TRƯỜNG ĐẠI HỌC BÁCH KHOA ĐÀ NẴNG
KHOA ĐIỆN TỬ VIỄN THÔNG

BÁO CÁO
ĐỒ ÁN KỸ THUẬT MÁY TÍNH

ĐỀ TÀI:
ỨNG DỤNG DEEP LEARNING ĐỂ NHẬN DẠNG CẢM XÚC KHUÔN MẶT

Giảng Viên Hướng Dẫn: Hồ Phước Tiến

Thành viên thực hiện: Trương Văn Phước-18DT2

Đặng Tiểu Minh-18DT2

Đà Nẵng, 6/2022

TÓM TẮT

TÊN ĐỀ TÀI: ỨNG DỤNG DEEP LEARNING ĐỂ NHẬN DẠNG CẢM XÚC KHUÔN MẶT

Sinh viên thực hiện: Đặng Tiểu Minh

106180095

Trương Văn Phước

106180105

Lớp: 18DT2

Trong những năm gần đây lĩnh vực thị giác máy tính, xử lý ảnh đã đem lại nhiều ứng dụng lớn trong cuộc sống con người, cùng với sự phát triển vượt bậc của khoa học kỹ thuật thì ngành khoa học này lại trở thành một trong các lĩnh vực được phát triển rất nhanh và thu hút được quan tâm đặc biệt của nhiều nhà nghiên cứu khoa học. Bằng chứng là xử lý ảnh đã chiếm một vai trò quan trọng trong các ứng dụng về khoa học kỹ thuật cũng như trong cuộc sống thường ngày như: sản xuất và kiểm tra chất lượng, sự di chuyển của Robot, an ninh và giám sát, nhận dạng mặt, các ứng dụng trong y học nổi trội trong những số đó như nhận diện khuôn mặt, phát hiện giám sát đối tượng,…

Nhận dạng cảm xúc khuôn mặt người là một bài toán được đặt ra nhiều năm trước đây với việc phân tích cảm xúc con người thông qua biểu hiện của khuôn mặt qua đó được ứng dụng nhiều trong việc thay thế cho việc giám sát yêu cầu sự tập trung cao độ trong thời gian dài như: khảo sát, thống kê,…;

Chính vì vậy tôi thực hiện đề tài " Nhận dạng cảm xúc khuôn mặt người sử dụng Deep learning '' sử dụng những kiến thức mình học được về Machine Learning kết hợp sử dụng ngôn ngữ lập trình Python để tạo ra một chương trình hỗ trợ nhận dạng cảm xúc khuôn mặt. Nội dung của đồ án chia thành 4 chương như sau :

Chương 1: GIỚI THIỆU TỔNG QUAN VỀ XỬ LÝ ẢNH VÀ NHẬN DẠNG CẢM XÚC TRÊN KHUÔN MẶT.

Chương 2: LÝ THUYẾT CƠ SỞ VÀ BÀI TOÁN NHẬN DẠNG CẢM XÚC KHUÔN MẶT.

Chương 3: XÂY DỰNG MẠNG NƠ-RON TÍCH CHẬP CHO BÀI TOÁN NHẬN DẠNG CẢM XÚC KHUÔN MẶT.

Chương 4: XÂY DỰNG THỰC NGHIỆM VÀ KẾT QUẢ.

Đề tài sử dụng Deep learning trong việc phân tích, dự đoán cảm xúc của con người thông qua webcam. Bên cạnh đó, trình bày các tiêu chí đánh giá và so sánh, kết quả đạt được sau khi tiến hành thực nghiệm đối với các mô hình đã xây dựng.

DANH MỤC CÁC TỪ VIẾT TẮT

Từ viết tắt	Từ đầy đủ
AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
CNN	Convolutional Neural Network
VGG	Visual Geometry Group
CSDL	Cơ sở dữ liệu
TP	True positive
FP	False positive
FN	False Negative

MỤC LỤC

DANH MỤC CÁC HÌNH VẼ

LỜI MỞ ĐẦU

Khuôn mặt của con người biểu hiện nhiều cảm xúc mà không cần phải nói ra. Đó là một trong những phương tiện mạnh mẽ và tự nhiên nhất để con người truyền đạt thể hiện cảm xúc. Không giống như các hình thức giao tiếp phi ngôn ngữ khác, cảm xúc trên khuôn mặt nó phổ quát. Hiện nay, nhận dạng và phân tích cảm xúc khuôn mặt tự động là một vấn đề thú vị và đầy thách thức, có ảnh hưởng to lớn đến xã hội. Cảm xúc trên khuôn mặt và hành động của chúng ta là phương tiện giao tiếp phi ngôn ngữ, bao gồm 93% cảm xúc giao tiếp của con người, trong đó 55% thể hiện cử chỉ khuôn mặt và hành động của con người.

Cảm xúc khuôn mặt có thể được phân tích dễ dàng thông qua hình ảnh khuôn mặt và máy tính có thể tương tác với con người, như cách con người tương tác với nhau. Đó là lý do tại sao nhận dạng cảm xúc qua khuôn mặt ngày càng được sự quan tâm trong mọi lĩnh vực. Các nhà nghiên cứu đã chỉ ra rằng cảm xúc trên khuôn mặt là phổ quát và bẩm sinh trong tất cả các chủng tộc, giới tính và độ tuổi. Thêm cảm xúc trung tính là có bảy cảm xúc cơ bản, gồm: trung tính, giận dữ, ghê tởm, sợ hãi, hạnh phúc, buồn và bất ngờ. Nhận dạng cảm xúc qua khuôn mặt có ứng dụng trong các lĩnh vực khác nhau:

- Giáo dục: Phản ứng của người học trong thời gian thực và sự tham gia vào nội dung là giáo dục là một thước đo lường cho hiệu quả của bài giảng.
- Tiếp thị: Đây là một cách tuyệt vời để các công ty kinh doanh phân tích cách khách hàng phản hồi với quảng cáo, sản phẩm, bao bì và thiết kế cửa hàng của họ.
- Chơi game: Với sự ra đời của game thực tế ảo gần với trải nghiệm thực tế. Nhận dạng cảm xúc khuôn mặt đóng một vai trò quan trọng để cải thiện trải nghiệm chơi trò chơi.
- Bảo mật: Nó có thể giúp xác định hành vi đáng ngờ trong đám đông và có thể được sử dụng để ngăn chặn tội phạm và những kẻ khủng bố tiềm năng.
- Chăm sóc sức khỏe: Nó có thể hữu ích trong việc tự động hóa dịch vụ y tế. Cả sức khỏe thể chất và tinh thần có thể được phân tích thông qua ứng dụng này.
- Dịch vụ khách hàng: Quản lý dịch vụ khách hàng có thể hiệu quả hơn bằng cách sử dụng hệ thống nhận dạng cảm xúc khuôn mặt. Phân tích phản hồi của khách hàng và phản ứng của máy tính sẽ đảm bảo tương tác máy tính với con người trong cuộc sống thực. Hệ thống nhận diện cảm xúc khuôn mặt được sử dụng nhiều trong cuộc sống: điều trị y tế, giao tiếp song ngôn ngữ, đánh giá đau của bệnh nhân, phát hiện nói dối, giám sát trạng thái của người lái xe phát hiện trạng thái buồn ngủ dựa vào cảm xúc trên khuôn mặt được phát triển để cảnh báo cho người lái xe khi thấy dấu hiệu buồn ngủ, mệt mỏi.

Khoa học ngày càng phát triển, những năm gần đây, "trí tuệ nhân tạo" – AI (Artificial Intelligence) và cụ thể hơn là "học máy" - Machine Learning nổi lên như một bằng chứng của cuộc cách mạng công nghiệp lần thứ tư. Trí tuệ nhân tạo đang len lỏi vào mọi lĩnh vực trong đời sống. Xe tự hành của Google và Tesla, hệ thống tự tag khuôn mặt trong ảnh của

Facebook, trợ lý ảo Siri của Apple, hệ thống gợi ý sản phẩm của Amazon, hệ thống gợi ý phim của Netflix, máy chơi cờ vây AlphaGo của Google DeepMind,… là một vài trong vô vàn những ứng dụng của AI/Machine Learning.

Mạng nơ-ron tích chập – CNN (Convolutional Neural Networks) là một trong những mô hình Deep Learning tiên tiến giúp chúng ta hiện thực hóa Machine Learning cũng như chinh phục trí tuệ nhân tạo. CNN phổ biến nhất và có ảnh hưởng nhiều nhất trong cộng đồng "thị giác máy tính" - Computer Vision (một lĩnh vực bao gồm các phương pháp thu nhận, xử lý ảnh kỹ thuật số, phân tích và nhận dạng các hình ảnh). CNN được dùng trong nhiều bài toán như nhân dạng ảnh, phân tích video, ảnh MRI, hoặc cho bài các bài của lĩnh vực xử lý ngôn ngữ tự nhiên và hầu hết đều giải quyết tốt các bài toán này.

Đề tài "Nhận dạng cảm xúc khuôn mặt người với Deep learning" chúng tôi áp dụng hết những kiến thức đã học về Machine Learning kết hợp sử dụng ngôn ngữ lập trình Python để tạo ra một chương trình hỗ trợ nhận dạng cảm xúc khuôn mặt người.

Trong Đồ án này, tôi sẽ thiết kế một mô hình, mô hình này sẽ sử dụng các thuật toán để giám sát các cảm xúc của khuôn mặt con người. Nội dung của đồ án chia thành 4 chương như sau:

Chương 1: GIỚI THIỆU TỔNG QUAN VỀ XỬ LÝ ẢNH VÀ NHẬN DẠNG CẢM XÚC TRÊN KHUÔN MẶT.

Chương 2: LÝ THUYẾT CƠ SỞ VÀ BÀI TOÁN NHẬN DẠNG CẢM XÚC KHUÔN MẶT.

Chương 3: XÂY DỰNG MẠNG NƠ-RON TÍCH CHẬP CHO BÀI TOÁN NHẬN DẠNG CẢM XÚC KHUÔN MẶT.

Chương 4: XÂY DỰNG THỰC NGHIỆM VÀ KẾT QUẢ

Đề tài sử dụng Deep learning trong việc phân tích, dự đoán cảm xúc của con người thông qua webcam.

CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN VỀ XỬ LÝ ẢNH VÀ NHẬN DẠNG CẢM XÚC TRÊN KHUÔN MẶT

1.1 Giới thiệu chương

Chương một tập trung vào giới thiệu tổng quan về xử lý ảnh, các ứng dụng trong các hệ thống thực tế, trình bày tổng quan về nhận dạng cảm xúc khuôn mặt và các ứng dụng của nhận dạng cảm xúc khuôn mặt.

1.2 Xử lý ảnh

1.2.1 Giới thiệu

Không có sự thống nhất trong sự phân chia giữa xử lý ảnh và các lĩnh vực liên quan đến nó như phân tích ảnh (image analysis) hay thị giác nhân tạo (computer vision). Một số tác giả cho rằng xử lý ảnh liên quan đến quá trình xử lý khi đầu vào và đầu ra đều là bức ảnh. Thị giác nhân tạo quan tâm việc mô phỏng thị giác con người, dùng mô hình toán học để biểu diễn và hiểu hình ảnh thực tế.

Đối với mảng xử lý ảnh chúng ta có thể phân chia thành 3 mức xử lý sau:

Mức thấp: khử nhiễu, tăng chất lượng ảnh, đầu vào và ra đều là ảnh**.**

Mức trung bình: phân vùng ảnh, chia ảnh thành các vùng hay các đối tượng, mô tả các đối tượng để hỗ trợ cho việc phân loại hay nhận dạng bằng máy tính. Đầu vào thường là ảnh, đầu ra là các thuộc tính trích ra từ ảnh (ví dụ cạnh, đường biên).

Mức cao: hiểu các đối tượng được nhận dạng và thực hiện các chức năng nhận thức.

1.2.2 Ứng dụng

Ngày nay, xử lý ảnh được áp dụng rất rộng rãi trong đời sống, trong nhiều lĩnh vực khác nhau như:

- Y khoa (ảnh tia gamma, X-quang) trong nhận dạng bệnh giúp các bác sĩ chuẩn đoán bệnh

- Thiên văn học, không gian giúp cá nhà nghiên cứu, con người có thể biết được các hiện tượng bên ngoài vũ trụ**.**

- Trong công nghiệp xử lý ảnh góp phần rất quan trọng trong việc phát hiện sản phẩm lỗi, quản lý các dây chuyền sản xuất,…

- Tìm kiếm hình ảnh bằng internet, mạng xã hội; xe tự hành một ứng dụng mảng xử lý ảnh rất được chú tâm gần đây**.**

- Hệ thống an ninh như nhận dạng biển số, nhận dạng khuôn mặt, nhận dạng chữ viết,…

1.3 Nhận dạng

1.3.1 Giới thiệu

Nhận dạng ảnh là giai đoạn cuối cùng của các hệ thống xử lý ảnh**.**

Nhận dạng là quá trình phân loại các đối tượng được biểu diễn theo một mô hình nào đó và gán cho chúng vào một lớp dựa theo những quy luật và các mẫu chuẩn. Quá trình học dựa vào các mẫu học biết trước gọi là nhận dạng có thầy hay học có thầy (supervised learning). Trong trường hợp ngược lại gọi là học không có thầy (no supervised learning).

Trong lý thuyết nhận dạng ảnh có ba cách tiếp cận khác nhau:

- Nhận dạng dựa vào phân hoạch không gian.
- Nhận dạng cấu trúc.
- Nhận dạng dựa vào kỹ thuật mạng nơron.

Hai cách tiếp cận đầu là các kỹ thuật kinh điển. Các đối tượng ảnh quan sát và thu được phải trải qua giai đoạn tiền xử lý nhằm tăng cường chất lượng ảnh, làm nổi các chi tiết, trích chọn và biểu diễn các đặc trưng, và cuối cùng là nhận dạng.

Cách tiếp cận thứ ba hoàn toàn khác, dựa vào cơ chế đoán nhận, lưu trữ và phân biệt đối tượng mô phỏng theo hoạt động của hệ thần kinh con người. Do cơ chế đặc biệt, các đối tượng thu nhận bởi thị giác người không cần qua giai đoạn cải thiện mà chuyển ngay sang giai đoạn tổng hợp, đối sánh với các mẫu đã lưu trữ để nhận dạng.

1.3.2 Bản chất của quá trình nhận dạng

Quá trình nhận dạng gồm 3 giai đoạn chính :

- Lựa chọn mô hình biểu diễn đối tượng.
- Lựa chọn luật ra quyết định (phương pháp nhận dạng) và suy diễn quá trình học.
- Học nhận dạng.

Khi mô hình biểu diễn đối tượng đã được xác định, quá trình nhận dạng chuyển sang giai đoạn học. Học là giai đoạn rất quan trọng. Thao tác học nhằm cải thiện,điều chỉnh việc phân hoạch tập đối tượng thành các lớp.

Việc nhận dạng chính là tìm ra quy luật và các thuật toán để có thể gán đối tượng vào một lớp.

❖ Học có giám sát (supervised learning)

Kỹ thuật phân loại nhờ kiến thức biết trước gọi là học có giám sát. Đặc điểm cơ bản

của kỹ thuật này là người ta có một thư viện các mẫu chuẩn. Mẫu cần nhận dạng sẽ được đem sánh với mẫu chuẩn để xem nó thuộc loại nào.

Thí dụ như trong một ảnh viễn thám, người ta muốn phân biệt một cánh đồng lúa, một cánh rừng hay một vùng đất hoang mà đã có các mô tả về các đối tượng đó. Vấn đề chủ yếu là thiết kế một hệ thống để có thể đối sánh đối tượng trong ảnh với mẫu chuẩn và quyết định gán cho chúng vào một lớp. Việc đối sánh nhờ vào các thủ tục ra quyết định dựa trên một công cụ gọi là 'hàm phân lớp' hay 'hàm ra quyết định'.

❖ Học không có giám sát (unsupervised learning).

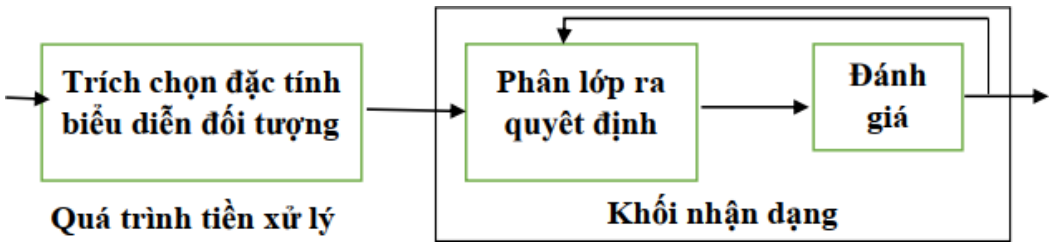
Kỹ thuật học này phải tự định ra các lớp khác nhau và xác định các tham số đặc trưng cho từng lớp.

Do số lớp và những đặc trưng của các lớp không được biết trước, nên kỹ thuật này khó khăn hơn.

Kỹ thuật này nhằm tiến hành mọi cách gộp nhóm có thể và chọn lựa cách tốt nhất.

Bắt đầu từ tập dữ liệu, nhiều thủ tục xử lý khác nhau nhằm phân lớp và nâng cấp dần để đạt được một phương án phân loại.

❖ Một hệ thống nhận dạng có thể tóm tắt theo sơ đồ:



Hình 1.1: Sơ đồ hệ thống nhận dạng.

1.4 Nhận dạng cảm xúc khuôn mặt

1.4.1 Giới thiệu

Khuôn mặt của con người biểu hiện nhiều cảm xúc mà không cần phải nói ra. Đó là một trong những phương tiện mạnh mẽ và tự nhiên nhất để con người truyền đạt thể hiện cảm xúc. Không giống như các hình thức giao tiếp phi ngôn ngữ khác, cảm xúc trên khuôn mặt nó phổ quát. Hiện nay, nhận dạng và phân tích cảm xúc khuôn mặt tự động là một vấn đề thú vị và đầy thách thức, có ảnh hưởng to lớn đến xã hội. Cảm xúc trên khuôn mặt và hành động của chúng ta là phương tiện giao tiếp phi ngôn ngữ, bao gồm 93% cảm xúc giao tiếp của con người, trong đó 55% thể hiện cử chỉ khuôn mặt và hành động của con người. Cảm xúc khuôn mặt có thể được phân tích dễ dàng thông qua hình ảnh khuôn mặt và máy tính có thể tương tác với con người, như cách con người tương tác với nhau. Đó là lý do

tại sao nhận dạng cảm xúc qua khuôn mặt ngày càng được sự quan tâm trong mọi lĩnh vực. Các nhà nghiên cứu đã chỉ ra rằng cảm xúc trên khuôn mặt là phổ quát và bẩm sinh trong tất cả các chủng tộc, giới tính và độ tuổi. Thêm cảm xúc trung tính là có bảy cảm xúc cơ bản, gồm: trung tính, giận dữ, ghê tởm, sợ hãi, hạnh phúc, buồn và bất ngờ.

1.5 Ứng dụng

Nhận dạng cảm xúc qua khuôn mặt có ứng dụng trong các lĩnh vực khác nhau:

- Giáo dục: Phản ứng của người học trong thời gian thực và sự tham gia vào nội dung là giáo dục là một thước đo lường cho hiệu quả của bài giảng.
- Tiếp thị: Đây là một cách tuyệt vời để các công ty kinh doanh phân tích cách khách hàng phản hồi với quảng cáo, sản phẩm, bao bì và thiết kế cửa hàng của họ.
- Chơi game: Với sự ra đời của game thực tế ảo gần với trải nghiệm thực tế. Nhận dạng cảm xúc khuôn mặt đóng một vai trò quan trọng để cải thiện trải nghiệm chơi trò chơi.
- Bảo mật: Nó có thể giúp xác định hành vi đáng ngờ trong đám đông và có thể được sử dụng để ngăn chặn tội phạm và những kẻ khủng bố tiềm năng.
- Chăm sóc sức khỏe: Nó có thể hữu ích trong việc tự động hóa dịch vụ y tế. Cả sức khỏe thể chất và tinh thần có thể được phân tích thông qua ứng dụng này.
- Dịch vụ khách hàng: Quản lý dịch vụ khách hàng có thể hiệu quả hơn bằng cách sử dụng hệ thống nhận dạng cảm xúc khuôn mặt. Phân tích phản hồi của khách hàng và phản ứng của máy tính sẽ đảm bảo tương tác máy tính với con người trong cuộc sống thực.

1.6 Kết luận chương

Sau khi có cái nhìn tổng quan về xử lý ảnh, nhận dạng và nhận dạng cảm xúc khuôn mặt, chúng ta sẽ tìm hiểu rõ hơn về bài toán nhận dạng cảm xúc khuôn mặt ở các chương tiếp theo.

CHƯƠNG 2: LÝ THUYẾT CƠ SỞ VÀ BÀI TOÁN NHẬN DẠNG CẢM XÚC KHUÔN MẶT

2.1 Giới thiệu chương

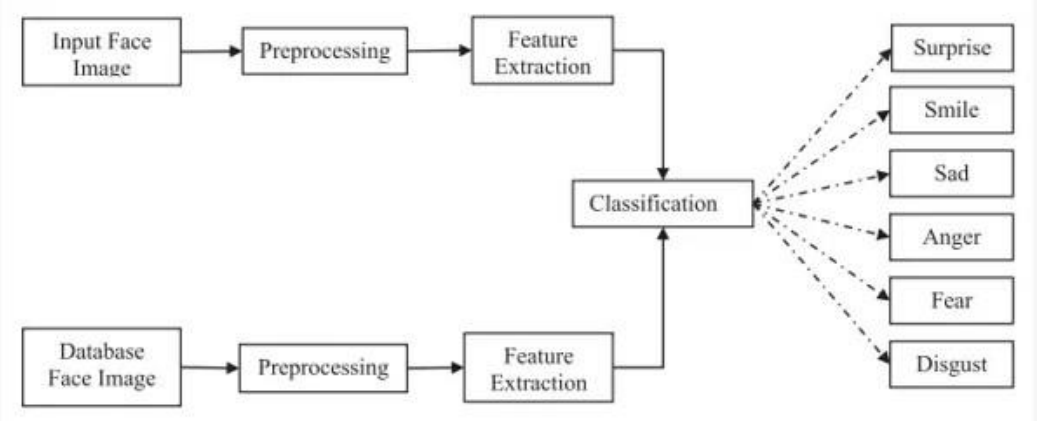
Chương này tập trung tìm hiểu các vấn đề quan trọng liên quan đến bài toán nhận dạng cảm xúc khuôn mặt, lý thuyết cơ bản về mạng nơ ron tích chập (Convolutional Neural Network) và tìm hiểu tổng quan các phương pháp nhận dạng cảm xúc khuôn mặt cũng như một vài kiến trúc phổ biến mạng nơ ron trong nhận dạng cảm xúc khuôn mặt.

2.2 Bài toán nhận dạng cảm xúc khuôn mặt

2.2.1 Giới thiệu bài toán nhận dạng cảm xúc khuôn mặt

2.2.1.1 Phương pháp truyền thống

Hệ thống nhận dạng cảm xúc qua khuôn mặt với phương pháp truyền thống thì xử lý bài qua các giai đoạn: tiền xử lý hình ảnh khuôn mặt, trích xuất đặc trưng và phân loại.



Hình 2.1:Kiến trúc nhận dạng cảm xúc khuôn mặt phương pháp truyền thống[2]

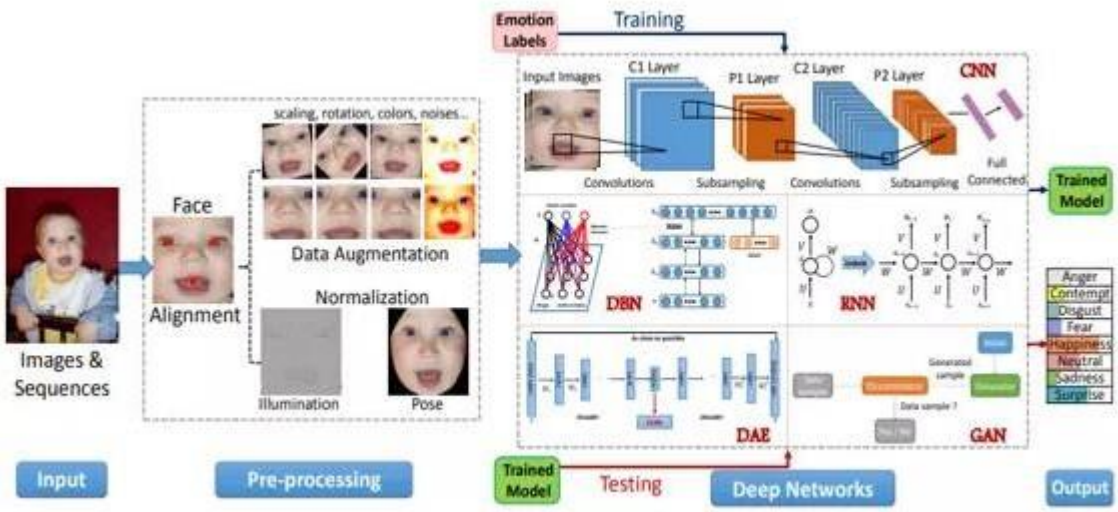
Tiền xử lý là quá trình được sử dụng để cải thiện hiệu suất của hệ thống nhận dạng cảm xúc qua khuôn mặt và được thực hiện các loại quy trình khác nhau: căn chỉnh độ rõ, chia tỷ lệ hình ảnh, điều chỉnh độ tương phản và sử dụng các quy trình nâng cao

Trích xuất đặc trưng trong thị giác máy tính là một giai đoạn quan trọng, nó phát hiện ra việc chuyển từ mô tả đồ họa sang mô tả dữ liệu ẩn, trích chọn những đặc trưng riêng nhất của hình ảnh, sau đó những mô tả dữ liệu này có thể được sử dụng làm đầu vào cho bài toán phân loại.

Phân loại là giai đoạn cuối cùng của hệ thống nhận diện cảm xúc qua khuôn mặt (FER), để phân loại ra các loại cảm xúc trên khuôn mặt: hạnh phúc, buồn bã, bất ngờ, tức giận, sợ hãi, ghê tởm và bình thường. Sử dụng các phương pháp phân loại như: Cây quyết định (ID3), SVM, HMM (Hidden Markov Model)… thì phân loại SVM cho độ chính xác và phân loại tốt nhất.

2.2.1.2 Phương pháp hiện đại

Trong phần này, sẽ mô tả các bước chính phổ biến trong hệ thống nhận dạng cảm xúc qua khuôn mặt thực hiện qua các giai đoạn: tiền xử lý, phân lớp sử dụng học sâu. Những năm gần đây, học sâu có độ chính xác hơn phương pháp truyền thống vì nó không phải qua bước trích xuất các đặc trưng một cách tường minh, nó sẽ thực hiện đi kèm với phương pháp phân loại.



Hình 2.2: Kiến trúc nhận dạng cảm xúc khuôn mặt phương pháp deep learning[2]

Tiền xử lý ảnh: cũng phải xử lý 1 số vấn đề của ảnh đầu vào hệ thống, xử lý trước quá trình trainning. Các bước thực hiện: Căn chỉnh khuôn mặt để phát hiện khuôn mặt, tăng dữ liệu hình ảnh đảm bảo đủ dữ liệu training, cuối cùng là chuẩn hóa dữ liệu khuôn mặt. Sử dụng các phương pháp CNN, DBN, DAE, RNN, GAN…

Phân loại: Trong phương pháp truyền thống bước trích xuất đặc trưng và bước phân loại tính năng là độc lập với nhau, trong Deep learning có thể thực hiện FER theo cách từ đầu đến cuối. Một lớp mất được thêm vào cuối mạng để điều chỉnh lỗi lan truyền ngược, sau đó xác suất dự đoán của từng mẫu có thể được mạng trực tiếp xuất ra.

2.3 Mạng nơ-ron tích chập (Convolutional neural network)

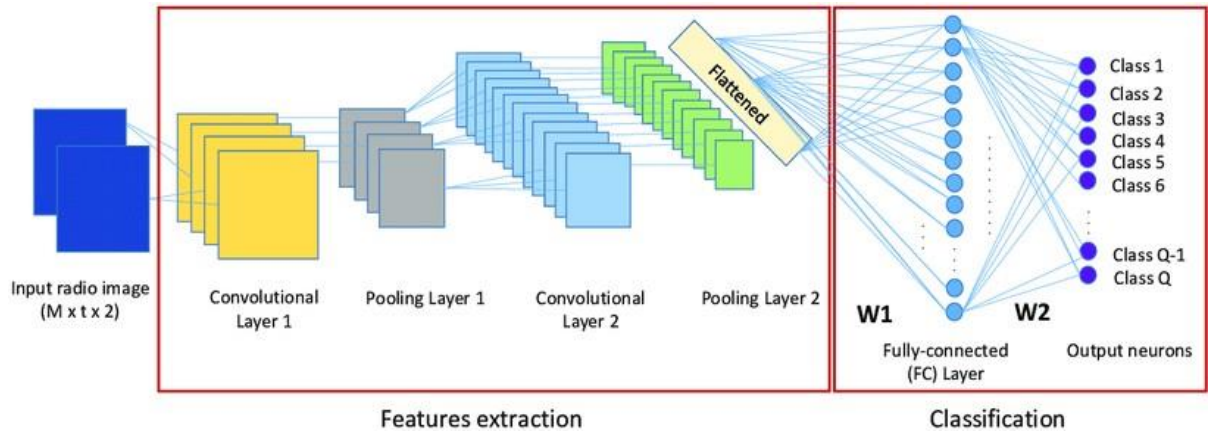
2.3.1 Giới thiệu mạng nơ-ron tích chập (Convolutional neural network)

Mạng nơ-ron tích chập (CNN) là một trong những mô hình mạng học sâu phổ biến nhất hiện nay, có khả năng nhận dạng và phân loại hình ảnh với độ chính xác rất cao mô hình này đã và đang được phát triển, ứng dụng vào các hệ thống xử lý ảnh lớn của Facebook, Google hay Amazon… cho các mục đích khác nhau như các thuật toán gắn thẻ tự động, tìm kiếm ảnh hoặc drone giao hàng tự động.

Sự ra đời của mạng nơ-ron tích chập là dựa trên ý tưởng cải tiến cách thức các mạng nơ-ron nhân tạo truyền thống học thông tin trong ảnh. Do sử dụng các liên kết đầy đủ giữa các điểm ảnh vào nút, các mạng nơ-ron nhân tạo truyền thẳng bị hạn chế rất nhiều bởi kích thước của ảnh, ảnh càng lớn thì số lượng liên kết càng tăng nhanh và kéo theo sự bùng nổ khối lượng tính toán.

2.3.2 Cấu trúc mạng CNN

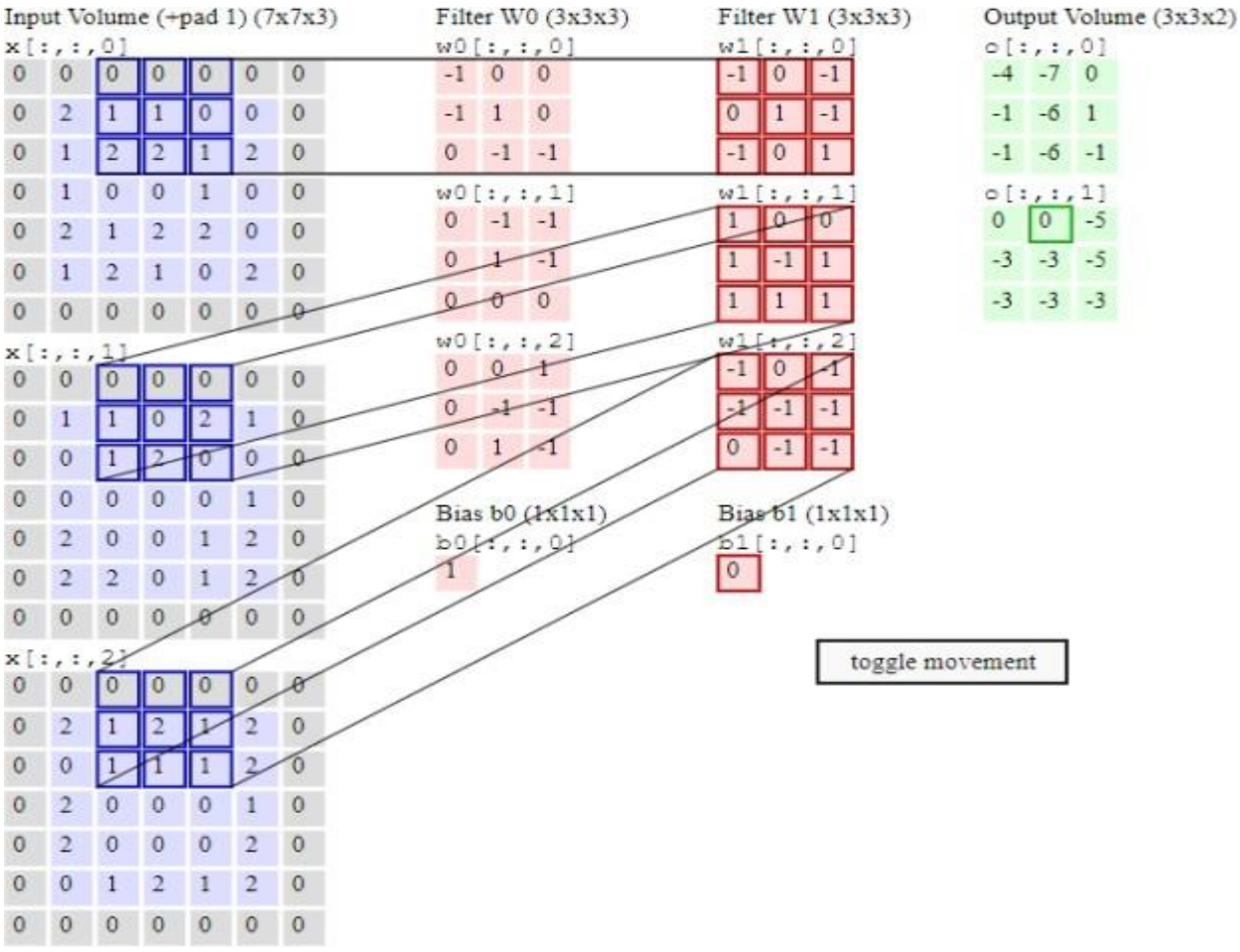
Các lớp cơ bản trong một mạng nơ-ron tích chập bao gồm: lớp tích chập (Convolutional layer) , lớp gộp (Pooling layer) , lớp kích hoạt (Activation layer) và lớp liên kết đầy đủ (Fully-connected layer), được thay đổi về số lượng và cách sắp xếp để tạo ra các mô hình huấn luyện phù hợp cho từng bài toán khác nhau.



Hình 2.3:Ví dụ về cấu trúc mạng CNN[3]

2.3.2.1 Lớp tích chập (Convolution layer)

Là lớp quan trọng nhất trong cấu trúc của mạng nơ-ron tích chập. Lớp này sẽ giúp trích xuất được những thông tin quan trọng từ dữ liệu. Để dễ hình dung, có thể xem tích chập như một cửa sổ trượt áp đặt lên một ma trận sau đó trích xuất những đặc trưng, thông tin quan trọng của ma trận đưa ra bản đồ đặc trưng.



Hình 2.4:Kenel 3x3 cho phép chập [6]

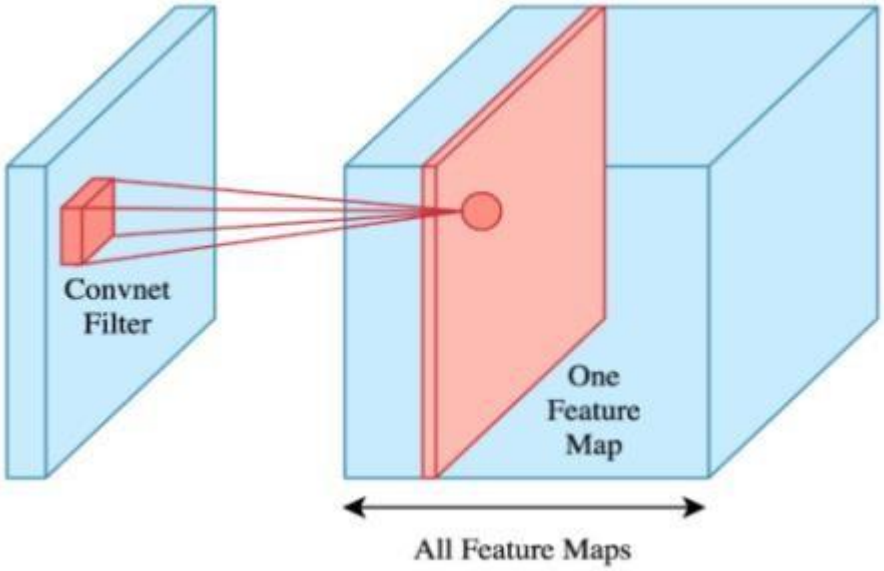
⇨ Convolution hay tích chập là nhân từng phần tử bên trong ma trận 3 × 3 với ma trận bên trái. Kết quả được một ma trận gọi là convoled feature được sinh ra từ việc nhân ma trận filter với ma trận ảnh 5 × 5 bên trái.

Sử dụng một kernel 3 × 3 chạy qua tất cả các phần tử của bức ảnh (cửa sổ trượt) và sử dụng tích chập tương ứng với các phần tử của bức ảnh trùng với vị trí của kernel 3 × 3 sau đó cộng tổng tất cả các điểm lại sẽ được một giá trị của ma trận đặc trưng.

Tổng số cột của ma trận đặc trưng bằng với tổng số lần tích chập được thực hiện khi đi hết một hàng và tổng số hàng của ma trận đặc trưng bằng với tổng số lần tích chập được thực hiện khi đi hết một cột.

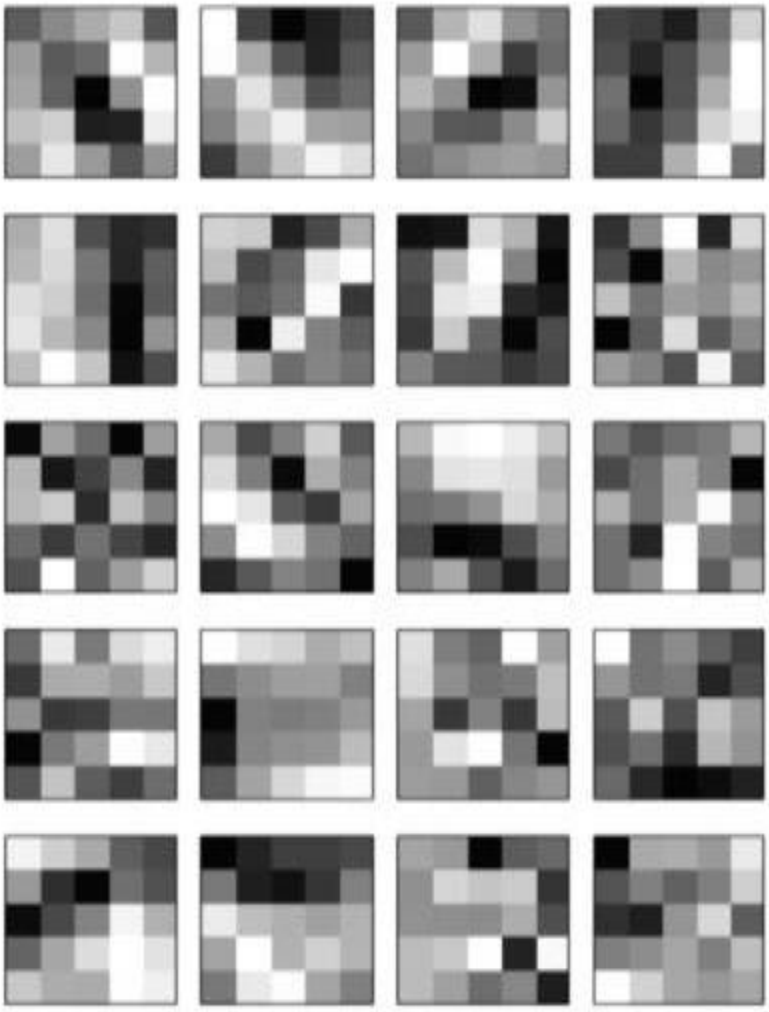
⇨ Lớp tích chập là phần cốt lõi của một mạng nơ ron tích chập, và khối lượng tính toán của lớp này chiếm đa số trong một mạng nơ ron tích chập.

Giống như các lớp ẩn khác, lớp tích chập lấy dữ liệu đầu vào, thực hiện các phép chuyển đổi để tạo ra dữ liệu đầu vào cho lớp kế tiếp (đầu ra của lớp này là đầu vào của lớp sau). Phép biến đổi được sử dụng là phép tính tích chập. Mỗi lớp tích chập chứa một hoặc nhiều bộ lọc - bộ phát hiện đặc trưng cho phép phát hiện và trích xuất những đặc trưng khác nhau của ảnh. Và sau khi thực hiện xong phép tính tích chập, thông tin sẽ được lưu tại bản đồ đặc trưng (feature map).



Hình 2.5: Mô tả sau khi thực hiện phép tích chập tại lớp tích chập

Ngoài ra, độ phức tạp của đặc trưng được phát hiện bởi bộ lọc tỉ lệ thuận với độ sâu của lớp tích chập mà nó thuộc về. Trong mạng CNN, những lớp tích chập đầu tiên sử dụng bộ lọc hình học để phát hiện những đặc trưng đơn giản như cạnh ngang, dọc, chéo của bức ảnh. Tiếp sau đó, những lớp tích chập tiếp đến được dùng để phát hiện đối tượng nhỏ hơn, mang tính đặc trưng rõ ràng của đối tượng xuất hiện trong ảnh đầu vào như là mắt, mũi, và các loại bộ phận trên cơ thể đối với vật thế là người và động vật, hoặc là những điểm đặc trưng nằm trên các đồ vật, vật thể đối với đối tượng mà nó nằm trên. Và cuối cùng, những lớp tích chập sâu nhất sẽ được dùng để phát hiện đối tượng hoàn chỉnh như là con người, ôtô, hoặc các loại của đối tượng nằm trong bức ảnh đầu vào.



Hình 2.6: Đầu ra trên lớp tích chập

Ở hình 2.6 trên là 20 ảnh tương ứng với 20 bản đồ đặc trưng khác nhau (hay còn gọi là bộ lọc, hay là nhân) đầu ra của một lớp tích chập. Mỗi bản đồ được thể hiện là một hình khối kích thước 5×5. Những hình ảnh trên cho thấy các kiểu đặc trong mà lớp tích chập đáp ứng.

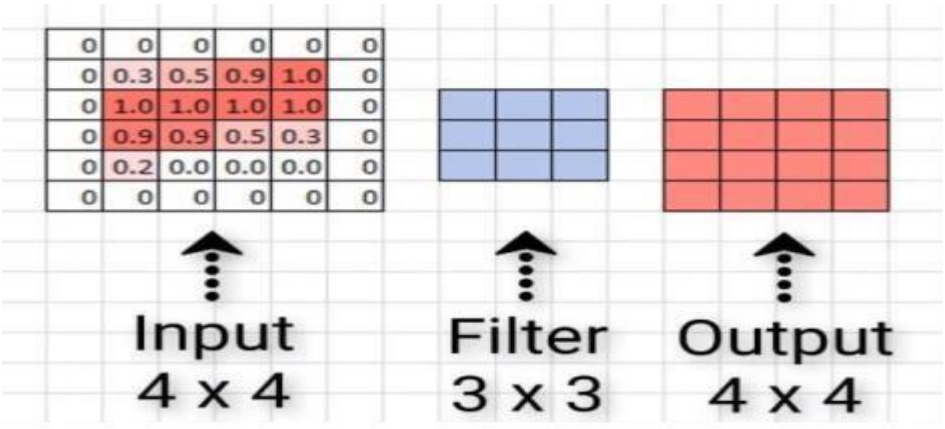
a. Padding

Ở đây chúng ta có ma trận đầu vào kích thước 4 × 4. Nếu ta chập với bộ lọc có kích thước 3 × 3, kết quả ta thu được một ma trận có kích thước 2 × 2.Thấy rằng kết quả đầu ra 2 × 2, trong khi đầu vào là 4 × 4, một lần nữa giống như ví dụ lớn hơn với hình ảnh số chúng ta thấy rằng đầu ra thực sự nhỏ hơn đầu vào về mặt kích thước. Chúng ta có thể biết trước về việc các kích thước của chúng ta sẽ thu hẹp bao nhiêu.

Tổng quát, nếu ta chập ma trận đầu vào kích thước $n \times n$ với một bộ lọc kích thước $f \times f$, kết quả thu được là một ma trận kích thước $(n - f + 1) \times (n - f + 1)$.

Mỗi một lần áp dụng phép chập, kích thước của ảnh bị giảm xuống. Khi tiếp tục thực hiện quá trình chập, số lượng thực hiện chỉ một vài lần trước khi kích thước của ảnh quá nhỏ khiến chúng ta không thể thực hiện được. Ngoài ra, điểm ảnh ở khoảng trung tâm của ma trận đầu vào được bao phủ bởi rất nhiều vùng 3 × 3 nghĩa là được sử dụng để tính nhiều giá trị đầu ra, trong khi những điểm ảnh ở góc hoặc cạnh chỉ được sử dụng 1 hoặc 2 lần vì chỉ bị bao phủ bởi 1 hoặc 2 vùng 3 × 3. Vì thế rất nhiều thông tin quan trọng tại các vùng gần cạnh của ảnh sẽ bị mất.

Để khắc phục nhược điểm này, một padding bằng không được thêm xung quanh ma trận đầu vào. Việc thêm padding làm tăng kích thước ma trận đầu vào dẫn tới sẽ tăng kích thước ma trận đầu ra. Từ đó độ chênh lệch ma trận đầu vào và ma trận đầu ra giảm. Những ô nằm trên cạnh/ góc của ma trận đầu vào gốc cũng lùi sâu vào bên trong hơn, dẫn tới được sử dụng nhiều hơn trong việc tính toán ma trận đầu ra, tránh được việc mất các thông tin.



Hình 2.7: Ma trận khi sử dụng Padding(giá trị bằng 0)

Trong hình 2.7, ma trận đầu vào kích thước 4x4 được thêm vào padding kích thước 1 ($p = 1$), trở thành ma trận 6x6. Khi chập ma trận này với bộ lọc 3×3, chúng ta thu được ma trận đầu ra 4×4. Kích thước của ma trận đầu vào (gốc) được duy trì. Những điểm ảnh nằm ở cạnh của ma trận đầu vào gốc được sử dụng nhiều lần hơn (4 lần với những điểm ảnh ở góc).

Theo quy ước, các ô trên padding có giá trị bằng không, p là kích thước của padding. Trong hầu hết các trường hợp, padding đối xứng trái-phải, trên-dưới so với ma trận gốc, vì thế kích thước của ma trận đầu vào được tăng lên $2p$ mỗi chiều. Ma trận đầu ra do đó có kích thước.

$$(n + 2p - f + 1) * (n + 2p - f + 1)$$

với fxf là kích thước của bộ lọc, giả dụ với bộ lọc 3 × 3 thì ta sẽ có f = 3.
Tuỳ theo giá trị của p, chúng ta có hai trường hợp chính:
Nhân chập không dùng padding (valid convolution) - NO padding:

$$(n * n) * (f * f) => (n - f + 1) * (n - f + 1)$$

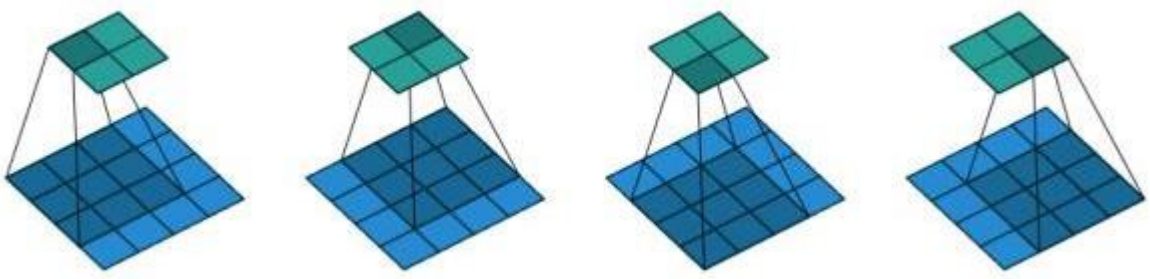
Nhân chập không làm thay đổi kích thước đầu vào (same convolution): Kích thước padding được tính theo công thức:

$$n + 2p - f + 1 = n => p = \frac{f - 1}{2}$$

Theo quy ước, kích thước của bộ lọc f thường là số lẻ bởi vì hai lý do chính sau:

- Nếu $\boldsymbol{f}$ là số chẵn, chúng ta phải thêm vào bên trái của ma trận đầu vào nhiều hơn bên phải (hoặc ngược lại), việc này dẫn tới hệ đầu vào không đối xứng.
- Nếu $\boldsymbol{f}$ là số lẻ, ma trận đầu vào có một điểm ảnh ở trung tâm. Trong lĩnh vực thị giác máy tính, việc có một nhân tố khác biệt - một điểm đại diện cho vị trí của bộ lọc thường mang lại hiệu năng cao cho bài toán.

b. Strided convolution



Hình 2.8: Strided convolution[8]

Phép chập sải có nghĩa là sau khi ta thực hiện phép chập ở một vị trí nào đó xong, nó sẽ thực hiện stride đến vị trí mới. Độ dài bước (length of stride) sải này sẽ dựa vào thông số sải (stride parameter) mà ta định nghĩa trước khi thực hiện phép chập.

Trong phép chập ở hình 2.8, bộ lọc trượt trên ma trận đầu vào với thông số sải bằng 1, có nghĩa là sau khi thực hiện xong phép chập ở lần đầu tiên thì lần tiếp đến nó sẽ bỏ qua 1 ô để đến với ô ngang tiếp theo là lần chập tiếp theo của bộ lọc đối với lần trượt là ngang,

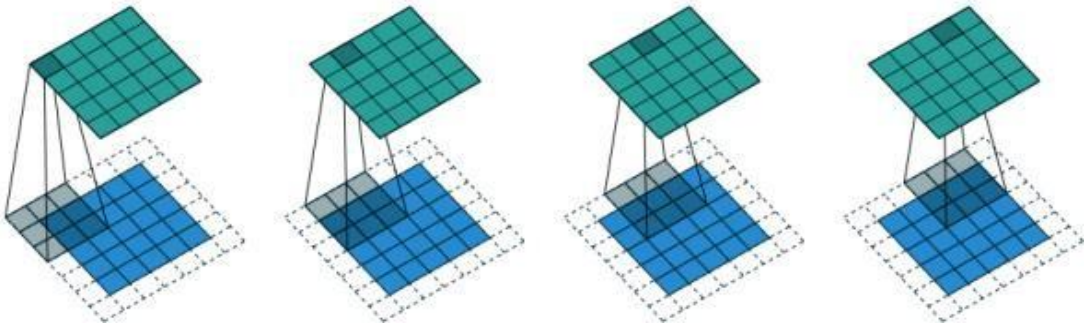
và bỏ qua một 1 ô để đến với ô dọc tiếp theo đối với lần trượt là dọc. Ngoài giá trị 1 thì giá trị này cũng có thể bằng 2, 3 hoặc lớn hơn tuỳ vào mục đích của người sử dụng.

Vậy, để thuận tiện trong việc tổng quát hoá, ta định nghĩa số hàng/ cột mà bộ lọc trượt qua trong một bước di chuyển ký hiệu là s. Kích thước ma trận đầu ra lúc này được tính bởi:

$$\left(\frac{n+2p-f}{s}\right) * \left(\frac{n+2p-f}{s}\right)$$

Với n : kích thước đầu vào, p : số lượng padding, f : kích thước của bộ lọc

Trong trường hợp mà biểu thức $n + 2p - f$ không chia hết cho s thì ta sẽ tiến hành lấy cận dưới của biểu thức đó. Ví dụ: với kích thước đầu vào là $n = 6$, kích thước của bộ lọc là $f = 3$, padding của nó là $p = 1$ và giá trị sải của nó là $s = 2$ thì sau khi tính toán ta có $\frac{6+2*1-3}{2} = 2.5$ sau khi tiến hành lấy cận dưới của nó thì kích thước đầu ra thu được sẽ là 2×2.



Hình 2.9: Strided convolution a 3x3 kernel over a 5x5 input, s = 1, p =1[8]

2.3.2.2 Lớp kích hoạt (Activation layer)

Lớp kích hoạt (activation layer) mô phỏng tỷ lệ truyền xung qua axon của một neuron thần kinh. Trong một mạng nơ-ron nhân tạo, hàm kích hoạt đóng vai trò là thành phần phi tuyến tại output của các nơ-ron.

Thông thường, cứ sau mỗi lớp tích chập, ta sẽ xét một lớp phi tuyến (lớp kích hoạt) ngay lập tức sau đó. Mục đích của việc sử dụng lớp này để giúp các nút có thể học các cấu trúc phức tạm trong bộ dữ liệu, điều không thể khi ta chỉ sử dụng hàm tuyến tính tại lớp kích hoạt.

Các hàm kích hoạt phổ biến

- Hàm Sigmoid
- Hàm Tanh
- Hàm ReLU
- Hàm Leaky ReLU
- Maxout

a. Hàm Sigmoid

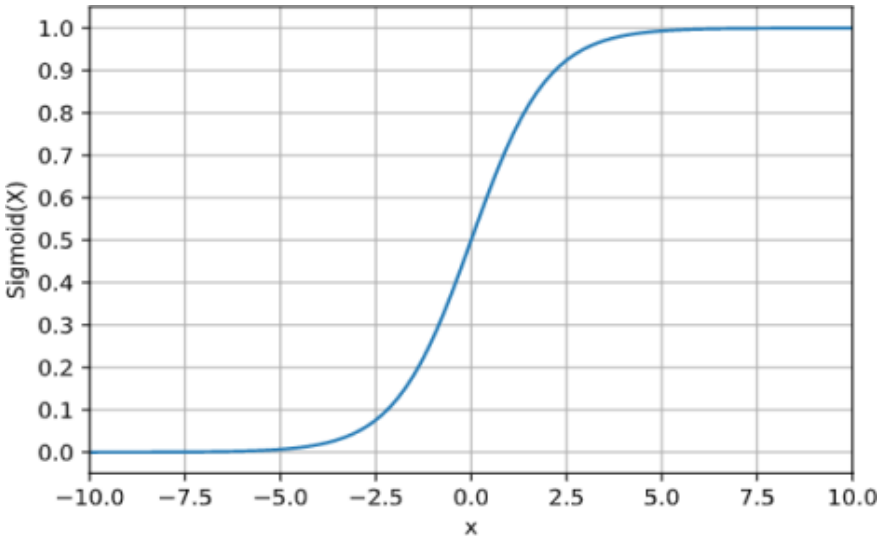
Hàm Sigmoid nhận đầu vào là một số thực và chuyển thành một giá trị trong khoảng (0;1). Đầu vào là số thực âm rất nhỏ sẽ cho đầu ra tiệm cận với 0, ngược lại, nếu đầu vào là một số thực dương lớn sẽ cho đầu ra là một số tiệm cận với 1. Trong quá khứ hàm Sigmoid hay được dùng vì có đạo hàm rất đẹp.

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Tuy nhiên hiện nay hàm Sigmoid rất ít được dùng vì những nhược điểm sau:

Hàm Sigmoid bão hòa và triệt tiêu gradient: Một nhược điểm dễ nhận thấy là khi đầu vào có trị tuyệt đối lớn (rất âm hoặc rất dương), gradient của hàm số này sẽ rất gần với 0. Điều này đồng nghĩa với việc các hệ số tương ứng với unit đang xét sẽ gần như không được cập nhật (còn được gọi là vanishing gradient).

Hàm Sigmoid không có trung tâm là 0 gây khó khăn cho việc hội tụ.



Hình 2.10:Đồ thị hàm Sigmoid[5]

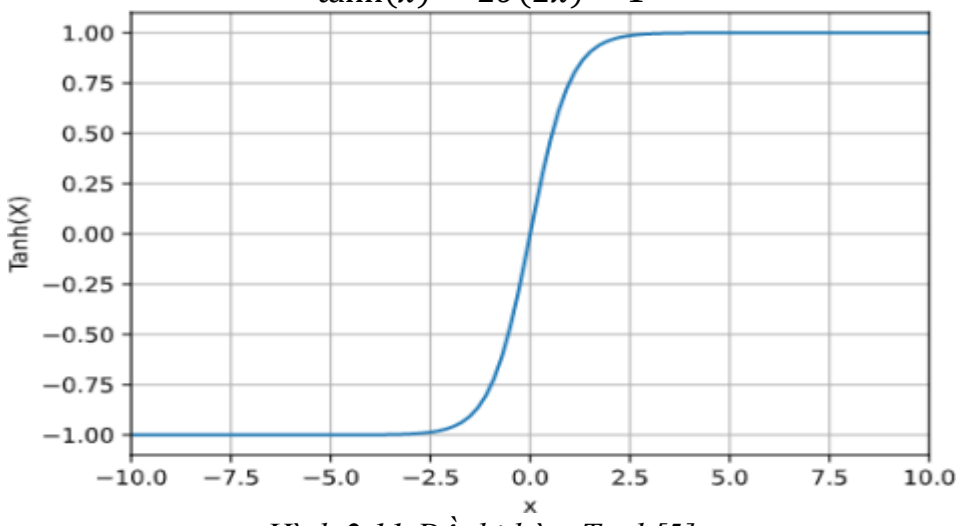
b. Hàm Tanh

Hàm Tanh nhận đầu vào là một số thực và chuyển thành một giá trị trong khoảng (-1; 1). Cũng như sigmoid, hàm tanh bị bão hoà ở 2 đầu (gradient thay đổi rất ít ở 2 đầu). Tuy nhiên hàm tanh lại đối xứng qua 0 nên khắc phục được một nhược điểm của Sigmoid.

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

Hàm Tanh còn có thể biểu diễn bằng hàm sigmoid như sau:

$$\tanh(x) = 2\sigma(2x) - 1$$



Hình 2.11:Đồ thị hàm Tanh[5]

c. Hàm ReLU

Hàm ReLU đang được sử dụng khá nhiều trong những năm gần đây khi huấn luyện các mạng neuron. ReLU đơn giản lọc các giá trị < 0. Nhìn vào công thức chúng ta dễ dàng hiểu được cách hoạt động của nó.

$$f(x) = \max(0, x)$$

Một số ưu điểm khá vượt trội của nó so với Sigmoid và Tanh:

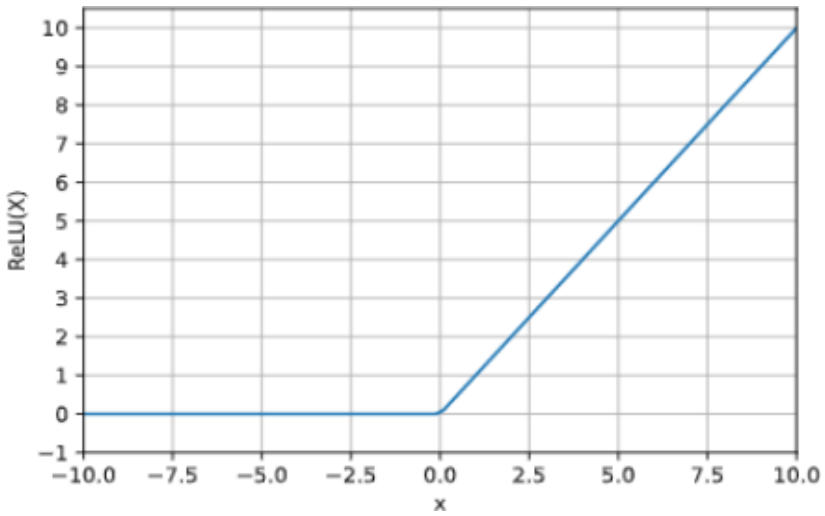
- Tốc độ hội tụ nhanh hơn hẳn. ReLU có tốc độ hội tụ nhanh gấp 6 lần Tanh. Điều này có thể do ReLU không bị bão hoà ở 2 đầu như Sigmoid và Tanh.
- Tính toán nhanh hơn. Tanh và Sigmoid sử dụng hàm exp và công thức phức tạp hơn ReLU rất nhiều do vậy sẽ tốn nhiều chi phí hơn để tính toán.

Tuy nhiên ReLU cũng có một nhược điểm:

- Với các node có giá trị nhỏ hơn 0, qua ReLU activation sẽ thành 0, hiện tượng đấy gọi là "Dying ReLU". Nếu các node bị chuyển thành 0 thì sẽ không có ý nghĩa với bước linear activation ở lớp tiếp theo và các hệ số tương ứng từ node đấy cũng không được cập nhật với gradient descent. => Leaky ReLU ra đời.

• Khi learning rate lớn, các trọng số (weights) có thể thay đổi theo cách làm tất cả neuron dừng việc cập nhật.

Hình 2.12:Đồ thị hàm ReLU[5]

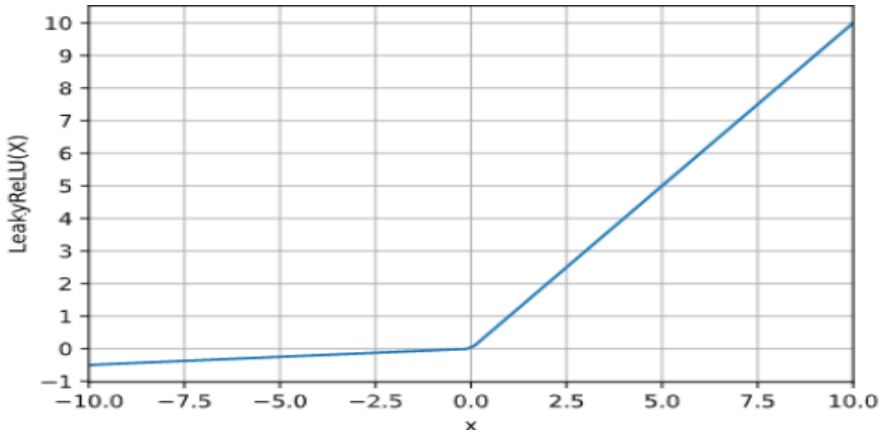


d. Hàm Leaky ReLU

Leaky ReLU là một cố gắng trong việc loại bỏ "dying ReLU". Thay vì trả về giá trị 0 với các đầu vào < 0 thì Leaky ReLU tạo ra một đường xiên có độ dốc nhỏ (xem đồ thị). Có nhiều báo cáo về việc hiệu Leaky ReLU có hiệu quả tốt hơn ReLU, nhưng hiệu quả này vẫn chưa rõ ràng và nhất quán.

$$f(x) = 1(x < 0)(\alpha x) + 1(x \geq 0)(x)$$

Hình 2.13:Đồ thị hàm Leaky ReLU[5]



e. Tổng quát

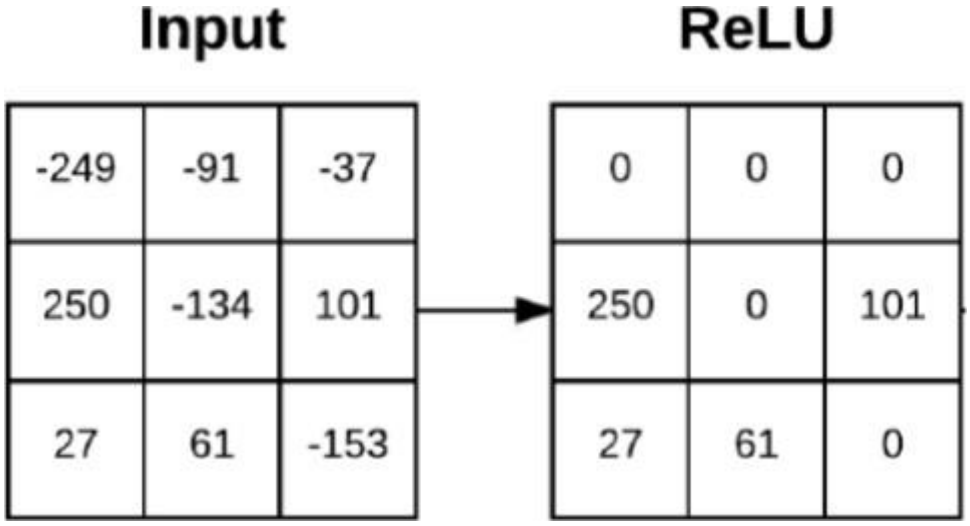
Trong quá khứ, người ta thường sử dụng các hàm phi tuyến như là sigmoid, tanh,nhưng trong những năm gần đây người ta nhận thấy rằng hàm ReLU khi đưa vào trong mạng thì mạng sẽ huấn luyện nhanh hơn, ngoài ra việc tính toán đạo hàm của nó cũng rất đơn giản. Bên cạnh đó, việc sử dụng hàm ReLU sẽ giúp cho mạng tránh được hiện tượng vanishing gradient do đặc, tức là hiện tượng mà đạo hàm không thay đổi khi đi ngược về

các lớp sau cùng, đối với quá trình lan truyền ngược, khiến cho sự thay đổi trọng số ở các lớp đó không đáng kể.

Vậy với hàm ReLU, công thức của nó là: $f(x) = \max(0, x)$

Từ công thức này, ta có thể thấy rằng với những giá trị âm thì hàm sẽ qui nó về giá trị không, và giữ nguyên những giá trị dương.

Ngoài ra với một bức ảnh đầu vào có kích thước là $WxHxC$ thì sau khi đi qua lớp kích hoạt thì kích thước của bức ảnh đầu ra vẫn sẽ được giữ nguyên, tức là $WxHxC$.



Hình 2.14:Ví dụ một ma trận sau khi đi qua lớp ReLU

2.3.2.3 Lớp gộp (Pooling layer)

Pooling layer thường được dùng giữa các convolutional layer, có chức năng làm giảm kích thước nhưng vẫn giữ được thuộc tính quan trọng. Kích thước dữ liệu giảm giúp việc tính toán trong model.

Gọi pooling size kích thước $K * K$. $Input$ của pooling layer có kích thước **H*W*D**, ta tách ra làm **D** ma trận kích thước **H*W.** Với mỗi ma trận, trên vùng kích thước $K * K$ trên ma trận ta tìm maximum hoặc average của dữ liệu rồi viết vào mà trận kết quả . Gọi pooling size kích thước **K*K.** $Input$ của pooling layer có kích thước **H*W*D**, ta tách ra làm D ma trận kích thước **H*W.** Với mỗi ma trận, trên vùng kích thước **K*K** trên ma trận ta tìm maximum hoặc average của dữ liệu rồi viết vào mà trận kết quả.Quy tắc về stride và padding áp dụng như phép tính convolution.

Nhưng hầu hết khi dùng pooling layer thì sẽ dùng size=(2,2), stride=2, padding=0. Khi đó output width height của dữ liệu giảm đi một nửa, depth thì vẫn giữ nguyên.

Thêm nữa, lớp pooling còn tóm tắt các đặc trưng xuất hiện trong một vùng của bản đồ đặc trưng sau khi thực hiện với lớp tích chập.

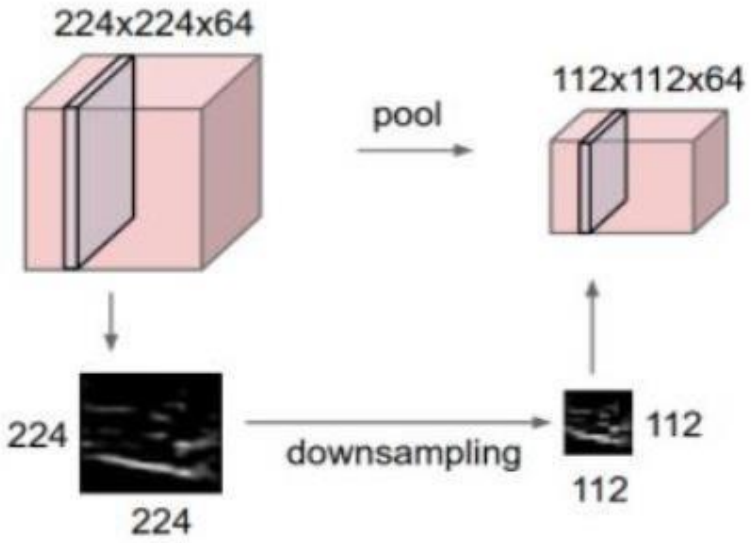
Nhằm để thuận lợi cho việc tính toán, ta tiến hành tổng quát hoá việc tính toán đầu ra của lớp pooling. Kích thước đầu ra của ma trận lúc này là:

$$\left(\frac{n+2p-f}{s}+1\right)*\left(\frac{n+2p-f}{s}+1\right)$$

Với n : kích thước đầu vào, p : số lượng padding, f: kích thước của bộ lọc

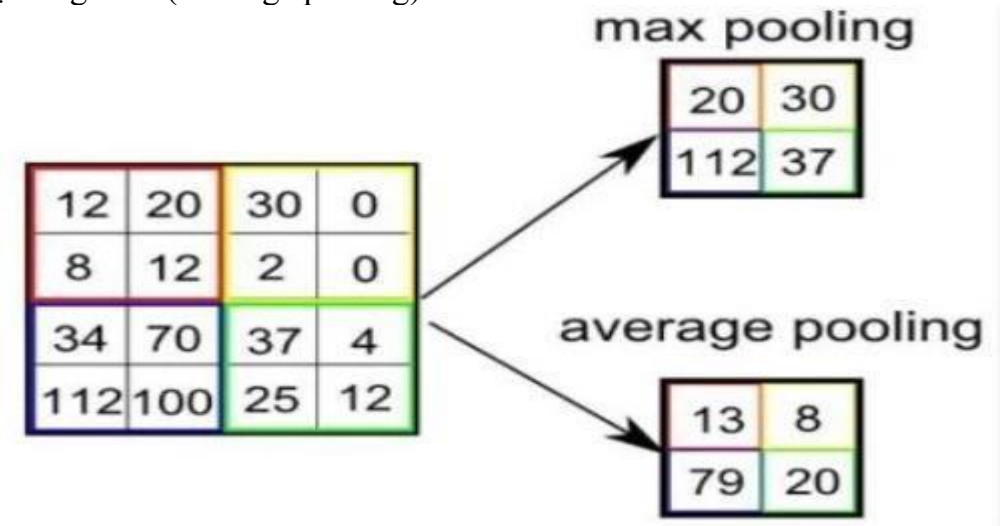
Với kích thước đầu ra này đối với trường hợp số lẻ thì ta cũng tiến hành thực hiện lấy cận dưới của nó.

Thông thường, lớp pooling sẽ được xét ngay sau lớp tích chập và lớp ReLU.



Hình 2.15:Mô tả Pooling layer[6]

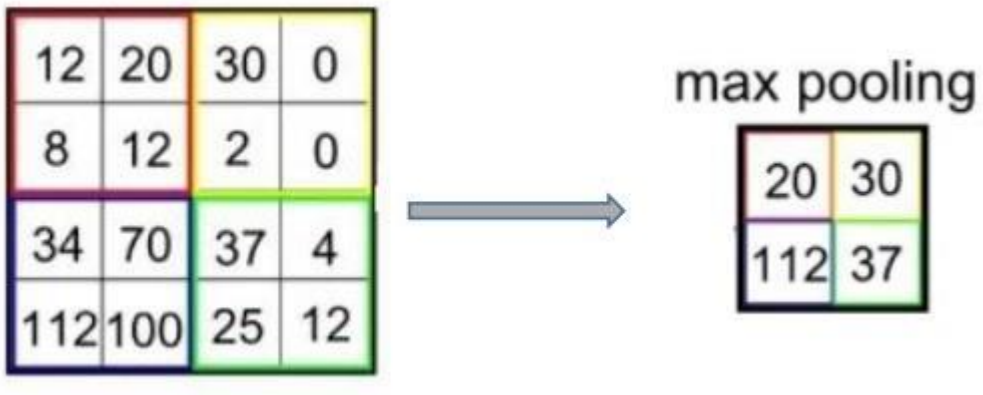
Ngoài ra, có 2 loại pooling layer phổ biến: lấy giá trị cực đại (Max pooling) và lấy giá trị trung bình (Average pooling).



Hình 2.16:Mô tả Max pooling và Average Pooling

a.Max pooling

Phép toán này mô tả khi bộ lọc trượt tại một vị trí bất kì nào đó thì giá trị cao nhất trong vùng sẽ được giữ lại và những giá trị còn lại bị loại bỏ. Max pooling được sử dụng rất rộng rãi trong các mạng nơ ron tích chập.

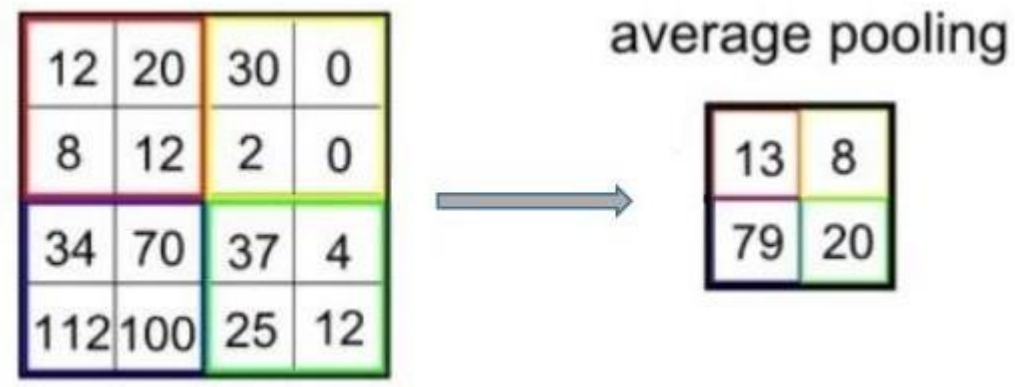


Hình 2.17:Mô tả Max pooling[6]

Ví dụ, trong hình 2.17, ta có thể thấy rằng sau khi áp dụng phép max pooling thì tại vị trí mà nó bị trượt bởi bộ lọc, các điểm ảnh nằm trong vị trí trượt thì sẽ qui định cùng một ô viền màu, thì những giá trị cao nhất vị trí đó sẽ giữ lại, còn những điểm ảnh khác sẽ bị loại bỏ đi.

b.Average Pooling

Trong phép toán này, thay vì lấy giá trị lớn nhất của một lần trượt, ta sẽ lấy trung bình của tất cả những điểm ảnh bị bao phủ bởi bộ lọc khi nó trượt trên từng phần của ma trận đầu vào. Nên, trong khi max pooling sẽ lấy những điểm đặc trưng nhất trong một phần của ma trận đầu vào thì average pooling sẽ lấy giá trị trung bình của đặc trưng nằm trong phần đó.



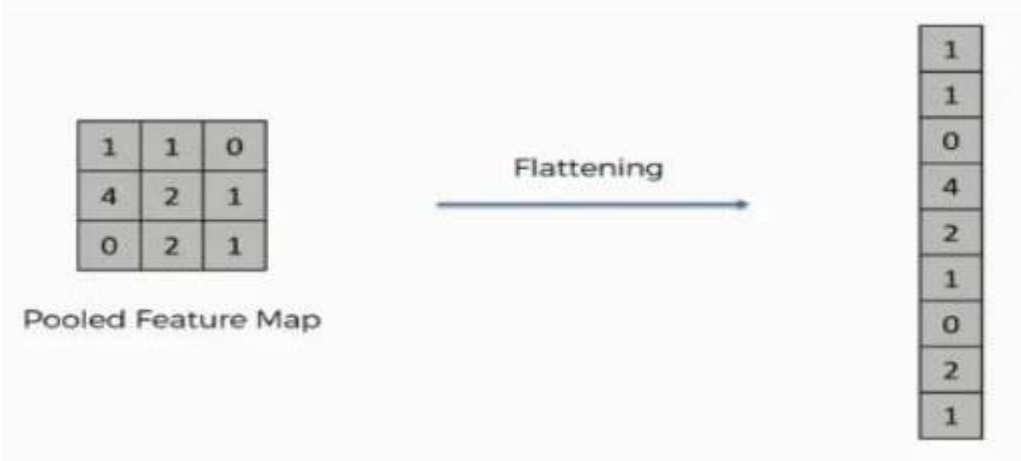
Hình 2.18:Mô tả Average Pooling[7]

Ví dụ, trong hình 2.18, ta có thể thấy rằng khi thực hiện phép Average pooling thì điểm viền màu vàng thay vì lấy giá trị lớn nhất là 30, thì trong phép này nó lấy trung bình cộng của bốn điểm nên giá trị sau khi thực hiện phép tính là 8.

2.3.2.4 Lớp liên kết đầy đủ (Fully-connected layer)

Sau khi ảnh được truyền qua nhiều convolutional layer và pooling layer thì model đã học được tương tối các đặc điểm về ảnh(ví dụ mắt,mũi,khung mặt,..) thì tensor của output của layer cuối cùng, kích thước $H * W * D$, sẽ được chuyển về 1 vector kích thước ($H * W * D$).

Sau đó ta dùng các fully connected layer để kết hợp các đặc điểm của ảnh để ra được output của model.



Hình 2.19:Quá trình chuyển về một vector[7]

Bây giờ vector này sẽ được đưa vào mạng neu-ron. Nói cách khác, vector này bây giờ sẽ trở thành lớp đầu vào của mạng nơ ron vào mạng nơ ron tích tụ hiện có.

2.3.3 Batch Normalization

Trong việc huấn luyện mạng Deep Learning, có hai vấn đề đáng chú ý là non- zero mean (hiện tượng dữ liệu không phân bố quanh giá trị 0, mà dữ liệu có phần nhiều giá trị lớn hơn 0 hoặc nhỏ hơn 0) và high variance (độ lệch chuẩn cao) khiến dữ liệu có nhiều thành phần rất lớn hoặc rất nhỏ. Tuỳ vào cách khởi tạo trọng số mà khi huấn luyện mạng càng sâu thì độ lệch càng nghiêm trọng. Độ lệch lớn này buộc các layer phía sau phải học theo các thay đổi trong trọng số và bias của layer phía trước (vì output của layer trước là input của layer sau). Hậu quả là thời gian huấn luyện tăng nhưng độ ổn định lại giảm. Ngoài ra vấn đề này còn ảnh hưởng đến các hàm kích hoạt phi tuyến tính như tanh, sigmoid, ReLU. Các hàm này đều có các vùng bão hoà (tức giá trị đầu ra không phụ thuộc giá trị đầu vào). Khi dữ liệu quá lớn hoặc quá nhỏ và rơi vào vùng bão hoà của hàm activation thì giá trị đầu ra rất giống nhau. Phương pháp rất hiệu quả để giải quyết vấn đề này là chuẩn

hoá dữ liệu về trạng thái zero mean (giá trị trung bình bằng 0) và unit variance (độ lệch chuẩn bằng 1). Phương pháp này được gọi là Batch Normalization.

Trong việc huấn luyện mạng học sâu, có việc thay đổi trọng số và bias của một lớp có thể dẫn đến sự thay đổi lớn ở lớp sau đó, khiến việc huấn luyện mất thời gian và không hiệu quả. Batch normalization ra đời nhằm khắc phục tình trạng này.

Trong quá trình huấn luyện, lớp Batch Normalization hoạt động như sau: Đầu tiên, tính giá trị trung vị và phương sai của dữ liệu đầu vào:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \sigma^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

Tiếp đến, chuẩn hóa dữ liệu đầu vào bằng các thông số trên:

$$\widehat{x_l} = \frac{x_i - \mu_B}{\sqrt{\sigma^2 + \epsilon}}$$

Cuối cùng, tính dữ liệu đầu ra:

$$y_i = \gamma\widehat{x_l} + \beta$$

Trong đó: x là dữ liệu đầu ra, $\hat{x}$ là giá trị đã chuẩn hoá, μ_B là giá trị trung bình, $\sigma 2$ phương sai và γ, β là tham số học được.

Trong quá trình test, giá trị trung bình và độ lệch chuẩn được tính từ những giá trị tương ứng trong quá trình train:

$$E_x = \frac{1}{m}\sum_{i=1}^{j} \mu_B$$

$$Var_x = \left(\frac{m}{m-1}\right)\frac{1}{m}\sum_{i=1}^{j} {\sigma^2}_B$$

$$y = \frac{\gamma}{\sqrt{Var_x + \epsilon}}x + (\beta + \frac{\gamma E_x}{\sqrt{Var_x + \epsilon}})$$

Với m là số điểm dữ liệu trong mỗi batch, j là số batch.

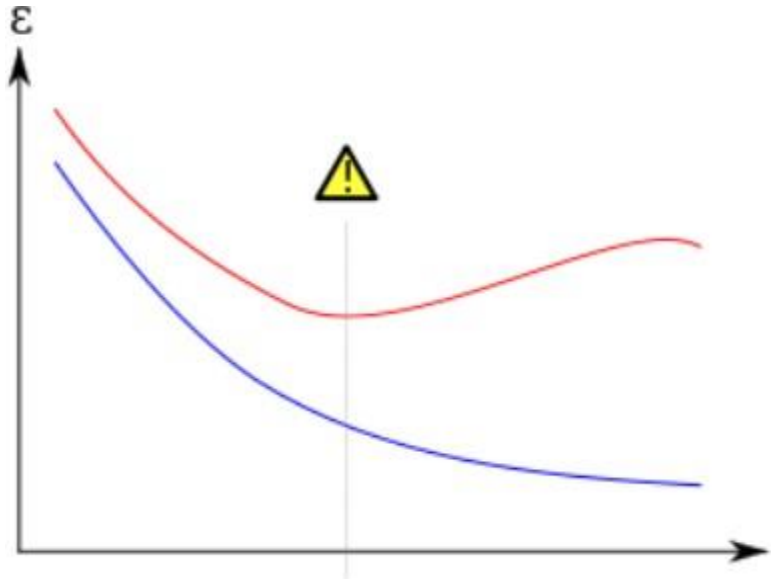
2.3.4 Early Stopping

Khi training model thì không phải lúc nào (hàm mất mát) loss của tập train và tập test cũng đồng thời giảm, tới một epoch nào đó thì loss của tập train sẽ tiếp tục giảm nhưng loss của tập test không giảm mà tăng trở lại => Đó là hiện tượng overfitting. Vì vậy để ngăn chặn nó, thì ngay tại thời điểm đó người ta sẽ dừng việc training (vì để chương trình tiếp tục training thì cũng không cải thiện được gì mà lại tốn tài nguyên). Early stopping tức dừng thuật toán trước khi hàm mất mát đạt giá trị quá nhỏ, giúp tránh overfitting.

Vậy dừng khi nào là phù hợp?

Một kỹ thuật thường được sử dụng là tách từ training set ra một tập validation set như trên. Sau một (hoặc một số, ví dụ 50) vòng lặp, ta tính cả train error và validation error, đến khi validation error có chiều hướng tăng lên thì dừng lại, và quay lại sử dụng mô hình tương ứng với điểm và validation error đạt giá trị nhỏ.

Hình 2.20:Early stopping



Hình 2.20 mô tả cách tìm điểm stopping . Đường màu xanh là train error, đường màu đỏ là validation error. Trục x là số lượng vòng lặp, trục y là error. Mô hình được xác định tại vòng lặp mà validation error đạt giá trị nhỏ nhất.

2.3.5 Softmax

Softmax là một cách ràng buộc đầu ra của các mạng nơ ron phải có tổng bằng 1. Qua đó, các giá trị đầu ra của hàm softmax có thể được coi như là một phân phối xác suất của các biến đầu ra. Nó rất hữu ích trong bài toán phân loại đa lớp. Softmax là một loại của hàm kích hoạt - activation function chúng ta đã bàn luận ở phần phía trên với điều đặc biệt đó là kết quả đầu ra của nó sẽ có tổng bằng 1. Để làm được điều này hàm softmax sẽ chuyển đổi giá trị đầu ra của mạng nơ ron bằng cách chia cho tổng giá trị. Lúc này đầu ra có thể coi là một vector của xác suất dự đoán của các class. Chúng ta có thể thấy rõ hơn trong công thức sau:

$$Softmax\ x_i = \frac{e^{a_i}}{\sum e^{a_i}}$$

2.3.6 Cross Entropy

Cross-entropy sử dụng để so sánh khoảng cách giữa các giá trị đầu ra của softmax và one-hot encoding. Cross-entropy là một hàm loss và giá trị của nó có thể được cực tiểu hoá (minimized). Điều này giúp cho một neural networks đánh giá được xác suất (độ chắc chắn) của phép dự đoán một mẫu dữ liệu tương ứng với một class. Xác suất sẽ là lớn nhất đối với biến mục tiêu của chúng ta. Cross entropy là tổng của các xác suất logarit âm. Chúng ta có thể định nghĩa nó theo công thức sau:

$$L = -\sum_{i=1}^{n} (y_j * Log(\hat{y}_j))$$

L ký hiệu hàm mất mát, y_j là kết quả thực, $\hat{y}_j$ là kết quả dự đoán

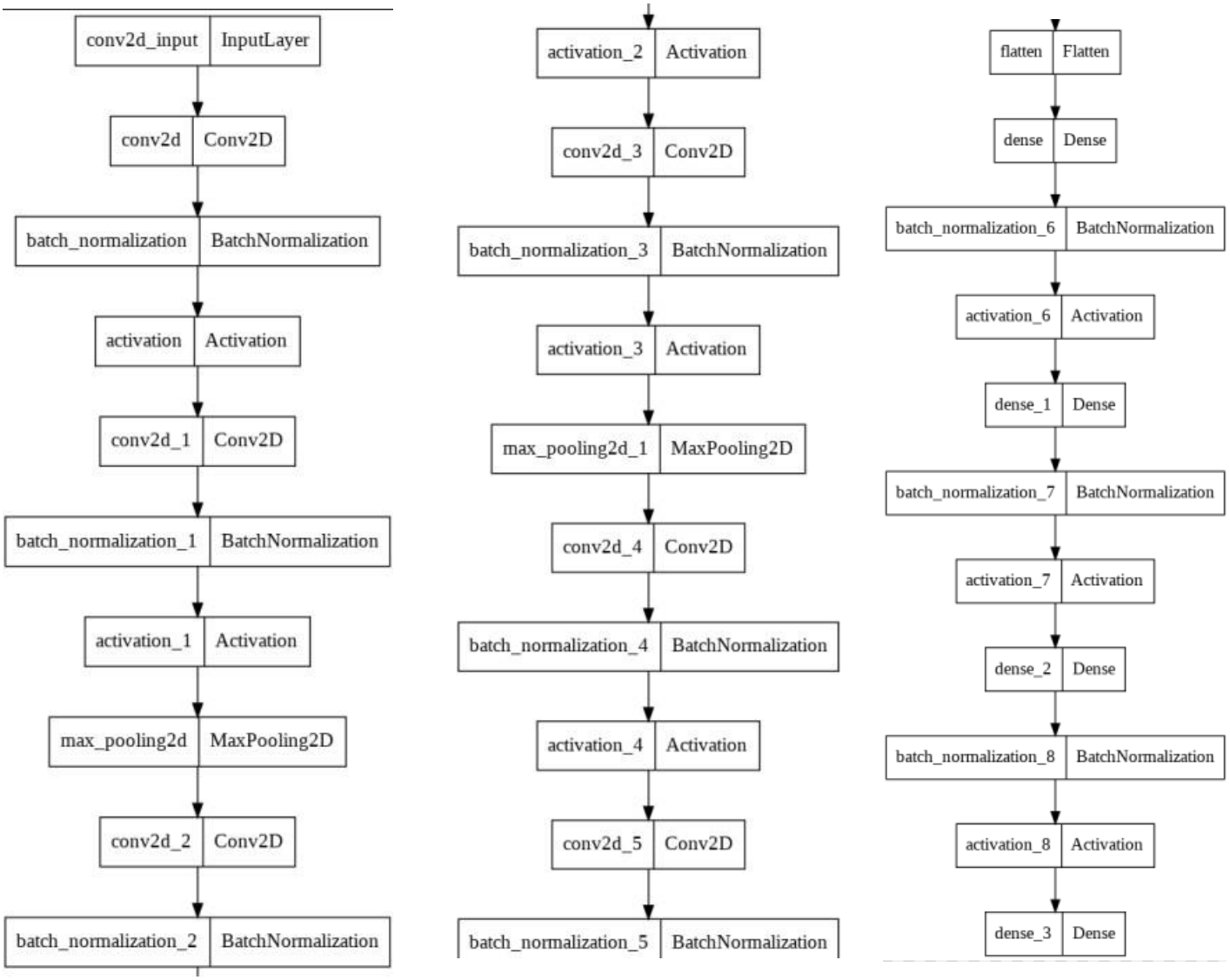
$$CrossEntropy = -[ylog(p) + (1-y)\log(1-p)]$$

Hàm logarit với giá trị âm được sử dụng để minimize hàm loss (maximize hàm logarit trong công thức tương ứng với việc minimize giá trị âm của nó). Mục đích của việc huấn luyện môt mạng nơ-ron nhân tạo tức là chúng ta sẽ tìm ra được một tập các tham số tối ưu cho bài toán của mình từ các dữ liệu đầu vào. Việc đó tương đương với các thao tác tính toán và cập nhật các trọng số - weights. Công việc tính toán và cập nhật các trọng số đó được gọi là lan truyền ngược - backpropagation. Thủ tục để cực tiểu hóa hàm lỗi - loss function được gọi là tối ưu hóa – optimization.

CHƯƠNG 3 :XÂY DỰNG MẠNG NƠ-RON TÍCH CHẬP CHO BÀI TOÁN NHẬN DẠNG CẢM XÚC KHUÔN MẶT

3.1 Xây dựng mạng nơ-ron tích chập

3.1.1 Mô tả kiến trúc mạng sử dụng trong bài toán



Hình 3.1: Kiến trúc mạng Deep CNN

Mạng Deep CNN có 6 lớp tích chập và 3 lớp full connected. Trong lớp tích chập đầu tiên được sử dụng 256 filter 3x3, đầu ra của lớp đầu tiên có kích thước 46*46*256 giảm đi so với ảnh đầu vào 48*48*256. Đối với lớp tích chập thứ 2 thì sử dụng 256 filter 3x3 và lớp padding 2 vì vậy kích thước đầu ra của lớp tích chập thứ 2 được bảo toàn bằng đầu ra của lớp thứ nhất, lớp maxpooling với kích thước filter là 2x2 và hệ số trượt strides 2x2 đầu ra của lớp này giảm đi một nữa là 23*23*128. Lớp tích chập thứ 3 sử dụng 128 filter 3x3 và padding bằng 2, lớp tích chập thứ 4 sử dụng số lượng filter là 128, padding bằng 2 và max

pooling 2x2, kích thước đầu ra của của 3 lớp này là 11*11*128. Tiếp đó là lớp tích chập thứ 5,6 với 64 filter 3x3 cùng với padding bằng 2 max pooling 2x2 và kích thước đầu ra chỉ còn 5*5*64. Đối với mạng Deep CNN ở sau mỗi lớp tích chập đều sử dụng batch nomarlization, và hàm kích hoạt Rectified Linear Unit (ReLU).

Đối với lớp full connected sử dụng trong mạng này có 3 lớp ẩn với số nơ-ron lần lượt là 512, 256, 128, sử dụng Rectified Linear Unit (ReLU) là hàm kích hoạt và sử dụng softmax là hàm tối ưu hàm mất mát

Siêu tham số sử dụng trong huấn luyện model

STT	Siêu tham số	Giá trị
1	Learning rate	0.001
2	regularization	1e-7
3	Batch size	64

Bảng 2.1: Giá trị các siêu tham số

- Learning rate – Tốc độ học là một siêu tham số sử dụng trong việc huấn luyện các mạng nơ ron. Giá trị của nó là một số dương, thường nằm trong khoảng giữa 0 và 1. Tốc độ học kiểm soát tốc độ mô hình thay đổi các trọng số để phù hợp với bài toán. Tốc độ học lớn giúp mạng nơ ron được huấn luyện nhanh hơn nhưng cũng có thể làm giảm độ chính xác.
- Chính quy hoá (regularization) là một kĩ thuật giúp giảm lỗi khớp quá bằng cách thêm một phần chính quy hoá vào hàm lỗi như sau:

$$j(v) = E_x(v) + \lambda E_v(v)$$

$E_X(v)$ là hàm lỗi ban đầu và cụm $\lambda E_v(v)$ mới thêm vào là số hạng chính quy hoá đóng vai trò như một biện pháp phạt lỗi (penalization).

- Trong đó, hệ số chính quy hoá λ được chọn từ trước để cân bằng giữa $E_x(v)$ và $E_v(v)$. λ càng lớn thì ta càng coi trọng $E_v(v)$, ít coi trọng tham số cho hàm lỗi ban đầu hơn, dẫn tới việc các tham số v ít có ảnh hưởng tới mô hình hơn. Hay nói cách khác là mô hình bớt phức tạp đi tránh lỗi quá khớp.

3.1.2 Thuật toán tối ưu Adam

Adam là một thuật toán tối ưu hóa có thể được sử dụng thay cho gradient descent cổ điển để cập nhật lặp đi lặp lại trọng số mạng dựa trên dữ liệu huấn luyện.

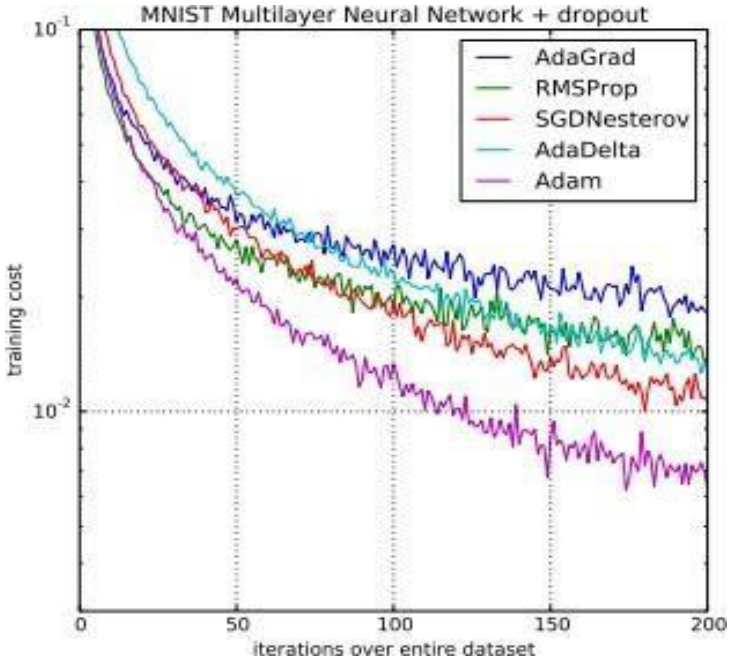
Những ưu điểm của thuật toán Adam

- Dễ dàng thực hiện
- Hiệu quả về mặt tính toán
- Yêu cầu ít bộ nhớ
- Thích hợp với các bài toán có độ biến thiên không ổn định và dữ liệu traning phân mảnh.

- Các siêu tham số được biến thiên một cách hiệu quả và yêu cầu ít điều chỉnh

Đối với thuật toán tối ưu gradien decent thì tốc độ học (learning rate) là một giá trị cố định trong quá trình cập nhật trọng số và không thay đổi trong quá trình huấn luyện model.

Adam điều chỉnh learning rate trong quá trình huấn luyện model, thuật toán này tính toán learning rate thích hợp cho từng tham số khác nhau, bằng việc ước tính thời điểm đầu tiên và tiếp theo của gradients.



Hình 3.2:So sánh thuật toán tối ưu Adam với thuật toán khác cùng huấn luyện perceptron nhiều lớp[4]

3.2 Giới thiệu Python, OpenCV, TensorFlow và Keras

Python

Ngôn ngữ Python là một trong những ngôn ngữ lập trình được nhiều người sử dụng nhất hiện nay, nhất là những người mới bắt đầu học lập trình. Không chỉ dễ dàng sử dụng mà ngôn ngữ Python còn mang tính hướng đối tượng.

Python có cấu trúc dữ liệu cao cấp và hệ thống thư viện lớn nhưng lại có thể tiếp cận đơn giản và vô cùng dễ hiểu. Điểm cộng lớn nhất của ngôn ngữ này chính là sự đơn giản, linh động, và có thể kết hợp với bất kỳ ngôn ngữ lập trình khác, được sử dụng trên nhiều nền tảng và sử dụng để phát triển nhiều ứng dụng.

Các đặc điểm của Python

- Cú pháp rất tường minh, dễ đọc.
- Các khả năng tự xét mạnh mẽ.
- Hướng đối tượng trực giác.
- Cách thể hiện tự nhiên mã thủ tục.
- Hoàn toàn mô-đun hóa, hỗ trợ các gói theo cấp bậc.
- Xử lý lỗi dựa theo ngoại lệ

- Kiểu dữ liệu động ở mức rất cao.
- Các thư viện chuẩn và các mô-đun ngoài bao quát hầu như mọi việc.
- Phần mở rộng và mô-đun dễ dàng viết trong C, C++.
- Có thể nhúng trong ứng dụng như một giao dạng kịch bản (scripting interface).
- Python mạnh mẽ và thực hiện rất nhanh.

OpenCV

OpenCV (Open Computer Vision) là một thư viện mã nguồn mở hàng đầu cho xử lý về thị giác máy tính, machine learning, xử lý ảnh. 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 cộng đồng hơn 47 nghìn người dùng và số lượng download vượt quá 6 triệu lần.

OpenCV có rất nhiều ứng dụng:

- Nhận dạng ảnh
- Xử lý hình ảnh
- Phục hồi hình ảnh /video
- Thực tế ảo

TensorFlow

Tensorflow là thư viện mã nguồn mở được phát triển bởi nhóm Google Brain, dành cho tính toán số học và Machine Learning quy mô lớn. Tensorflow kết hợp một loạt các mô hình và thuật toán học máy, học sâu (hay còn gọi là mạng thần kinh) giúp cho quá trình thu thập cơ sở dữ liệu, huấn luyện mô hình, hỗ trợ dự đoán và điều chỉnh kết quả dễ dàng hơn.

Kiến trúc của Tensorflow làm việc trong ba phần:

- Tiền xử lý dữ liệu.
- Dựng mô hình.
- Huấn luyện và ước tính mô hình.

Nó được gọi là Tensorflow bởi vì nó nhận đầu vào là một mảng đa chiều, còn được gọi là tensors, các tensor chảy qua một danh sách các toán tử và đi ra ở đầu bên kia.

Các thành phần chính của Tensorflow:

- Variables (Biến): giữ lại giá trị giữa các phiên (session), sử dụng cho trọng số (weights)/ độ lệch (bias).

- Nodes (Nút): các phéo tính (operation).
- Tensors: các tín hiệu truyền từ các nút hoặc các tín hiệu truyền đến các nút.
- Placeholders: được sử dụng để gửi dữ liệu giữa chương trình của người lập trình và biểu đồ tensorflow.

Keras

Keras là một API cao cấp được viết bằng Python, chạy trên nền tảng học máy Tensorflow. API cao cấp hơn có nghĩa là Keras có thể đóng vai trò là giao diện người dùng như một front end và Theano (người phát triển chính Yoshua Bengio) hoặc Tensor- flow.

Keras được phát triển với trọng tâm là phép thử nghiệm nhanh, với mục tiêu cho phép viết script mà không cần tìm hiểu chi tiết về phần phụ trợ. Keras là một giao diện cao cấp sử dụng Theano hoặc Tensorflow cho phần phụ trợ của nó.

Keras hỗ trợ hầu hết các mô hình của mạng neural – fully connected, tích chập (convolutional), gộp (pooling), lặp lại (recurrent), nhúng (embedding), … Các mô hình này có thể được kết hợp để xây dựng mô hình phức tạp hơn.

Keras hoạt động như một wrapper cho các thư viện cấp thấp. Bản thân Keras sử dụng Tensorflow cho phần back-end.

Keras sẽ không hoạt động nếu thực hiện các thay đổi cấp thấp (low-level) cho mô hình. Để có thể thực hiện được thì cần có Tensorflow.

3.3 Kết luận chương

Chương này đã nêu lên được vấn đề cơ bản đối với bài toán nhận dạng cảm xúc khuôn mặt trong thực tế, đồng thời đưa ra hướng tiếp cận đối với bài toán nhận dạng cảm xúc khuôn mặt và cũng như các phương pháp phổ biến thường được sử dụng.

Từ đó lựa chọn phương pháp phù hợp cho bài toán nhận dạng cảm xúc khuôn mặt ta đã tìm hiểu về lý thuyết mô hình kiến trúc được sử dụng trong đề tài cũng như xây dựng kiến trúc , các thuật toán cho bài toán nhận dạng cảm xúc khuôn mặt .

Giới thiệu ngôn ngữ, thư viện hỗ trợ cho việc xây dự kiến trúc mô hình, huấn luyện mô hình.

CHƯƠNG 4: XÂY DỰNG THỰC NGHIỆM VÀ KẾT QUẢ

4.1 Giới thiệu chương

Chương này tập trung xây dựng thực nghiệm và đưa ra kết quả từ kiến trúc đã xây dựng dựa trên cơ sở của chương 2 và chương 3**.**

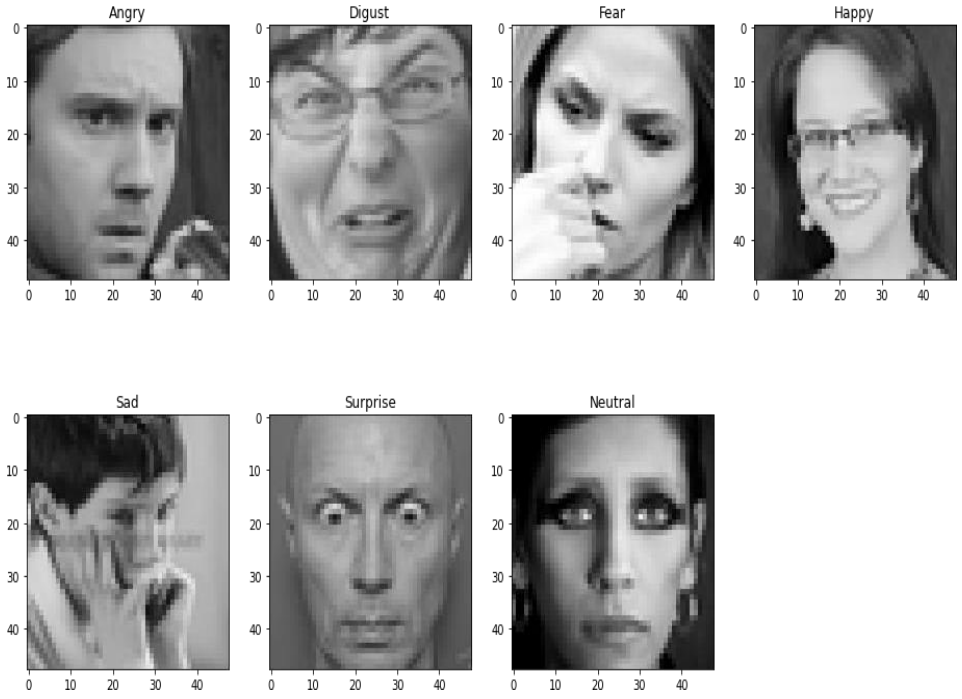
4.2 Kết quả thực nghiệm

4.2.1 Cơ sở dữ liệu và xử lý dữ liệu

Với bài toán nhận diện cảm xúc khuôn mặt lần này, em đã tận dụng cơ sở dữ liệu FER 2013**.**

Dữ liệu bao gồm các hình ảnh xám 48x48 pixel của khuôn mặt. Phân loại từng khuôn mặt dựa trên cảm xúc thể hiện trên nét mặt thành một trong bảy loại (0 = Giận dữ, 1 = Kinh tởm , 2 = Sợ hãi, 3 = Hạnh phúc, 4 = Buồn, 5 = Bất ngờ, 6 = Trung lập).

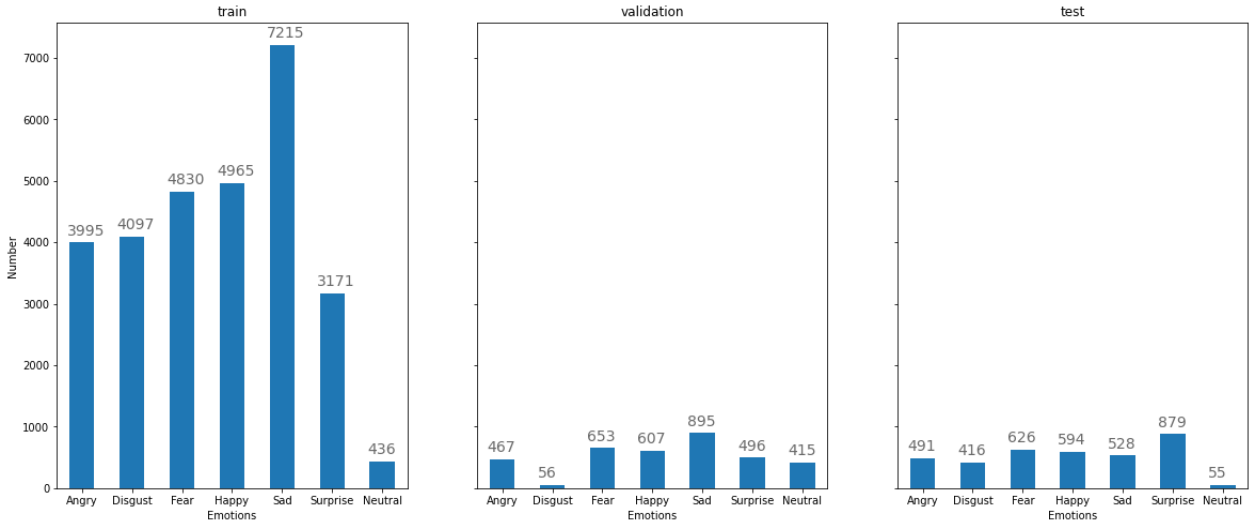
Sau đó dữ liệu sẽ được resize để tất cả các ảnh về cùng kích thước vì đầu vào của mô hình CNN là cố định . Sau khi ảnh được resize thì tất cả sẽ được gắn nhãn các loại cảm xúc và đưa vào mô hình để huấn luyện.



Hình 4.1 : Kết quả sau khi xử lý dữ liệu

4.2.2 Huấn luyện mô hình

Để thuật toán có thể đưa ra đầu ra mong muốn từ tập dữ liệu đầu vào, cần một quá trình huấn luyện sử dụng các dữ liệu huấn luyện. Dữ liệu đầu vào được chia thành 3 tập Train, Test, Validation .



Hình 4.2:Dữ liệu đầu vào

4.2.3 Kết quả huấn luyện và đánh giá

Kết quả huấn luyện mô hình:

```
Epoch 23/50
448/448 - 45s - loss: 0.7915 - accuracy: 0.7031 - val_loss: 1.0601 - val_accuracy: 0.6261 - 45s/epoch - 100ms/step
Epoch 24/50
448/448 - 44s - loss: 0.7803 - accuracy: 0.7101 - val_loss: 1.0065 - val_accuracy: 0.6431 - 44s/epoch - 98ms/step
Epoch 25/50
448/448 - 44s - loss: 0.7655 - accuracy: 0.7111 - val_loss: 1.0219 - val_accuracy: 0.6509 - 44s/epoch - 99ms/step
Epoch 26/50
448/448 - 44s - loss: 0.7535 - accuracy: 0.7145 - val_loss: 0.9915 - val_accuracy: 0.6495 - 44s/epoch - 98ms/step
Epoch 27/50
448/448 - 44s - loss: 0.7459 - accuracy: 0.7206 - val_loss: 1.0191 - val_accuracy: 0.6461 - 44s/epoch - 98ms/step
Epoch 28/50
448/448 - 44s - loss: 0.7344 - accuracy: 0.7230 - val_loss: 1.0986 - val_accuracy: 0.6269 - 44s/epoch - 98ms/step
Epoch 29/50
448/448 - 44s - loss: 0.7193 - accuracy: 0.7308 - val_loss: 0.9970 - val_accuracy: 0.6503 - 44s/epoch - 98ms/step
Epoch 30/50
448/448 - 44s - loss: 0.7117 - accuracy: 0.7306 - val_loss: 1.0022 - val_accuracy: 0.6545 - 44s/epoch - 98ms/step
Epoch 31/50
448/448 - 44s - loss: 0.6926 - accuracy: 0.7385 - val_loss: 1.0304 - val_accuracy: 0.6445 - 44s/epoch - 98ms/step
Epoch 32/50
448/448 - 44s - loss: 0.6797 - accuracy: 0.7445 - val_loss: 1.0408 - val_accuracy: 0.6403 - 44s/epoch - 98ms/step
Epoch 33/50
448/448 - 44s - loss: 0.6748 - accuracy: 0.7473 - val_loss: 1.0420 - val_accuracy: 0.6422 - 44s/epoch - 97ms/step
Epoch 34/50
448/448 - 44s - loss: 0.6622 - accuracy: 0.7503 - val_loss: 1.0774 - val_accuracy: 0.6339 - 44s/epoch - 99ms/step
Epoch 35/50
448/448 - 44s - loss: 0.6535 - accuracy: 0.7547 - val_loss: 1.0821 - val_accuracy: 0.6436 - 44s/epoch - 97ms/step
Epoch 36/50
448/448 - 44s - loss: 0.6442 - accuracy: 0.7567 - val_loss: 1.0697 - val_accuracy: 0.6367 - 44s/epoch - 98ms/step
```

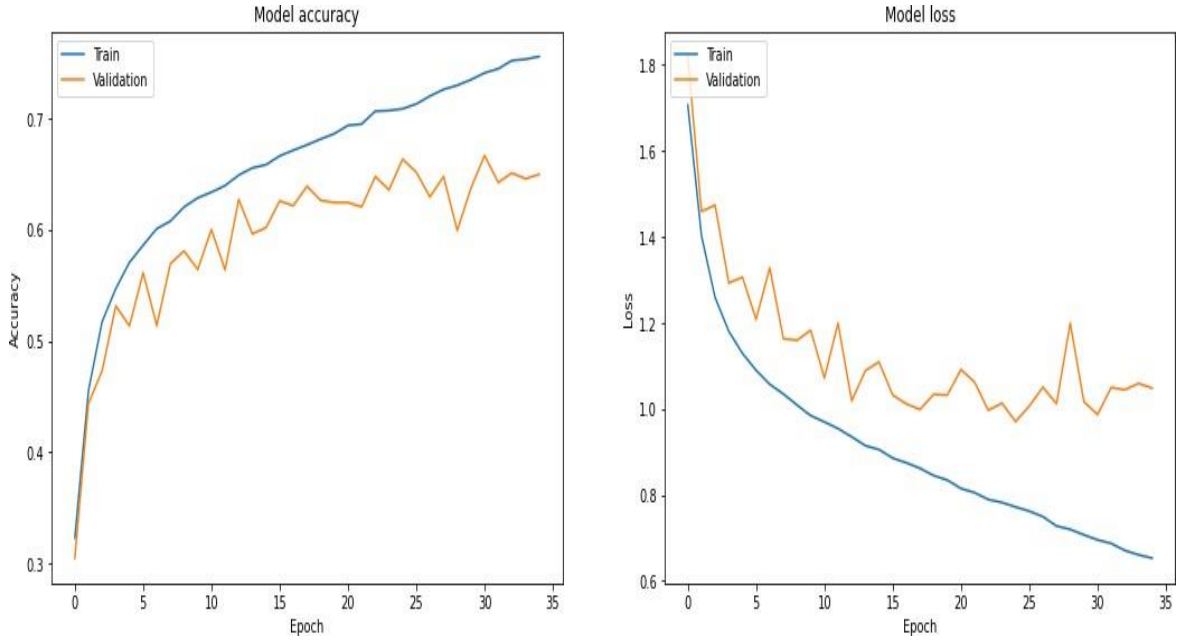
Hình 4.3: Quá trình huấn luyện mô hình

Kết quả cuối cùng:

```
[ ]  1  test_true = np.argmax(test_Y, axis=1)
     2  test_pred = np.argmax(model.predict(test_X), axis=1)
     3  print("CNN Model Accuracy on test set: {:.4f}".format(accuracy_score(test_true, test_pred)))
```

```
CNN Model Accuracy on test set: 0.6659
```

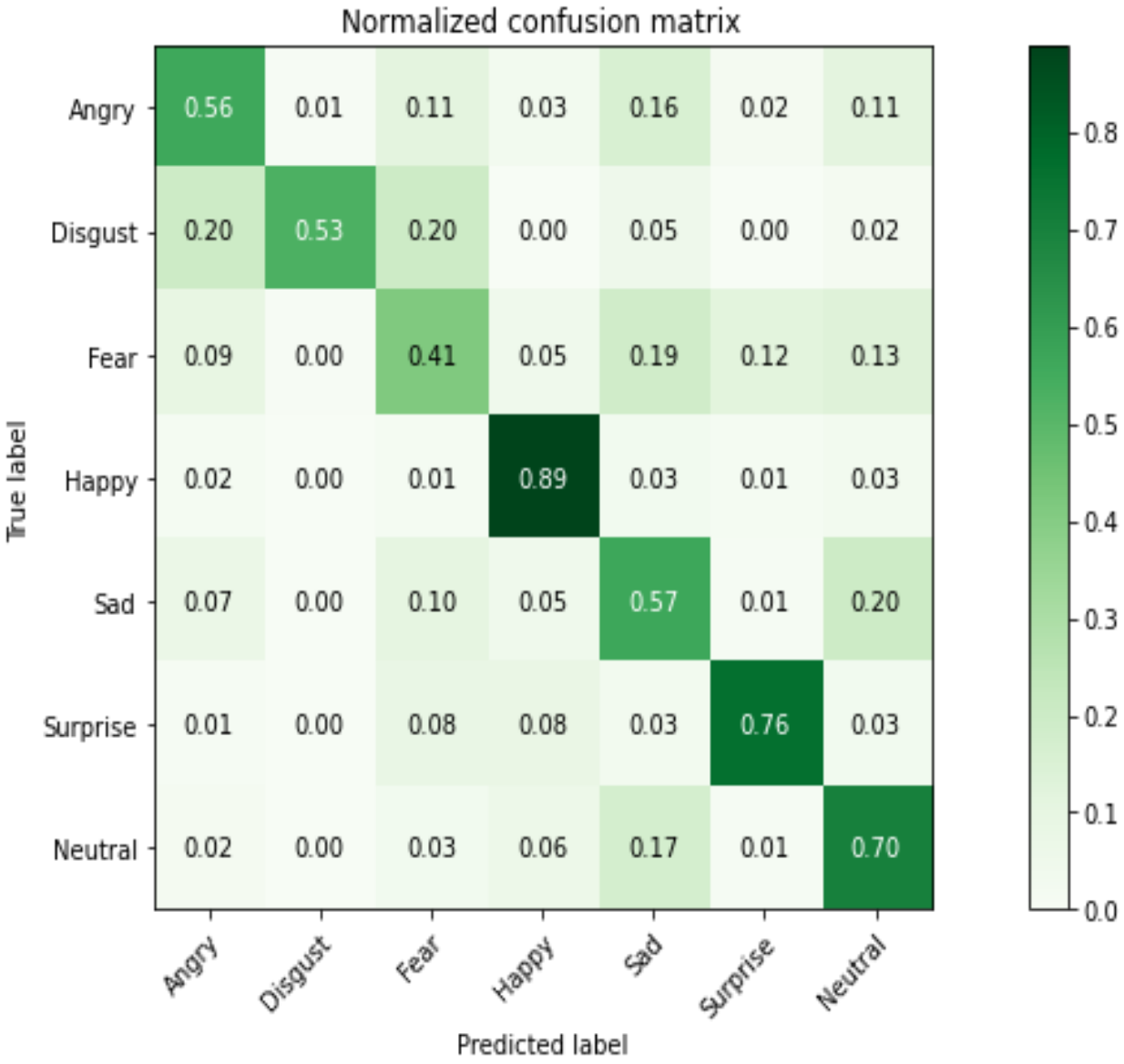
Hình 4.4:Kết quả huấn luyện trên tập test



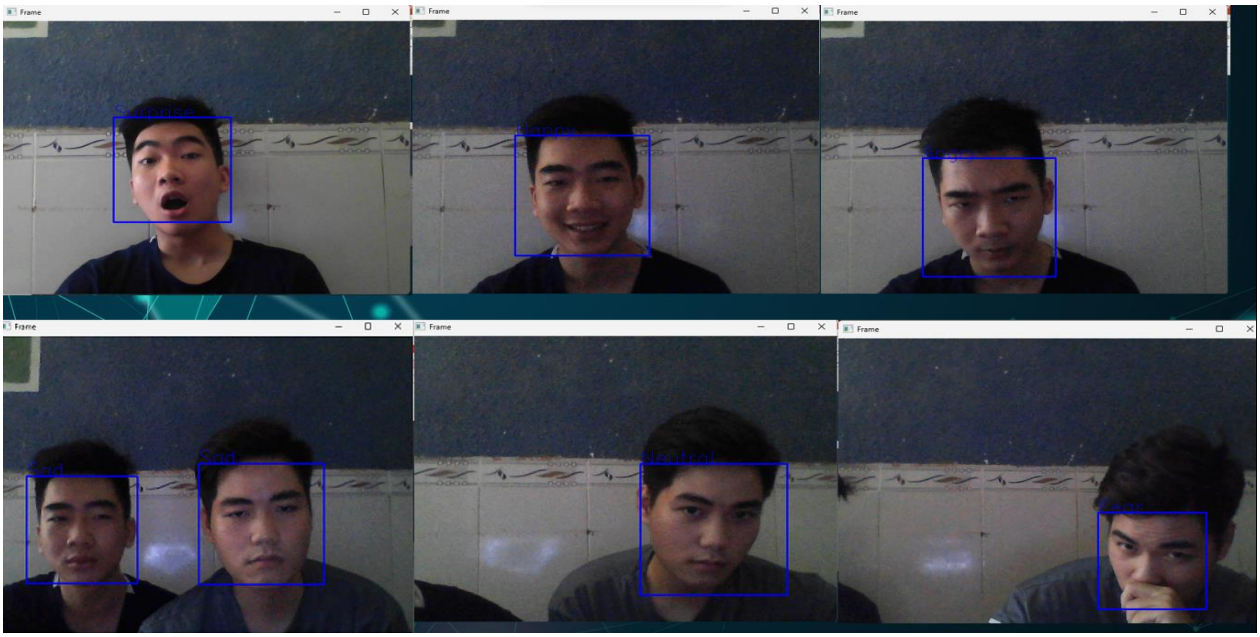
Hình 4.5: Đồ thị accuracy, loss theo epoch (số lần lặp)

Dựa vào biểu đồ có thể thấy lịch sử huấn luyện của model với độ chính xác tăng lên, model loss giảm theo sự tăng lên của epoch . Độ chính xác của tập cơ sở huấn luyện là 74% và trên tập validation là 66% .

Confusion matrix (ma trận nhầm lẫn) là một phương pháp đánh giá kết quả của những bài toán phân loại với việc xem xét cả những chỉ số về độ chính xác và độ bao quát của các dự đoán cho từng lớp. Dựa vào ma trận nhầm lẫn có thể đánh giá khả năng dự đoán model trên mỗi class.



Hình 4.6: Ma trận nhầm lẫn



Hình 4.7: Kết quả nhận dạng cảm xúc khuôn mặt

Từ confusion matrix (ma trận nhầm lẫn) có thể nhận thấy rằng các lớp Giận dữ, Kinh Tởm, Sợ Hãi có xu hướng lẫn lộn với nhau. Điều này là hợp lý bởi trong thực tế cảm xúc con người là một hình thái phức tạp và giữa những cảm xúc có sự đa xen với nhau. Tuy nhiên với những cảm xúc khác nhau hoàn toàn như giữa Hạnh Phúc và Giận dữ mô hình cho được kết quả rất tốt.

Đánh giá mô hình: Từ độ chính xác trên tập train và tập validation cũng như đồ thị hàm mất mất mát và độ chính xác qua các epochs, ta có thể thấy được mô hình đã hội tụ và không có dấu hiệu của overfitting. Độ chính xác trên tập validation là 66% được đánh giá là khá cao nếu đặt trong một nhiệm vụ đầy thách thức như phân loại cảm xúc con người.

4.3 Kết luận chương

Chương đã trình bày được quá trình huấn luyện dựa trên kiến trúc lý thuyết , từ đó đưa ra được kết quả .

Bên cạnh đó , chương cũng nêu ra được các đánh giá từ kết quả trong quá trình huấn luyện , các kết quả thực nghiệm được trình bày bằng các hình ảnh trực quan

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN ĐỀ TÀI

❖ Kết luận

Để tài qua tìm hiểu và thực nghiệm đã phát huy hiệu quả, đáp ứng được những yêu cầu đặt ra lúc bắt đầu thực hiện đề tài . Model dự đoán cảm xúc khá tốt nhưng vẫn còn nhầm lẫn nhiều ở một số classes. Hệ thống vận hành hiệu quả nhưng không tránh khỏi khuyết điểm cần nâng cao tính ổn định và chính xác hơn .

❖ Hướng phát triển

- Cải thiện độ chính xác để việc áp dụng bài toán vào thực tế tốt hơn.

- Huấn luyện thêm model nhận dạng khuôn mặt để hệ thống vừa nhận dạng khuôn mặt, vừa nhận dạng cảm xúc khuôn mặt .

- Hệ thống có thể ứng dụng làm thiết bị hỗ trợ cho người khiếm thị tích hợp thêm các thiết bị thông minh khác như cảm biến vật cản, GPS, ...v.v.

TÀI LIỆU THAM KHẢO

[1] Giới thiệu về CNN https://topdev.vn/blog/thuat-toan-cnn-convolutional-neural- network/

[2] Nghiên cứu và ứng dụng các kĩ thuật nhận dạng cảm xúc qua khuôn mặt https://viblo.asia/p/nghien-cuu-va-ung-dung-cac-ky-thuat-nhan-dang-cam-xuc qua-khuon-mat-ORNZqdmeK0n

[3] introduction to CNN https://towardsdatascience.com/convolutional-neural-network17fb77e76c05#:~:text=Fully%20Connected%20Layer%20is%20simply,into% 20the 2 0fully%20connected%20layer.

[4] introduction to Adam optimizer Algorithm https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

[5] "Các hàm kích hoạt (activation function) trong neural network" https://aicurious.io/posts/2019-09-23-cac-ham-kich-hoat-activationfunction-trong- neural-networks/.

[6] "CS231n Convolutional Neural Networks for Visual Recognition", http://cs231n.github.io/convolutional-networks/, [3/12/2019].

[7] "CNN–Introduction to Pooling Layer", https://www.geeksforgeeks.org/cnnintroduction-to-pooling-layer/, [4/12/2019].

[8] "A guide to convolution arithmetic for deep learning Vincent Dumoulin and Francesco Visin" 1603.07285.pdf (arxiv.org).

[9] "Giới thiệu Haar Cascade",https://otohanquoc.vn/haar-cascade-la-gi/.

[10] "https://jte.hcmute.edu.vn/index.php/jte/article/download/38/46"