❖ **Lớp thứ tư**

- Làm phẳng chuyền từ tensor sang dạng vector

❖ **Lớp thứ 5 (Kết nối đầy đủ)**

- Với 128 nút
- Hàm kích hoạt: relu

❖ **Lớp thứ 5 (Kết nối đầy đủ)**

- Với 10 nút
- Hàm kích hoạt: softmax

3.4.2. Huấn luyện mô hình

Để giúp cho việc huấn luyện mô hình trở nên tốt hơn trong hàm compile sử dụng để training model em tiến hành sử dụng thuật toán tối ưu optimizers cụ thể là Stochastic Gradient Descent (SGD) , tuy SGD sẽ làm giảm tốc độ của 1 epoch, nhưng sau vài lần epoch SGD sẽ hội tụ rất nhanh, cùng với đó là momentum giúp cho giải quyết được bài toán Gradient Descent không tiến được tới điểm global minimum mà chỉ dừng lại ở local mimum. Ở hàm loss , sử dụng categorical_crossentropy để có thể dùng classifier nhiều class. Và để đánh giá accuracy của mô hình em sẽ dụng accuracy. Code huấn luyện mô hình.

```
from tensorflow.keras.optimizers import SGD
opt = SGD(learning_rate = 0.001, momentum = 0.9)
model.compile(optimizer = opt, loss ='categorical_cros
sentropy', metrics = ['accuracy'])
history=model.fit(train_dataset,batch_size=batch,epoch
s=200, verbose=1,validation_data=validation_dataset)
```

Tập dữ liệu sau khi được chia thành 1 tập huấn luyện và 1 tập để kiểm tra, ta tiến hành huấn luyện mô hình với batch_size là 32 , epochs là 200 và verbose là 1 để có thể hiển thị tiến trình hoạt động của việc huấn luyện. Sau quá trình huấn luyện, mô hình đạt độ chính xác 91.61%

```
Epoch 194/200
116/116 [==============================] - 28s 244ms/step - loss: 0.0580 - accuracy: 0.9604 - val_loss: 0.4915 - val_accur
acy: 0.9096
Epoch 195/200
116/116 [==============================] - 28s 240ms/step - loss: 0.0575 - accuracy: 0.9553 - val_loss: 0.5686 - val_accur
acy: 0.8922
Epoch 196/200
116/116 [==============================] - 28s 245ms/step - loss: 0.0585 - accuracy: 0.9582 - val_loss: 0.4681 - val_accur
acy: 0.9139
Epoch 197/200
116/116 [==============================] - 29s 248ms/step - loss: 0.0577 - accuracy: 0.9580 - val_loss: 0.5013 - val_accur
acy: 0.9063
Epoch 198/200
116/116 [==============================] - 30s 256ms/step - loss: 0.0575 - accuracy: 0.9563 - val_loss: 0.4699 - val_accur
acy: 0.9139
Epoch 199/200
116/116 [==============================] - 29s 253ms/step - loss: 0.0586 - accuracy: 0.9525 - val_loss: 0.5169 - val_accur
acy: 0.8987
Epoch 200/200
116/116 [==============================] - 29s 250ms/step - loss: 0.0571 - accuracy: 0.9591 - val_loss: 0.4535 - val_accur
acy: 0.9161
```

Hình 3.7. *Mô hình sau huấn luyện*

3.4.3. Lưu mô hình huấn luyện

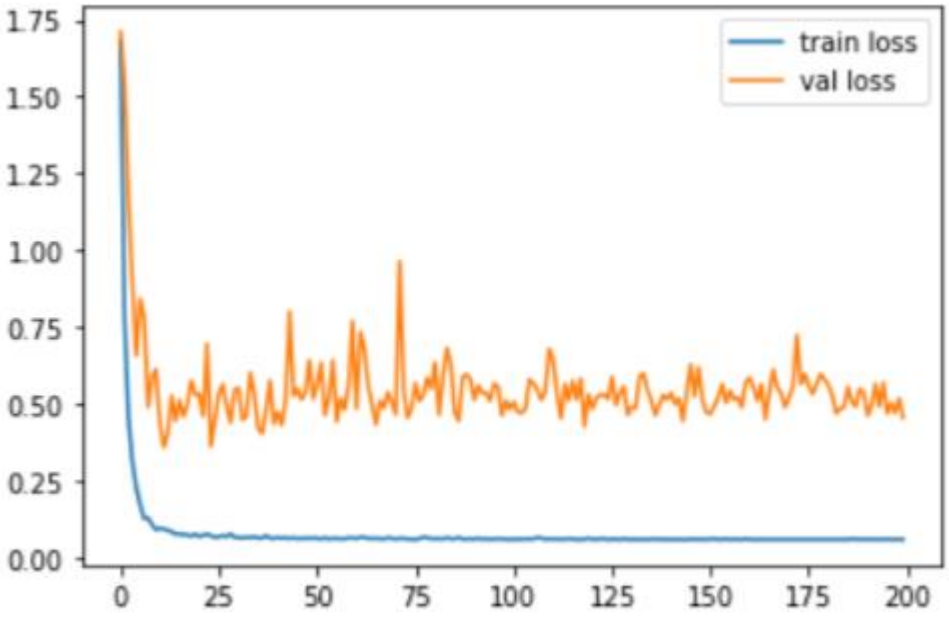
Sau khi huấn luyện xong mô hình, ta tiến hành lưu mô hình với đoạn code như sau

```
# Save model Mask
model.save('Final_10Main2206.h5')
```

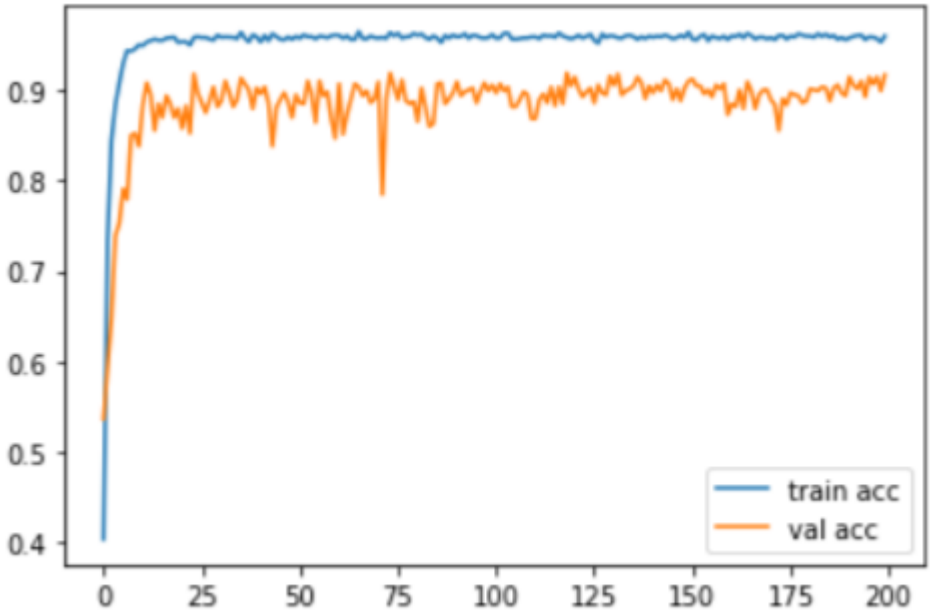
3.4.4. Vẽ các đồ thị

Ta tiến in ra các đồ thị Loss và Accuracy cho mô hình. Code huấn luyện mô hình như sau:

```
# plot the loss
import matplotlib.pyplot as plt
history = history
plt.plot(history.history['loss'], label='train loss')
plt.plot(history.history['val_loss'], label='val loss'
)
plt.legend()
plt.show()
plt.savefig('LossVal_loss')
# plot the accuracy
plt.plot(history.history['accuracy'], label='train acc
')
plt.plot(history.history['val_accuracy'], label='val a
cc')
plt.legend()
plt.show()
plt.savefig('AccVal_acc')
```



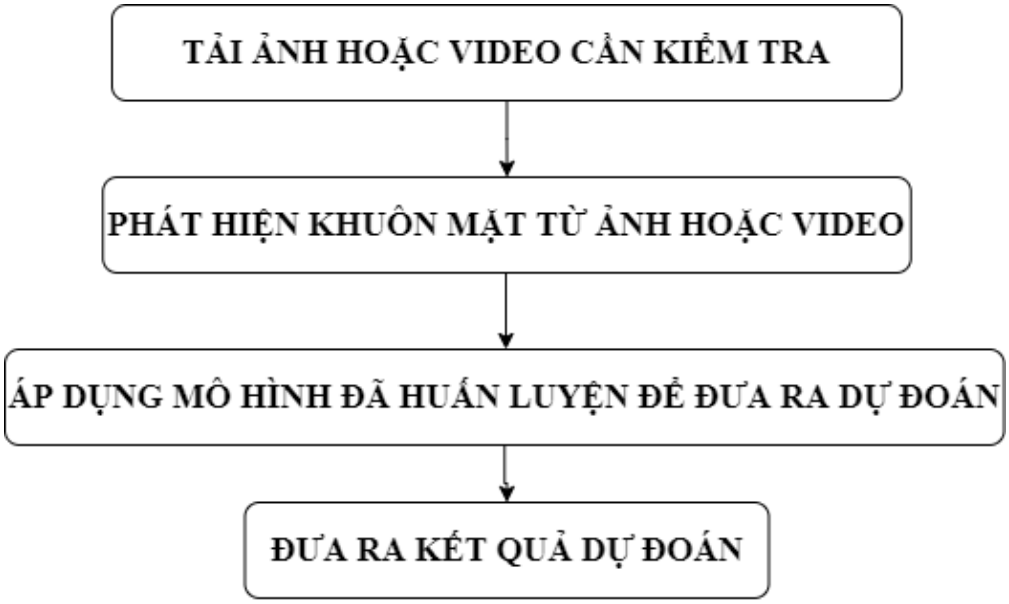
Hình 3.8. *Sai số mất mát của mô hình*



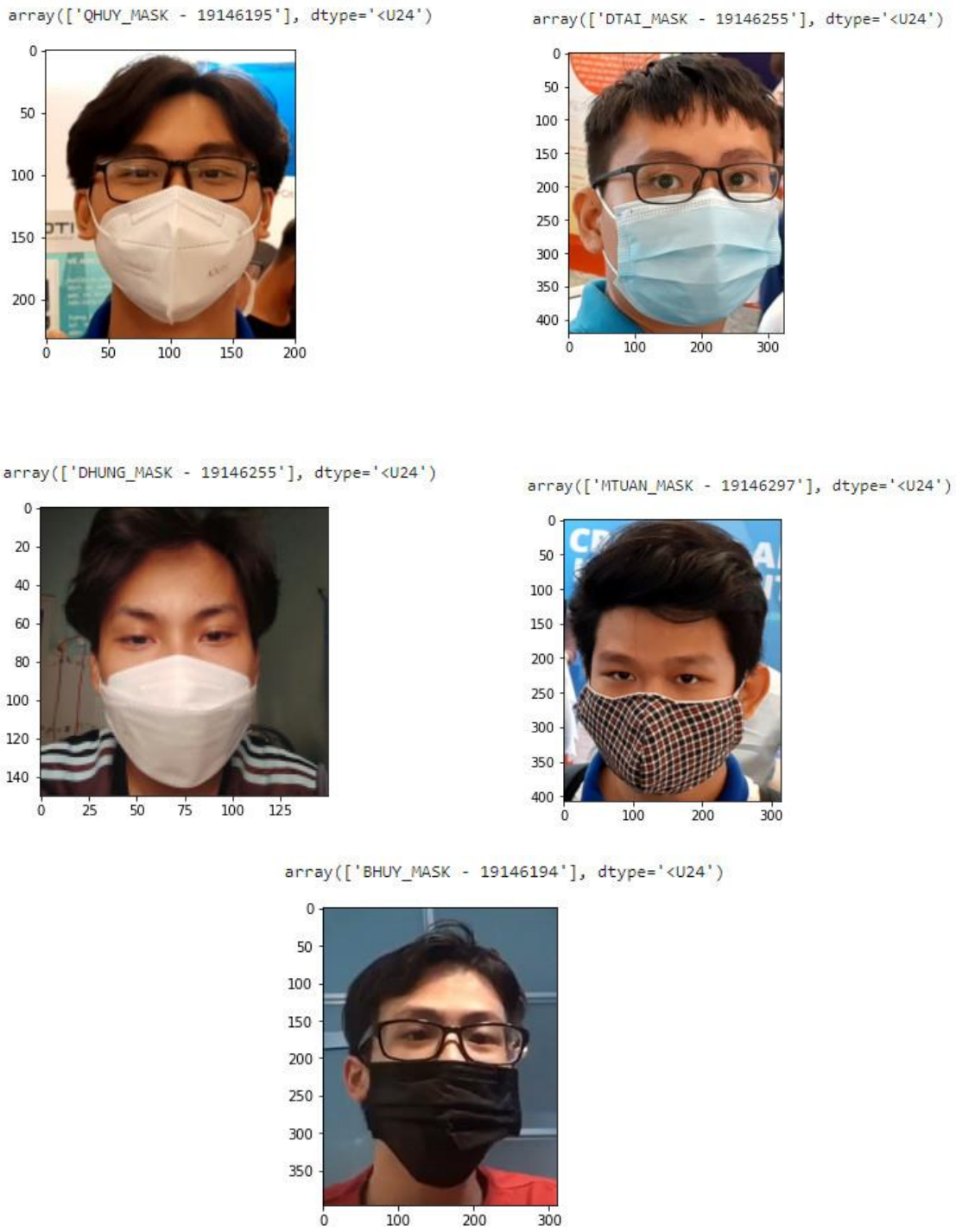
Hình 3.9. *Sai số độ chính xác của mô hình*

3.4.5. Áp dụng mô hình để dự đoán

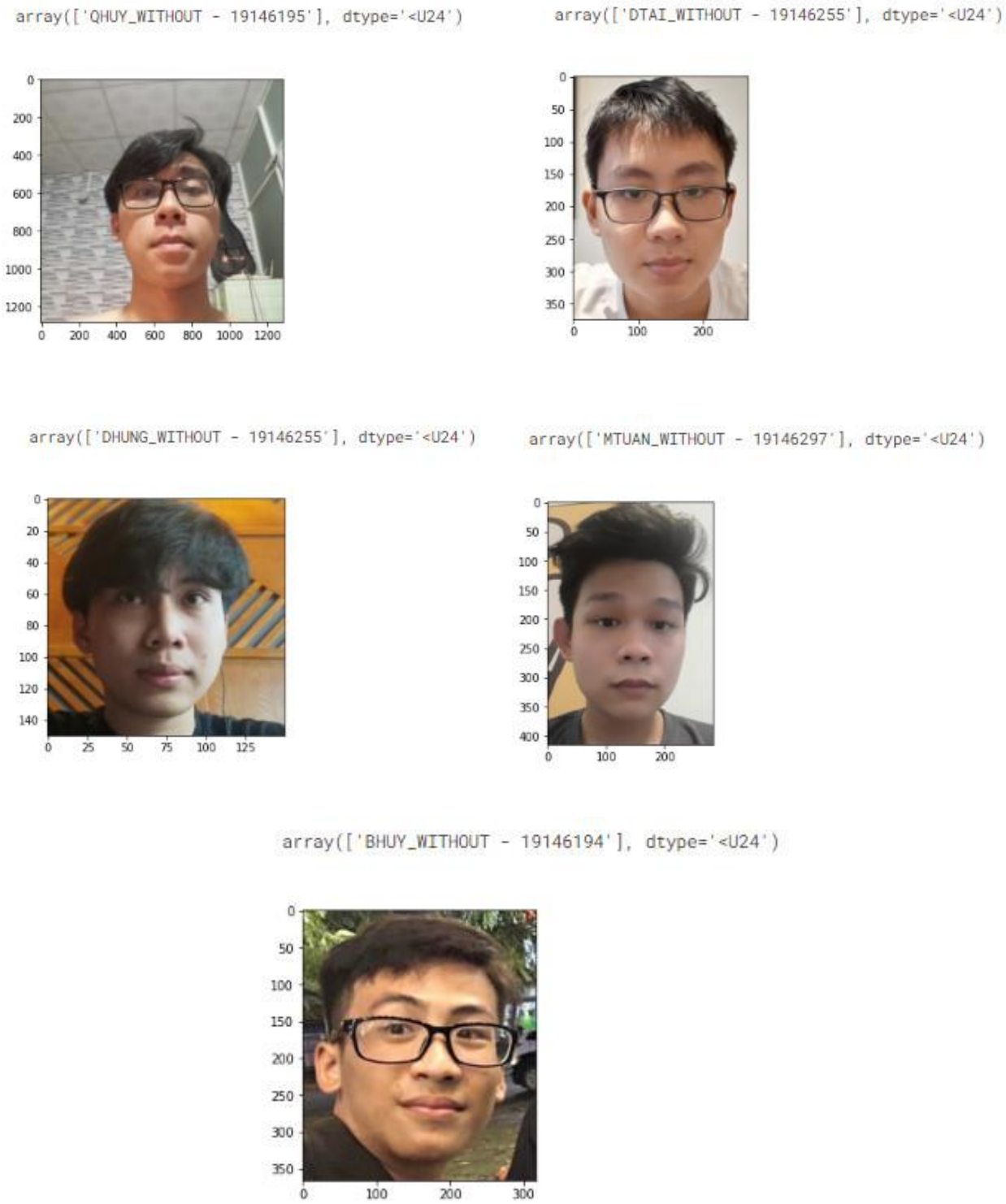
Sau khi huấn luyện mô hình, tiến hành kiểm tra chất lượng của mô hình bằng cách đưa từng tấm ảnh khuôn mặt của 5 người khi đeo và không đeo khẩu trang, những tấm ảnh này phải khác với các tấm ảnh đã sử dụng để huấn luyện mô hình. Đầu tiên, những tấm ảnh này sẽ được đưa về kích thước 150 x 150 để phù hợp với mô hình, chuyển đổi tỷ lệ ảnh từ số nguyên 0-255 và dạng float 0-1. Sau đó tiến hành dự đoán và in kết quả.



Hình 3.10. *Lưu đồ thuật toán cho kiểm nghiệm mô hình*



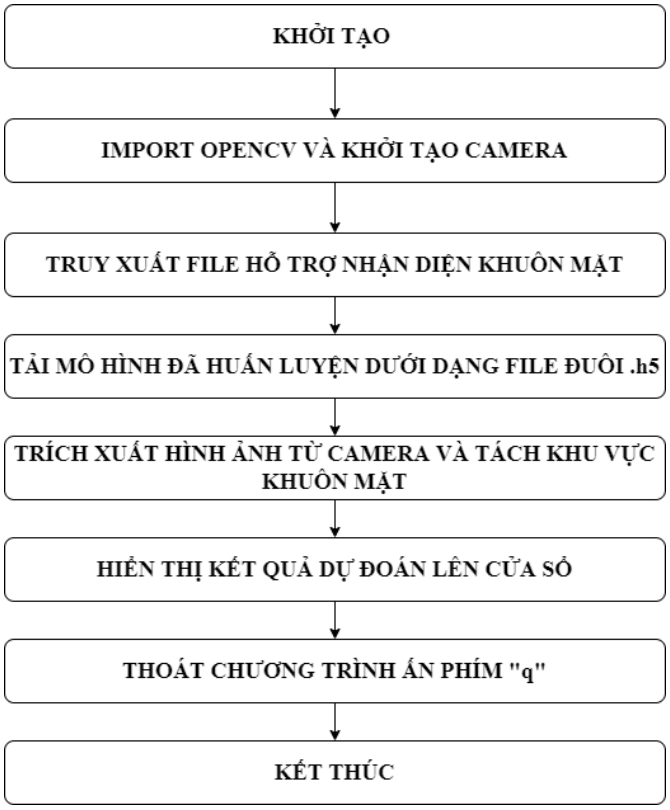
Hình 3.11. *Mô hình dự đoán những khuôn mặt đeo khẩu trang*



Hình 3.12. *Mô hình dự đoán những khuôn mặt không đeo khẩu trang*

3.5. Kiểm nghiệm mô hình trên thời gian thực

3.5.1. Thuật toán phát hiện người đeo khẩu trang trong thời gian thực



Hình 3.13. *Lưu đồ thuật toán cho kiểm nghiệm mô hình thời gian thực*

Để phát hiện khuôn mặt, thuật toán được sử dụng là thuật toán phát hiện khuôn mặt Viola-Jones [2]. Sau khi ảnh được cắt từ video, nó sẽ được định dạng lại kích thước giống với kích thước của dữ liệu trong mô hình huấn luyện. Sau đó, ảnh này sẽ được chuyển đổi thành những tham số có định dạng giống mô hình mẫu. Khi đó các tham số sinh ra được từ tập dữ liệu huấn luyện sẽ được sử dụng để thẩm định lại tính thích hợp của mô hình trên tập dữ liệu của hình ảnh vừa được trích xuất. Dựa vào kết quả thu được ta tiến hành hiển thị lên màn hình kết quả của 1 trong 5 người có đeo khẩu trang hay không.

3.5.2. Khởi tạo các thư viện cần thiết

Để có thể xử lý các tấm ảnh từ video, ta sử dụng thư viện open cv để có thể phát hiện khuôn mặt người thông qua thuật toán phát hiện khuôn mặt Viola-Jones. Và khởi tạo mô hình đã huấn luyện dưới biến model_final. Cụ thể được thể hiển bởi đoạn code dưới đây:

```
import numpy as np
import cv2
from keras.models import load_model
facedetect = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
threshold=0.90
cap=cv2.VideoCapture(0)
cap.set(3, 1000)
cap.set(4, 1000)
font=cv2.FONT_HERSHEY_COMPLEX
model_final = load_model('Final_10Main2206.h5')
```

Sau đó tiến hành tạo các nhãn để sau khi thuật toán phát hiện được khuôn mặt , các nhãn sẽ được gán và đưa ra dự đoán. . Cụ thể được thể hiển bởi đoạn code dưới đây:

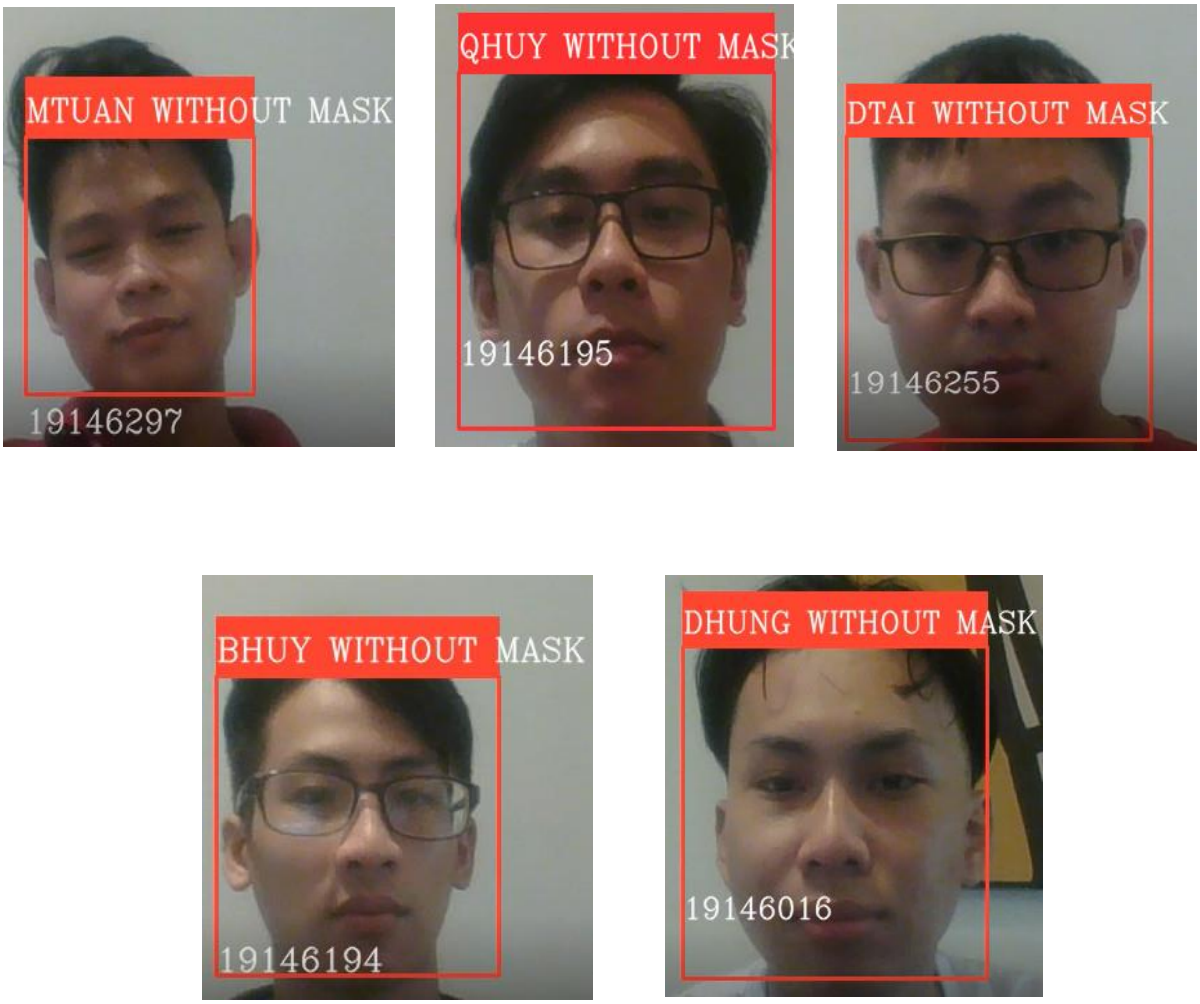
```
def wearMask_className(result):
    if result == 0:
        return "QHUY WITHOUT MASK"
    elif result == 1:
        return "QHUY MASK"
```

Sau khi phát hiện được khuôn mặt, ta sẽ đưa ảnh đã cắt được về kích thước 150x150 để phù hợp với mô hình đã huấn luyện. Cùng với đó sẽ tạo khung bao quanh khuôn mặt và gán nhãn để đưa ra kết quả cuối cùng:

```
while True:
  sucess, imgOrignal=cap.read()
  faces = facedetect.detectMultiScale(imgOrignal,1.3,5
)
  for x,y,w,h in faces:
    crop_img=imgOrignal[y:y+h,x:x+h]
    img=cv2.resize(crop_img, (150,150))
    img=img.reshape(1, 150, 150, 3)
    img = img.astype('float32')
    img = img/255
    prediction=model_final.predict(img)
    result_model =np.argmax(model_final.predict(img),a
xis=-1)
    probabilityValue=np.amax(prediction)
  if probabilityValue>threshold:
      if result_model==8:
        cv2.rectangle(imgOrignal,(x,y),(x+w,y+h),(0,25
5,0),2)
        cv2.rectangle(imgOrignal, (x,y-
40),(x+w, y), (0,255,0),-2)
        cv2.putText(imgOrignal, str(wearMask_className
(result_model)),(x,y-
10), font, 0.75, (255,255,255),1, cv2.LINE_AA)
        cv2.putText(imgOrignal, str("19146195"),(x,y+2
00), font, 0.75, (255,255,255),1, cv2.LINE_AA)
```

3.5.3. Kết quả thu được

***Hình 3.14**. Kết quả khi phát hiện danh tính người đeo khẩu trang trong thời gian thực*



Hình 3.15. *Kết quả khi phát hiện danh tính người không đeo khẩu trang trong thời gian thực*

CHƯƠNG 4: KẾT LUẬN

Bài báo cáo đã xây dựng thành công mô hình nhằm xác định danh tính của 5 người khi đeo và không đeo khẩu trang. Tuy nhiên điều kiện ánh sáng kèm việc phát hiện khẩu trang vẫn chưa thực sự hiệu quả. Chương trình khi phân biệt người đeo khẩu trang và không đeo khẩu trang đôi lúc vẫn chưa phân biệt được đúng. Chương trình được viết bằng ngôn ngữ Python và sử dụng một số thư viện mã nguồn mở như Keras, Python, OpenCV,… Dựa vào kết quả thu được, để có thể giúp cho mô hình tốt hơn, tập dữ liệu phải được tăng cường mạnh mẽ hơn nữa. Trong tương lai chương trình có thể kết hợp với các thiết bị phần cứng như Raspberrypi, Arduino, STM32,…[3] để xây dựng các hệ thống giám sát và phát hiện danh tính người, nhắc nhở người dân đeo khẩu trang ở những nơi đông đúc như trường học, trung tâm thương mại,…

TÀI LIỆU THAM KHẢO

[1] https://viblo.asia/p/deep-learning-tim-hieu-ve-mang-tich-chap-cnn-maGK73bOKj2. [Online].

[2] P. Viola and M. J. Jones, Robust real-time face detection,, Int. J. Comput. Vision, Vol.57, No.2, pp.137-154, May 2004.

[3] W. Gay, Book Raspberrypi Hardware Reference,, Apress 2014.

[4] https://123docz.net/document/7269131-phat-hien-khau-trang-su-dung-mo-hinh-hoc-sau-mobilenetv2.htm. [Online].

[5] https://www.kaggle.com/datasets/infloop1/testdataset-cl3b-ver2. [Online].

[6] https://en.wikipedia.org/wiki/Convolutional_neural_network. [Online].

[7] https://pbcquoc.github.io/cnn/. [Online].

LINK GITHUB PROJECT

https://github.com/infloop1/TRANQUANGHUY_19146195_FINALPROJECTAI

LINK KAGGLE VÀ DATASET

https://www.kaggle.com/code/infloop1/facemaskid-tranquanghuy-19146195

https://www.kaggle.com/datasets/infloop1/cl3dataset-final

LINK YOUTUBE

https://youtu.be/YxIsOwgugXs