BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH

KHOA ĐÀO TẠO CHẤT LƯỢNG CAO

BỘ MÔN CƠ ĐIỆN TỬ

BÁO CÁO CUỐI KỲ

TRÍ TUỆ NHÂN TẠO

ĐỀ TÀI: DỰ ĐOÁN TUỔI TÁC, CẢM XÚC, GIỚI TÍNH DỰA TRÊN KHUÔN MẶT REAL-TIME

GVHD: **PGS. TS. Nguyễn Trường Thịnh**

SVTH: **Bùi Minh Tú** MSSV:**19146295**

Thủ Đức, thứ năm ngày 23 tháng 6 năm 2022

NHẬN XÉT CỦA GIẢNG VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Chữ ký giảng viên

MỤC LỤC

MỤC LỤC HÌNH ẢNH

Thông tin môn học

- Tên môn học: Trí tuệ nhân tạo (Mã môn học: ARIN337629)
- Mã lớp học phần: ARIN337629_21_2_05CLC
- Số tín chỉ: 3
- Điều kiện tiên quyết: Không
- Tóm tắt nội dung học phần: Học phần này trang bị cho sinh viên ngành Công nghệ kỹ thuật cơ điện tử kiến thức cơ bản về trí tuệ nhân tạo, vai trò của trí tuệ nhân tạo trong nghiên cứu khoa học, cũng như sự cần thiết của việc ứng dụng trí tuệ nhân tạo sản xuất và đời sống. Môn học cung cấp cho sinh viên cách giải quyết vấn đề bằng các thuật toán tìm kiếm, biểu diễn tri thức và lập luận (kiến thức và kĩ năng để biểu diễn tri thức, xây dựng một hệ chuyên gia), máy học (kiến thức tổng quan để xây dựng những hệ thống tự động rút trích tri thức từ dữ liệu). Đồng thời, sinh viên cũng được tiếp cận các ứng dụng và thực nghiệm các kiến thức đã được học từ lý thuyết từ đó áp dụng vào thực tiễn. Sau mỗi buổi tham dự tại trường hoặc tại doanh nghiệp, sinh viên sẽ viết báo cáo, khoa xác nhận và cử giảng viên chấm điểm.

1 CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI

1.1 Lý do chọn đề tài

Ngày nay, với sự tiến bộ vượt bậc của công nghệ, ngày càng có những ứng dụng của công nghệ trong nhiều lĩnh vực của cuộc sống. Trong lĩnh vực trí tuệ nhân tạo những năm gần đây, một trong những bài toán nhận được nhiều sự quan tâm và tốn nhiều công sức là bài toán nhận dạng. Tuy mới xuất hiện chưa lâu nhưng nó đã rất được quan tâm vì tính ứng dụng thực tế cũng như sự phức tạp của nó. Bài toán nhận dạng có rất nhiều lĩnh vực như nhận dạng khuôn mặt, nhận dạng giọng nói, nhận dạng chữ viết, nhận dạng vật chất,... Trong đó nhận dạng khuôn mặt được ứng dụng rất rộng rãi trong đời sống hằng ngày như các hệ thống giám sát, an ninh, mở khóa các thiết bị bằng khuôn mặt,...Không giới hạn giải pháp sử dụng, vận dụng được kiến thức trong nhiều lĩnh vực và đưa ra những ứng dụng có ích trong thực tế. Với mong muôn tìm hiểu, nghiên cứu, khám phá và mong muôn xây dựng một ứng dụng có khả năng nhận diện giới tính, độ tuổi và cảm xúc của con người nên em quyết định chọn đề tài "Dự đoán tuổi tác,cảm xúc, giới tính, dựa trên khuôn mặt real-time" cho project cuối kỳ của mình.

1.2 Mục tiêu của đề tài

Dùng thuật toán Convolutional Neural Network (CNN) xây dựng model để dự đoán tuổi, cảm xúc, giới tính dựa trên khuôn mặt real-time. Để đạt được mục tiêu, đề tài tập trung vào các nhiệm vụ:

+ Nghiên cứu tổng quan về nhận diện khuôn mặt.

+ Thu thập datasets đầy đủ để xây dựng các model có độ chính xác cao.

+ Nghiên cứu về real-time và xây dựng giao diện trên app.

2 CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1 Tổng quát về Trí tuệ nhân tạo

Trí Tuệ Nhân Tạo - AI (Artificial Intelligence) hoặc trí thông minh nhân tạo là công nghệ mô phỏng các quá trình suy nghĩ và học tập của con người cho máy móc, đặc biệt là hệ thống máy tính. Trí tuệ nhân tạo này do con người lập trình ra với mục đích tự động hóa các hành vi thông minh như con người, từ đó cắt giảm bớt nhân công là con người và có tính chuẩn xác cao hơn. Cụ thể, trí tuệ nhân tạo giúp máy tính có được những trí tuệ của con người như: biết suy nghĩ và lập luận để giải quyết vấn đề, biết giao tiếp do hiểu ngôn ngữ, tiếng nói, biết học và tự thích nghi,…[1]

Theo dòng chảy của cuộc cách mạng 4.0, trí tuệ nhân tạo ngày càng được phổ biến và ứng dụng rộng rãi trong mọi lĩnh vực của cuộc sống, mặc dù được John McCarthy – nhà khoa học máy tính người Mỹ đề cập lần đầu tiên vào những năm 1950 nhưng đến ngày nay thuật ngữ trí tuệ nhân tạo mới thực sự được biết đến rộng rãi và được các "ông lớn" của làng công nghệ chạy đua phát triển.

AI là công nghệ sử dụng đến kỹ thuật số có khả năng thực hiện những nhiệm vụ mà bình thường phải cần tới trí thông minh của con người, được xem là phổ biến nhất. Đặc trưng của công nghệ AI là năng lực "tự học" của máy tính, do đó có thể tự phán đoán, phân tích trước các dữ liệu mới mà không cần sự hỗ trợ của con người, đồng thời có khả năng xử lý dữ liệu với số lượng rất lớn và tốc độ cao. Hiện mỗi ngày trên toàn cầu có khoảng 2,2 tỷ Gb dữ liệu mới (tương đương 165.000 tỷ trang tài liệu) được tạo ra và được các công ty, như Google, Twitter, Facebook, Amazon, Baidu, Weibo, Tencent hay Alibaba thu thập để tạo thành "dữ liệu lớn" (big data). Trí tuệ nhân tạo là một lĩnh vực liên quan đến chuyên ngành khoa học máy tính và công nghệ thông tin, bản chất của trí tuệ nhân tạo vẫn do con người làm ra, họ xây dựng các thuật toán, lập trình bằng các công cụ phần mềm công nghệ thông tin, giúp các máy tính có thể tự động xử lý các hành vi thông minh như con người.

Trí tuệ nhân tạo có khả năng tự thích nghi, tự học và tự phát triển, tự đưa ra các lập luận để giải quyết vấn đề, có thể giao tiếp như người…tất cả là do AI được cài một cơ sở dữ liệu lớn, được lập trình trên cơ sở dữ liệu đó và tái lập trình trên cơ sở dữ liệu

mới sinh ra. Cứ như vậy cấu trúc của AI luôn luôn thay đổi và thích nghi trong điều kiện và hoàn cảnh mới. Dự báo đến năm 2030 của công ty kiểm toán và tư vấn tài chính PwC, GDP toàn cầu có thể tăng trưởng thêm 14% từ sự hỗ trợ của trí tuệ nhân tạo, AI đã xuất hiện trong nhiều ngành, từ cung cấp dịch vụ mua sắm ảo và ngân hàng trực tuyến đến giảm chi phí đầu tư trong sản xuất và hợp lý hóa chẩn đoán trong chăm sóc sức khỏe. AI đã thúc đẩy hầu hết các ngành công nghiệp tiến lên và thay đổi cuộc sống của nhiều người.

Nhìn chung, đây là một ngành học rất rộng, bao gồm các yếu tố tâm lý học, khoa học máy tính và kỹ thuật. Một số ví dụ phổ biến về AI có thể kể đến ô tô tự lái, phần mềm dịch thuật tự động, trợ lý ảo trên điện thoại hay đối thủ ảo khi chơi trò chơi trên điện thoại.

2.2 Tổng quát về Deep Learning

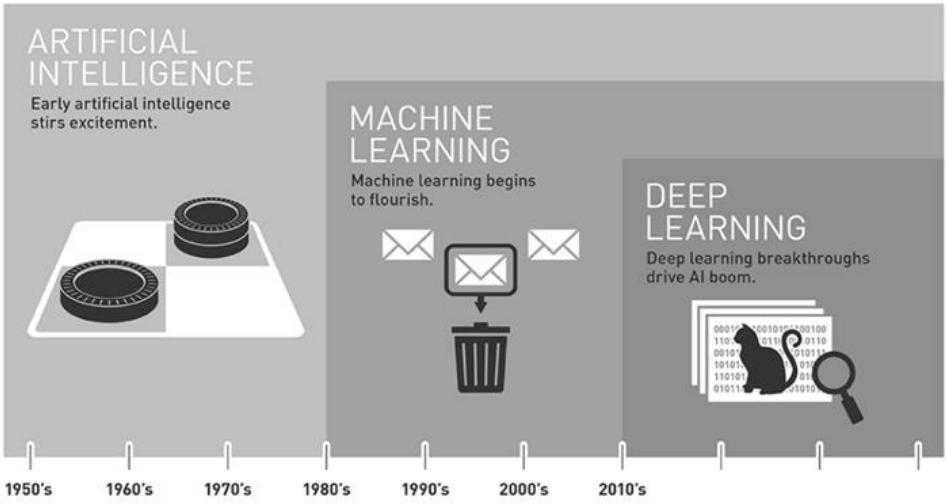
Deep Learning (DL) là tập hợp con của Machine Learning và nó có tác dụng hỗ trợ cho máy tính tự huấn luyện chính nó để có thể thực hiện mọi tác vụ tương tự như con người. Điều này chính là giúp máy tính bắt chước con người cách học hỏi và suy nghĩ.[2]

Học máy (machine learning, ML) là một tập con của trí tuệ nhân tạo. Machine learning là một lĩnh vực nhỏ trong khoa học máy tính, có khả năng tự học hỏi dựa trên dữ liệu được đưa vào mà không cần phải được lập trình cụ thể.

Những năm gần đây, sự phát triển của các hệ thống tính toán cùng lượng dữ liệu khổng lồ được thu thập bởi các hãng công nghệ lớn đã giúp machine learning tiến thêm một bước dài. Một lĩnh vực mới được ra đời được gọi là học sâu (deep learning, DL). Deep learning đã giúp máy tính thực thi những việc vào mười năm trước tưởng chừng là không thể: phân loại cả ngàn vật thể khác nhau trong các bức ảnh, tự tạo chú thích cho ảnh, bắt chước giọng nói và chữ viết, giao tiếp với con người, chuyển đổi ngôn ngữ, hay thậm chí cả sáng tác văn thơ và âm nhạc.

Các hệ thống của Deep Learning có khả năng cải thiện được những hiệu suất của chúng với quyền truy cập vào dữ liệu sẽ được nhiều hơn. Deep Learning có hỗ trợ cho việc dịch ngôn ngữ, phân loại các hình ảnh, nhận dạng giọng nói. Chính vì thế, nó

có thể được ứng dụng để giải quyết mọi nhu cầu cần nhận dạng mẫu mà không cần đến sự can thiệp của con người.



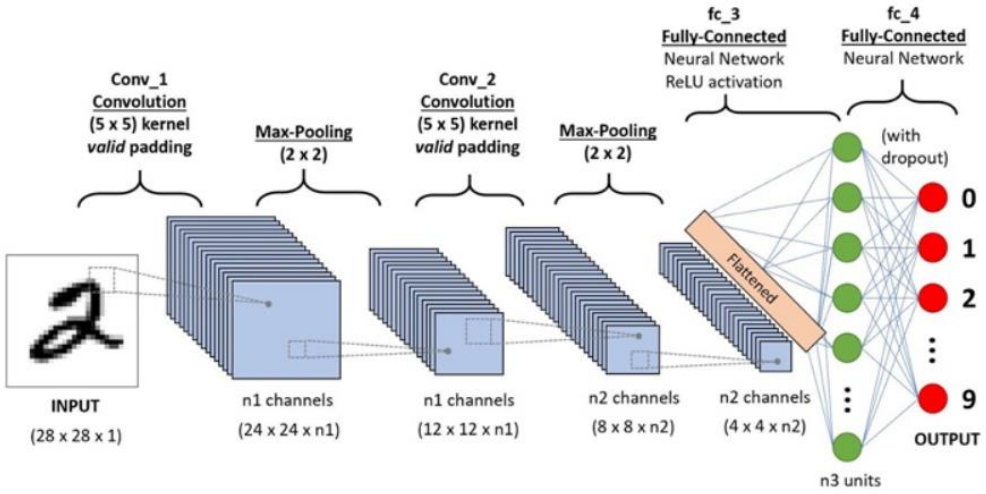
Hình 1. Mối quan hệ giữa AI, ML và DL

2.3 Thuật toán Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNN) là một trong những mô hình deep learning phổ biến nhất và có ảnh hưởng nhiều nhất trong cộng đồng Computer Vision. CNN được dùng trong trong nhiều bài toán như nhân dạng ảnh, phân tích video, ảnh MRI, hoặc cho bài các bài của lĩnh vự xử lý ngôn ngữ tự nhiên,và hầu hết đều giải quyết tốt các bài toán này.[3]

Cấu trúc của CNN: Mạng CNN là tập hợp nhiều lớp Convolutional chồng lên nhau, sử dụng các hàm Nonlinear Activation và tanh để kích hoạt các trọng số trong các node. Ở mỗi lớp CNN, sau khi được các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho những lớp tiếp theo. Mỗi Layer kết tiếp sẽ là kết quả Convolution từ Layer trước đó nên chúng ta có được các kết nối cục bộ.

Thông qua quá trình huấn luyện mạng, các lớp Layer CNN tự động học các giá trị được thể hiện qua các lớp Filter.



Hình 2.Cấu trúc mạng CNN

Trong mô hình CNN có 2 khía cạnh cần quan tâm là tính bất biến (Location Invariance) và tính kết hợp (Compositionality). Với cùng một đối tượng, nếu đối tượng này được chiếu theo các gốc độ khác nhau (translation, rotation, scaling) thì độ chính xác của thuật toán sẽ bị ảnh hưởng đáng kể.

2.4 Dữ liệu mất cân bằng - Imbalanced data

Mất cân bằng dữ liệu (Imbalanced dataset) là tập dữ liệu có tỷ lệ categories khác nhau, thường là chênh nhau khá xa. Ví dụ dữ liệu y tế, chắc hẳn dữ liệu về một số bệnh sẽ có nhiều mẫu âm tính nhiều hơn nhiều so với mẫu dương tính, có thể lên tới tỷ lệ 98%-2%. Điều này ảnh hưởng rất lớn tới mô hình dự đoán, khi mà sự mất cân bằng như vậy sẽ khiến mô hình dự báo kém chính xác trên nhóm thiểu số, bởi đa phần kết quả dự báo thường thiên về 1 nhóm là nhóm đa số.[4]

Các phương pháp xử lý:

- Random Oversampling and Undersampling : kỹ thuật Oversampling ở đây đơn giản là tăng số lượng mẫu ở Lớp thiểu số, giả sử class A có 100 mẫu, class B có 1000 mẫu ta sẽ tăng số lượng mẫu của class A bằng cách Random vị trí từ tập dữa liệu có trong mẫu và đưa vào training set. Đi cùng với oversampling thì là Undersampling. Ý tưởng tương tự như Over nhưng thay vì tăng thì sẽ là giảm data ở Lớp đa số cũng bằng cách Random như trên và delete dữ liệu đó đi.

- SMOTEL: Ý tưởng chung của SMOTE là tạo ra dữ liệu tổng hợp giữa mỗi mẫu của lớp thiểu số và "k" hàng xóm gần nhất của nó. Có nghĩa là, đối với mỗi một trong các mẫu của lớp thiểu số, "k" các láng giềng gần nhất của nó được chọn (theo mặc định là k = 5). Sau đó giữa các cặp điểm được tạo bởi mẫu và từng láng giềng của nó thì ta sẽ có được data mới. Ví dụ như hình bên dưới, với P1 là dữ liệu mẫu với k=3 --> P2, 3, 4 ta sẽ ramdom giá trị giữa từng khoảng cách từ P2, 3, 4 đến P. Và cũng từ P2, 3, 4 làm tương tự như thế.

2.5 Các phương pháp đánh giá mô hình

2.5.1 Accuracy

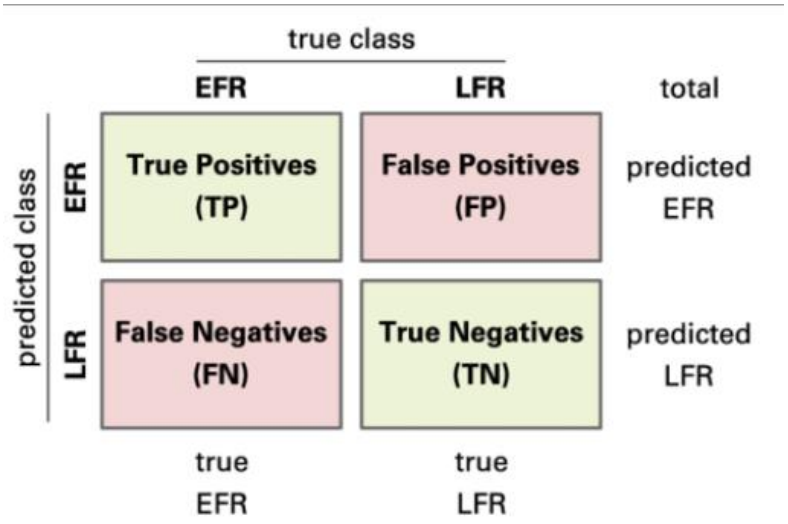
Cách đơn giản và hay được sử dụng nhất là accuracy (độ chính xác). Cách đánh giá này đơn giản tính tỉ lệ giữa số điểm được dự đoán đúng và tổng số điểm trong tập dữ liệu kiểm thử. Độ chính xác giúp ta đánh giá hiệu quả dự báo của mô hình trên một bộ dữ liệu. Độ chính xác càng cao thì mô hình của chúng ta càng chuẩn xác.[5]

2.5.2 Confusion matrix

Cách tính sử dụng accuracy như ở trên chỉ cho chúng ta biết được bao nhiêu phần trăm lượng dữ liệu được phân loại đúng mà không chỉ ra được cụ thể mỗi loại được phân loại như thế nào, lớp nào được phân loại đúng nhiều nhất, và dữ liệu thuộc lớp nào thường bị phân loại nhầm vào lớp khác. Để có thể đánh giá được các giá trị này, chúng ta sử dụng một ma trận được gọi là confusion matrix.[6]

Confusion matrix là một kỹ thuật đánh giá hiệu năng của mô hình cho các bài toán phân lớp. Confusion matrix là một ma trận thể hiện số lượng điểm dữ liệu thuộc vào một class và được dự đoán thuộc vào class.

Nhược điểm của Accuracy là chỉ cho ta biết độ chính xác khi dự báo của mô hình, nhưng không thể hiện mô hình đang dự đoán sai như thế nào, vì vậy chúng ta cần phương pháp Confusion Matrix. Trong những bài toán này, người ta thường định nghĩa lớp dữ liệu quan trọng hơn cần được xác định đúng là lớp Positive (P-dương tính), lớp còn lại được gọi là Negative (N-âm tính). Ta định nghĩa True Positive (TP), False Positive (FP), True Negative (TN), False Negative (FN) dựa trên confusion matrix.



Hình 3. Các đối tượng của confusion matrix

- True Positive (TP): đối tượng ở lớp Positive, mô hình phân đối tượng vào lớp Positive (dự đoán đúng).
- True Negative (TN): đối tượng ở lớp Negative, mô hình phân đối tượng vào lớp Negative (dự đoán đúng).
- False Positive (FP): đối tượng ở lớp Negative, mô hình phân đối tượng vào lớp Positive (dự đoán sai).
- False Negative (FN): đối tượng ở lớp Positive, mô hình phân đối tượng vào lớp Negative (dự đoán sai).

2.5.3 Percision

Precision trả lời cho câu hỏi trong các trường hợp được dự báo là positive thì có bao nhiêu trường hợp là đúng. Precision được định nghĩa là tỉ lệ số điểm true positive trong số những điểm được phân loại là positive.[7]

Công thức tính Precision:

$$\text{Precision} = \frac{\text{TP}}{\text{total predicted positive}} = \frac{\text{TP}}{\text{TP+FP}}$$

2.5.4 Recall

Recall giúp chúng ta biết được có bao nhiêu điểm dữ liệu thực sự ở lớp Positive được mô hình phân lớp đúng trong mọi điểm dữ liệu thực sự ở lớp Positive. Recall

được định nghĩa là tỉ lệ số điểm true positive trong số những điểm thực sự là positive.[8]

Công thức tính Recall:

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}$$

2.5.5 F1-score

F1-Score là trung bình điều hòa giữa precision và recall. Do đó nó đại diện hơn trong việc đánh giá độ chính xác trên đồng thời precision và recall. Một mô hình tốt khi cả Precision và Recall đều cao, thể hiện cho mô hình ít phân loại nhầm giữa các lớp cũng như tỉ lệ bỏ sót các đối tượng thuộc lớp cần quan tâm là thấp. Tuy nhiên, hai giá trị Precision và Recall thường không cân bằng với nhau (giá trị này tăng thì giá trị kia thường có xu hướng giảm). Để đánh giá cùng lúc cả Precision và Recall, ta sử dụng F1-Score.[9]

Công thức F1-score:

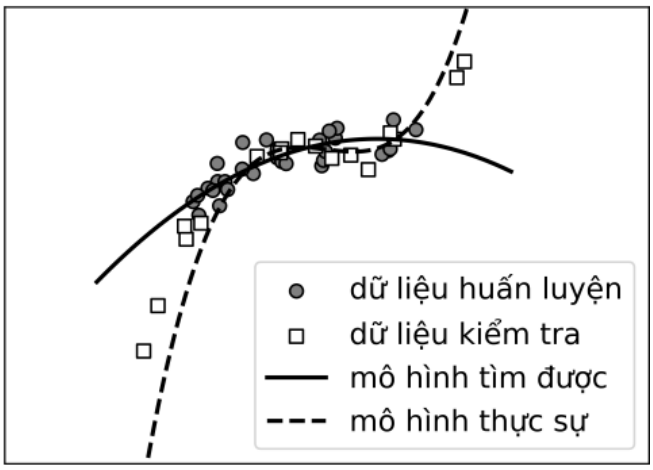
$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

2.5.6 Learning Curve

Learning Curve (đường cong học tập) cho thấy điểm xác nhận và đào tạo của một ước tính cho số lượng mẫu đào tạo khác nhau. Đây là một công cụ để tìm hiểu xem người dùng sẽ được hưởng lợi bao nhiêu từ việc thêm nhiều dữ liệu để huấn luyện hơn và liệu ước tính có bị lỗi phương sai hay lỗi thiên vị hay không.[10]

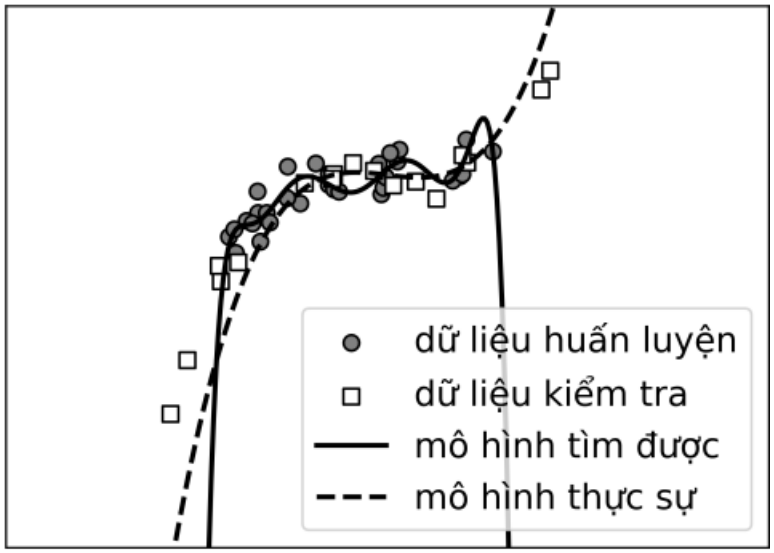
Có 3 dạng có thể quan sát được ở Learning Curve:

+ Underfitting (chưa khớp): là hiện tượng khi mô hình xây dựng chưa có độ chính xác cao trong tập dữ liệu huấn luyện cũng như tổng quát hóa với tổng thể dữ liệu.



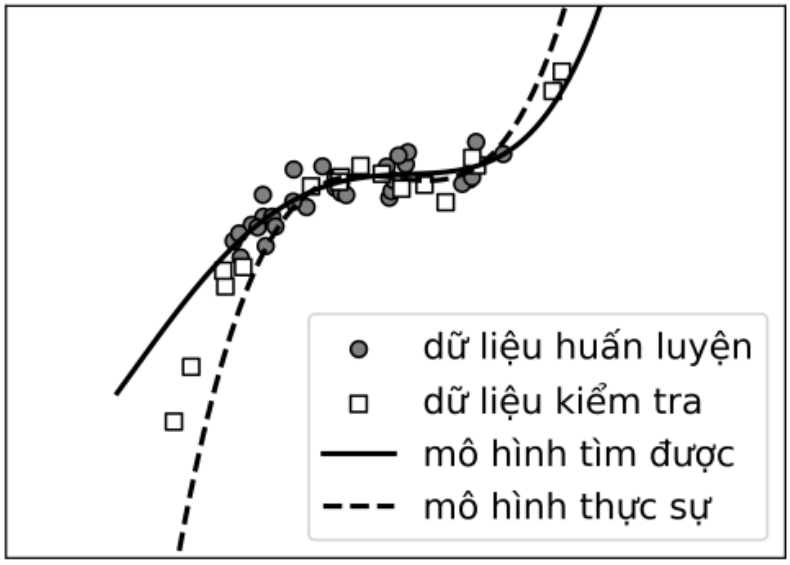
Hình 4. Underfitting

+ Overfitting (quá khớp): là một lỗi mô hình hóa xảy ra khi một hàm quá phù hợp với một tập hợp giới hạn các điểm dữ liệu. Mô hình đã học bộ dữ liệu huấn luyện quá tốt, bao gồm cả những dữ liệu nhiễu hoặc những biến động ngẫu nhiên trong bộ dữ liệu.



Hình 5. Overfitting

+ Good fitting (vừa khớp): nằm giữa Underfitting và Overfitting. Mô hình cho ra kết quả hợp lý với cả tập dữ liệu huấn luyện và các tập dữ liệu mới. Đây là mô hình lý tưởng mang tính tổng quát và khớp được với nhiều dữ liệu mẫu và cả các dữ liệu mới.Good Fitting là mục tiêu của mỗi bài toán. Tuy nhiên, trên thực tế, vấn đề này rất khó thực hiện. Để tìm được điểm Good Fitting, ta phải theo dõi hiệu suất của thuật toán học máy theo thời gian khi thuật toán thực hiện việc học trên bộ dữ liệu huấn luyện. Ta có thể mô tả và thể hiện các thông số mô hình, độ chính xác của mô hình trên cả hai tập dữ liệu huấn luyện và đào tạo.



Hình 6. Good fitting

2.6 Các thư viện sử dụng trong đề tài

2.6.1 Thư viện Scikit-learn

Scikit-learn (Sklearn) là thư viện mạnh mẽ nhất dành cho các thuật toán học máy được viết trên ngôn ngữ Python. Thư viện cung cấp một tập các công cụ xử lý các bài toán machine learning và statistical modeling gồm: classification, regression, clustering, và dimensionality reduction.[11]

Thư viện được cấp phép bản quyền chuẩn FreeBSD và chạy được trên nhiều nền tảng Linux. Scikit-learn được sử dụng như một tài liệu để học tập. Scikit-learn hỗ trợ mạnh mẽ trong việc xây dựng các sản phẩm. Nghĩa là thư viện này tập trung sâu trong việc xây dựng các yếu tố: dễ sử dụng, dễ code, dễ tham khảo, dễ làm việc, hiệu quả cao.

2.6.2 Thư viện Keras

Keras là một open source cho Neural Network được viết bởi ngôn ngữ Python. Nó là một library được phát triển vào năm 2005 bởi Francois Chollet, là một kỹ sư nghiên cứu Deep Learning. Keras là một thư viện tương đối dễ sử dụng đối với người mới bắt đầu. Nó cung cấp các hàm số cần thiết với cú pháp đơn giản.[12]

Một số ưu điểm của Keras:

- Dễ sử dụng, xây dựng model nhanh.

- Run được trên cả CPU và GPU.
- Hỗ trợ xây dựng CNN , RNN hoặc cả hai.

2.6.3 Thư viện Tensorflow

Tensorflow là một thư viện mã nguồn mở cung cấp khả năng xử lí tính toán số học dựa trên biểu đồ mô tả sự thay đổi của dữ liệu, trong đó các node là các phép tính toán học còn các cạnh biểu thị luồng dữ liệu.[13]

Một chương trình Tensorflow được chia thành hai phần chính. Phần thứ nhất là xây dựng mô hình tính toán (được gọi là construction phase), phần thứ hai là chạy mô hình vừa mới xây dựng (được gọi là execution phase).

Ưu điểm của Tensorflow:

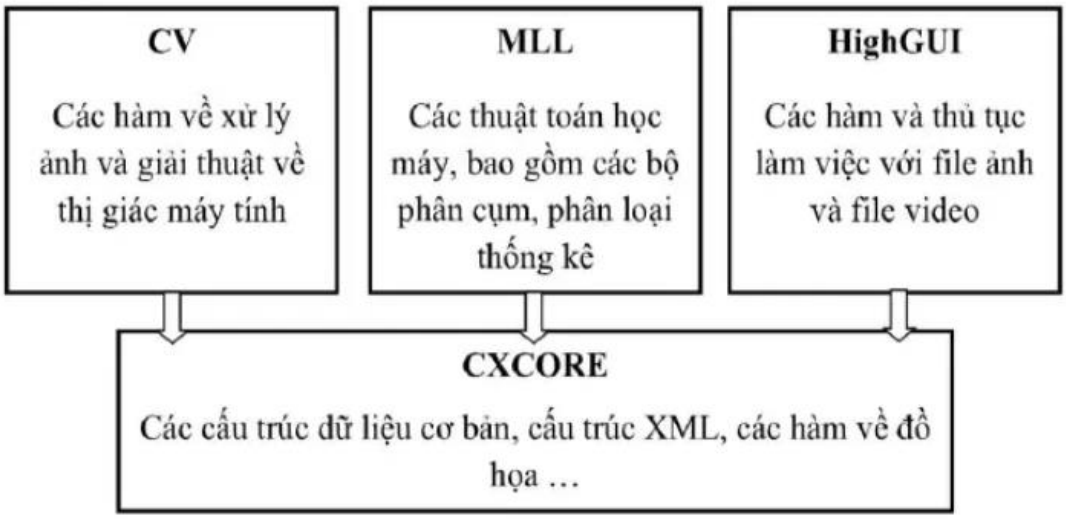
- Abstraction: người dùng chỉ cần tập trung vào phần logic tổng thể thay vì phải đối phó với việc triển khai thuật toán rườm rà.
- Chế độ eager execution cho phép đánh giá và sửa đổi từng operation của biểu đồ 1 cách riêng biệt và minh bạch, thay vì phải dựng toàn bộ biểu đồ dưới dạng 1 đối tượng độc lập vốn khá mơ hồ hay phải đánh giá chung tổng thể.
- TensorBoard cho phép quan sát 1 cách trực quan những gì TensorFlow đang làm.

2.6.4 Thư viện OpenCV

OpenCV (Open Computer Vision) là một thư viện nguồn mở cho máy tính. Cụ thể hơn OpenCV là kho lưu trữ các mã nguồn mở được dùng để xử lý hình ảnh, phát triển các ứng dụng đồ họa trong thời gian thực.[14]

OpenCV cho phép cải thiện tốc độ của CPU khi thực hiện các hoạt động real time. Nó còn cung cấp một số lượng lớn các mã xử lý phục vụ cho quy trình của thị giác máy tính hay các learning machine khác. OpenCV đươc viết bằng C/C++, vì vậy có tốc độ tính toán rất nhanh, có thể sử dụng với các ứng dụng liên quan đến thời gian thực. Opencv có các interface cho C/C++, Python Java vì vậy hỗ trợ được cho Window, Linux, MacOs lẫn Android, iOS.

OpenCV có các ứng dụng như nhận dạng hình ảnh, xử lý hình ảnh, phục hồi hình ảnh, video,...



Hình 7. Cấu trúc các phần của OpenCV

3 CHƯƠNG 3. NỘI DUNG ĐỀ TÀI

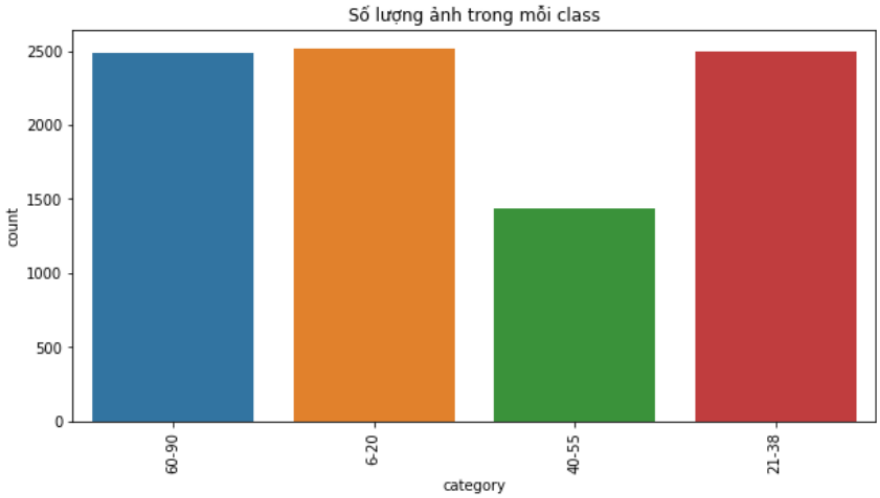
3.1. Mô tả đề tài

Nhận dạng khuôn mặt người là một công nghệ được ứng dụng rộng rãi trong đời sống hằng ngày của con người như các hệ thống giám sát, quản lý vào ra, tìm kiếm thông tin một người nổi tiếng,…Có rất nhiều phương pháp nhận dạng khuôn mặt để nâng cao hiệu suất tuy nhiên dù ít hay nhiều những phương pháp này đang vấp phải những thử thách về độ sáng, hướng nghiêng, kích thước ảnh, hay ảnh hưởng của tham số môi trường. Và ở đề tài này, em đã sử dụng thuật toán CNN (Mạng nơ-ron tích chập) để xây dựng 3 mô hình giới tính, tuổi và cảm xúc để dự đoán dựa trên khuôn mặt real-time. Để thực hiện đề tài này, em đã thu thập hình ảnh từ nhiều nguồn khác nhau với khoảng 47000, 9000 và 28000 hình ảnh tương ứng với mỗi mô hình. Công việc thực hiện bao gồm thay đổi kích thước hình ảnh, giải mã và tiêu chuẩn hóa rồi sau đó sẽ được đào tạo và kiểm tra để xác định độ chính xác của các mô hình.

3.2. Giới thiệu về datastes

3.2.1. "Age recognition dataset "

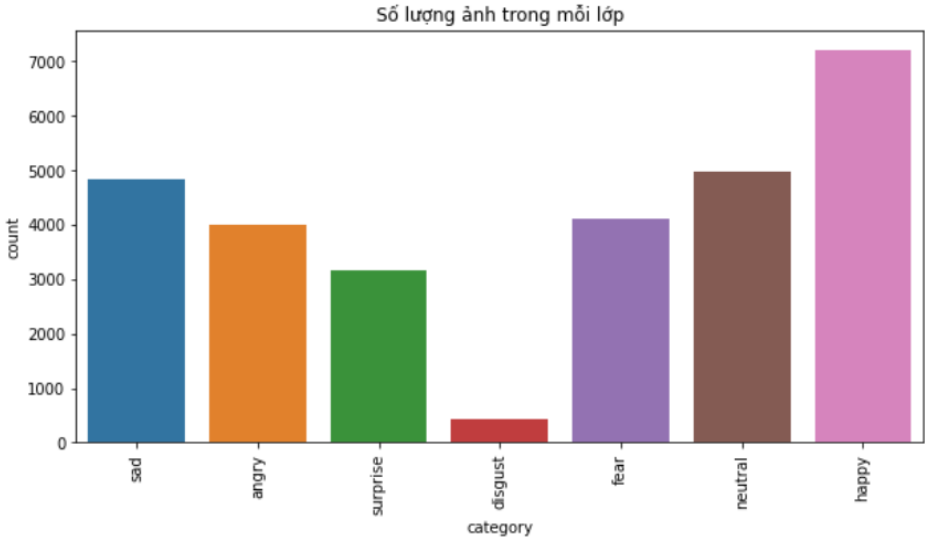
Đây là tập dữ liệu về tuổi tác được đăng lên Kaggle bởi Rashik Rahman. Gồm có 4 classes bao gồm: 6-20, 25-30, 42-48, 60-98. Do các độ tuổi trong các lớp không sát nhau nên em đã lọc, tìm kiếm thêm và phân lại các độ tuổi giúp việc dự đoán dễ hơn. Sau khi phân lại thì tập cũng gồm 4 classes: 6-20, 21-38, 40-55, 60-90.



Hình 8. Biểu đồ thể hiện số lượng ảnh trong mỗi lớp trong tập "Age recognition dataset"

3.2.2. "FER-2013''

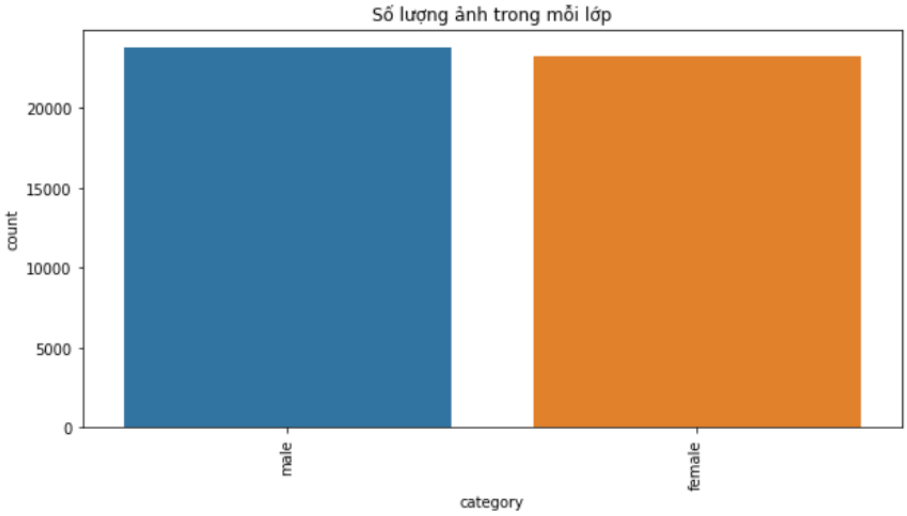
Đây là tập dữ liệu về cảm xúc được đăng lên Kaggle bởi Mansas Sambare. Gồm có 7 classes bao gồm: angry, disgust, fear, happy, neutral, sad, surprise. Theo biểu đồ bên dưới ta có thể thấy dữ liệu trong các lớp không đều và đang gặp tình trạng imbalanced.



Hình 9. Biểu đồ số lượng ảnh trong mỗi lớp của tập "FER-2013"

3.2.3. "Gender Classification Dataset"

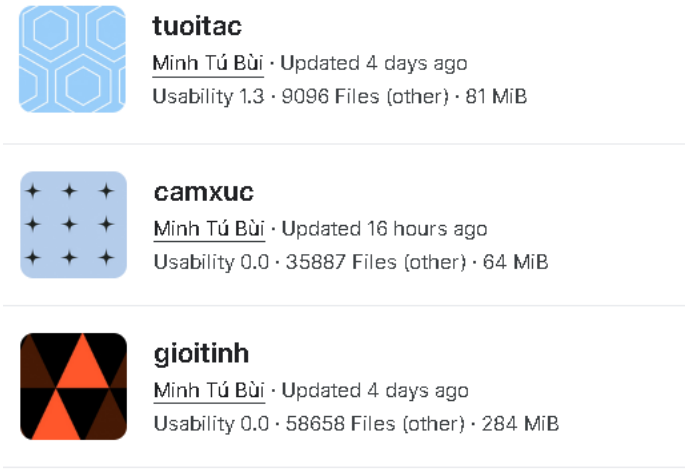
Đây là tập dữ liệu về giới tính gồm 2 lớp female và male, được đăng lên Kaggle bởi Ashutosh Chauhan.



Hình 10. Biểu đồ số lượng ảnh trong 2 lớp của "Gender Classification Dataset"

3.3. Xử lý dữ liệu, huấn luyện và đánh giá mô hình

Để giúp việc tải dữ liệu lên colab nhanh chóng và hiệu quả thì các tập dữ liệu sẽ được tải lên Kaggle trước, sau đó tại colab sẽ dùng các câu lệnh cần thiết để tải dữ liệu từ Kaggle lên.



Hình 11. Các tập dữ liệu tuổi, cảm xúc, giới tính được tải lên Kaggle

```
[ ]  import os
     import json
     kaggleAPIToken = {"username":"minhtbi","key":"5c090ffe0e6f413dc7d7d970d54b1f76"}
     with open('/content/kaggle/kaggle.json', 'w') as file:
         json.dump(kaggleAPIToken, file)
```

```
[ ]  !chmod 600 /content/kaggle/kaggle.json
     !sudo mkdir ~/.kaggle
     !cp /content/kaggle/kaggle.json ~/.kaggle/kaggle.json
     !mkdir dataset
     !kaggle datasets download -d minhtbi/dataset-tuoi
```

```
Downloading dataset-tuoi.zip to /content
 99% 75.0M/75.8M [00:01<00:00, 78.1MB/s]
100% 75.8M/75.8M [00:01<00:00, 51.9MB/s]
```

```
[ ]  !unzip /content/dataset-tuoi.zip
```

Khai báo các thư viện cần dùng để huấn luyện:

```
import tensorflow
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential, save_model, load_model
from tensorflow.keras.layers import Conv2D, MaxPool2D
from tensorflow.keras.layers import Activation, Dropout, Flatten, Dense
,BatchNormalization
import numpy as np
from tensorflow.keras.callbacks import TensorBoard
```

```
import time
import matplotlib.pyplot as plt
import datetime
import cv2
import tensorflow as tf
import tensorflow.keras
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
from keras.callbacks import EarlyStopping,ModelCheckpoint
from sklearn.metrics import confusion_matrix,classification_report
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

3.3.1. Tập "Age recognition dataset"

Đồng bộ lại kích thước của tập dataset là 96x96.

```
class_name = ['21-38','40-55','6-20','60-90']
path = '/content/Dataset/'
target_name = {'21-38': 0, '40-55': 1,'6-20': 2, '60-90': 3}
X, y = [], []
for age in class_name:
    for img_name in os.listdir(path+age):
        img = cv2.imread(path + age +'/'+ img_name)
        scalled_raw_img = cv2.resize(img, (96, 96))/255.0
        X.append(scalled_raw_img)
        y.append(target_name[age])
```

Sử dụng thư viện để phân tích và khai thác dữ liệu. Một phân đoạn sẽ được thực hiện để phân tách bộ dữ liệu thành hai gồm train và test.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
, random_state=42)
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Xây dựng model:

```
model = Sequential()
model.add(base_model)
model.add(Dropout(0.5))
model.add(Dense(4, activation='softmax'))
model.compile(loss ='categorical_crossentropy', optimizer ='adam', metr
ics =['accuracy'])
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 xception (Functional)       (None, 2048)              20861480

 dropout (Dropout)           (None, 2048)              0

 dense (Dense)               (None, 4)                 8196

=================================================================
Total params: 20,869,676
Trainable params: 20,815,148
Non-trainable params: 54,528
_________________________________________________________________
```

Sử dụng EarlyStopping để lưu lại model tốt nhất:

```
my_calls = [EarlyStopping(monitor="val_accuracy",patience=5),
            ModelCheckpoint("Model_tuoi.h5",verbose= 1 ,save_best_only=
True)]
```

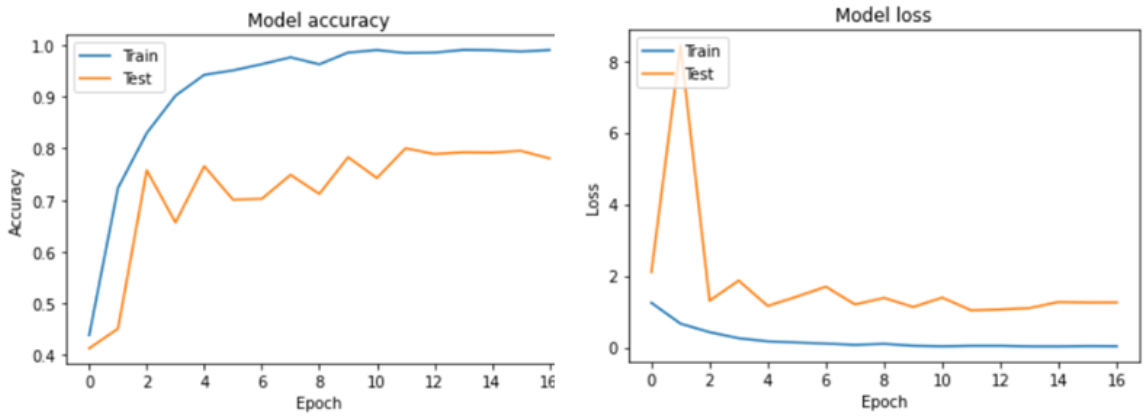
Tiến hành train model:

```
model1 = model.fit(x=X_train, y=y_train, batch_size=256, epochs=20, val
idation_data=(X_test, y_test),callbacks=my_calls)
```

Model tốt nhất được lưu lại:

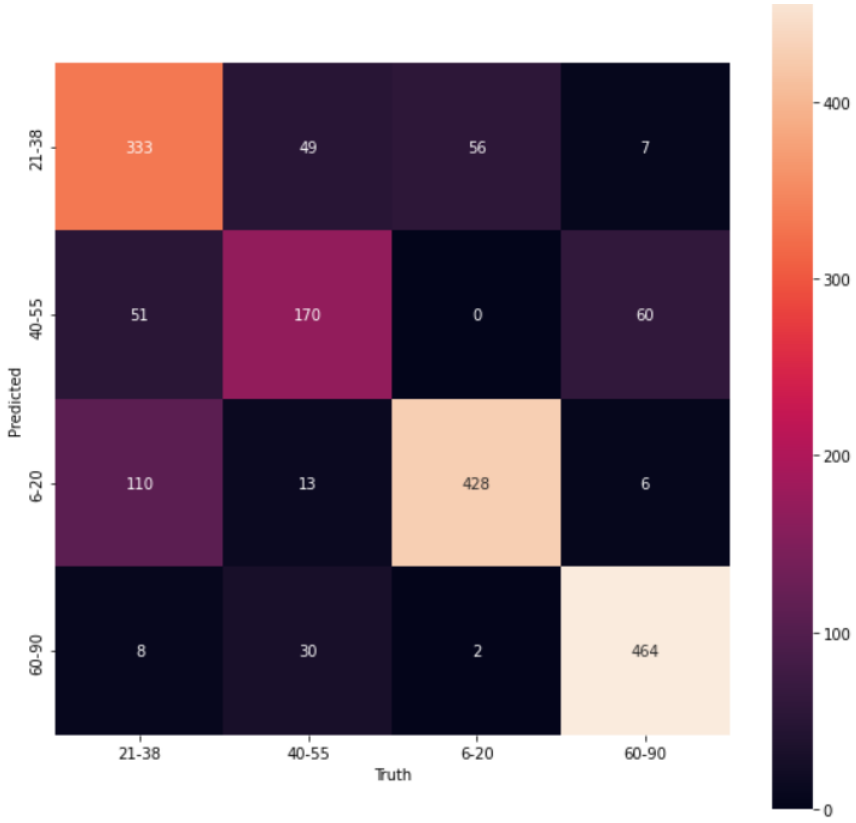
```
Epoch 12: val_loss improved from 1.12688 to 1.03149, saving model to Model_tuoi.h5
28/28 [==============================] - 120s 4s/step - loss: 0.0427 - accuracy: 0.9846 - val_loss: 1.0315 - val_accuracy: 0.8002
```

Confusion matrix, training curve, precision, recall, f1 score, support vector machine được dùng để đánh giá mô hình. Tuy nhiên thì độ chính xác "accuracy" là công cụ đánh giá chuẩn xác và nhanh nhất.



Classification Report

	precision	recall	f1-score	support
21-38	0.75	0.66	0.70	502
40-55	0.60	0.65	0.63	262
6-20	0.77	0.88	0.82	486
60-90	0.92	0.86	0.89	537
accuracy			0.78	1787
macro avg	0.76	0.76	0.76	1787
weighted avg	0.78	0.78	0.78	1787



3.3.2. Tập "FER-2013 "

Đồng bộ lại kích thước tập train và test là 96x96:

```
class_name = ['angry','disgust','fear','happy','neutral','sad','surpris
e']
path_train = '/content/camxuc/train/'
path_test = '/content/camxuc/test/'
target_name = {'angry': 0, 'disgust': 1,'fear': 2, 'happy': 3,'neutral'
:4,'sad': 5,'surprise':6}
X, y = [], []
for emotion in class_name:
    for img_name in os.listdir(path_train+emotion):
```

```
        img = cv2.imread(path_train + emotion +'/'+ img_name)
        scalled_raw_img = cv2.resize(img, (96, 96))
        X.append(scalled_raw_img)
        y.append(target_name[emotion])

for emotion in class_name:
    for img_name in os.listdir(path_test+emotion):
        img = cv2.imread(path_test + emotion +'/'+ img_name)
        scalled_raw_img = cv2.resize(img, (96, 96))
        X.append(scalled_raw_img)
        y.append(target_name[emotion])
```

Tăng cường dữ liệu giúp khắc phục vấn đề "không đủ dữ liệu", khắc phục các điểm overfitting và giúp cho mô hình hoạt động tốt hơn với đa dạng mẫu dữ liệu. Data Augmentation (Tăng cường dữ liệu) là một kỹ thuật được sử dụng để mở rộng kích thước của tập huấn luyện bằng cách tạo thêm dữ liệu đã được sửa đổi từ dữ liệu ban đầu. Trong project này, các phương thức được sử dụng bao gồm:

horizontal_flip, vertical_flip, rescale=1./255, validation_split=0.2, brightness_range= [1.1, 1.5], zoom_range=0.2, width_shift_range = 0.1, height_shift_range = 0.1.

```
train_datagen = ImageDataGenerator(width_shift_range = 0.1,
                                   height_shift_range = 0.1,
                                   horizontal_flip = True,
                                   rescale = 1./255,
                                   zoom_range = 0.2,
                                   validation_split = 0.2)
validation_datagen = ImageDataGenerator(rescale = 1./255, validation_sp
lit = 0.2)
```

Sử dụng fow_from_directory:

```
train_generator = train_datagen.flow_from_directory(directory = "/conte
nt/camxuc/train",
target_size = (96,96),
batch_size = 64,
color_mode = "rgb",
class_mode = "categorical",
subset = "training")
validation_generator = validation_datagen.flow_from_directory( director
y = "/content/camxuc/test",
```

```
target_size = (96,96),
batch_size = 64,
color_mode = "rgb",
class_mode = "categorical",
subset = "validation")
```

Trong đó:

- **directory**: phải đặt đường dẫn có các classes của folder.
- **target_size**: là size của các ảnh input đầu vào, mỗi ảnh sẽ được resized theo kích thước này.
- **color_mode**: Nếu hình ảnh là màu đen và màu trắng hoặci là grayscale thì set "grayscale" hoặc nếu nó gồm 3 channels thì set "rgb"
- **batch_size** : Số lượng ảnh được yielded từ generator cho mỗi lô batch.
- **class_mode** : set "binary" nếu có 2 classes để dự đoán, nếu không thì set "categorical". trong trường hợp nếu lập trình một hệ thống tự động Autoencoder, thì cả input và output đều là ảnh, trong trường hợp này thì set là input.
- **shuffle**: set True nếu bạn muốn đổi thứ tự hình ảnh, ngược lại set False.
- **seed** : Random seed để áp dụng tăng hình ảnh ngẫu nhiên và xáo trộn thứ tự của hình ảnh.

Xây dựng model:

```
model = tf.keras.Sequential([
        tf.keras.layers.Conv2D(32,(3,3),activation = 'relu',input_shape
=(96,96,3)),
        tf.keras.layers.Conv2D(64,(3,3), padding='same', activation='re
lu' ),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Dropout(0.25),

        tf.keras.layers.Conv2D(128,(5,5), padding='same', activation='r
elu'),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.MaxPooling2D(2,2),
```

```
        tf.keras.layers.Dropout(0.25),


        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(256, activation='relu'),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Dropout(0.25),

        tf.keras.layers.Dense(7, activation='softmax')])

optimiser = tf.keras.optimizers.Adam(lr=0.0001)
model.compile(optimizer=optimiser,loss='categorical_crossentropy',metri
cs=['accuracy'])

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 94, 94, 32)        896

conv2d_1 (Conv2D)            (None, 94, 94, 64)        18496

batch_normalization (BatchN  (None, 94, 94, 64)        256
ormalization)

max_pooling2d (MaxPooling2D  (None, 47, 47, 64)        0
)

dropout (Dropout)            (None, 47, 47, 64)        0

conv2d_2 (Conv2D)            (None, 47, 47, 128)       204928

batch_normalization_1 (Batc  (None, 47, 47, 128)       512
hNormalization)

max_pooling2d_1 (MaxPooling  (None, 23, 23, 128)       0
2D)

dropout_1 (Dropout)          (None, 23, 23, 128)       0

flatten (Flatten)            (None, 67712)             0

dense (Dense)                (None, 256)               17334528

batch_normalization_2 (Batc  (None, 256)               1024
hNormalization)

dropout_2 (Dropout)          (None, 256)               0

dense_1 (Dense)              (None, 7)                 1799

=================================================================
Total params: 17,562,439
Trainable params: 17,561,543
Non-trainable params: 896
```

Sử dụng EarlyStopping để lưu lại model tốt nhất:

```
my_calls = [EarlyStopping(monitor="val_accuracy",patience=8),
            ModelCheckpoint("Model_camxuc.h5",verbose= 1 ,save_best_onl
y=True)]
```

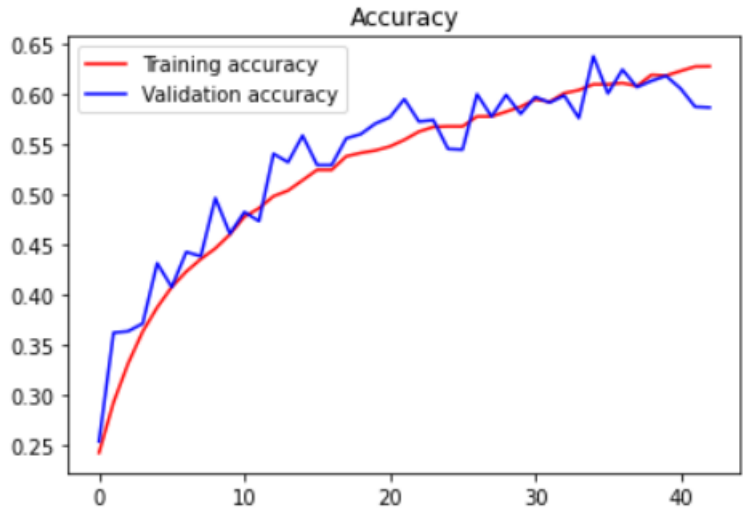
Tiến hành train model:

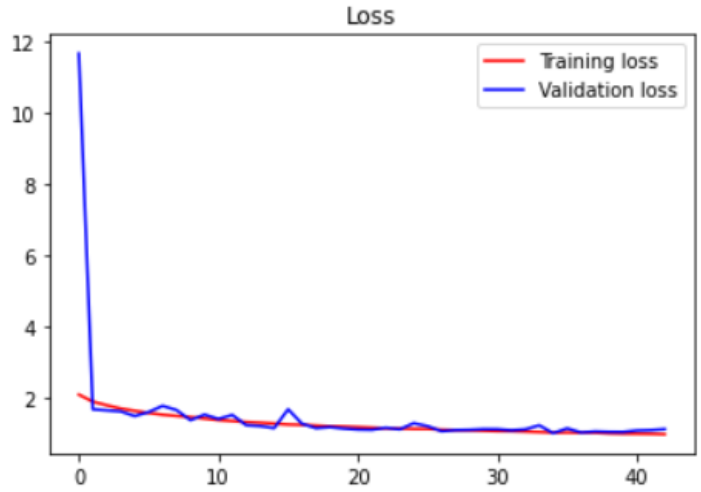
```
history = model.fit(train_generator,epochs = 50,validation_data = valid
ation_generator,callbacks=my_calls)
```

Model tốt nhất được lưu lại:

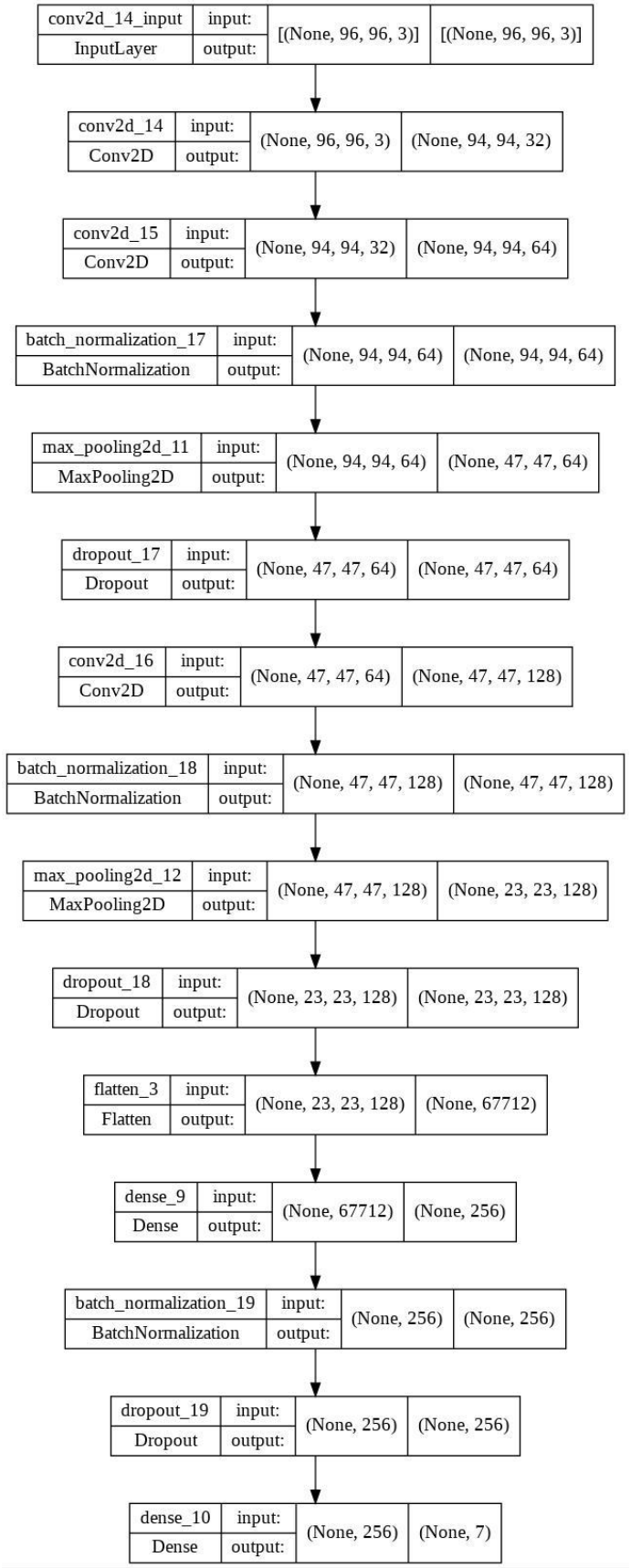
```
Epoch 35: val_loss improved from 1.08196 to 1.02823, saving model to Model_camxuc.h5
359/359 [==============================] - 62s 174ms/step - loss: 1.0429 - accuracy: 0.6089 - val_loss: 1.0282 - val_accuracy: 0.6369
```

Dùng confusion matrix, training curve, precision, recall, f1 score, accuracy để đánh giá mô hình:





Lưu đồ:



3.3.3. Tập "Gender Classification Dataset"

Tăng cường dữ liệu và sử dụng flow_from_directory:

```
train_datagen = ImageDataGenerator(rescale=1./255,
                 shear_range = 0.2,
                 zoom_range = 0.2,
                 width_shift_range = 0.2,
                 height_shift_range = 0.2,
                 fill_mode="nearest",
                 validation_split=0.15)

test_datagen = ImageDataGenerator(rescale=1./255)


trainds = train_datagen.flow_from_directory("/content/gioitinh/Training
",
                                            target_size=(96,96),
                                             seed=123,
                                             batch_size=32,
                                             subset="training")
valds = train_datagen.flow_from_directory("/content/gioitinh/Training",
                                            target_size=(96,96),
                                             seed=123,
                                             batch_size=32,
                                             subset="validation")
testds = test_datagen.flow_from_directory("/content/gioitinh/Validation
",
                                          target_size=(96,96),
                                          seed=123,
                                          batch_size=32,
                                          shuffle=False)
```

Xây dựng model:

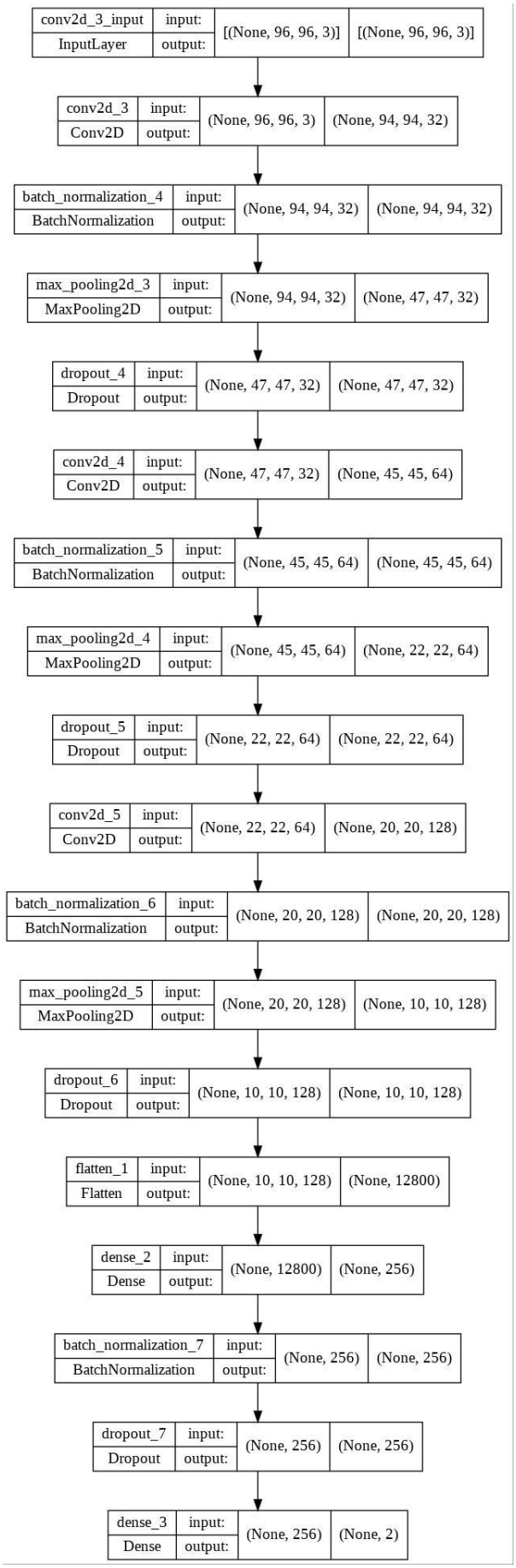
```
model = Sequential()
model.add(Conv2D(32,(3, 3), activation='relu', input_shape=(96,96,3)))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.2))

model.add(Conv2D(64,(3,3), activation='relu' ))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.2))

model.add(Conv2D(128,(3,3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.2))
model.add(Flatten())
```

```
model.add(Dense(256,activation = 'relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))

model.add(Dense(2, activation='sigmoid'))
model.compile(optimizer="adam",loss="categorical_crossentropy",metrics=
["accuracy"])
model.summary()
```

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 94, 94, 32)        896

batch_normalization_4 (Batc  (None, 94, 94, 32)        128
hNormalization)

max_pooling2d_3 (MaxPooling  (None, 47, 47, 32)        0
2D)

dropout_4 (Dropout)          (None, 47, 47, 32)        0

conv2d_4 (Conv2D)            (None, 45, 45, 64)        18496

batch_normalization_5 (Batc  (None, 45, 45, 64)        256
hNormalization)

max_pooling2d_4 (MaxPooling  (None, 22, 22, 64)        0
2D)

dropout_5 (Dropout)          (None, 22, 22, 64)        0

conv2d_5 (Conv2D)            (None, 20, 20, 128)       73856

batch_normalization_6 (Batc  (None, 20, 20, 128)       512
hNormalization)

max_pooling2d_5 (MaxPooling  (None, 10, 10, 128)       0
2D)

dropout_6 (Dropout)          (None, 10, 10, 128)       0

flatten_1 (Flatten)          (None, 12800)             0

dense_2 (Dense)              (None, 256)               3277056

batch_normalization_7 (Batc  (None, 256)               1024
hNormalization)

dropout_7 (Dropout)          (None, 256)               0

dense_3 (Dense)              (None, 2)                 514

=================================================================
Total params: 3,372,738
Trainable params: 3,371,778
Non-trainable params: 960
_________________________________________________________________
```

Tạo lưu đồ:

```
tf.keras.utils.plot_model(model, 'model_gioitinh.jpg', show_shapes=True)
```


conv2d_3_input | InputLayer | input: | output: | [(None, 96, 96, 3)] | [(None, 96, 96, 3)]
conv2d_3 | Conv2D | input: | output: | (None, 96, 96, 3) | (None, 94, 94, 32)
batch_normalization_4 | BatchNormalization | input: | output: | (None, 94, 94, 32) | (None, 94, 94, 32)
max_pooling2d_3 | MaxPooling2D | input: | output: | (None, 94, 94, 32) | (None, 47, 47, 32)
dropout_4 | Dropout | input: | output: | (None, 47, 47, 32) | (None, 47, 47, 32)
conv2d_4 | Conv2D | input: | output: | (None, 47, 47, 32) | (None, 45, 45, 64)
batch_normalization_5 | BatchNormalization | input: | output: | (None, 45, 45, 64) | (None, 45, 45, 64)
max_pooling2d_4 | MaxPooling2D | input: | output: | (None, 45, 45, 64) | (None, 22, 22, 64)
dropout_5 | Dropout | input: | output: | (None, 22, 22, 64) | (None, 22, 22, 64)
conv2d_5 | Conv2D | input: | output: | (None, 22, 22, 64) | (None, 20, 20, 128)
batch_normalization_6 | BatchNormalization | input: | output: | (None, 20, 20, 128) | (None, 20, 20, 128)
max_pooling2d_5 | MaxPooling2D | input: | output: | (None, 20, 20, 128) | (None, 10, 10, 128)
dropout_6 | Dropout | input: | output: | (None, 10, 10, 128) | (None, 10, 10, 128)
flatten_1 | Flatten | input: | output: | (None, 10, 10, 128) | (None, 12800)
dense_2 | Dense | input: | output: | (None, 12800) | (None, 256)
batch_normalization_7 | BatchNormalization | input: | output: | (None, 256) | (None, 256)
dropout_7 | Dropout | input: | output: | (None, 256) | (None, 256)
dense_3 | Dense | input: | output: | (None, 256) | (None, 2)

Sử dụng EarlyStopping để lưu lại model tốt nhất:

```
ModelCheckpoint("Model_gioitinh.h5",verbose= 1 ,save_best_only=True)]
```

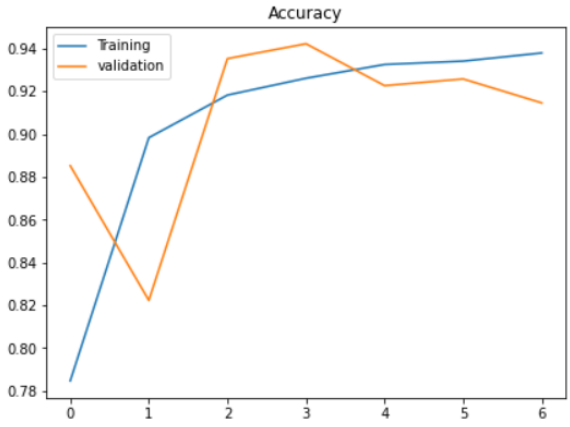
Tiến hành train model:

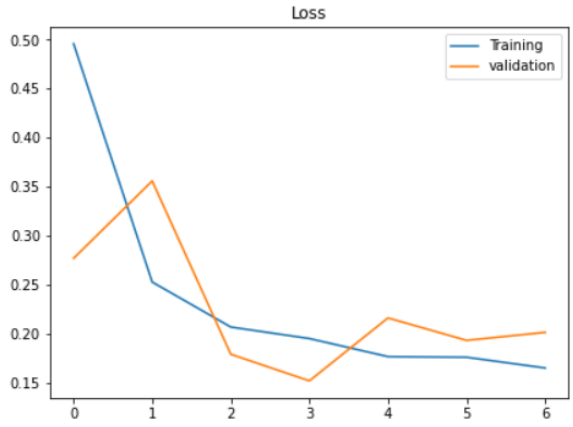
```
hist = model.fit(trainds,epochs=22,validation_data=valds,callbacks=my_c
alls)
```

Model tốt nhất được lưu lại:

```
Epoch 4: val_loss improved from 0.17914 to 0.15202, saving model to Model_gioitinh.h5
1249/1249 [==============================] - 191s 153ms/step - loss: 0.1952 - accuracy: 0.9261 - val_loss: 0.1520 - val_accuracy: 0.9423
```

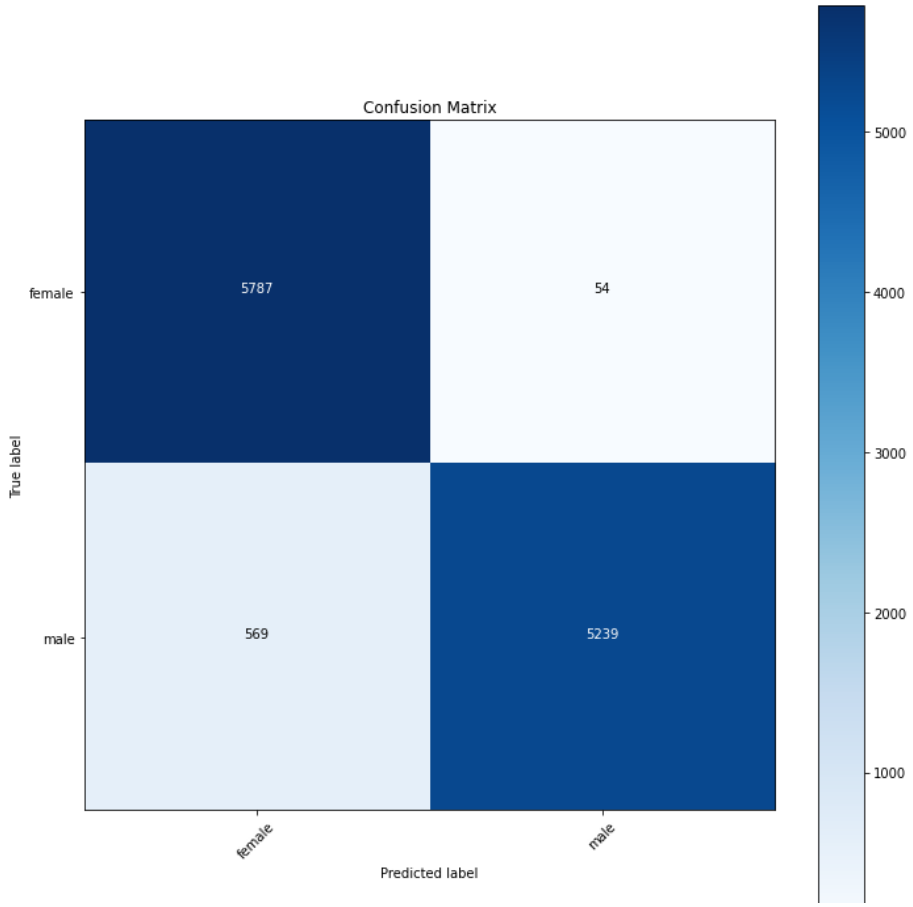
Dùng confusion matrix, training curve, precision, recall, f1 score, accuracy để đánh giá mô hình:





Classification Report

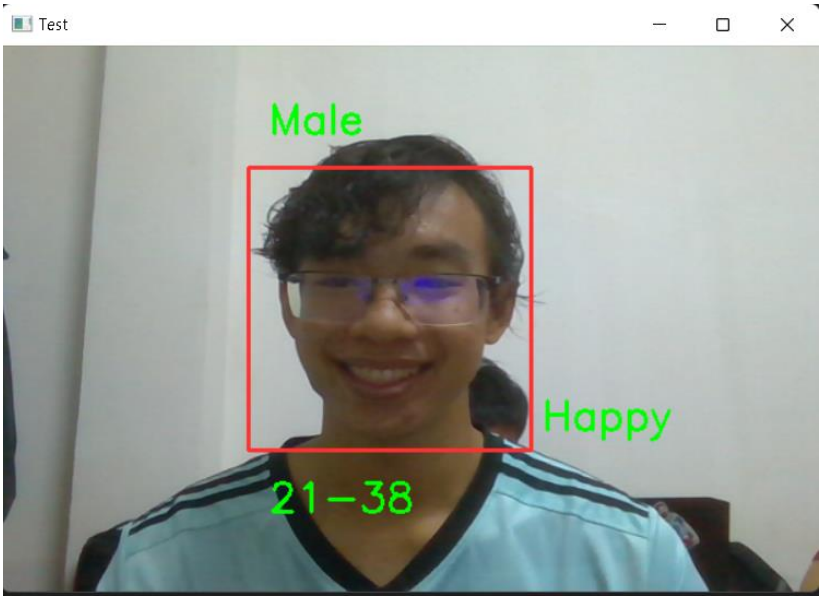
	precision	recall	f1-score	support
female	0.91	0.99	0.95	5841
male	0.99	0.90	0.94	5808
accuracy			0.95	11649
macro avg	0.95	0.95	0.95	11649
weighted avg	0.95	0.95	0.95	11649


Confusion Matrix
5787
54
569
5239
female
male
True label
female
male
Predicted label
5000
4000
3000
2000
1000

4 CHƯƠNG 4. CHẠY THỰC NGHIỆM ĐỀ TÀI

4.1 Chạy thực nghiệm trên Visual Studio Code

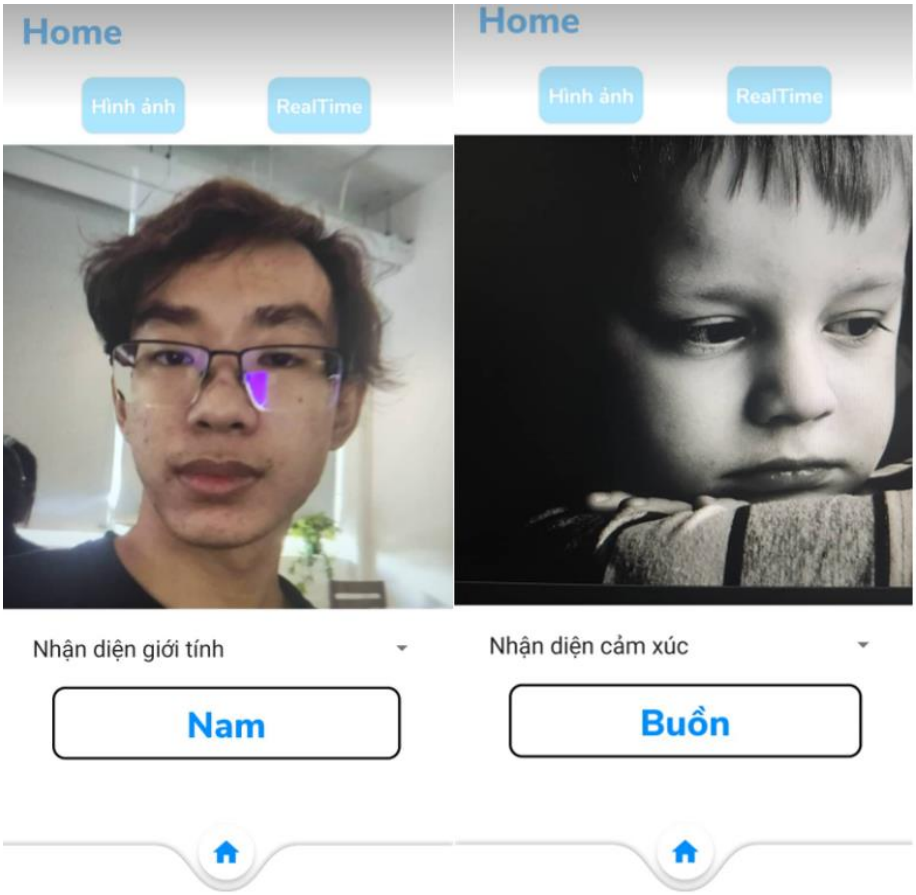
Để trích chọn đặc trưng cho mỗi khuôn mặt, trước tiên ta cần tìm ra vị trí khuôn mặt trong bức hình. Vì bộ dữ liệu sẽ bao gồm nhiều ảnh có điều kiện ánh sáng cũng như các góc độ của khuôn mặt khác nhau, chính vì vậy việc lựa chọn face detector cũng rất quan trọng để đảm hiệu quả cao nhất cho hệ thống. Vì vậy việc sử dụng haarcascade xml để phát hiện khuôn mặt là điều cần thiết.



Hình 12. Phát hiện khuôn mặt với Haar cascade

4.2 Chạy thực nghiệm mô hình trên app android

Để mô hình có thể sử dụng một cách thuận lợi đối với mọi người thì nó được triển khai trên ứng dụng điện thoại. Ứng dụng điện thoại được hướng đến ở đây là ứng dụng điện thoại Android được viết trên ngôn ngữ JAVA. Hiện tại TensorFlow (thư viện mã nguồn mở cho machine learning) có phiên bản hỗ trợ triển khai mô hình học máy lên thiết bị Android.



Hình 13. Chạy thực nghiệm trên app

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết luận

Sau khi triển khai, khả năng dự đoán giới tính khá tốt khi đa phần là đúng, khả năng dự đoán tuổi tương đối tốt khi vẫn dự đoán được với tỉ lệ khá cao, còn dự đoán cảm xúc là chỉ ở mức tạm ổn khi độ chính xác không cao, dự đoán sai tương đối nhiều và cần được cải thiện nhiều trong tương lai.

Các kết quả đạt được đã đáp ứng được mục tiêu đề ra. Tuy nhiên, kết quả này vẫn chưa thực sự tốt do còn nhiều điểm thiếu sót khi độ chính xác của mô hình dự đoán cảm xúc chỉ là 50%. Nguyên nhân là do mạng CNN và việc xử lý dữ liệu đầu vào chưa đủ tốt. Điều này dẫn đến app android hoạt động chưa thực sự đáp ứng được yêu cầu khi dự đoán.

Hướng phát triển

Từ những thiếu sót trên đã nhận thấy được rằng đề tài cần được tiếp tục cải thiện nhiều trong tương lai.

Định hướng phát triển cho đề tài: Sàng lọc dataset, thu thập thêm dữ liệu phù hợp, tạo nhiều lớp hơn, cải thiện mạng CNN, thử các phương pháp xử lý dữ liệu khác, nâng cấp giao diện và chức năng trên ứng dụng để nhiều người có thể tiếp cận.

Tài liệu tham khảo

1. “ Tổng quan về AI- Artifical intelligence”, Charlie, https://insights.magestore.com/posts/tong-quan-ai-artificial-intelligence
2. https://nordiccoder.com/blog/deep-learning-la-gi/
3. https://hocdauthau.com/convolutional-neural-network-la-gi-cach-chon-tham-so-cho-convolutional-neural-network-chuan-chinh/
4. https://phamdinhkhanh.github.io/2020/02/17/ImbalancedData.html
5. 6. 7. 8. 9. https://phamdinhkhanh.github.io/2020/08/13/ModelMetric.html
10. Sách “Machine Learning cơ bản” – Vũ Hữu Tiệp
11. https://phamdinhkhanh.github.io/2020/08/13/ModelMetric.html
12. https://machinelearningcoban.com/2018/07/06/deeplearning/
13. https://wiki.tino.org/tensorflow-la-gi/
14. https://teky.edu.vn/blog/opencv-la-gi/