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

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*



**BÁO CÁO ĐỒ ÁN 2**

**NHẬN DIỆN KHUÔN MẶT CỦA 10 ĐỐI TƯỢNG TỰ CHỌN BẰNG MẠNG CNN**

SVTH: HOÀNG VIẾT HƯNG

MSSV: 18119082

Khóa: 18

Ngành: CÔNG NGHỆ KĨ THUẬT MÁY TÍNH

GVHD: THS. HUỲNH HOÀNG HÀ

Tp Hồ Chí Minh --- tháng 11, năm 2022

|  |  |
| --- | --- |
| LOGO CLC.JPG | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM  Độc lập – Tự do – Hạnh phúc  ----\*\*\*---- |

Tp. Hồ Chí Minh, ngày 27 tháng 11 năm 2021

**NHIỆM VỤ ĐỒ ÁN MÔN HỌC**

|  |  |  |  |
| --- | --- | --- | --- |
| Họ và tên sinh viên: Hoàng Viết Hưng | | MSSV: 18119082 | |
| Ngành: Công nghệ kỹ thuật máy tính | | Lớp: 18119CL3B | |
| Giảng viên hướng dẫn: Huỳnh Hoàng Hà | | ĐT: 0866507210 | |
| Ngày nhận đề tài: 22/09/2021 | | Ngày nộp đề tài: 27/11/2021 | |
|  | |  | |
| 1. Tên đề tài : Nhận diện khuôn mặt của 10 đối tượng tự chọn bằng CNN | |  | |
| 2. Các số liệu, tài liệu ban đầu: Các kiến thức liên quan đến AI, mạng học sâu. Kiến về Computer Vision, Am hiểu các kiến trúc mạng CNN đặc biệt là mạng VGG. | |  | |
| 1. Nội dung thực hiện đề tài:  * Tóm tắt lý thuyết về AI, Deep Learning, Computer Vision * Ứng dụng phương pháp xử lý ảnh, tạo tập datasets * Xây dựng một mạng CNN để trích xuất và nhận diện đối tượng * Huấn luyện tập dữ liệu và cải thiện độ chính xác của mô hình * Tạo giao diện ứng dụng cho việc dữ đoán kế quả từ model đã huấn luyện * Kiếm tra toàn hệ thống * Viết báo cáo trình bày kết quả | |  | |
| 1. Sản phẩm:   Hệ thống nhận diện khuôn mặt | |  | |
|  | | GIẢNG VIÊN HƯỚNG DẪN | |
| LOGO CLC.JPG | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM  Độc lập – Tự do – Hạnh phúc  ----\*\*\*---- |

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Họ và tên Sinh viên: Hoàng Viết Hưng MSSV: 18119082

Ngành: Công nghệ kỹ thuật Máy tính

Tên đề tài: Nhận diện khuôn mặt của 10 đối tượng tự chọn bằng CNN

Họ và tên Giáo viên hướng dẫn: Huỳnh Hoàng Hà

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện:

1. Ưu điểm:

1. Khuyết điểm:

1. Đề nghị cho bảo vệ hay không?

1. Đánh giá loại:

1. Điểm:……………….(Bằng chữ: )

Tp*. Hồ Chí Minh, ngày 27 tháng 11 năm 2021*

Giáo viên hướng dẫn

*(Ký & ghi rõ họ tên)*

|  |  |
| --- | --- |
| Logo CLC | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM  Độc lập – Tự do – Hạnh phúc  ----\*\*\*---- |

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Họ và tên Sinh viên: Hoàng Viết Hưng MSSV: 18119082

Ngành: Công nghệ kỹ thuật Máy tính

Tên đề tài: Nhận diện khuôn mặt của 10 đối tượng tự chọn bằng CNN

Họ và tên Giáo viên phản biện: Huỳnh Hoàng Hà

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện:

1. Ưu điểm:

1. Khuyết điểm:

1. Đề nghị cho bảo vệ hay không?

1. Đánh giá loại:

1. Điểm:……………….(Bằng chữ: )

Tp*. Hồ Chí Minh, ngày tháng năm 2021*

Giáo viên phản biện

*(Ký & ghi rõ họ tên)*

**MỤC LỤC**

**NHIỆM VỤ ĐỒ ÁN MÔN HỌC 2**

**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 3**

**[PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN 4](#_Toc21791)**

**DANH MỤC CÁC TỪ VIẾT TẮT 7**

**DANH MỤC CÁC BẢNG BIỂU 8**

**DANH SÁCH CÁC HÌNH ẢNH, BIỂU ĐỒ 9**

**[CHƯƠNG 1 1](#_Toc3166)0**

**[TỔNG QUAN 1](#_Toc14653)0**

[1.1. GIỚI THIỆU ĐỀ TÀI. 1](#_Toc6913)0

[1.2. LÝ DO CHỌN ĐỀ TÀI. 1](#_Toc23553)0

[1.3. MỤC TIÊU VÀ NHIỆM VỤ ĐỀ TÀI. 1](#_Toc18377)1

[1.4. GIỚI HẠN ĐỂ TÀI 1](#_Toc4868)1

[1.5. PHẦN MỀM SỬ DỤNG. 1](#_Toc29900)2

1.6. BỐ CỤC 12

**[CHƯƠNG 2 1](#_Toc30584)3**

**CƠ SỞ LÝ THUYẾT 13**

[2.1. CÁC KHÁI NIỆM CƠ BẢN VỀ COMPUTER VISION, FACE DETECTION, FACE RECOGNITION.](#_Toc24604) **[1](#_Toc24604)3**

[2.1.1. Lý thuyết về Computer Vision. 1](#_Toc23078)3

[2.1.2. Lý thuyết về Face Detection. 1](#_Toc32343)4

2.1.3. Lý thuyết về Face Recognition 15

[2.2. MỘT SỐ PHƯƠNG PHÁP VỀ TIỀN XỬ LÝ ẢNH. 1](#_Toc4167)6

[2.2.1. Khái niệm về xử lý ảnh. 1](#_Toc32412)6

[2.2.2. Phương pháp xử lý ảnh trên miền không gian. 1](#_Toc10273)6

[2.3. PHÁT HIỆN KHUÔN MẶT BẰNG PHƯƠNG PHÁP HAAR-CASCADE. 20](#_Toc27441)

[2.3.1. Khái niệm Haar-Cascade 2](#_Toc19348)0

[2.3.2. Nguyên lý hoạt động 2](#_Toc13412)0

[2.4. KIẾN TRÚC MẠNG CNN. 2](#_Toc2037)2

[2.4.1. Khái niệm. 2](#_Toc8047)2

[2.4.2. Quá trình phát triển trong kiến trúc mạng CNN. 2](#_Toc11759)3

[2.4.3. Các lớp phổ biến trong mạng CNN 2](#_Toc8539)4

[2.5. KIẾN TRÚC VỀ MẠNG VGG-FACE (VGG-16). 2](#_Toc20801)6

[2.5.1. Giới thiệu. 2](#_Toc11842)6

[2.5.2. Kiến trúc mạng VGG-16. 2](#_Toc28560)7

2.5.3. Một số lỗi thường gặp trong quá trình huấn luyện dữ liệu. 28

**[CHƯƠNG 3 3](#_Toc17274)0**

**[THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG 3](#_Toc9018)0**

[3.1. TIẾN TRÌNH THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG. 3](#_Toc9788)0

[3.2. SƠ ĐỒ GIẢI THUẬT. 3](#_Toc14629)5

**[CHƯƠNG 4](#_Toc19972) 37**

**[KẾT QUẢ, NHẬN XÉT VÀ ĐÁNH GIÁ](#_Toc17909) 37**

[4.1. KẾT QUẢ](#_Toc6899) 37

[4.2. NHẬN XÉT VÀ ĐÁNH GIÁ.](#_Toc6265) 41

**[CHƯƠNG 5](#_Toc20888) 42**

**[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN](#_Toc23451) 42**

[5.1. KẾT LUẬN](#_Toc32289) 42

[5.2. HƯỚNG PHÁT TRIỂN](#_Toc29395) 43

**PHỤ LỤC 44**

**[TÀI LIỆU THAM KHẢO](#_Toc26144) 45**

**DANH MỤC CÁC TỪ VIẾT TẮT**

AI: Artificial Intellgence

CNN: Convolution Neural Netwo

VGG: Visual Geometry Group

VGG-Face: Visual Geometry Group-Face of Convolution Neural Network

VGG-16: Visual Geometry Group, Convolution Neural Network with 16 Convolution Layers

Open-CV: Open Source Computer Vision Library

1. CNN: Region with Convolution Neural Network

SSD: Single Shot Detection of Convolution Neural Network

XLA: Xử Lý Ảnh

HOG: Histogram of Oriented Gradients

SIFT: Scale-Invariant Feature Transform

Conv-Net: Convolution Network

Res-Net: Residual Neural Network

GPU: Graphics Processing Unit

CPU: Central Processing Unit

ReLU: Rectified Linear Unit

Resnet-50: Residual Neural Network with 50 Convolution Layers

Senet-50: Scientific and Engineering Network with 50 Convolution Layers

**DANH MỤC CÁC BẢNG BIỂU**

**DANH SÁCH CÁC HÌNH ẢNH, BIỂU ĐỒ**

Hình 2.1. Công thức tích chập 16

Hình 2.2. Công thức tích chập 2 chiều 17

Hình 2.3. Mô phỏng quá trình tích chập 1 17

Hình 2.4. Mô phỏng quá trình tích chập 2 18

Hình 2.5. Bộ lọc trung bình 18

Hình 2.6. Hiệu chỉnh giá trị kết quả từ bộ lọc trung bình 19

Hình 2.7. Công thức tính vị trí 19

Hình 2.8. Bộ lọc Laplace 20

Hình 2.9. Các bộ lọc Haar-Like phổ biến 20

Hình 2.10. Mô phỏng hoạt động của bộ lọc Haar\_Cascade 21

Hình 2.11. Hình ảnh minh họa phương pháp Adaboost 22

Hình 2.12. Tiến trình phát triển CNN 23

Hình 2.13. Công thức 24

Hình 2.14. Mô tả quá trình tính toán của một lớp tích chập 25

Hình 2.15. Mô tả hoạt động của MaxPooling 25

Hình 2.16. Mô tả hoạt động của AveragePooling 26

Hình 2.17. Mô phỏng lớp kết nối đầy đủ 26

Hình 2.18. Các kiến trúc mạng VGG 27

Hình 2.19. Hiện tượng Overfitting 28

Hình 2.20. Hiện tượng Underfitting 29

Hình 3.1. Tiến trình xây dựng và thiết kế hệ thống 30

Hình 3.2. Các lớp trong model nhận diện đối tượng 33

Hình 3.3. Giải thuật nhận diện khuôn mặt 35

Hình 4.1. Các epoch cuối cùng của quá trình huấn luyện 37

Hình 4.2. Giao diện ứng dụng 38

Hình 4.3. Kết quả test 1 38

Hình 4.4. Kết quả test 2 39

Hình 4.5. Kết quả test 3 39

Hình 4.6. Kết quả nhận diện sai 1 40

Hình 4.7. Kết quả nhận diện sai 2 40

CHƯƠNG 1

**TỔNG QUAN**

* 1. GIỚI THIỆU.

Đồ án môn học một là bước đệm cho các kiến thức cơ bản về mạch điện, điện tử cơ bản, kỹ thuật số và một số ngôn ngữ lập trình cơ bản như C, C++. Kế tiếp đó, đồ án môn học thứ hai sẽ kiểm tra về các kiến thức liên quan đến các lĩnh vực như Iots, hệ thống nhúng, vi mạch, lập trình di động, web hay AI. Vậy nên trong đồ án môn học thứ hai này, em sẽ xây dựng một hệ thống ứng dụng AI, cũng có thế nói AI sẽ là nền tảng cho hướng đi sau này của em.

Với sự phát triển của khoa học hiện đại, AI được ứng dụng khá phổ biến, từ lĩnh vực y học với các thành tựu về các thiết bị chuẩn đoán bệnh ung thư, hay lĩnh vực giao thông với thành tựu những chiếc ôtô tự hành, hay các camera giám xác được đặt khắp nơi đền đường phố. Hoặc là đến cả những chiếc SmartPhone mà bản thân mỗi người đều có, chúng được những nhà sản xuất kết hợp ứng dụng AI vào việc bảo mật như nhận diện dấu, hay nhận diện khuôn mặt, hay đến các trợ lý ảo chatbox để người dùng có thể giao tiếp và sử dụng dể dàng hơn. Từ các ứng dụng trên, em đã chọn việc xây dựng một ứng dụng nhận diện khuôn mặt để làm để tài cho việc thực hiện đồ án hai. Tuy nhiên ở ứng dụng lần này, sẽ được giới hạn số lượng khuôn mặt được dùng để nhận diện sau cho phù hợp với vai trò của đồ án hai cũng như phù hợp với bộ xử lý của máy tính.

* 1. LÝ DO CHỌN ĐỀ TÀI.

Đề tài được chọn dựa trên lĩnh vực mà sau này bản thân em định theo đuổi và làm việc. Hiện tại AI là một lĩnh vực trọng yếu và được các nhà đầu tư tập trung đào tạo nhân viên cũng như sản xuất ra các ứng dụng có tích hợp AI ở trong đó. Vậy nên AI sẽ là hướng đi phù hợp và có tính phát triển mạnh mẽ trong hơn chục năm nữa.

Nhận diện khuôn mặt luôn có một vai trò quan trọng trong các ứng dụng về AI như các ứng dụng về hệ thống bảo mật, chỉnh sửa hình ảnh, chuẩn đoán bệnh thông qua khuôn mặt, nhận diện cảm xúc,… Nhận diên khuôn mặt là một trong những lĩnh vực trong việc nghiên cứu của Computer Vision. Với Computer Vision ta có thể dể dàng phát hiện được khuôn mặt của con người bằng phương pháp được dùng khá phổ biến đó là phương pháp trích xuát đặt trưng (features extraction). Với phương pháp này ta có thể phân biệt được khuôn mặt của con người với các khuôn mặt của các con vật khác. Bên cạnh đó ta sẽ kết hợp phương pháp học sâu (Deep Learning) để xây dựng một kiến trúc mạng CNN nhằm phân biệt được các khuôn mặt của từng người khác nhau. Với hệ thống này, nếu được xây dựng thành công, ta có thể định danh được bất kì đối tượng nào thông qua khuôn mặt của họ mà không cần phải tìm kiếm thông tin hay thông qua giao tiếp để nhận biết.

* 1. MỤC TIÊU VÀ NHIỆM VỤ ĐỀ TÀI.

Mục tiêu để tài là thành công xây dựng được một hệ thống nhận diện khuôn mặt với các chức năng chính sau:

* Hệ thống có thể phát hiện được khuôn mặt của con người
* Hệ thống có thể nhận diện và phân loại được khuôn mặt của 10 đối tượng khác nhau.

Nhiệm vụ của để tài là:

* Tìm hiểu được các giải thuật cho việc phát hiện và nhận dạng đối tượng
* Tìm hiểu về kiến trúc của mạng CNN, đặc biệt là kiến trúc VGG-Face
* Xây dựng được một tập dữ liệu (datasets) ứng dụng thư viện OpenCV để xử lý ảnh được đưa vào.
* Xây dựng model CNN để huấn luyện cho tập dữ liệu được tạo ra, liên tiếp cải thiện model, để đưa ra độ chính xác cao hơn.
* Hoàn thành hệ thống bằng việc xây dựng giao diện ứng dụng nhận diện khuôn mặt cho kết quả dự đoán được từ model đã train sẵn.
  1. GIỚI HẠN ĐỂ TÀI.

Đề tài nhận diện khuôn mặt sẽ chỉ dừng ở mức nghiên cứu về kiến trúc mạng CNN đơn giản như mạng VGG16, là một dạng mạng thực thi theo kiến trúc đường thẳng, không áp dụng các mạng CNN có các kiến trúc theo kiểu phân nhánh hay rút gọn.

Đồng thời đối tượng nhận dạng ở đây sẽ là ngẫu nhiên nhưng chỉ dừng ở mức là 10 đối tượng, vì bộ xử lý của máy tính không phù hợp các việc huấn luyện có qua nhiều đối tượng vì điều đó làm cho bộ xử lý trong máy tính không chịu nỗi.

Đề tài sẽ giúp ta có thêm kiến trúc về mạng CNN, cách hoạt động cũng như cách xây dựng một mạng CNN như thế nào là phù hợp với đối tượng phân loại và phù hợp với tập dữ liệu tự xây dựng.

* 1. PHẦN MỀM SỬ DỤNG.

* Ngôn ngữ lập trình: Python
* Phần mềm sử dụng: Visual Code
* Môi trường lập trình: Python 3.8
* Các thư viện mã nguồn mở sẽ được tìm hiểu và áp dụng là OpenCV, Tensorflow, Keras, Tkinter.
  1. BỐ CỤC.

Báo cáo đồ án 2 sẽ được trình bày theo bố cục gồm 5 phần chính:

* **Chương 1: Tổng quan.** Giới thiệu khái quát và nêu lý do chọn đề tài, mục tiêu và nhiệm vụ chính cần hoàn thành, giới hạn nghiên cứu cho đề tài. Cuối cùng là phần mêm sử dụng trong để tài và bố cục báo cáo đồ án hai.
* **Chương 2: Cơ sở lý thuyết.** Trình bày các kiến thức cơ bản về Computer Vision, Face Detection, Face Recognition. Xây dựng các thuật toán phát hiện và nhận diện khuôn mặt.
* **Chương 3: Thiết kế và xây dựng hệ thống.** Xây dựng các chương trình xử lý ảnh, phát hiện khuôn mặt, model nhận diện khuôn mặt, giao diện kết quả.
* **Chương 4: Kết quả, nhận xét và đánh giá.** Thực thi chương trình, hiển thị kết quả đạng được, phân tích và đánh giá kết quả.
* **Chương 5: Kết luận và hướng phát triển.** Tóm lược kết quả đạt được, so sánh kết quả với yêu cầu ban đầu đã đặt ra của hệ thống. Nêu lên những ưu và nhược điểm của hệ thống, từ đó đưa ra định hướng, hướng phát triển cho bản thân.

CHƯƠNG 2

**CƠ SỞ LÝ THUYẾT**

1. CÁC KHÁI NIỆM CƠ BẢN VỀ COMPUTER VISION, FACE DETECTION, FACE RECOGNITION.
2. Lý thuyết về Computer Vision.
3. *Định nghĩa.*

Thị giác máy tính là một mảng trong lĩnh vực AI, cho phép các máy tính hoặc hệ thống tự định nghĩa được các thông tin dựa trên những hỉnh ảnh, video kỹ thuật số, …Tự đó máy tính hoặc hệ thống có thể thực hiện, phản hổi dựa trên các thông tin đã phân tích.

Thị giác máy tính khá giống với thị giác con người, tuy nhiên con người có một thời gian rất dài trong cuộc sống để quan sát và tiếp thu sự vật, sự việc biến đổi, để rồi nhận thức được đó là vật gì, đó là tình huống gì. Còn đối với máy tính những thị giác, võng mạc, hệ thống dây thần kinh được thay bằng các camera, dự liệu và các thuật toán. Quá trình học của máy tính phải tương đối nhanh hơn con người để có thể trả ra kết quả một cách nhanh và chính xác nhất.

1. *Nguyên lý hoạt động.*

Cách mà computer vision hoạt động cơ bản là dùng một lượng lớn dữ liệu được đưa vào trong máy tình, từ đó tiến hành phân tích, quá trình này sẽ lặp đi lặp lại cho đến khi nhận ra được sự khác biệt giữa các dữ liệu (bức ảnh).

Và để máy tính thực hiện được việc này cần hai yếu tố chủ yếu hỗ trợ đó là machine learning và mạng nơ tron tích chập CNN.

Với machine learning, may tính sẽ được xây dựng một thuật toán để có thể xử lý được các ảnh đầu vào, nhận biết được ngữ cảnh của các dữ liệu, phân biệt được ngữ cảnh và vật thể trong dữ liệu đầu vào. Sau đó máy tính sẽ đẩy các dữ liệu đó vào một cái mô hình để huấn luyện (hay nói cách khác là để cho máy tính “nhìn”).

Với CNN, CNN là thuật toán trong Deep Learning ra đời năm 1980, của Pháp Yan LeCun. Sau khi đưa dữ liệu vào mô hình huấn luyện, thuật toán CNN sẽ từ các dữ liệu đưa vào trích xuất ra các đặc trưng riêng biệt của từng bức ảnh, cụ thể là máy tính sẽ xem các dữ liệu thành là một tập các pixel được dán nhãn (label). Sử dụng các label để thực hiện tích chập để tự dự đoán ra đối tượng đang “nhìn” là cái gì. Các bức ảnh sẽ được đưa qua nhiều lớp như vậy trong mạng nơ tron, đưa ra được độ chính xác với các dự đoán qua nhiều lần tích chập cho đến khi dữ đoán đưa ra là chính xác.

1. *Ứng dụng.*

Thị giác máy tính là một công nghệ thông minh được ứng dụng rộng rãi có thể kể đến như dùng để Image classification, Object Detection, Object Tracking, Content-base image retrieval, automatics cars, Face Detection, Optical Character Recognition.

1. Lý thuyết về Face Detection.
2. *Định nghĩa.*

Phát hiện khuôn mặt là một ứng dụng của Computer Vision, và cũng là một phần không thể thiếu trong quá trình nhận diện khuôn mặt (Face recognition). Phát hiện khuôn mặt sẽ giúp máy tính nhận dạng được khuôn mặt của con người sau khi qua huấn luyện với tập dữ liệu rất lớn về mặt người.

1. *Nguyên lý hoạt động.*

Như đã nói các bài toán của Computer Vision là một thuật toán với đầu vào là một lượng lớn dữ liệu được sử dụng, và dữ liệu được sử dụng ở đây là các hình ảnh kỹ thuật số về khuôn mặt người. Các thuật toán phát hiện khuôn mặt luôn bắt đầu với việc tìm kiếm mắt vì nó là dể nhất sau đó sẽ tiếp tục với lông mày, mũi, miệng,... Sau khi đã kiểm tra xong các đặt điểm trên, máy tính sẽ xác nhận đó là khuôn mặt người hay không phải. Mỗi một bức ảnh ta để làm như thế để huấn luyện mô hình từ đó làm tăng thêm đọ chính xác trong việc phát hiện khuôn mặt.

Các phương pháp dùng trong phát hiện khuôn mặt là:

Như nói trên, ta sử dụng các đặt trưng như mắt mũi miệng để thực hiện nhận diện khuôn mặt, đây cũng là một cách phát hiện khuôn mặt tuy nhiên có nhược điểm là sẽ bị ảnh hưởng bởi các nhiễu và ánh sáng. Phương pháp thứ hai sẽ là phát hiện khuôn mặt dựa trên các hình ảnh khuôn mặt mẫu tiêu chuẩn cao mang đi so sánh. Nhưng cách này sẽ bị ảnh hưởng bởi các khuôn mặt có tỷ lệ khác nhau, hình dạng kì lạ, … Phương pháp thức ba, sử dụng các đặc điểm để trích xuất các đặc trưng liên quan đến hình ảnh khuôn mặt.

Một số kĩ thuật cần lưu ý trong phát hiện khuôn mặt như remove background, đây là kỹ thuật phổ biến dùng trong các bài toán xử lý ảnh, xóa đi nền sẽ giúp việc trích xuất trở nên chính xác hơn tránh có nhiễu. Kĩ thuật cần lưu ý nữa là việc phát hiện khuôn mặt trong các video, khi đó khuôn mặt sẽ chuyển động không theo quy luật rất khó phân biệt vậy nên ta cần tính toán ra vùng hoạt động của khuôn mặt để có thể cho ra kết quả tốt nhất.

Một số thuật toán Deep learning dùng để phát hiện khuôn mặt nói riêng và phát hiện vật thế nói chung là CNN, R-CNN, SSD(Single Shot Detection).

1. *Ứng dụng cụ thể.*

Phát hiện khuôn mặt được sử dụng trong nhiều lĩnh vực như bảo mật, trắc sinh học, phát luật, giải trí và thông tin cá nhân. Nó hỗ trợ nhận diện khuôn mặt, nhận diện cảm xúc, phân tích khuôn mặt,…

1. *Ưu điểm và nhược điểm.*

Ưu điểm: tăng cường bảo mật, dể dàng tích hợp với các ứng dụng thực tế, tự động nhận diện được khuôn mặt người.

Nhược điểm: dữ liệu quá lớn, phát hiện có thể không hoàn toàn chính xác bởi các loại khuôn mặt khác biệt lớn hay các góc máy ảnh, có khả năng xâm nhập quyền riêng tư của người khác.

1. Lý thuyết về Face Recognition.

Nhận diện khuôn mặt cùng phát hiện khuôn mặt luôn là đi chung với nhau để tạo ra các ứng dụng thực tế. Tuy nhiên phát hiện khuôn mặt chỉ dừng lại ở mức phát hiện được mặt của con người, còn nhận diện khuôn mặt không chỉ phát hiện được khuôn mặt của con người mà còn phân biệt, nhận diện được đây là người nào, người này có đặc điểm gì khác so với những khuôn mặt khác.

Đó chính là khác nhau cơ bản nhất giữa phát hiện và nhận diện khuôn mặt. Còn về thuật toán thì cả hai vấn để này để có cùng một cơ sở cách giải quyết, đề là dựa trên việc trích xuất các đặc điểm khuôn mặt. Tuy nhiên việc trích xuất để nhận diện được khuôn mặt sẽ phức tạp hơn, sử dụng nhiều bộ lọc phức tạp nhiều chiều để trích xuất ra được các đặc điểm phức tạp, đa chiều. Và việc trích xuất các đặc điểm đó trong Deep Learning người ta gọi là việc tạo ra các Feature Map. Các feature map càng nhiều, càng phức tạp thì độ chính xác đầu ra sẽ càng cao.

1. MỘT SỐ PHƯƠNG PHÁP VỀ TIỀN XỬ LÝ ẢNH.
2. Khái niệm về xử lý ảnh.

Xử lý ảnh (XLA) là một lĩnh vực của tin học ứng dụng, được sử dụng nhiều trong các ứng dụng về thị giác máy tính. Quá trình xử lý ảnh sẽ là việc biến đổi từ ảnh ban đầu đưa vào và lấy ra một ảnh đầu ra mới, có đặc tính mà người dùng mong muốn.

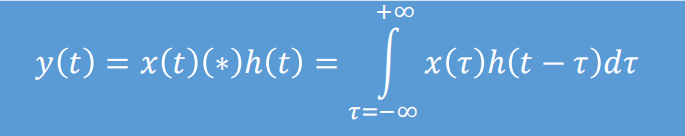
Mục đích của việc xử lý ảnh trong các ứng dụng học sâu là điều chỉnh kích thước của ảnh, thay đổi kích thước về chiều dài, chiều rộng của ảnh (từ ảnh có kích thước 700x300 chuyển về ảnh có kích thước 224x224), hay thậm chí là chiều sâu của ảnh (từ ảnh RGB chuyển sang ảnh xám). Việc chuyển đổi kích thước ảnh này là nhằm phù hợp với kiến trúc mạng học sâu được sau dựng sau này. Bên cạnh việc thay đổi kích thước của ảnh, xử lý ảnh còn giúp tăng cường chất lượng ảnh như giảm nhiễu hay tăng độ tương phản cho ảnh nhằm cải thiện tập ảnh cho quá trình huấn luyện sau này.

1. Phương pháp xử lý ảnh trên miền khuôn gian.
2. Khái niệm.

Để xử lý được ảnh trong miền không gian, ta sử dụng các bộ lọc có các đặc tính và kích thước nhất định, và thực hiện phép tích chập (Convolution) trên từng channel của ảnh.

1. Phép tính tích chập (convolutions).

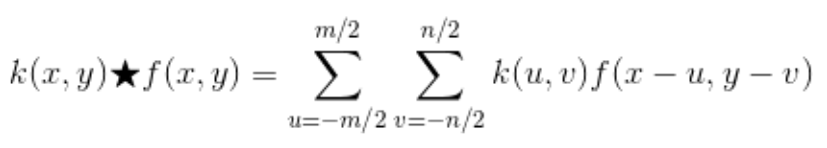
Theo toán học tích chập là kết quả của tính toán của một hàm dựa trên hai hàm có sắn. Giả sử có hai hàm số x(t) và h(t), ta sẽ thực hiện tích chập hai hàm này và cho ra kết quả y(t), với y(t) là là tập hợp các giá trị x(t) nhưng chỉ được chắc lọc lại các giá trị có đặc điểm tương ứng với các đặc điểm mà hàm h(t) có. Hay nói cách khác là hàm x(t) sau khi được sửa đổi bởi hàm h(t) sẽ cho ra hàm y(t). Ta có công thức sau:



*Hình 2.1. Công thức tích chập*

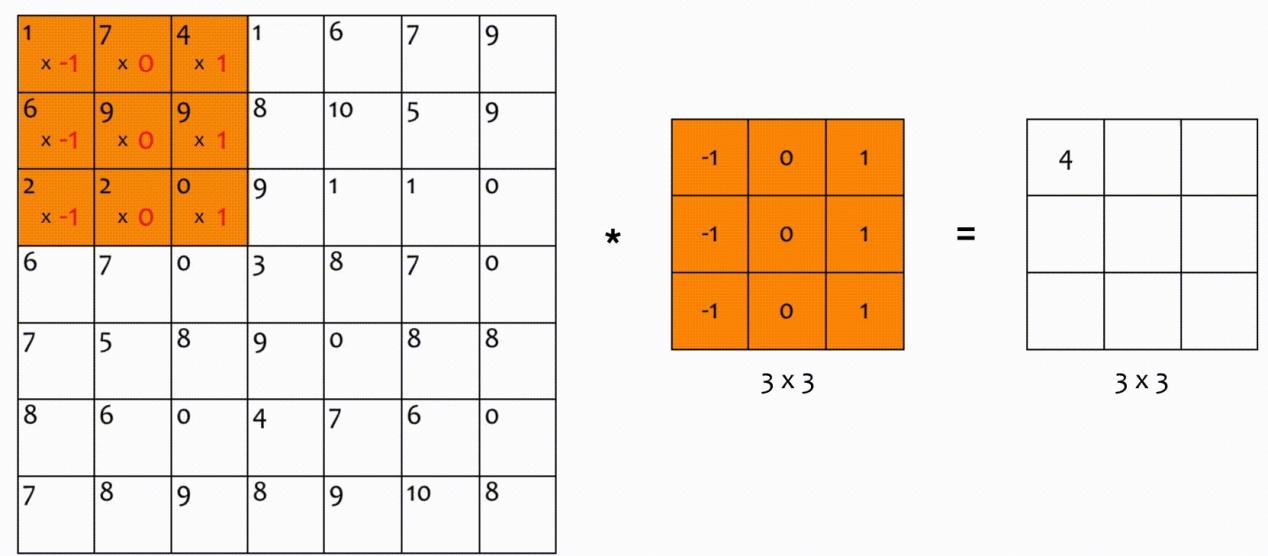
Trên công thức là việc tích chập hàm x(t) và hàm h(t) theo τ trên một khoảng là t. Để nhân chập được hai hàm trên, đầu tiên ta thay t bằng τ vào thành x(τ) và h(τ), sau đó ta đảo ngược h(τ) theo trục tung thành h(-τ). sau đó ta dịch h(-τ) phải một đoạn là t thành h(-τ + t). Cuối cùng là nhân hai hàm số h(-τ + t) và x(τ) và cho ra được kết quả của phép tính tích chập.

Tích chập trong không gian 2 chiều là: Giả sử ta có một hàm ảnh f(x, y) và bộ lọc k(x, y) (kích thước mxn), ta có công thức



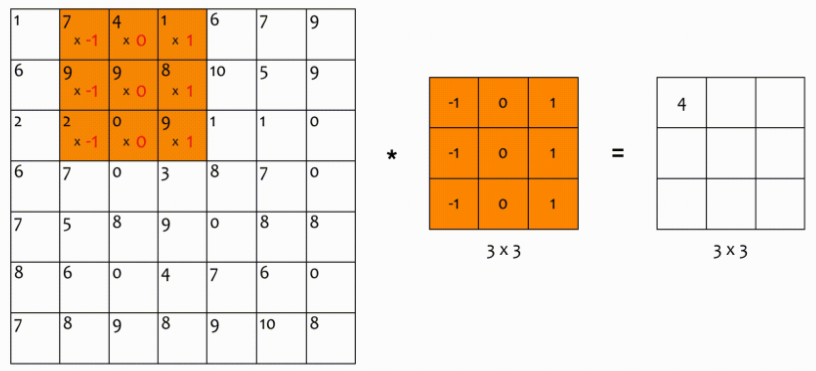
*Hình 2.2. Công thức tích chập 2 chiều*

Công thức tích chập hai chiều, được sử dụng chủ yếu trong xử lý ảnh, ta sẽ tích chập các ảnh 2D thậm chí 3D với các bộ lọc để trích xuất đặc trưng hoặc loại bỏ nhiễu trong ảnh.



*Hình 2.3. Mô phỏng quá trình tích chập 1*

Hình ảnh trên mô tả quá trình tích chập giữa một bộ lọc với một tấm ảnh. Ta có một tấm ảnh 7x7 pixels, và một bộ lọc 3x3 pixel màu cam như trên hình, ta sẽ trượt bộ lọc trên bức ảnh bắt đầu vào vị trí như trên hình. Kết quả sẽ là tổng giá trị thu được từ việc nhân giá trị từng ô trong bộ lọc với các giá trị tương ứng từng ô trên ảnh. Và kết quả ban đầu cho ra bằng 4.



0

*Hình 2.4. Mô phỏng quá trình tích chập 2*

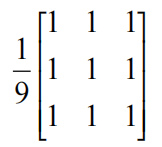
Tiếp theo ta trước bộ lọc sang trái một ô và cho ra kết qua là 0, cứ tiếp tục trược như vậy với bước trượt bằng 1.

Và đó chính là phép tính tích chập dùng cho việc xử lý ảnh trong miền không gian. Nhưng điểm khác biệt ở đây là so với khái niệm tích chập thông thường ở tích chập này ta chỉ đưa bộ lọc vào rồi nhân cho ra kết quả còn đối với tích chập thông thường ta sẽ đảo một trong hai hàm lại theo trục tung rồi mới thực hiện phép nhân hai hàm với nhau cho ra kết quả. Chính vì điểm khác biệt này nên các phép tính tích chập trong không gian 2 chiều còn được gọi là tương quan chép (cross-correlation), nơi mà ta có thể nhân chập hai hàm mà không cần đảo ngược hàm theo trục tung.

1. Một số bộ lọc phổ biến dùng trong xử lý ảnh.

Một số bộ lọc phổ biến được sử dụng nhiều trong xử lý ảnh:

* Bộ lọc trung bình (Median Filter): bộ lọc được xây dựng dựa trên ý tưởng là thay đổi tất cả giá trị các điểm ảnh, mà mỗi điểm sau khi qua bộ lọc sẽ có giá trị bằng giá trị trung bình của các điểm ảnh xung quanh điểm ảnh đó.



*Hình 2.5. Bộ lọc trung bình*

Thuật toán cho bộ lọc trung bình: Ta thực hiện tích chập ảnh với bộ lọc trung bình như hình bên trên. Lấy kết quả ta được chia cho 9 để tính trung bình từng điểm giá trị của kết quả.



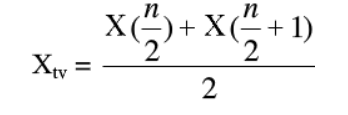
*Hình 2.6. Hiệu chỉnh giá trị kết quả từ bộ lọc trung bình*

Sau đó ta thực hiện hiêu chỉnh kết quả, bằng cách lấy giá trị ban đầu trừ cho giá trị kết quả tại cùng một vị trí pixel trên hình, nếu lớn hơn θ thì ta nhận kết quả mới làm giá trị cho điểm đó, nếu không ta vẫn sẽ lấy giá trị ban đầu là giá trị cho điểm đó. Ở hình tren I(x, y) là giá trị cũ và Itb(x, y) là giá trị mới và θ là ngưỡng có thể có hoặc không tùy thuộc vào mục đich người sử dụng.

Đây là một bộ lọc hỗ trợ việc giảm nhiễu muối tiêu, nhiễu possion, bảo vệ biên ảnh không bị làm mờ.

* Bộ lọc trung vị (Mean Filter): Bộ lọc trung vị là một bộ lọc phi tuyến, giá trị của một điểm ảnh sẽ vẫn phụ thuộc vào các điểm ảnh xung quanh. Giá trị của điểm ảnh được xác định bằng việc lấy tất cả các điểm ảnh xung quanh xếp theo thứ tự tăng dần, sau đó lấy giá trị nằm ở giữa trong dãy số tăng dần đó làm giá trị cho điểm ảnh đang xét.

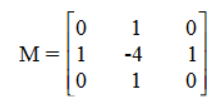
Thông thường các bộ lọc thường có kích thước là 3x3, 5x5, 7x7 nên rất dễ tìm ta được điểm chĩnh giữa, nhưng nếu là các bộ lọc có kích thước là số chắn thì áp dụng công thức:

 Với Xtv là vị trí cần chọn, và n là số lượng pixels của bộ lọc.

*Hình 2.7. Công thức tính vị trí*

Đây là một bộ lọc cũng hỗ trợ giảm nhiễu muối tiêu nhưng vẫn đảm bỏa được độ phân giải, và làm mịn ảnh, phép lọc trung vị thực hiện chậm hơn so với phép lọc trung bình.

* Bộ lọc Laplace: Bộ lọc Laplace là bộ lọc dựa trên ý tưởng lấy đạo hàm bậc hai trên từng điểm ảnh



*Hình 2.8. Bộ lọc Laplace*

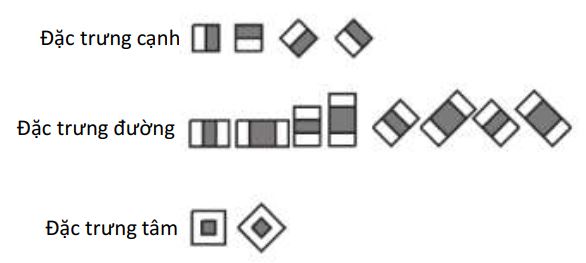
Bộ lọc laplace hỗ trợ phát hiện và làm nổi bật lên các cạnh cục bộ trong ảnh. Từ ta có thể thấy rõ được các đừng viền của các đối tượng. Với bộ lọc này ta có thể dùng cho ứng dụng Sharpen ảnh để tạo ra những bức ảnh có độ sắc nét cao hơn và rõ ràng hơn.

1. PHÁT HIỆN KHUÔN MẶT BẰNG PHƯƠNG PHÁP HAAR-CASCADE
2. Khái niệm Haar-Cascade.

Haar\_cascade là một mô module phát hiện đối tượng do OpenCV trong một dự án viết ra, trong đó có “**haarcascade\_frontalface\_default.xml**” là một model được xây dựng và đóng gói thành một file .xml, model này được sử để hổ trợ để nhận diện khuôn mặt con người.

1. Nguyên lý hoạt động.

Nói một cách đơn giản là model này sẽ sử dụng các bộ lọc Haar, các bộ lọc này chỉ trích xuất các đặc trưng Haar (các đặc trưng của khuôn mặt), ta sẽ trượt (cascade) bộ lọc này trên một bức ảnh, việc này giống với các trượt của sổ bộ lọc trong mô hình nơ tron tích chập CNN. Qua đó ta có thể trích xuất ta được các đặc điểm trên khuôn mặt từ đó phát hiện được khuôn mặt.



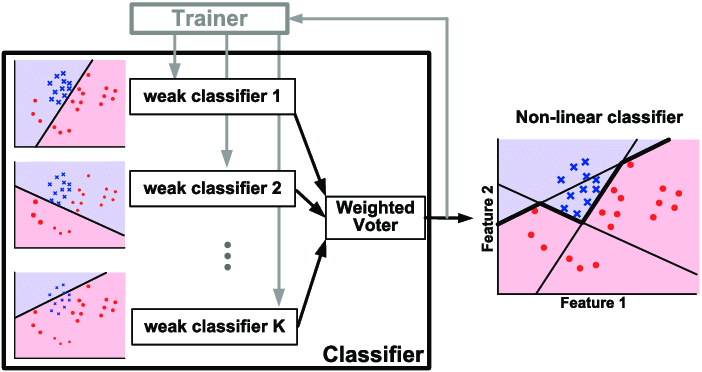
*Hình 2.9. Các bộ lọc Haar-Like phổ biến*

Model sẽ sử dụng một cửa sổ trượt có kích thước khá lớn tùy thuộc vào kích thước khuôn mặt của ảnh đầu vào, trong cửa sổ trước sẽ chứa các bộ lọc Haar khác nhau, bộ lọc Haar cũng không khác bộ lọc bình thường, chỉ là nó sẽ trích suất ra các đặc trưng quan trong của khuôn mặt mà thôi. Ví dụ như bộ lọc cạnh ngang sẽ được đặt ở vị trí giữa con mắt với lông mày vì ở đâu có sự khác này về màu sắc hay bộ lọc cạnh thẳng sẽ đặt ở sống mũi vì nó khác với hai bên má về góc độ ánh sáng hay bộ lọc lấy phần trung tâm đặt ở con mắt (*mô tả ở hình 2.10*), … các bộ lọc như thế sẽ được đặt trong cửa sổ trượt, và cứ mỗi lần di chuyển của sổ trước các bộ lọc sẽ thực thi nhiệm vụ của nó. Nếu tất cả bộ lọc để bắt được hình ảnh có chứa khuôn mặt thì nó sẽ ngừng khung cửa sổ trượt lại và lưu các đầu ra của từng bộ lọc lại, sau đó mới tiếp tục trượt, nếu không thì sẽ bỏ qua và bỏ kết quả các bộ lọc. Sau khi trượt quả một lần và thu lại kết quả, cửa sổ trượt và bộ lọc có thể sẽ tiến hành thu nhỏ đi để trượt thêm một lần nữa trên kết quả vừa rồi hoặc là trượt lại từ đầy ảnh đầu vào tùy thuộc vào giải thuật, đến khi bộ lọc nhỏ vừa đủ với kích thước khuôn mặt trên ảnh đâu vào thì ngừng lại và thực hiện so sánh các kết quả lưu được, kết quả nào phù hợp có chỉ số IoU cao nhất sẽ được chọn làm kết quả đầu ra cao nhất.



*Hình 2.10. Mô phỏng hoạt động của bộ lọc Haar\_Cascade*

Trên thực tế có rất nhiều loại bộ lọc khoảng 160 nghìn bộ lọc khác nhau nhưng để chọn một lại bộ lọc tốt thì không dể. Vì lẽ đó thay vì chọn bộ lọc ta sẽ tiến hành cải tiến nó để cho ra kết quả tốt hơn và phương pháp cải tiến là Adaboost. Phương pháp này sẽ thực hiện cải tiến các bộ lọc bằng cách gộp các bộ lọc lại với nhau (đa phần là các bộ lọc tuyến tính) để tạo thành bộ lọc phi tuyến tính, từ đó việc lọc có độ chính xác trở nên cao hơn. Kết quả của bộ lọc phi tuyến có ứng dụng của Adaboost được thể hiện ở hình 2.11 bên dưới đây:

*Hình 2.11. Hình ảnh minh họa phương pháp Adaboost*

Để rõ ràng hơn về phương pháp phát hiện khuôn mặt bằng haar\_cascade, ta có thể xem quá trình thực hiện của nó thông qua *clip [\*]*:

1. KIẾN TRÚC MẠNG CNN.
2. Khái niệm.

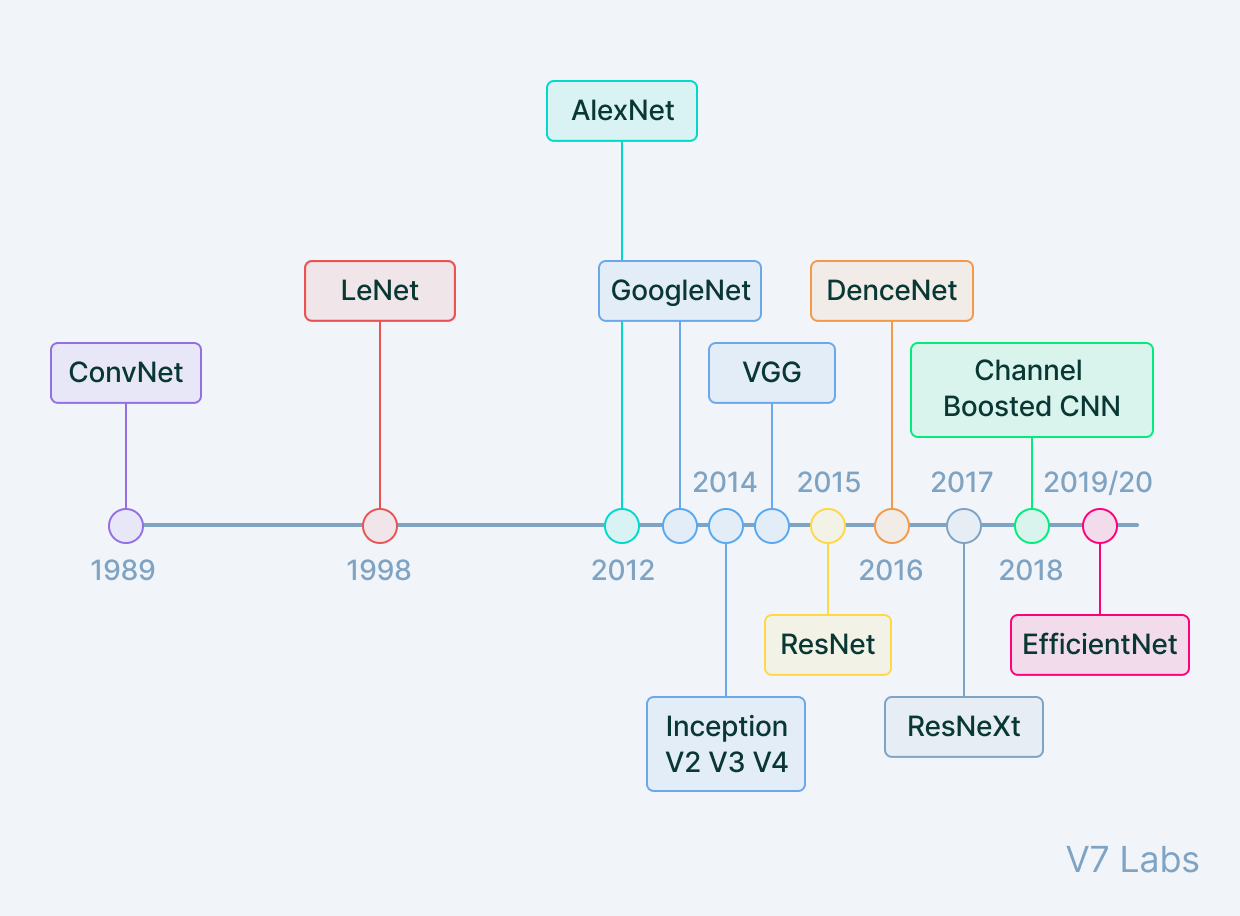
Như ta đã được học, tích chập là kĩ thuật phổ biến trong trong lĩnh vực xử lý ảnh với các phương pháp lâu đời như Haar-cascade, HOG hay SIFT SURE,… Và mạng tích chập CNN là một mạng học sâu, sự ra đời của CNN mang đến sự thúc đẩy mạnh mẽ cho quá trình phát triển của ngành thị giác máy tính.

Mạch CNN hay còn gọi là Conv-Net, lấy các lớp tích chập là chủ yếu để xây dựng lên các kiến trúc mạng khác nhau, phục vụ cho các mục đích khác nhau. Trong đó các lớp tích chập sẽ đóng vai trò như một lớp để tạo ra dữ liệu với việc trích xuất các đặc trưng theo cấp độ, từ việc trích xuất các đặc trưng đơn giản, một chiều, đến việc trích xuất các đặc trưng phức tạp hơn, đa chiều. Đương nhiêu phương pháp phép tích chập để trích xuất đặc trưng vẫn dựa trên lý thuyết cơ bản từ các phương pháp tích chập trước đó. Sau khi đã trích xuất được các đặc trưng của dữ liệu đầu vào qua các lớp tích chập trên, thì dữ liệu đã được trích xuất(hay còn gọi là các feature map) sẽ được đưa qua các lớp kết nối đầy đủ (fully-connected), các lớp này dựa trên cấu tạo của mạng nơ tron thần kinh trong bộ não con người để tạo ra, nhằm huấn luyện các feature map, và từ đó cập nhật các trọng số để nâng cao độ chính xác cho đầu ra của mạng.

***Chú thích: Clip [\*] là link youtube to phần tài liệu tham khảo***

1. Quá trình phát triển của kiến trúc mạng CNN.

Quá trình phát triển của CNN được khái quát bởi các cột mốc chính sau:



*Hình 2.12. Tiến trình phát triển CNN*

Kiến trúc mạng CNN ra đời từ rất lâu vào năm 1989, nhưng phái đến gần một thập kỉ sau đó thì người ta mời áp dụng kiến trúc mạng CNN cho các ứng dụng cụ thể như nhận diện chữ viết tay hay nhận diện khuôn mặt. Đó là lúc mà mạng LeNet ra đời kiến trúc mạng đầu tiên được nhà khoa học Yann LeCun phát triển. Tuy vậy phải đợi được đến năm 2012 trở đi, nhưng kiến trúc mạng CNN mới bắt đầu trở lại và có nhiều bản cải tiến tốt hơn như AlexNet (2012), GoogleNet hay còn được gọi là InceptionNet (2014), VGG(2015), ResNet, DenseNet, …

Đi cùng với sự phát triển của các kiến trúc về mạng, là sự phát triển về phần cứng của máy tính như GPU, TPU, với kỹ thuật phân tán và vận hành song song nhiều GPU cùng một lúc, giúp cho việc huấn luyện trở nên nhanh hơn và hiệu quả hơn đặc biệt là đối với việc huấn luyện các model mạng có kiến trúc phức tạp, nhiều lớp và số lớp có thể lên đến hàng trăm. Ngoài ra, các framework hổ trợ cho quá trình huấn luyện trở nên nhiều hơn và tiên ích hơn như pytorch của facebook hay tensorflow, keras của google. Trong đó đặc biệt nhất là pytorch, công cụ hỗ trợ lập trình với ngôn ngữ python như được xây dựng hầu hết bằng ngôn như C++, có tích hợp sẵn kết nối GPU trong quá trình huấn luyện.

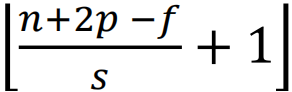
1. Các lớp phổ biến dùng trong mạng CNN.
2. Lớp tích chập (Convolution Layer).

Lớp tích chập được xuất hiện rất nhiều trong một kiến trúc mạng CNN. Trong một lớp tích chập ta có thể có 1 hoặc nhiều cửa sổ chập hay còn được gọi là các bộ lọc (kernel), các bộ lọc trong cùng một lớp có kích thước giống nhau, đều là 3x3 hoặc 5x5 hoặc 7x7 hoặc cũng có thể là 11x11. Mỗi bộ lọc này sẽ có giúp mạng trích xuất ra được một đặc trưng riêng của dữ liệu đầu vào. Và còn một điều nữa là số lượng các bộ lọc trong một lớp tích chập các nhiều (hay còn gọi là số lượng channel càng nhiều), đặc trưng được trích xuất trong mạng kiến trúc CNN (có sử dụng framework) sẽ càng phức tạp. Về phương pháp tính phép chập sẽ là Cross-Correlation (tương quan chéo), đây là phương pháp ta cho bộ lọc trượt trên dữ liệu đầu vào.

Trong quá trình trượt của bộ lọc thường có 2 thông số đáng chú ý là:

* Padding: có nghĩa là canh về, tùy vào kích thước của bộ lọc mà feature map thu được sau khi được trượt qua của ảnh sẽ có kích thước giảm đi nhiều hay ít so với ảnh gốc. Vì vậy trước khi bắt đầu trượt ta sẽ thêm một lớp pixel đều có giá trị bằng 0 ở bên ngoài ảnh đầu vào, để khi trượt xong thì feature map có cùng kích thước với ảnh đầu vào. Và trong framework của tensorflow, padding thường có 2 giá trị đó là same và valid, với giá trị “same” dữ liệu đầu ra sẽ có cùng kích thước như ban đầu, còn giá trị “valid” thì sẽ không có padding và dữ liệu đầu ra sẽ có kích thước nhỏ hơn ban đầu. (kích thước ở đâu chỉ tính chiều dài và chiều rộng của dự liệu không tích chiều sâu)
* Strike: có nghĩa là bước trượt, có thể bằng 1 hoặc bằng 2, 3,…

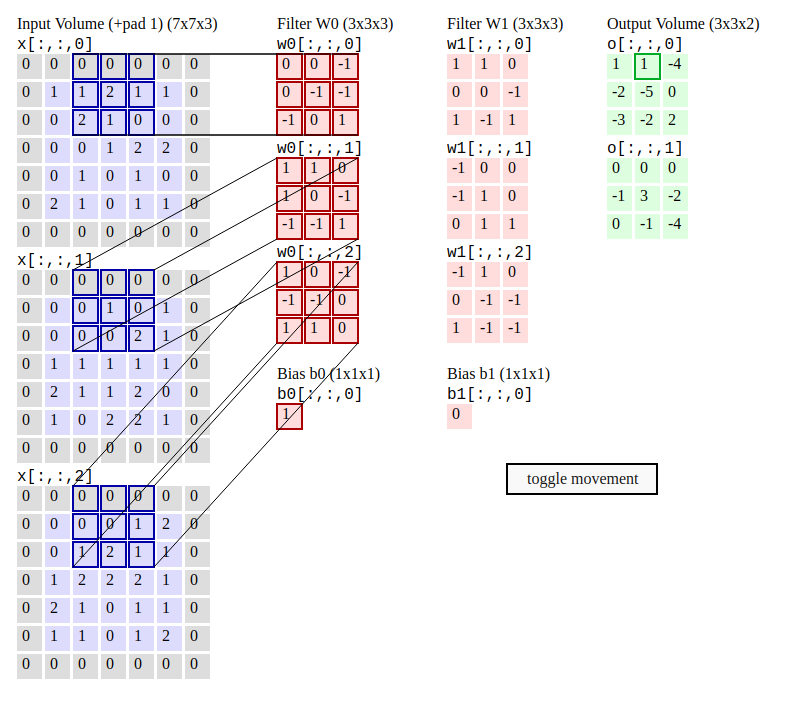
Công thức tính kích thước đầu ra của của một lớp tích chập với sự ảnh hưởng của padding và strike

Trong đó: n là kích thước đầu vào của ảnh

p là chỉ số padding

s là số bước trượt

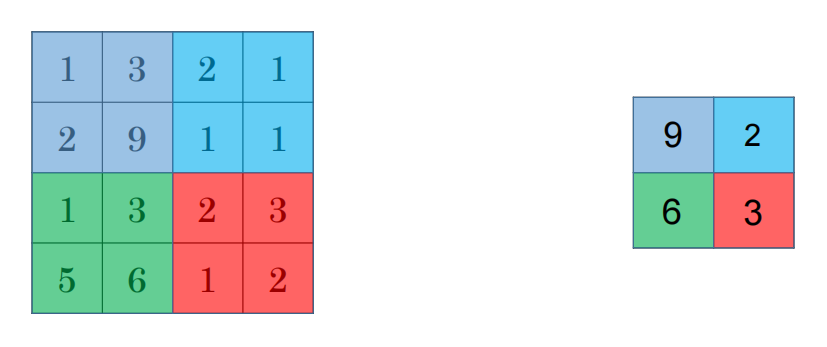
*Hình 2.13. Công thức* f là kích thươc bộ lọc



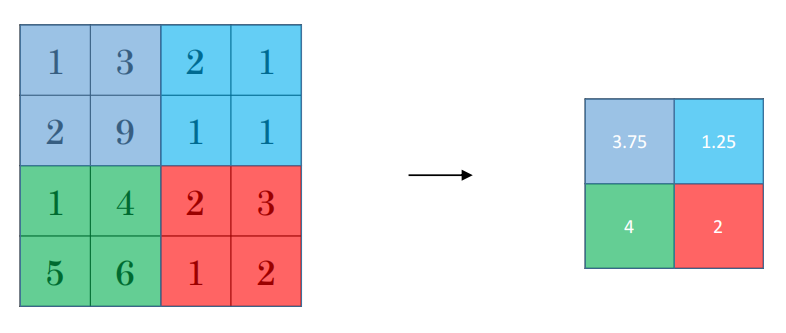
*Hình 2.14. Mô tả quá trình tính toán của một lớp tích chập*

1. Lớp Pooling (Pooling Layer).

Pooling layer thường dùng để làm giảm tích kích thước về chiều dài và chiều rộng của dữ liệu nhưng vẫn giữ lại được các đặc trưng quan trọng. Có 2 loại pooling là MaxPooling và AveragePooling. Quá trình Pooling cũng giống như có trình tích chập, đều cần một lớp kernel với tích thước phổ biến là 2x2, và lớp kernel này sẽ trượt trên dữ liệu đầu vào, với Maxpooling thì pixel trên ảnh có giá trị lớn nhất trong mỗi lần kernel trượt qua sẽ được giữ lại, phần còn lại sẽ bị loại bỏ, còn với AveragePooling thì giá trị trung bình của các pixel trong mỗi lần kernel trượt tới sẽ được giữa lại.



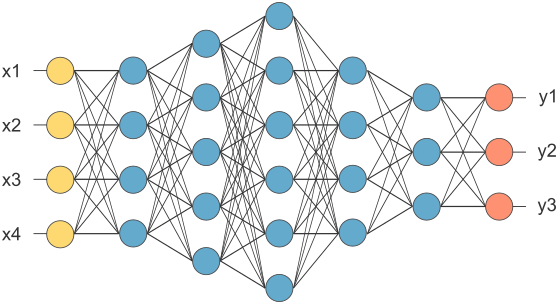
*Hình 2.15. Mô tả hoạt động của MaxPooling*



*Hình 2.16. Mô* *tả hoạt động của AveragePooling*

1. Lớp kết nối đầy đủ (Fully-connected Layer).

Sau khi đã có các feature map với các đặc trưng nổi bật của ảnh đầu vào. Ta đưa dữ liệu đó vào trong lớp kết nối đầu đủ. Ở đầu các feature map với kích thước là một ma trận sẽ được làm phẳng tình các vector một chiều với từng pixel được đưa vào từng neutral, qua từng lớp hidden kèm với các hàm kích hoạt phi tuyến tính như ReLU, tanh, sigmoid hay softmax. Quá trình này sẽ giúp huấn luyện dữ liệu, cập nhật các trọng số để cải thiện độ chính xác dữ liệu.



*Hình 2.17. Mô phỏng lớp kết nối đầy đủ*

1. KIẾN TRÚC VỀ MẠNG VGG-FACE (VGG-16).

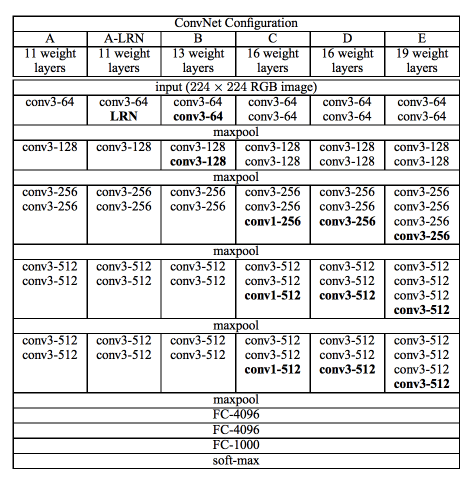
1. Giới thiệu.

VGG-Face là một kiến trúc mạng phổ biến dùng cho việc nhận diện khuôn mặt vào những năm 2016-2017. VGG-Face không phải là một kiến trúc cụ thể nào mà nó là tập lớp các kiến trúc mạng khác như VGG-16, Resnet-50 và Senet-50. Đây là các kiến trúc được dùng phổ biến cho việc nhận diện khuôn mặt nên người ta hay gọi chung nó là VGG-Face. Và ở đồ án lần này, kiến trúc VGG-16 sẽ được chọn là kiến trúc mạng chủ yếu cho việc nghiên cứu và tìm hiểu của em.

1. Kiến trúc mạng VGG-16.

Điểm khác biệt giữa kiến trúc VGG với các kiến trúc mạng đời trước LeNet và AlexNet là: Nếu mạng LeNet và AlexNet là sự kết hợp liên tiếp của một lớp Convolution và một lớp AveragePooling thì kiến trúc mạng VGG-Net sẽ là sự kết hợp liên tiếp của 2 hoặc 3 lớp tích chập lại với nhau rồi mới dùng đến lớp MaxPooling. Điều này giúp cho việc trích xuất dữ liệu trở nên nhiều hơn, các đặc trưng trích xuất được sẽ nhiều hơn nên khi ta thực hiện MaxPooling để giảm kích thước sẽ không lấy đi quá nhiều đặc điểm của feature map, giúp cho độ chính xác sau này sẽ tăng lên cao hơn. Và bên cạnh đó việc thay lớp MaxPooling thay vì AveragePooling sẽ giúp cho việc tích toán ở lớp Pooling sẽ nhanh hơn nhưng ngược lại quá trình tính toán ở lớp Convolution sẽ chập hơn, lâu hơn qua thuật toán kết hợp nhiều lớp tích chập là phức tạp hơn.

Và kiến trúc của VGG-16 sẽ là:

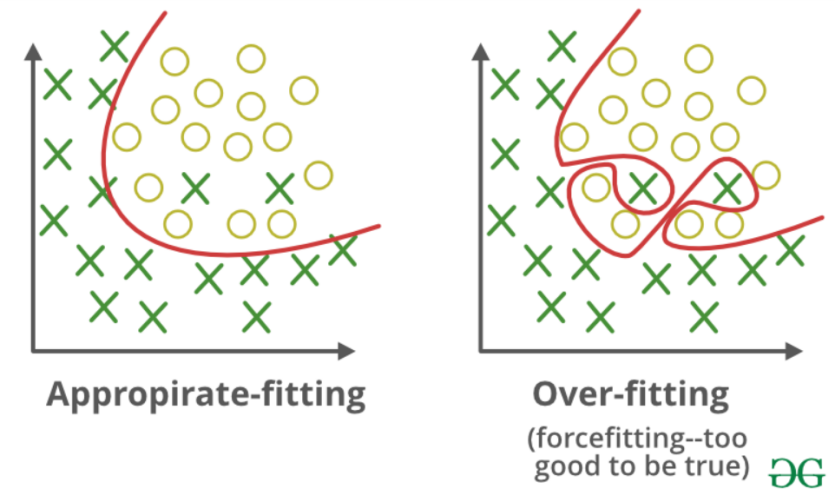


*Hình 2.18. Các kiến trúc mạng VGG*

Như hình trên mô tả các kiến trúc của mạng VGG trong đó cột C và cột D sẽ là kiến trúc của VGG-16, và cột D chính là kiến trúc của VGG-Face có 16 lớp convolution. Ta có thể thấy rõ ràng là càng về sau, số lượng lớp tích chập liên tiếp này càng nhiều từ 2 đến 3 lớp tích chập liên tiếp nhau, đồng thời số bộ lọc trong các lớp tích chập tăng cũng càng lớn theo lũy thừa của 2, vậy nên các đặc trưng trích xuất được của các lớp tích chập sau này sẽ càng phức tạp và đa chiều hơn.

1. Một số lỗi thường gặp trong quá trình huấn luyện dữ liệu.
2. Hiện tượng overfitting.

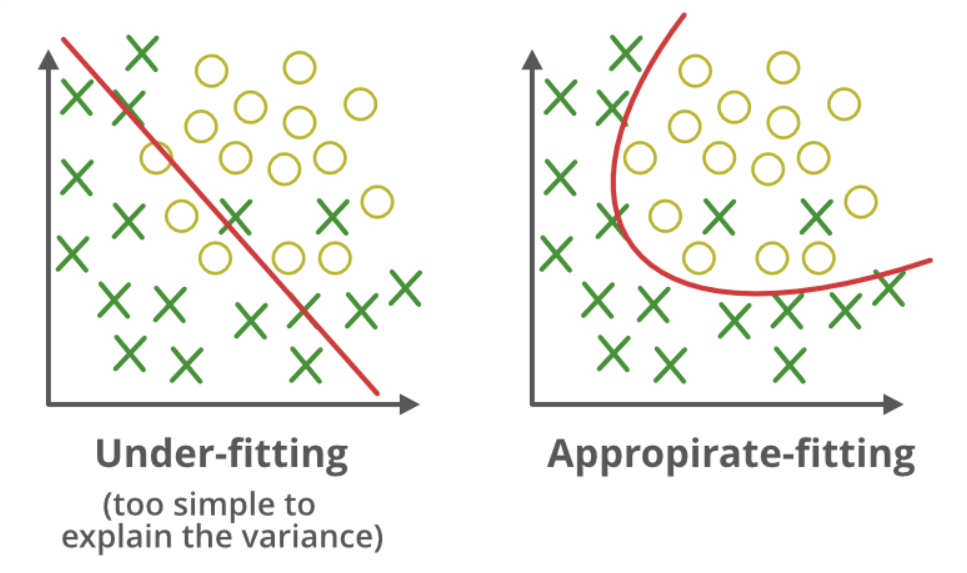
Overfitting là hiện tượng độ chính xác đối với tập huấn luyện (train) cao nhưng độ chính xác đối với tập kiểm tra (test) thấp. Điều này xảy ra là model xây dựng quá sâu, thuật toán phân loại đối tượng quá phức tạp, làm cho model huấn luyện quá chính xác (hay còn được gọi trong cuộc sống là học vẹt). Không chỉ do ảnh hưởng của thuật toán, tập dữ liệu quá ít cũng là một nguyên nhân dẫn đến hiện tượng overfitting, khi tập dữ liệu của đối tượng nhân dạng quá ít hay tập dữ liệu không được xáo trộn trước khi huấn luyện sẽ làm cho việc học của từng ảnh trở nên đơn giản hơn, độ phức tạp thấp,… quá trình huấn luyên thuận lợi cho tập huấn luyện nhưng sẽ gây khó khăn cho tập kiểm tra. Dưới đây là hình ảnh trình bày hiện tượng overfitting:



*Hình 2.19. Hiện tượng Overfitting*

1. Hiện tượng underfitting.

Trái ngược với Overfitting, thì Underfitting là hiện tượng độ chính xác trên tập huấn luyện (train) nhỏ và độ chính xác trên tập kiểm tra (test) cũng nhỏ. Nguyên do xảy ra hiện tượng Underfitting là khi thuật toán phân loại quá đơn giản, không đáp ứng đủ để phân biệt mức độ khác nhau của 2 đối tượng (tùy thuộc vào các đối tượng có độ giống nhau bao nhiêu thì thuật toán dùng để phân loại phải có độ phức tạp bấy nhiêu). Dưới đây là hình ảnh trình bày bày hiện tượng Underfitting:



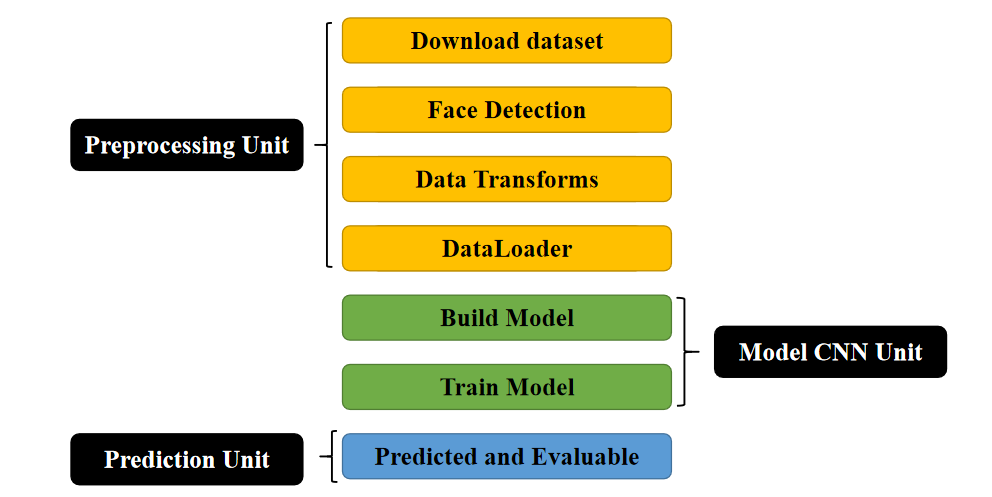
*Hình 2.20. Hiện tượng Underfitting*

CHƯƠNG 3

**THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG**

1. TIỀN TRÌNH THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG

Quá trình xây dựng hệ thống một hệ thống AI luôn luôn bao gồm ba bước cơ bản là tiền xử lý ảnh, xây dựng model huấn luyện và dự đoán kết quả thu được từ model. Và đồ án xây dựng một hệ thống nhận dạng khuôn mặt của 10 người khác nhau cũng áp dụng các bước trên để hoàn thành một hệ thống hoàn chỉnh, và chi tiết của từng bước được thể hiện ở hình ảnh dưới đây



*Hình 3.1. Tiến trình xây dựng và thiết kế hệ thống*

Dựa và trên hình ba bước chính của hệ thống được thể hệ bởi ba màu sắc khác nhau. Bước đầu tiên là tiền xử lý ảnh (Preprocessing datasets Unit), mục đích của bước này là xây dựng tập ảnh đầu vào cho quá trình huấn luyện, quá trình đánh giá và quá trình kiểm tra (test). Công việc tiền xử lý đối với đề tài nhận diện khuôn mặt bao gồm 4 bước nhỏ sau:

**Bước 1: Tải tập dữ liệu (Download dataset).** Ta sẽ chọn 10 người bất kì trên mạng (lời khuyên là nên người nổi tiếng để tập dữ liệu thu được nhiều) tải ảnh của người đó và tạo ra một thư mục riêng với tên thư mục là tên của người đó nhằm mục đích là để đánh nhãn cho từng đối tượng nhận diện, mỗi đối tượng cần ít nhất 30 đến 40 ảnh để tránh sinh ra hiện tượng underfitting trong quá trình huấn luyện vì lượng dữ liệu huấn luyện quá ít trong khi đối tượng huấn luyện là quá nhiều. Sau đó xây dựng các thư mục dành riêng cho việc huấn luyện (Train), đánh giá (Validation) và kiểm tra (Test).

**Bước 2: Phát hiện khuôn mặt (Face Detection).** Điều hiển nhiên sẽ xảy ra trong quá trình thu thập ảnh là các ảnh tải về sẽ không cố định chỉ là khuôn mặt của đối tượng mà nó có thể bao gồm chân, tay, hoặc các vật thể khác không liên qua đến đối tượng nhận dạng. Điều ta cần là là loại bỏ các vật thể không liên quan đến đối tượng nhận dạng đó, và do ban đầu ta đã xác định các đối tượng nhận dạng chính là khuôn mặt của con người vậy nên ta sẽ sử dụng model Haar-Cascade, model này sẽ giúp ta phát hiện và chắc lọc từng khuôn mặt trong từng bức ảnh. Kết quả là sau bước 2 ta đã thu thập được tập dữ liệu trong đó từng bức ảnh chỉ chứa đúng khuôn mặt của đối tượng nhận diện.

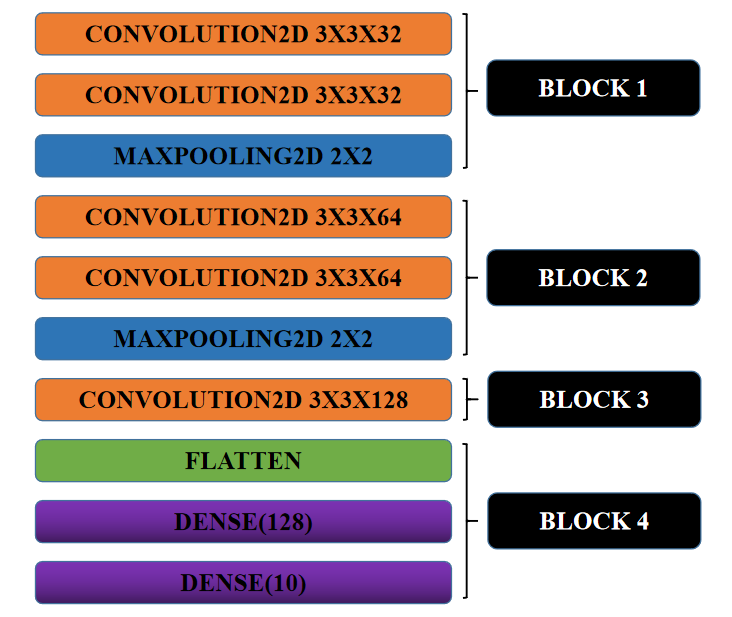
**Bước 3: Biến đổi tập dữ liệu (Data Transform).**  Đây là bước mà ta sẽ chuẩn hóa tập dữ liệu để nó có cùng kích thước với ngõ vào của model huấn luyện. Do model huấn luyện sẽ xây dựng dựa trên kiến trúc của model VGG-Face (VGG-16), vậy nên kích thước tập ảnh ngõ vào sẽ là 224x224. Sau khi đã resize cho kích thước tập ảnh ngõ vào, ta tiến hành lọc bỏ nhiễu trong tập ảnh, theo như kiến thức thu được trong quá trình học bộ môn xử lý ảnh, thì đa phần các tấm ảnh đề sẽ bị ảnh hưởng bởi nhiễu muối tiêu và nhiễu gause (nhất là các tấm ảnh bị crop đi một phần), vậy nên để loại bỏ nhiễu ta sẽ dùng bộ lọc trung vị để loại bỏ các nhiễu đó. Tuy nhiên bộ lọc trung vị còn có chức năng là làm mịn ảnh, vậy nên sau khi loại bỏ được nhiễu thì ảnh có khả năng sẽ bị nhòa, bị làm mờ nhưng không quá nhiều, nhưng để cải thiện điều đó ta sẽ dùng bộ lọc Sharpen để làm sắc nét lại các đường viền trong ảnh.

Quá trình biến đổi tập dữ liệu không chỉ dừng ở mức chỉnh sủa kích thước và lọc nhiễu, ta còn cần phải là đa dạng tập dữ liệu lên bằng việc lật ngang (horizon flip), lật dọc (vertical flip), phóng to (zoom), xoay (rotation), dịch ngang (horizon shift), dịch dọc (vertical shift),… Việc này, sẽ làm cho lập dữ liệu được đa dạng hơn, giúp quá trình huấn luyện được chính xác hơn. Bên cạnh đó ta còn cần phải rescale lại giá trị từng pixels trong ảnh, nếu từng pixels trong ảnh có giá trị từ 0 đến 255, ta sẽ đưa giá trị đó trở về trong khoảng từ 0 đến 1 bằng cách lấy từng giá trị ban đầu chia cho 255. Tất cả các điều này đều có công cụ hỗ trợ là class ImageDataGenerator do framework Tensorflow cung cấp.

**Bước 4: Bộ tải tập dữ liệu (DataLoader).** Tại bước này ta sẽ tiến hành xây dựng các dataloader cho cả tập dữ liệu huấn luyện và tập dữ liệu đánh giá, mục đích của việc xây dựng các dataloader này, là giúp ta có thể tải tập dữ liệu vào trong model để huấn luyện. Trong dataloader ta có thể điều chỉnh kích số lượng batch-size, tức là mỗi lần huấn luyện của model ta sẽ đưa dữ liệu vào model với số lượng ảnh tối đa là batch-size. Bên cạnh đó, ta có thể xáo trộn tập dữ liệu để quá trình huấn luyện tốt hơn, cố định được kích thước tập ảnh ngõ vào, và đặt biệt là có thể gán nhãn cho từng đối tượng trong tập dữ liệu dựa trên tên của thư mục của ảnh của đối tượng nhận diện đó. Công cụ hỗ trợ chính là hàm flow\_from\_directory trong class ImageDataGenerator cũng là do framework Tensorflow cung cấp.

Với 4 bước trên , ta đã hoàn thành được quá trình tiền xử lý ảnh, tiếp theo ta sẽ đến với quá trình xây dựng và huấn luyện model (Model CNN Unit), và quá trình này sẽ trải qua 2 bước chính sau:

**Bước 1: Xây dựng model (Build Model).** Dựa trên kiến trúc của model VGG-16, thì ta sẽ xây dựng một model với các lớp tích chập liên tiếp nhau thay vì mỗi lớp tích chập sẽ xen kẽ một lớp pooling. Bên cạnh đó, lớp tích chập dùng trong model sẽ sử dụng kernel với kích thước đều là 3x3, lớp pooling được sử dụng sẽ là MaxPooling2D với kích thước kernel đều là 2x2. Vậy nên kiến trúc model em xây dựng sẽ bao gồm các lớp được sắp xếp theo thứ tự như sau:



*Hình 3.2. Các lớp trong model nhận diện đối tượng*

Để dễ dàng phân tích ta phân model thành 4 block như trên hình. Bắt đầu với block 1 sẽ là sự kết hợp của 2 lớp tích chập, với kích thước lớp kernel sử dụng là 3x3x32, vậy nên thông số học thu được sau khi đi qua hai lớp tích chập là (3x3x3 + 1)x32 + (3x3x32 + 1)x32 = 10,144 parameters, và trong mỗi lớp tích chập thì sẽ luôn đi kèm với hàm kích hoạt (activation function) của nó là hàm Rectified Linear Unit hay còn viết tắt là hàm ReLU, kế tiếp sẽ là một lớp maxpooling2D với kích thước kernel là 2x2 để làm giảm kích thước ảnh từ 224x224 xuống còn 112x112, lớp maxpooling này có thông số học bằng 0 tại vì nó chỉ có chắc năng là làm giảm kích thước chứ không có chức năng học. Kết thúc của block 1 sẽ tạo ra một feature map với kích thước là 112x112x32.

Tiếp tục ta sẽ đế block 2, tại đây model sẽ đi qua 2 lớp tích chập liên tiếp với kích thước lớp kernel là 3x3x64, vậy nên thông số học thu được sau khi đi qua 2 lớp tích chập này là (3x3x32+1)x64 + (3x3x64 + 1)x64 = 55,424 parameters, mỗi lớp tích chập luôn đi kèm với hàm kích hoạt ReLU, kế tiếp là một lớp maxpooling2D với kernel có kích thước là 2x2, để làm giảm kích thước tập ảnh từ 112x112 thành 56x56. Kết thúc của block 2 sẽ tạo ra một feature map với kích thước là 56x56x64.

Tiếp tục với block 3, ở block 2 ta chỉ có một lớp tích chập với kích thước kernel là 3x3x128, vậy nên thông số học sau khi đi qua lớp tích chập này là (3x3x64 + 1)x128 = 73,856 parameters. Kết thúc lớp tích chập này sẽ tạo ra một feature map với kích thước là 56x56x128.

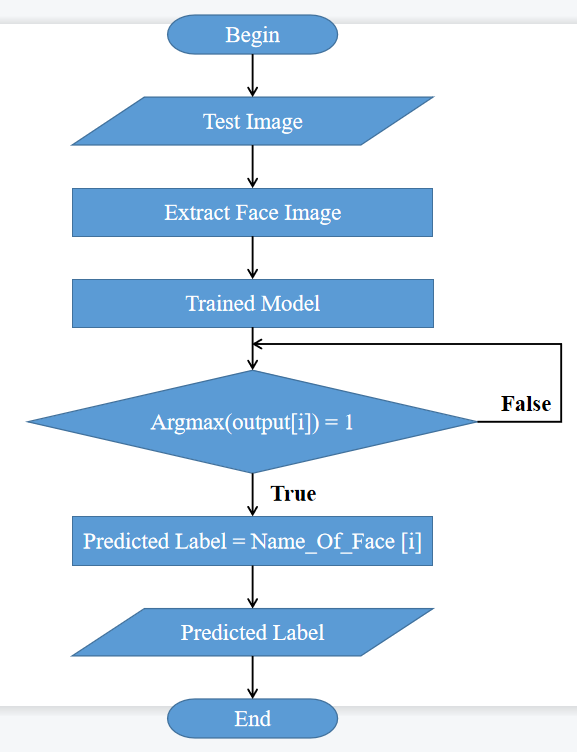
Cuối cùng là block 4, trong block thứ 4 này sẽ không còn các lớp như lớp tích chập hay với maxpooling mà là các lớp fully-connected. Đầu tiên ta sẽ tiến hành làm phảng các feature map lấy từ block thứ 3 về dạng vector 1 chiều. Vậy nên sau lớp flatten feature map sẽ có kích thước mới là 1x1x(56x56x128) = 1x1x401,408. Kế tiếp ta sẽ tạo ra một lớp fully-connected bằng hàm dense với số lượng neutral trong lớp này là 128, vậy nên kích thước feature map thu được sau khi qua lớp dense(128) sẽ là 1x1x128. Và kết thúc của model một lớp output trong hệ thống fully-connected, lớp output này được tạo ra cũng bởi hàm dense những số lượng neutral trong lớp này sẽ là 10, tại vì có 10 đối tượng khác nhau cần nhận diện (nếu số lượng đối tượng nhận diện lớp hơn ta sẽ thay đổi thông số này), đi kèm với hàm dense(10) này ta dùng hàm kích hoạt là hàm softmax. Hàm softmax sẽ trả về xác suất của từng đối tượng trong một tập 10 đối tượng khác nhau, dựa vào xác xuất này ta có thể dự đoán được kết quả đầu ra của model đối với từng tấm ảnh test.

**Bước 2: Huấn luyện model (Train Model).** Trước khi huấn luyện model ta phải xác định hàm lỗi (loss function) và hàm tối ưu (optimizer) cho model. Vì lớp cuối cùng của model có hàm kích hoạt là softmax nên dựa theo nguyên lý các bài toán softmax regression thì hàm loss được dùng phổ biến ở đây là cross entropy loss. Bên cạnh đó hàm tối ưu ta dùng để cập nhật trọng số trong quá trình huấn luyện là hàm adam. Và công cụ để tổng hợp hàm lỗi và hàm tối ưu là hàm compile(), công cụ để huấn luyện model là hàm fit(), cả hai đều là những công cụ được cung cấp trong framework Tensorflow.

Cuối cùng là quá trình dự đoán kết quả từ model đã train (Prediction Unit). Sau khi huấn luyện xong được mode ta sẽ lưu model lại dưới dạng file .h5. Từ đó ta sẽ viết một giao diện hỗ trợ quá trình nhận diện khuôn mặt được viết bằng ngôn ngữ python, sử dụng thư viện hỗ trợ là tkinter. Trong chương trình giao diện ta sẽ đưa ảnh đầu vào vào trong model và lấy ra giá trị xác suất cho từng đối tượng được tính từ hàm softmax. Các giá trị xác suất này sẽ được lưu dưới dạng mảng một chiều có tích thước là (1x10), ta sẽ dùng làm argmax() được cung cấp bởi thư viện numpy, hàm này sẽ trả về giá trị vị trí nào trong hàm có giá trị lớn nhất, từ đó ta có thể biết được ảnh đầu vào là thuộc đối tượng nào.

1. SƠ ĐỒ GIẢI THUẬT

Giải thuật dùng cho việc nhận diện đối tượng trong tập dữ liệu test là:



*Hình 3.3. Giải thuật nhận diện khuôn mặt*

Giải thuật nhận diện này được viết trong chương trình giao diện chạy kết quả dự đoán của đề tài. Giải thuật này chỉ được thực hiện khi model đã được huấn luyện thành công và được lưu lại.

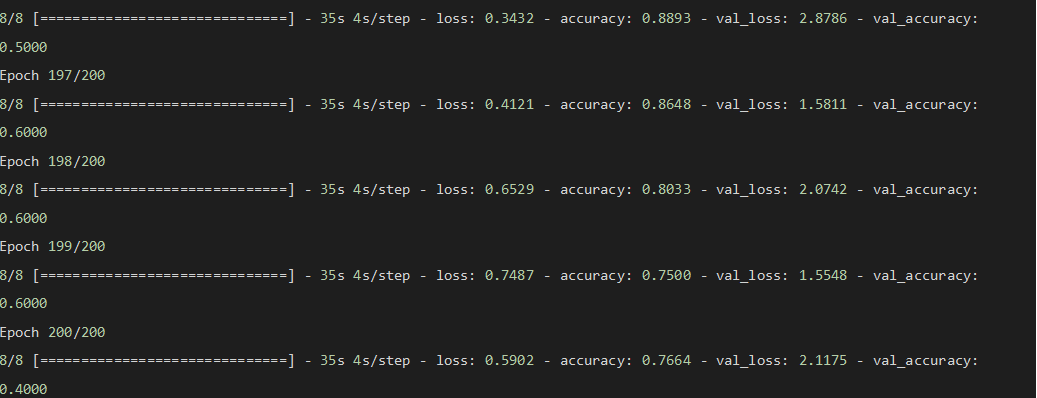
Ban đầu ta sẽ chọn một tấm ảnh nào đó của một trong số 10 đối tượng nhận diện. Sau đó tấm ảnh đó sẽ được đưa qua hàm trích xuất khuôn mặt để trích xuất ra được khuôn mặt trong ảnh (thuật toán trích xuất khuôn mặt vẫn là haar-cascade). Tiếp theo, ta sẽ đưa tấm ảnh đã được trích xuất vào trong model CNN đã được huấn luyện sẵn, model sẽ trả về một output dưới dạng mảng một chiều có kích thước lả (1, 10), trong đó 10 giá trị là phân bố xác suất cho độ chính xác của từng đối tượng trong 10 đối tượng nhận diện. Ta sẽ dùng hàm argmax của thư viện numpy để chọn ra vị trí đối tượng có phân bố xác suất cao nhất, tại vị trí đó sẽ được gán giá trị bằng 1, còn những vị trí khác được gán giá trị bằng 0. Ta lấy vị trí có giá trị bằng 1 đó, cho vào mảng những cái tên của các đối tượng đã được sắp xếp theo thứ tự nhận dạng trước đó, chọn ra được nhãn cho tấm ảnh ngõ vào và hiển trị ra giao diện chính.

CHƯƠNG 4

**KẾT QUẢ, NHẬN XÉT VÀ ĐÁNH GIÁ**

1. KẾT QUẢ

Quá trình huấn luyện tập dữ liệu sẽ trải qua 200 epochs và thời gian huấn luyện lên đến gần 3 tiếng đồng hồ để hoàn thành.



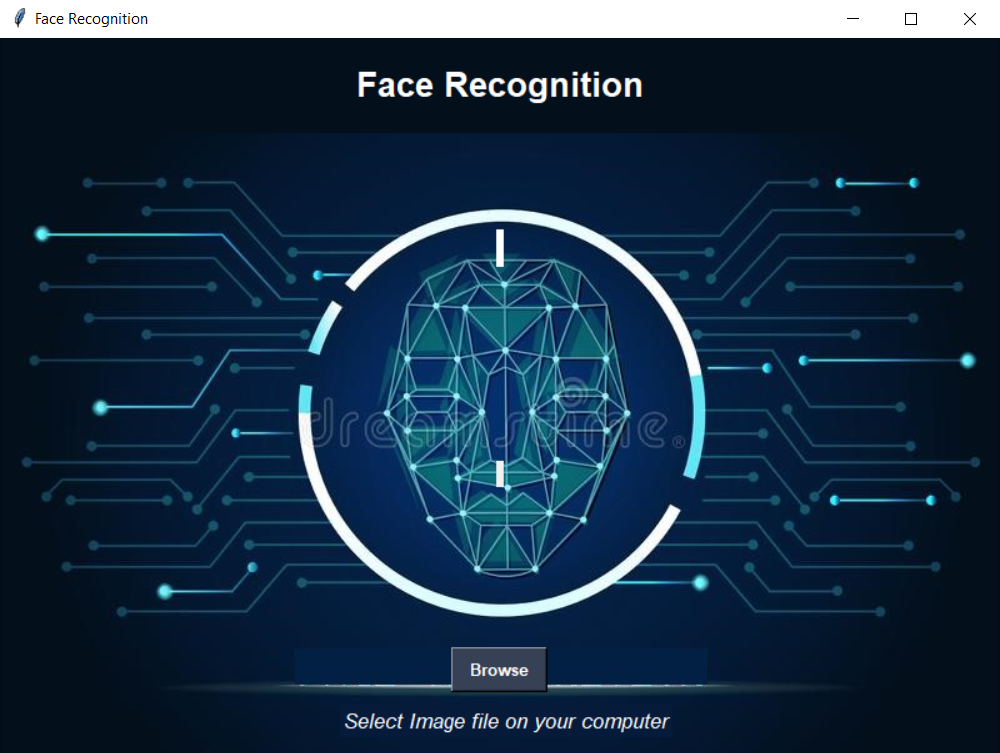
*Hình 4.1. Các epoch cuối cùng của quá trình huấn luyện*

Phía trên là hình ảnh của một số epochs được huấn luyện cuối cùng của hệ thống, epoch đạt độ chính xác cao nhất lên đến gần 89%.

Ban đầu, trong quá trình thiết kế, xây dựng một model để huấn luyện và kết quả thu được sau khi huấn luyện qua 200 epochs thì epoch đạt độ chính xác cao nhất cũng chỉ khoảng 76% mà thôi. Nhưng qua quá trình chỉnh sửa và hoàn thiện model thì độ chính xác dần dần tăng lên và lần cao nhất đạt được là 89% như trên.

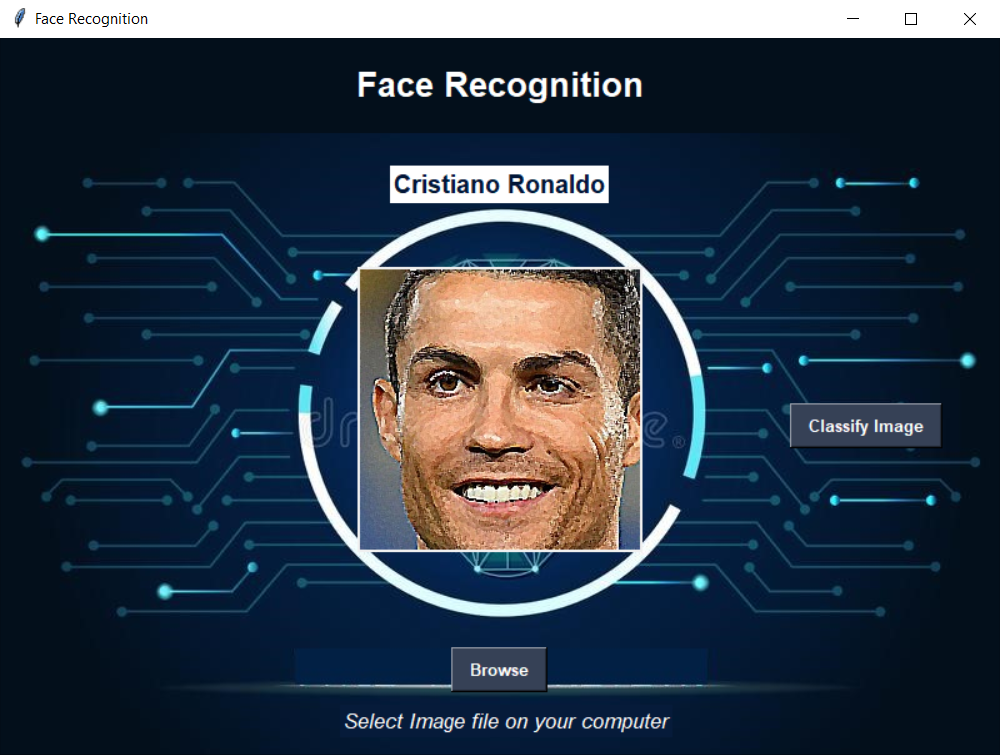
Các đối tượng được chọn để huấn luyện bao gồm: Albert Einstein, Cristiano Ronaldo, Donald Trump, Galileo Galilei, Joe Biden, Lionel Messi, Mark Zuckerberg, Nicola Tesla, Sơn Tùng MTP, Trấn Thành.

Giao diện cho ứng dụng nhận diện khuôn mặt được thiết kế như hình bên dưới:

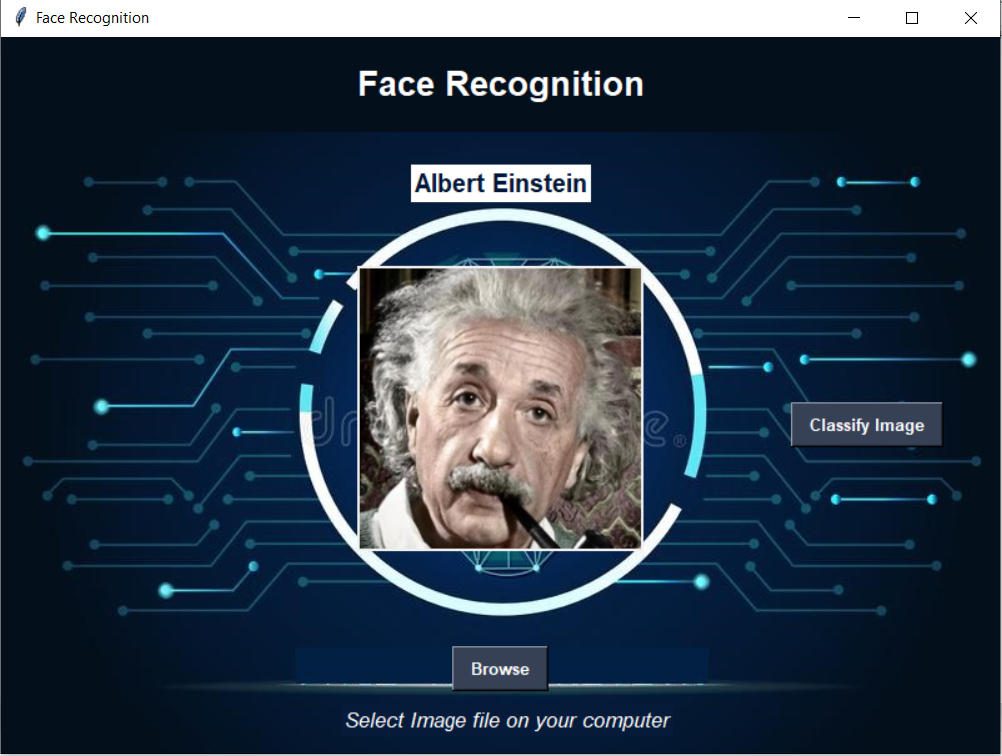


*Hình 4.2. Giao diện ứng dụng*

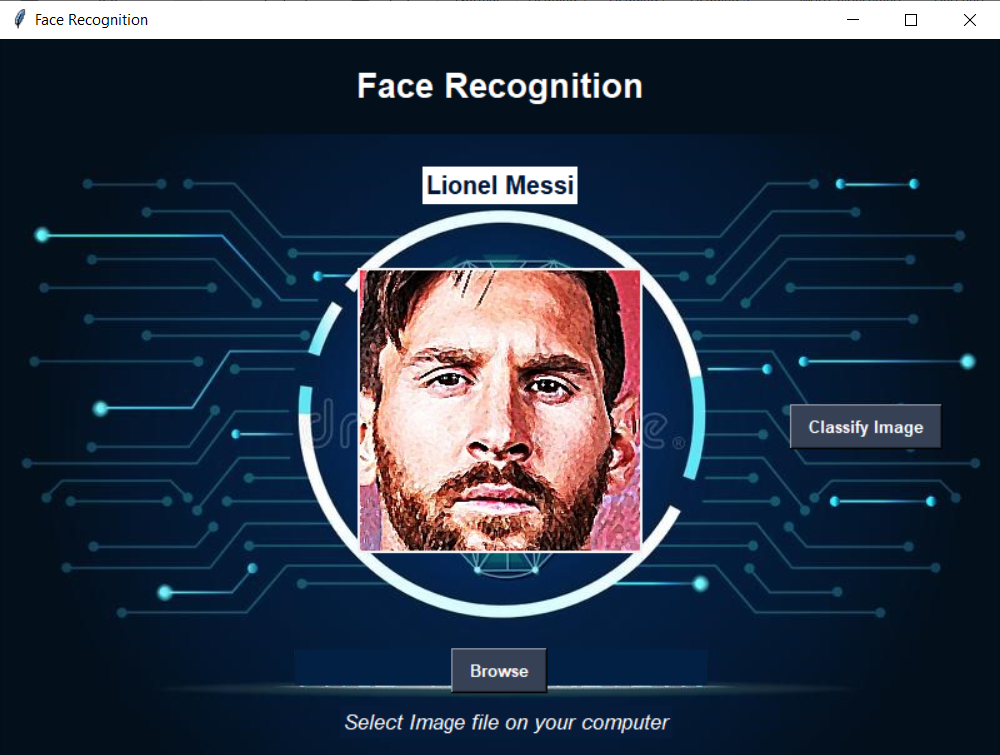
Sau đây, ta se cho một số ảnh test vào để cho hệ thống dự đoán và đưa ra kết quả:



*Hình 4.3. Kết quả test 1*

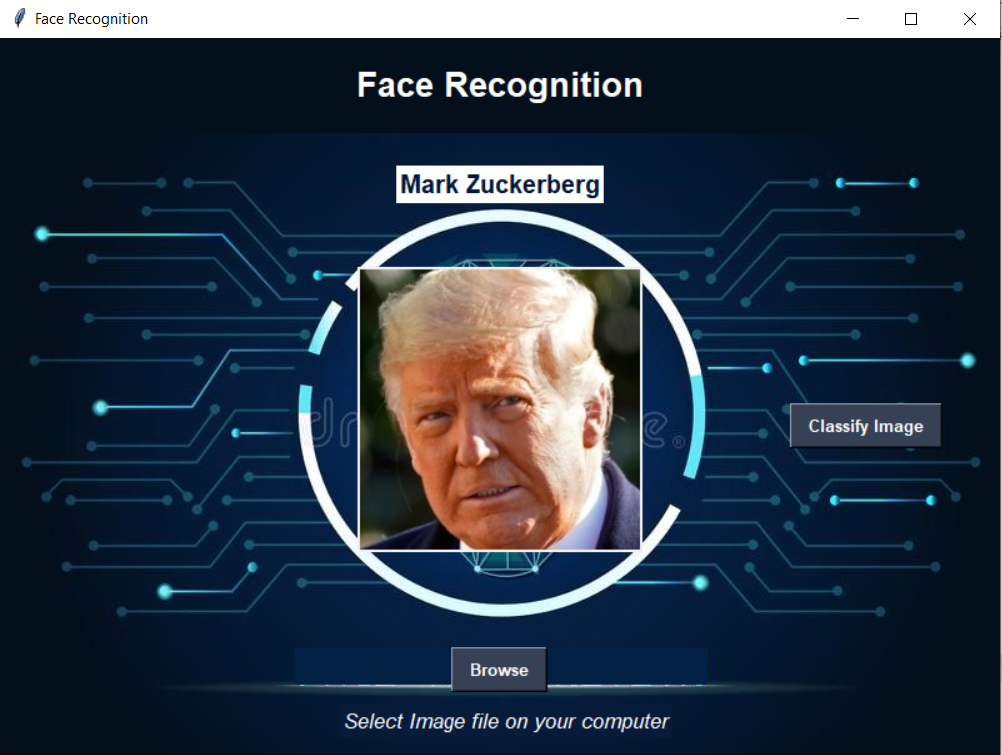


*Hình 4.4. Kết quả test 2*

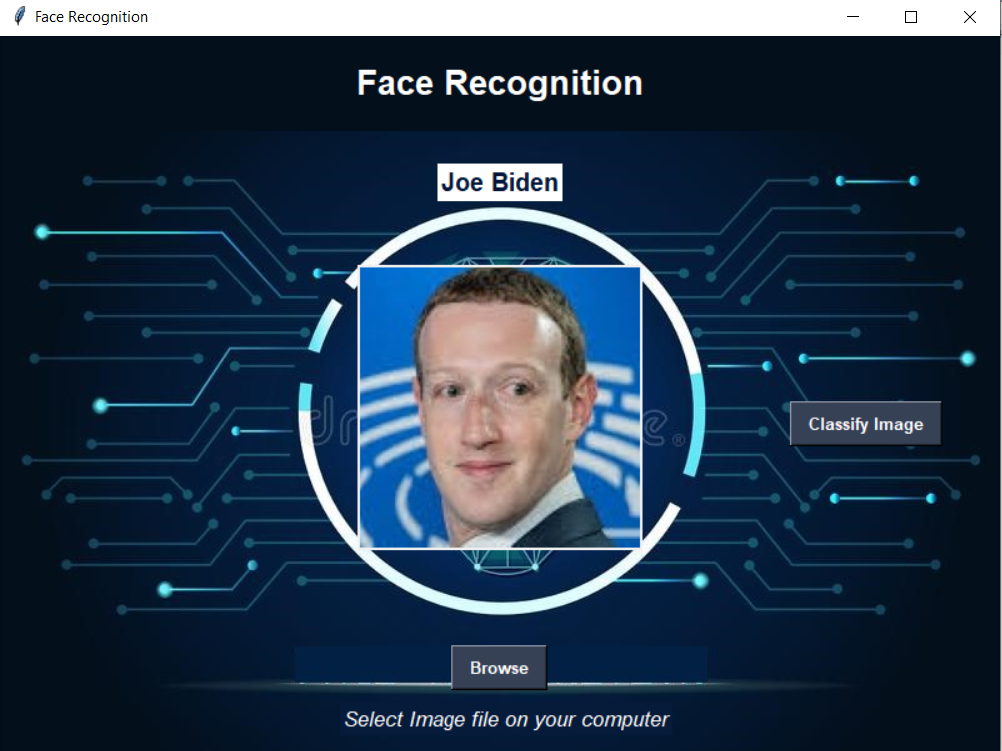


*Hình 4.5. Kết quả test 3*

Bên cạnh việc nhận diện đúng thì vẫn có vài trường hợp nhận diện chưa chính xác như:



*Hình 4.6. Kết quả nhận diện sai 1*



*Hình 4.7. Kết quả nhận diện sai 2*

1. NHẬN XÉT VÀ ĐÁNH GIÁ

Nhận xét và đánh giá:

* Tốc độ thực thi việc nhận diện không quá nhanh khoảng 1s.
* Độ chính xác chưa cao chỉ khoảng 89%, vẫn còn vài lần nhận diện sai.
* Trong quá trình huấn luyện, thời gian trung bình cho mỗi epoch chạy được là khoảng 57%, ban đầu các epoch mất hơn 80s để chạy, dần dần về sau thời gian tiêu tốn sẽ ít đi và luôn cố định ở mức 35s.

CHƯƠNG 5

**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

1. KẾT LUẬN.

Sau khi đã hoàn thành xong hệ thống nhận diện khuôn mặt, em đã rút ra một số kết luận như sau:

Ưu điểm:

* Xây dựng được một ứng dựng AI dùng có thể áp dụng trong thực tế.
* Hiểu được kiến trúc mạng CNN đặc biệt là kiến trúc mô hình VGG.
* Nắm bắt được cách xây dựng và sắp xếp các lớp trong một model một cách hợp lý.
* Có thể dựa vào thay đổi kiến trúc model hay thay đổi về tập dữ liệu huấn luyện để làm tăng độ chính xác.
* Sử dụng thành thạo được framework tensorflow.
* Lập trình được với python theo hướng OOP.
* Tốc độ nhận diện khá nhanh.

Nhược điểm:

* Hệ thống có độ chính xác chưa cao, còn cần được cải thiện.
* Số lượng đối tượng nhận diện không nhiều, cần tăng lên để có thể áp dụng hiệu quả hơn trong thực tế.
* Vấn để xử lý ảnh còn hạn chế, cần vài thiện việc này để tạo ra một tập dữ liệu huấn luyện có độ hiệu quả cao hơn.
* Thiết kế giao diện ứng dụng vẫn chưa đạt độ thẩm mỹ cao.
* Tốc độ huấn luyện còn khá chậm, cần hiệu chỉnh kiến trúc model cũng như tập dữ liệu để tăng tốc độ.
* Độ phân giải còn thấp, đối với các tấm ảnh test có độ phân giải thấp thì hệ thống có thể nhận diện sai.

Kinh nghiệm rút ra được trong quá trình xây dựng và huấn luyện:

* Tập dữ liệu cần phải đa dạng và nhiều, vì càng nhiều dữ liệu huấn luyện độ chính xác càng cao.
* Kích thước tập dữ liệu càng lớn thì quá trình trích xuất và huấn luyện càng lâu.
* Một tập dữ liệu với số lượng dữ liệu huấn luyện quá ít trong khi model xây dựng quá phức tạp, quá nhiều lớp, sẽ làm độ chính xác càng giảm.
* Số lượng lớp kernel càng nhiều trong quá mỗi lớp tích chập, thì việc trích xuất ta các đặc trưng càng phức tạp, các đặc trưng tích xuất được sẽ càng nhiều chiều.
* Việc loại bỏ nhiễu cho tập ảnh đầu vào sẽ giúp quá trình huấn luyện tập dữ liệu sẽ có độ chính xác cao hơn.

1. HƯỚNG PHÁT TRIỂN

Hướng phát triển cho đề tài:

* Xây dựng một hệ thống nhận diện khuôn mặt cho nhiều đối tượng hơn, tốt nhất là các đối tượng có cùng một lĩnh vực để hệ thống có thể dễ dàng ứng dựng trong thực tế.
* Cải thiện tốc độ nhận diện, ứng dụng realtime
* Ứng dựng thuật toán của các bài toán về Object Detection để có thể nhận diện được cho nhiều đối tượng khác nhau trong một khung ảnh.
* Tăng độ chính xác của model.
* Áp dụng thêm các loại kiến trúc khác nhau của CNN để tạo ra một model có độ phức tạp cao hơn và độ chính xác cũng tăng lên.
* Bên cạnh xử lý các tập dữ liệu 2D, ta sẽ huấn luyện các tập dữ liệu 3D, các khuôn mặt 3D.
* Xây dựng model nhận diện chính xác khuôn mặt 3D, loại bỏ nhận diện các khuôn mặt 2D để trách trộm cắp.

**PHỤ LỤC**

Hướng dẫn các bước chạy chương trình:

**Bước 1:** Tải phần mềm visual studio, và môi trường lập trình với python.

**Bước 2:** Mở folder trong Visual Studio: Chọn file->Open Folfer-> Chọn thư mục face\_recognition vừa tải về.

**Bước 3:** Kết nối với môi trường python trên Visual Code: nhập tổ hợp phím Ctrl+Shift+P, chọn Select Interpreter. Vào View-> Terminal, mở của sổ ternimal, nhấn chuột dấu cộng bên phải góc trên của cửa sổ Terminal, chọn Command Prompt.

**Bước 4:** Tải các thư viện cần thiết cho chương trình: nhập vào cửa sổ Terminal “pip install opencv-python matplotlib tensorflow keras pillow”

**Bước 5:** Chạy chương trình nhận diện: nhập vào cửa sổ Terminal “python UI\_recognition.py”, sau đó chọn một tấm ảnh trong thư mục Test Image rồi nhấn face-recognition để hệ thống tiến hành nhận diện khuôn mặt.

TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| **Tiếng Anh (Sách, tạp chí,… Bảng in)** | |
|  | Karen Simonyan & Andrew Zisserman, VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION, 10/04/2015.  Link: <https://arxiv.org/pdf/1409.1556.pdf/> |
|  | Srikanth Tammina, Transfer learning using VGG-16 with Deep  Convolutional Neural Network for Classifying Images, 10/2019  Link: https://www.researchgate.net/profile/Srikanth-Tammina/publication/337105858\_Transfer\_learning\_using\_VGG-16\_with\_Deep\_Convolutional\_Neural\_Network\_for\_Classifying\_Images/links/5dc94c3ca6fdcc57503e6ad9/Transfer-learning-using-VGG-16-with-Deep-Convolutional-Neural-Network-for-Classifying-Images.pdf/ |
|  | Senthamizh Selvi.R, Face Recognition Using Haar - Cascade  Classifier for Criminal Identification, 04/2019.  Link:https://www.researchgate.net/profile/Kanaga-Suba-Subramanian/publication/345896024\_22\_Face\_Recognition\_using\_Haar\_-\_Cascade\_Classifier\_for\_Criminal\_Identification/links/5fb10f6045851518fda6e4fc/22-Face-Recognition-using-Haar-Cascade-Classifier-for-Criminal-Identification.pdf/ |
|  | Wei-Lun Chao, Face Recognition  Link: <http://disp.ee.ntu.edu.tw/~pujols/Face%20Recognition-survey.pdf/> |
|  | [Sefik Ilkin Serengil](https://sefiks.com/), Deep Face Recognition with Keras, 08/2018 Link: <https://sefiks.com/2018/08/06/deep-face-recognition-with-keras/> |
|  | [Jason Brownlee](https://machinelearningmastery.com/author/jasonb/" \o "Posts by Jason Brownlee), How to Perform Face Recognition With VGGFace2 in Keras, 08/2020. Link: https://machinelearningmastery.com/how-to-perform-face-recognition-with-vggface2-convolutional-neural-network-in-keras/ |
|  | Clip [\*] <https://www.youtube.com/watch?v=L0JkjIwz2II/> |