

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN

=====***=====



ĐỒ ÁN TỐT NGHIỆP
NGÀNH KHOA HỌC MÁY TÍNH

ĐỀ TÀI: KỸ THUẬT NHẬN DIỆN KHUÔN MẶT
THEO THỜI GIAN THỰC VÀ ỨNG DỤNG TÌM
KIỂM NGƯỜI TRONG Đám ĐÔNG

Giảng viên hướng dẫn:	TS. Nguyễn Mạnh Cường
Sinh viên thực hiện:	Lý Trần Hồng Minh
Mã sinh viên:	2020605555
Lớp:	KHMT02
Khóa:	15

Hà Nội, Năm 2024

MỤC LỤC

DANH MỤC TỪ VIẾT TẮT.....	iv
DANH MỤC HÌNH ẢNH	v
DANH MỤC BẢNG BIỂU	vii
LỜI CẢM ƠN	viii
LỜI MỞ ĐẦU	ix
Chương 1. BÀI TOÁN NHẬN DIỆN KHUÔN MẶT THEO THỜI GIAN THỰC	1
1.1. Tổng quan về Deep Learning	1
1.2. Bài toán nhận diện khuôn mặt theo thời gian thực	2
1.2.1. Giới thiệu bài toán	2
1.2.2. Các khó khăn thách thức của bài toán	4
1.2.3. Đầu vào đầu ra của bài toán	5
1.2.4. Ứng dụng của bài toán.....	7
Chương 2. MỘT SỐ KỸ THUẬT HIỆN CÓ ĐỂ GIẢI QUYẾT BÀI TOÁN	9
2.1. Support Vector Machine	9
2.1.1. Tổng quan	9
2.1.2. Ưu điểm	14
2.1.3. Nhược điểm	14
2.2. Neural Network	15
2.2.1. Tổng quan	15
2.2.2. Cách thức hoạt động của mạng nơ-ron.....	16
2.2.3. Các loại mạng nơ-ron	18
2.2.4. Ưu điểm	19

2.2.5.	Nhược điểm	20
2.3.	YOLO (You only look one)	21
2.3.1.	Giới thiệu về YOLO	21
2.3.1.	Cách thức hoạt động	22
2.3.2.	YOLOv5	31
2.3.3.	Ưu điểm	39
2.3.1.	Nhược điểm	40
2.4.	ArcFace.....	41
2.4.1.	Tổng quan	41
2.4.2.	Additive Angular Margin Loss (ALLM) trong ArcFace	42
2.4.3.	Ưu và nhược điểm	45
Chương 3.	MỘT SỐ KẾT QUẢ THỰC NGHIỆM	46
3.1.	Thu thập và tiền xử lý dữ liệu	47
3.1.1.	Thu thập dữ liệu.....	47
3.1.2.	Tiền xử lý dữ liệu	49
3.2.	Huấn luyện mô hình và đánh giá độ chính xác	49
3.2.1.	YOLOv5	49
3.2.2.	ArcFace.....	53
3.3.	Kết hợp các model sau khi được huấn luyện vào hệ thống.....	54
3.3.1.	Phát hiện khuôn mặt từ camera và trích xuất đặc trưng.....	54
3.3.2.	So sánh đặc trưng khuôn mặt và nhận diện.....	56
Chương 4.	XÂY DỰNG HỆ THỐNG WEBSITE NHẬN DIỆN KHUÔN MẶT THEO THỜI GIAN THỰC	60
4.1.	Giới thiệu framework Streamlit	60
4.2.	Các bước xây dựng hệ thống tích hợp.....	61

4.2.1.	Phân tích yêu cầu.....	61
4.2.2.	Thiết kế giao diện người dùng và logic ứng dụng.....	61
4.3.	Phân tích thiết kế hệ thống tích hợp.....	62
4.3.1.	Vẽ biểu đồ use case	62
4.3.2.	Mô tả chi tiết use case	62
4.4.	Kiểm thử website nhận diện khuôn mặt theo thời gian thực	64
4.4.1.	Phân tích thiết kế kiểm thử.....	64
4.4.2.	Thực hiện kiểm thử.....	65
4.5.	Các kết quả đạt được và nhận xét.....	66
4.6.	Hạn chế và hướng phát triển	69
4.6.1.	Hạn chế	69
4.6.2.	Hướng phát triển.....	70
KẾT LUẬN.....		72
Tài liệu tham khảo.....		73

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Thuật ngữ tiếng Anh	Thuật ngữ tiếng Việt
SVM	Support vector machine	Máy vector hỗ trợ
CNN	Convolutional Neural network	Mạng nơ-ron tích chập
RNN	Recurrent Neural Network	Mạng nơ-ron Hồi Quy
YOLO	You only look once	Chỉ nhìn một lần
AI	Artificial intelligence	Trí tuệ nhân tạo
ML	Machine learning	Học máy
mAP	Mean average precision	Độ chính xác trung bình
AAM Loss (AAML)	Additive Angular Margin Loss	Hàm Mất Mất Biên Góc Cộng Thêm.
COCO	Common objects in context	Các đối tượng phổ biến trong bối cảnh
IOU	Intersection over Union	Giao trên hợp
CPU	Central processing unit	Bộ xử lý trung tâm
GPU	Graphics processing unit	Bộ xử lý đồ họa

DANH MỤC HÌNH ẢNH

Hình 1.1: Mô tả quá trình nhận diện khuôn mặt [11]	3
Hình 2.1: Các mặt phân cách hai classes linearly separable [3]	11
Hình 2.2: Margin của hai classes là bằng nhau và lớn nhất có thể	11
Hình 2.3: Sơ đồ của một nơ-ron trong mạng nơ-ron nhân tạo [18]	16
Hình 2.4: Sơ đồ một mạng nơ-ron nhân tạo [18]	16
Hình 2.5: Kiến trúc YOLO tổng quan [7]	22
Hình 2.6: Minh họa việc phát hiện người trong ảnh với YOLO [7]	24
Hình 2.7: Bức ảnh được chia thành 4x4 các ô vuông có kích thước bằng nhau [7]	25
Hình 2.8: Điểm số xác suất các ô màu đỏ và vàng [7]	26
Hình 2.9: Bounding box centers [7]	27
Hình 2.10: Minh họa quá trình lựa chọn lưới [7]	28
Hình 2.11: Location loss [7]	30
Hình 2.12: Minh họa kỹ thuật Mosaic Augmentation [8]	32
Hình 2.13: Mô tả phương pháp tăng cường sao chép-dán [8]	33
Hình 2.14: Mô tả kỹ thuật biến đổi ngẫu nhiên [8]	33
Hình 2.15: Minh họa kỹ thuật MixUp Augmentation [8]	34
Hình 2.16: Cơ chế định vị hộp giới hạn trong YOLO [15]	36
Hình 2.17: Đồ thị Loss Function trước và sau [8]	37
Hình 2.18: Tính tỉ lệ Ground truth box [8]	38
Hình 2.19: Kết hợp Ground truth box và Anchor box [8]	38
Hình 2.20: Gán các Anchor box phù hợp vào ô tương ứng [8]	39
Hình 2.21: Huấn luyện mô hình nhận diện khuôn mặt sâu bằng ArcFace Loss (K=1) và Sub-center ArcFace Loss (K=3) [9]	41

Hình 2.22: So sánh hàm Softmax thông thường và AAM Loss trong ArcFace [9]	44
Hình 3.1: Các giai đoạn trong mô hình xử lý nhận diện khuôn mặt theo thời gian thực	46
Hình 3.2: Một số lớp trong bộ dữ liệu COCO	48
Hình 3.3: Bộ dữ liệu WIDER FACE	49
Hình 3.4: Cấu trúc bộ dữ liệu.....	51
Hình 3.5: Kết quả đánh giá mô hình sau khi training	52
Hình 3.6: Ảnh kết quả test phát hiện khuôn mặt với mô hình YOLOv5	52
Hình 3.7: So sánh giá trị dự đoán với thực tế	54
Hình 3.8: Ảnh vùng khuôn mặt được cắt ra với bounding box bao quanh trên một khung hình.....	55
Hình 3.9: Thư mục chứa ảnh gốc và vector đặc trưng sau khi trích xuất.....	57
Hình 3.10: Luồng hoạt động của hệ thống.....	58
Hình 3.11: Kết quả nhận diện khuôn mặt	59
Hình 4.1: Biểu đồ use case.....	62
Hình 4.2: Kết quả kiểm thử thực tế với website	66
Hình 4.3: Giao diện màn hình trang chủ của website.....	66
Hình 4.4: Thông báo upload ảnh thành công	67
Hình 4.5: Confidence Threshold và IOU Threshold.....	68
Hình 4.6: Kết quả nhận diện khuôn mặt theo thời gian thực trên Website....	69

DANH MỤC BẢNG BIỂU

Bảng 3.1: Cấu hình phân cứng trên Kaggle	50
Bảng 4.1: Use case tải ảnh và tùy chỉnh tham số.....	62
Bảng 4.2: Use case hiển thị Video kết quả nhận diện.....	63

LỜI CẢM ƠN

Trước tiên với tình cảm sâu sắc và chân thành nhất, cho phép em được bày tỏ lòng biết ơn đến các thầy cô của trường Đại học Công Nghiệp Hà Nội, đặc biệt là các thầy cô khoa Công Nghệ Thông Tin của trường đã tạo điều kiện cho em. Và em cũng xin chân thành cảm ơn thầy giáo Tiến Sĩ Nguyễn Mạnh Cường – người đã tận tâm hướng dẫn em qua từng buổi nói chuyện, hướng dẫn, thảo luận về các lĩnh vực trong đề tài.

Trong quá trình thực hiện đề tài, cũng như là trong quá trình làm bài báo cáo khó tránh khỏi sai sót, em rất mong nhận được ý kiến đóng góp từ thầy cô để học thêm được nhiều kinh nghiệm và sẽ hoàn thành tốt hơn bài báo cáo sắp tới.

Em xin chúc thầy cô luôn dồi dào sức khỏe, luôn vui vẻ và thành công trong cuộc sống.

Sinh viên thực hiện

Lý Trần Hồng Minh

LỜI MỞ ĐẦU

Trong những năm gần đây ở trên thế giới nói chung cũng như ở Việt Nam nói riêng, việc ứng dụng các công nghệ AI, Machine Learning trong việc phân loại và xử lý hình ảnh nhằm ứng dụng vào các hệ thống phân loại và nhận diện đang ngày một phát triển. Các kết quả của nghiên cứu ứng dụng đã cho ra đời nhiều các hệ thống mang lại lợi ích to lớn trong cuộc sống. Vai trò của trí tuệ nhân tạo cũng như các kỹ thuật học máy đang ngày càng trở nên rõ rệt trong cuộc sống.

Trong thế giới ngày nay, nhu cầu xác định và theo dõi các cá nhân trong một tập hợp lớn người dùng ngày càng tăng cao. Từ việc quản lý an ninh công cộng đến tìm kiếm người mất tích, khả năng nhận diện khuôn mặt theo thời gian thực đã trở thành công cụ vô cùng quan trọng. Điều này đặc biệt trở nên cần thiết trong môi trường đông đúc, nơi mà việc theo dõi và nhận biết các cá nhân một cách tự động và nhanh chóng là không thể thiếu.

Đề tài này tập trung khám phá cách thức triển khai một hệ thống nhận diện khuôn mặt theo thời gian thực và ứng dụng nó vào việc tìm kiếm và nhận diện người trong đám đông. Các phương pháp và công nghệ hiện đại nhất sẽ được nghiên cứu để xây dựng một hệ thống có khả năng nhận diện chính xác và nhanh chóng ngay cả trong môi trường đám đông phức tạp nhất.

Qua nỗ lực nghiên cứu và phát triển này, em hy vọng có thể đóng góp một phần nhỏ vào việc cải thiện hiệu suất và tính tiện ích của các hệ thống nhận diện khuôn mặt theo thời gian thực, giúp cho công tác an ninh và quản lý đám đông trở nên hiệu quả hơn và tiện lợi hơn đối với xã hội.

Nội dung bài báo cáo sẽ gồm các chương như sau:

Chương 1: Phát biểu bài toán nhận diện khuôn mặt theo thời gian thực

Trong chương 1 tập trung tiến hành các cuộc nghiên cứu khảo sát về nhu cầu cấp thiết cần tới nhận diện khuôn mặt theo thời gian thực. Tiếp tới sẽ mô

tả tổng quan về bài toán nhận diện khuôn mặt theo thời gian thực và nêu lên những kỳ vọng của bài toán.

Chương 2: Một số kỹ thuật hiện có để giải quyết bài toán

Trong chương 2 tập trung trình bày về các kỹ thuật có thể dùng để giải quyết bài toán nhận diện khuôn mặt theo thời gian thực. Tổng quan về các thuật toán phân loại và nhận diện sẽ được trình bày, bao gồm ưu nhược điểm của từng thuật toán và các nghiên cứu nổi bật đã đạt thành công nhất định..

Chương 3: Một số kết quả thực nghiệm

Chương này sẽ trình bày việc sử dụng các kết quả nghiên cứu ở chương trước để tiến hành xây dựng chương trình sử dụng thuật toán YOLO nhằm xây dựng mô hình Deep learning dùng để training với bộ dữ liệu ảnh COCO và WIDER FACE, từ đó đưa ra các đặc trưng nhằm sử dụng để nhận diện khuôn mặt người. Tiếp đến sử dụng mô hình ArcFace để biểu diễn vector khuôn mặt của người cần nhận diện. Các mô hình sau khi hoàn thiện sẽ được kết hợp lại để xây dựng hệ thống nhận diện khuôn mặt theo thời gian thực từ camera..

Chương 4: Xây dựng hệ thống website nhận diện khuôn mặt theo thời gian thực

Chương này sẽ trình bày hệ thống website tích hợp chương trình nhận diện khuôn mặt đã được xây dựng ở chương 3..

Kết luận:

Cuối cùng trong phần kết luận, em tổng hợp các kết quả đạt được, các hướng phát triển và mở rộng đề tài nghiên cứu trong tương lai.

Qua quá trình thực hiện đề tài này, em đã đạt được cái nhìn tổng quan vững về ứng dụng của mạng nơ-ron nói chung cũng như mô hình YOLO nói riêng trong bài toán nhận diện khuôn mặt. Em đã được trải nghiệm quá trình xây dựng mô hình, thu thập dữ liệu, và tối ưu hóa mô hình để đạt được hiệu suất tốt nhất.

Đặc biệt hy vọng rằng báo cáo này không chỉ là kết quả của quá trình nghiên cứu cá nhân của mình mà còn là một tài liệu hữu ích cho những người quan tâm đến lĩnh vực học máy và thị giác máy tính. Bằng cách trình bày chi tiết về quy trình, các vấn đề mà đã gặp phải, và các giải pháp đã được thử nghiệm, em mong muốn đề tài này sẽ cung cấp một nguồn thông tin đầy đủ và hữu ích cho cộng đồng nghiên cứu.

Đề tài này có thể mang lại cái nhìn sâu sắc hơn về khả năng ứng dụng của YOLO và ArcFace trong việc nhận diện khuôn mặt theo thời gian thực, đồng thời mở ra những thách thức và hướng phát triển tiềm năng trong tương lai. Hy vọng rằng báo cáo này sẽ là một nguồn tài nguyên hữu ích và kích thích sự quan tâm và nghiên cứu trong lĩnh vực này

Chương 1. BÀI TOÁN NHẬN DIỆN KHUÔN MẶT THEO THỜI GIAN THỰC

1.1. Tổng quan về Deep Learning

Deep Learning là một phần của trí tuệ nhân tạo (AI) tập trung vào việc xây dựng và huấn luyện các mạng nơ-ron sâu (deep neural networks) để tự động học và hiểu các mẫu phức tạp từ dữ liệu. Deep Learning đã đạt được sự chú ý lớn trong những năm gần đây với sự phát triển của các mô hình mạng nơ-ron sâu và sự tăng cường của công nghệ tính toán. Các đặc điểm chính của Deep Learning có thể tóm tắt như sau:

Kiến trúc mạng nơ-ron sâu: Deep Learning sử dụng các mạng nơ-ron sâu với nhiều lớp ẩn (hidden layers) để học các đặc trưng từ dữ liệu. Các mạng nơ-ron sâu như Convolutional Neural Networks (CNNs) cho thị giác máy tính, Recurrent Neural Networks (RNNs) cho dữ liệu tuần tự như văn bản hoặc âm thanh, và Transformer Networks cho xử lý ngôn ngữ tự nhiên (NLP) đã đạt được nhiều thành công trong các ứng dụng thực tế.

Học đại diện và tự động hóa: Deep Learning cho phép học các biểu diễn cấp cao từ dữ liệu đầu vào, từ đó giúp tự động hóa quá trình phân tích và hiểu dữ liệu. Điều này giúp giảm sự phụ thuộc vào các phương pháp thủ công và giúp mô hình có khả năng tự động học từ dữ liệu một cách linh hoạt và hiệu quả.

Dữ liệu lớn và tính toán phân tán: Deep Learning đòi hỏi lượng dữ liệu lớn để huấn luyện các mạng nơ-ron sâu hiệu quả. Đồng thời, nó cũng yêu cầu tính toán phân tán mạnh mẽ để xử lý và huấn luyện các mô hình trên dữ liệu lớn. Công nghệ tính toán như GPU (Graphics Processing Unit) và TPU (Tensor Processing Unit) đã đóng vai trò quan trọng trong việc tăng cường hiệu suất của Deep Learning.

Ứng dụng rộng rãi: Deep Learning đã được áp dụng trong nhiều lĩnh vực, từ thị giác máy tính, ngôn ngữ tự nhiên, âm thanh, đến y học, tài chính, và tự động hóa công nghiệp. Các ứng dụng của Deep Learning bao gồm nhận diện hình ảnh, dịch máy, nhận diện giọng nói, xe tự lái, và nhiều hơn nữa.

Công cụ và thư viện: Các công cụ và thư viện phổ biến như TensorFlow, PyTorch, và Keras đã được phát triển để hỗ trợ việc xây dựng, huấn luyện và triển khai các mô hình Deep Learning một cách dễ dàng và hiệu quả.

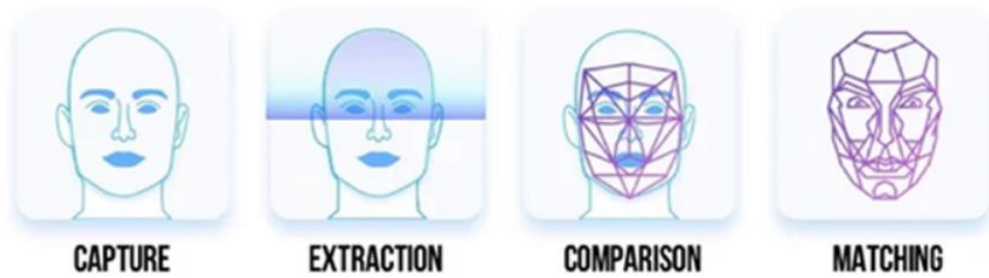
Tóm lại, Deep Learning là một lĩnh vực quan trọng của trí tuệ nhân tạo với tiềm năng lớn trong việc giải quyết nhiều vấn đề phức tạp trong thế giới thực. Sự kết hợp giữa kiến trúc mạng nơ-ron sâu, dữ liệu lớn và công nghệ tính toán phân tán đã tạo nên tiền đề cho sự phát triển và ứng dụng rộng rãi của Deep Learning.

1.2. Bài toán nhận diện khuôn mặt theo thời gian thực

1.2.1. Giới thiệu bài toán

Nhận diện khuôn mặt theo thời gian thực là một bài toán sử dụng trí tuệ nhân tạo (AI) và học sâu (Deep Learning) để phát hiện và nhận dạng khuôn mặt của con người từ video hoặc hình ảnh ngay lập tức khi dữ liệu đó được thu thập. Điều này có nghĩa là hệ thống có thể nhận diện và xác định danh tính của một người trong khoảng thời gian ngắn nhất có thể từ lúc hình ảnh được chụp hoặc video được quay, không có độ trễ đáng kể giữa lúc hình ảnh/video được thu thập và lúc kết quả nhận dạng được cung cấp. Với sự phát triển mạnh mẽ của các thuật toán và công nghệ, nhận diện khuôn mặt theo thời gian thực đã trở thành một bài toán quan trọng trong nhiều lĩnh vực, từ an ninh đến giải trí và giao thông vận tải.

Bài toán nhận diện khuôn mặt theo thời gian thực bao gồm nhiều bước liên quan, mỗi bước đều có vai trò quan trọng trong việc đảm bảo hệ thống hoạt động hiệu quả và chính xác. Dưới đây là mô tả quá trình nhận diện khuôn mặt:



Hình 1.1: Mô tả quá trình nhận diện khuôn mặt [11]

❖ Phát hiện khuôn mặt:

Đây là bước đầu tiên và quan trọng nhất trong quá trình nhận diện khuôn mặt. Các thuật toán phát hiện khuôn mặt có thể được sử dụng để xác định vị trí và ranh giới của các khuôn mặt trong các hình ảnh hoặc video. Phát hiện khuôn mặt thường sử dụng các phương pháp như Haar Cascades, HOG (Histogram of Oriented Gradients). Ngoài ra, các mô hình học sâu như Convolutional Neural Networks (CNNs) cũng đã được sử dụng phổ biến trong phát hiện khuôn mặt. Các mạng CNNs được huấn luyện trên các tập dữ liệu lớn với hàng ngàn hoặc thậm chí hàng triệu hình ảnh khuôn mặt để học được các đặc trưng và mẫu của khuôn mặt. Các mô hình như MTCNN (Multi-task Cascaded Convolutional Networks) và RetinaFace là những ví dụ điển hình về việc sử dụng CNNs trong phát hiện khuôn mặt.

❖ Trích xuất đặc trưng:

Sau khi khuôn mặt đã được phát hiện, các đặc trưng quan trọng như hình dạng của mắt, mũi, miệng, và các điểm đặc trưng khác cần được trích xuất. Các thuật toán trích xuất đặc trưng như Local Binary Patterns (LBP), Principal Component Analysis (PCA), hoặc các mạng neural chuyên dụng được sử dụng để trích rút thông tin từ các vùng quan trọng trên khuôn mặt như Resnet, Retina Face.

❖ So sánh đặc trưng:

Khi các đặc trưng đã được trích xuất, các thuật toán phân loại hoặc nhận dạng sẽ được áp dụng để xác định xem khuôn mặt thuộc về ai hoặc liệu có phải

là một người đã được đăng ký trong cơ sở dữ liệu hay không. Các phương pháp phân loại như Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), hoặc các mạng neural như Convolutional Neural Networks (CNNs) thường được sử dụng trong quá trình này.

❖ Nhận dạng và tích hợp hệ thống theo thời gian thực:

Xác định danh tính của khuôn mặt bằng cách tìm kiếm sự tương đồng cao nhất trong cơ sở dữ liệu. Dựa trên mức độ tương đồng giữa vector đặc trưng của khuôn mặt mới và các vector trong cơ sở dữ liệu, hệ thống sẽ xác định khuôn mặt mới có khớp với bất kỳ khuôn mặt nào trong cơ sở dữ liệu không. Đối với các ứng dụng yêu cầu nhận diện khuôn mặt theo thời gian thực, việc tích hợp hệ thống với các thiết bị và cảm biến thời gian thực là rất quan trọng. Các hệ thống như OpenCV, Dlib, hoặc TensorFlow, Pytorch đều cung cấp các công cụ và API để phát triển và tích hợp các hệ thống nhận diện khuôn mặt vào các ứng dụng thời gian thực.

1.2.2. Các khó khăn thách thức của bài toán

Bài toán nhận diện khuôn mặt theo thời gian thực hiện nay đang đối mặt với nhiều thách thức và khó khăn, dưới đây là một số điều cần xem xét:

Biến đổi và biến động của khuôn mặt: Các biến đổi như góc nhìn, tỉ lệ, ánh sáng và biểu cảm có thể làm thay đổi hình dạng của khuôn mặt. Ví dụ, một khuôn mặt có thể xuất hiện với góc độ nghiêng, hoặc một phần của khuôn mặt có thể bị che khuất bởi ánh sáng yếu. Các biến đổi này đòi hỏi mô hình nhận diện phải có khả năng linh hoạt và chịu được sự biến đổi.

Nhiều từ nền và các vật thể lạ: Dữ liệu hình ảnh thường chứa nhiều từ nền và các vật thể khác. Nhiễu này có thể làm giảm độ chính xác của quá trình nhận diện khuôn mặt. Các phương pháp nhận diện khuôn mặt phải có khả năng phát hiện và loại bỏ nhiễu một cách hiệu quả.

Số lượng lớn các khuôn mặt: Trong các tình huống thực tế, có thể có hàng trăm hoặc thậm chí hàng nghìn khuôn mặt xuất hiện trong cùng một khung

hình. Việc xử lý số lượng lớn các khuôn mặt này đòi hỏi một mô hình có hiệu suất tính toán cao và khả năng xử lý song song.

Bảo mật và quyền riêng tư: Bảo mật dữ liệu khuôn mặt và quyền riêng tư của cá nhân là một vấn đề nhạy cảm và quan trọng. Việc sử dụng dữ liệu khuôn mặt đòi hỏi sự chú ý đặc biệt đối với quy định về bảo mật dữ liệu và quyền riêng tư.

Hiệu suất và thời gian phản hồi: Trong các ứng dụng thời gian thực, hệ thống nhận diện khuôn mặt cần phải có thời gian phản hồi nhanh chóng và hiệu quả. Điều này đặt ra yêu cầu cao về hiệu suất tính toán và khả năng xử lý thời gian thực của hệ thống.

Độ chính xác và tính tổng quát: Để đảm bảo tính chính xác và tổng quát của mô hình, cần phải huấn luyện nó trên một tập dữ liệu đa dạng và đủ lớn. Tuy nhiên, việc thu thập và gắn nhãn dữ liệu cho các khuôn mặt có thể tốn kém và đòi hỏi sự chú ý đặc biệt đối với quy trình gắn nhãn.

Thách thức pháp lý và đạo đức: Việc sử dụng công nghệ nhận diện khuôn mặt có thể gây ra các tranh cãi về quyền riêng tư và an ninh. Các vấn đề liên quan đến quyền riêng tư và bảo mật dữ liệu cần phải được xem xét cẩn thận để đảm bảo rằng việc triển khai công nghệ là hợp pháp và đạo đức.

Nhìn chung, bài toán nhận diện khuôn mặt theo thời gian thực đòi hỏi sự kết hợp của kiến thức về thị giác máy tính, mạng nơ-ron sâu và xử lý ảnh cùng với sự quản lý thông tin và quyền riêng tư. Để giải quyết các thách thức này, cần có sự tiếp cận toàn diện và cẩn thận từ cả mặt kỹ thuật và pháp lý.

1.2.3. Đầu vào đầu ra của bài toán

❖ Đầu vào cho bài toán nhận diện khuôn mặt theo thời gian thực:

Hình ảnh hoặc video đầu vào: Hình ảnh hoặc video là nguồn dữ liệu chính được sử dụng để thực hiện bài toán nhận diện khuôn mặt. Các hình ảnh có thể được thu thập từ camera an ninh, camera điện thoại di động hoặc các

thiết bị ghi hình khác. Trong khi đó, video thường là một chuỗi các hình ảnh được chụp liên tiếp trong một khoảng thời gian nhất định.

Cơ sở dữ liệu khuôn mặt: Để có cơ sở so sánh và nhận diện khuôn mặt, đầu vào của bài toán cần có tập hợp các đặc trưng khuôn mặt đã được trích xuất và mã hóa từ các khuôn mặt đã biết, dùng để so sánh và nhận diện. Cơ sở dữ liệu này thường là các vector đặc trưng được lưu trữ trong các cấu trúc dữ liệu như mảng, danh sách, hoặc các cơ sở dữ liệu chuyên dụng.

Mô hình nhận diện khuôn mặt: Một phần không thể thiếu trong bài toán nhận diện khuôn mặt là các mô hình học máy hoặc học sâu được huấn luyện để phát hiện, trích xuất và so sánh đặc trưng khuôn mặt. Các mô hình có thể được lưu trữ dưới dạng tệp mô hình (.h5, .pth, .pb, vv) hoặc tải trực tiếp từ các thư viện như TensorFlow, PyTorch, Dlib.

❖ **Đầu ra cho bài toán nhận diện khuôn mặt theo thời gian thực:**

Vị trí của khuôn mặt: Đầu ra chính của bài toán là vị trí của các khuôn mặt được nhận diện trong hình ảnh hoặc video. Thông thường, vị trí được biểu diễn dưới dạng hình chữ nhật xác định ranh giới của khuôn mặt trên hình ảnh hoặc video.

Thông tin đặc trưng: Bên cạnh vị trí, hệ thống cũng có thể cung cấp các thông tin về các đặc trưng của khuôn mặt như hình dạng của mắt, mũi, miệng và các điểm đặc trưng khác. Các đặc trưng này có thể hữu ích trong việc nhận diện người dùng hoặc phân tích biểu cảm khuôn mặt.

Định danh hoặc nhận diện: Một phần mở rộng của bài toán là xác định danh tính của các khuôn mặt được nhận diện. Điều này có thể thực hiện thông qua việc so sánh các đặc trưng của khuôn mặt với cơ sở dữ liệu các khuôn mặt đã biết trước, hoặc thông qua các phương pháp nhận diện cá nhân.

Thời gian phản hồi: Trong các ứng dụng thời gian thực, đầu ra cần được tạo ra một cách nhanh chóng và liên tục để cung cấp phản hồi trong thời gian

ngắn nhất có thể, đảm bảo rằng hệ thống có thể hoạt động một cách hiệu quả và linh hoạt trong các tình huống thực tế.

1.2.4. Ứng dụng của bài toán

Ứng dụng của bài toán nhận diện khuôn mặt theo thời gian thực là vô cùng đa dạng và có sự phổ biến rộng rãi trong nhiều lĩnh vực khác nhau. Dưới đây là một số ứng dụng phổ biến của bài toán này:

An ninh và giám sát công cộng: Trong lĩnh vực an ninh, hệ thống nhận diện khuôn mặt được sử dụng để giám sát và theo dõi các hoạt động của mọi người trong các khu vực công cộng như sân bay, ga tàu, trạm xe buýt, trung tâm thương mại, bến cảng và các điểm kiểm soát biên giới. Điều này giúp các cơ quan thực thi pháp luật xác định và theo dõi các đối tượng có liên quan đến tội phạm hoặc hoạt động bất hợp pháp.

Quản lý quyền truy cập: Trong các tổ chức và doanh nghiệp, hệ thống nhận diện khuôn mặt được sử dụng để quản lý quyền truy cập và xác thực người dùng. Thay vì sử dụng thẻ hoặc mã PIN, nhân viên có thể xác minh danh tính của họ thông qua việc quét khuôn mặt, giúp tăng cường bảo mật và tiện lợi.

Xác thực người dùng trên các nền tảng trực tuyến: Trong lĩnh vực công nghệ, hệ thống nhận diện khuôn mặt có thể được sử dụng để xác thực người dùng trên các nền tảng trực tuyến như thiết bị di động, máy tính và các ứng dụng web. Điều này giúp tăng cường bảo mật và ngăn chặn các hành vi giả mạo hoặc truy cập trái phép vào hệ thống.

Dịch vụ khách hàng tự động: Trong lĩnh vực dịch vụ khách hàng, hệ thống nhận diện khuôn mặt có thể được tích hợp vào các ứng dụng tự động để nhận diện và phục vụ khách hàng một cách cá nhân hóa. Ví dụ, trong ngân hàng, khách hàng có thể được nhận dạng và chào đón khi đến gặp nhân viên, giúp tạo ra trải nghiệm khách hàng tốt hơn.

Quản lý sự kiện và đám đông: Trong tổ chức sự kiện và quản lý đám đông, hệ thống nhận diện khuôn mặt có thể được sử dụng để đếm số lượng

người tham dự, phân tích hành vi của khách hàng và xác định vị trí của họ trong không gian. Điều này giúp các tổ chức tối ưu hóa quản lý sự kiện và cải thiện trải nghiệm của khách hàng.

Y tế và y học: Trong lĩnh vực y tế, hệ thống nhận diện khuôn mặt có thể được sử dụng để xác định bệnh nhân và nhân viên y tế trong các bệnh viện và phòng khám. Điều này giúp tăng cường bảo mật thông tin y tế và đảm bảo rằng chỉ những người có quyền truy cập được phép truy cập vào thông tin nhạy cảm.

Những ứng dụng này chỉ là một phần nhỏ trong các cách mà bài toán nhận diện khuôn mặt theo thời gian thực có thể được áp dụng. Với sự phát triển của công nghệ và sự tăng cường nhận thức về tính hiệu quả của việc sử dụng công nghệ nhận diện khuôn mặt, chắc chắn sẽ có nhiều ứng dụng mới và sáng tạo được phát triển trong tương lai.

Chương 2. MỘT SỐ KỸ THUẬT HIỆN CÓ ĐỂ GIẢI QUYẾT BÀI TOÁN

Trong chương này, em sẽ giới thiệu một vài các kỹ thuật và phương pháp hiện đại để giải quyết bài toán nhận diện khuôn mặt theo thời gian thực. Bắt đầu từ các phương pháp cơ bản như SVM và Neural Network, sau đó đi sâu vào hai kỹ thuật tiên tiến hơn là YOLO và ArcFace.

2.1. Support Vector Machine

2.1.1. Tổng quan

Support Vector Machine (SVM) là một thuật toán học máy được sử dụng trong phân loại và hồi quy. Nó được xây dựng dựa trên lý thuyết về tối ưu hóa và giúp xác định ranh giới quyết định tối ưu giữa các lớp dữ liệu khác nhau.

Ý tưởng cơ bản của SVM là tìm ra một siêu phẳng (hyperplane) trong không gian đa chiều, chia dữ liệu thành hai lớp sao cho khoảng cách từ các điểm dữ liệu gần nhất đến siêu phẳng là lớn nhất.

Một trong những đặc điểm quan trọng của SVM là nó có khả năng xử lý cả dữ liệu tuyến tính và phi tuyến tính. Trong trường hợp dữ liệu không thể tách biệt hoàn hảo bằng một siêu phẳng tuyến tính, SVM sử dụng một kỹ thuật được gọi là kernel trick để ánh xạ dữ liệu vào một không gian cao hơn, nơi nó có thể được tách biệt bằng siêu phẳng tuyến tính.

SVM cũng có thể được mở rộng để giải quyết các bài toán hồi quy. Trong trường hợp này, SVM tìm cách tạo ra một siêu phẳng sao cho tổng bình phương sai số giữa các điểm dữ liệu thực tế và các điểm dữ liệu dự đoán trên siêu phẳng là nhỏ nhất.

Dựa trên bản chất của ranh giới quyết định, SVM có thể được chia thành hai phần chính:

- **SVM tuyến tính:** SVM tuyến tính sử dụng ranh giới quyết định tuyến tính để phân tách các điểm dữ liệu của các lớp khác nhau. Khi dữ liệu có

thể được phân tách tuyến tính một cách chính xác, các SVM tuyến tính rất phù hợp. Điều này có nghĩa là một đường thẳng đơn (ở dạng 2D) hoặc một siêu phẳng (ở các chiều cao hơn) hoàn toàn có thể chia các điểm dữ liệu thành các lớp tương ứng của chúng. Một siêu phẳng tối đa hóa lề giữa các lớp là ranh giới quyết định [3].

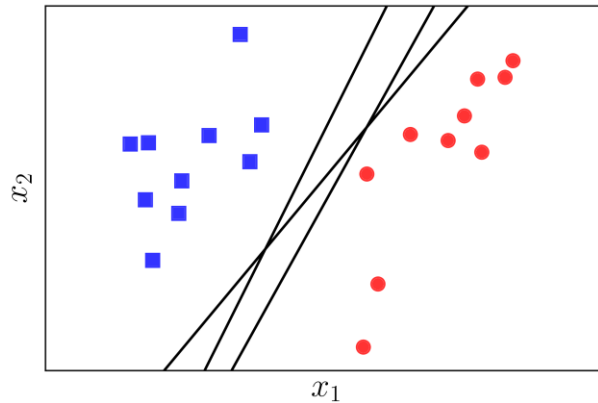
- **SVM phi tuyến tính:** SVM phi tuyến tính có thể được sử dụng để phân loại dữ liệu khi không thể tách nó thành hai lớp bằng một đường thẳng (trong trường hợp 2D). Bằng cách sử dụng các hàm kernel, các SVM phi tuyến tính có thể xử lý dữ liệu có thể phân tách phi tuyến tính. Dữ liệu đầu vào ban đầu được các hàm kernel này chuyển đổi thành không gian đặc trưng có chiều cao hơn, nơi các điểm dữ liệu có thể được phân tách tuyến tính. Một SVM tuyến tính được sử dụng để xác định ranh giới quyết định phi tuyến tính trong không gian được sửa đổi này [3].

Trên thực tế, SVM đã được áp dụng rộng rãi trong nhiều lĩnh vực như phân loại văn bản, nhận diện hình ảnh, nhận dạng ký tự và nhiều lĩnh vực khác, nhờ vào tính linh hoạt và hiệu quả của nó trong việc xử lý các bài toán phân loại và hồi quy.

Để hiểu rõ hơn về thuật toán SVM, chúng ta cùng đi qua tìm hiểu về bài toán cơ sở và bài toán tối ưu cho thuật toán này.

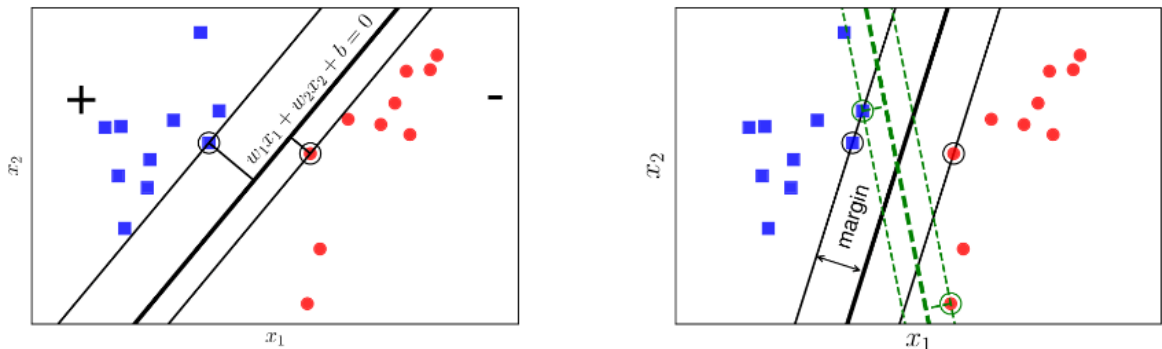
❖ Bài toán cơ sở:

Giả sử rằng có hai class khác nhau được mô tả bởi các điểm trong không gian nhiều chiều, hai classes này *linearly separable*, tức tồn tại một siêu phẳng phân chia chính xác hai classes đó. Tìm một siêu mặt phẳng phân chia hai classes đó, tức tất cả các điểm thuộc một class nằm về cùng một phía của siêu mặt phẳng đó và ngược phía với toàn bộ các điểm thuộc class còn lại. Biết rằng, thuật toán PLA có thể làm được việc này nhưng nó có thể cho chúng ta vô số nghiệm như dưới đây:



Hình 2.1: Các mặt phân cách hai classes linearly separable [3]

Trong ba đường thẳng minh họa trong hình phía trên, có hai đường thẳng khá *lệch* về phía class hình tròn đỏ. Điều này làm cho đường phân chia giữa các lớp chưa được tối ưu. Chúng ta cần tìm một tiêu chuẩn để đo sự phân chia của mỗi class, cụ thể một đường phân chia sao cho khoảng cách từ điểm gần nhất của mỗi class (các điểm được khoanh tròn) tới đường phân chia là như nhau, như thế thì mới công bằng. Khoảng cách như nhau này được gọi là margin (lẻ).



Hình 2.2: Margin của hai classes là bằng nhau và lớn nhất có thể

Vậy bài toán cơ sở của SVM là tìm một siêu phẳng tối ưu để phân loại các điểm dữ liệu với biên tối đa, điều này dẫn tới việc giải một bài toán tối ưu với các ràng buộc phân loại chính xác các điểm dữ liệu. Bài toán tối ưu trong Support Vector Machine (SVM) chính là bài toán đi tìm đường phân chia sao cho margin là lớn nhất. Đây cũng là lý do vì sao SVM còn được gọi là Maximum Margin Classifier.

❖ Bài toán tối ưu:

Để xây dựng bài toán tối ưu, giả sử rằng các cặp dữ liệu của training set là $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ với các vector $x_i \in R^d$ thể hiện đầu vào của một điểm dữ liệu và y_i là nhãn của điểm dữ liệu đó. d là số chiều dữ liệu và N là số điểm dữ liệu. Giả sử rằng *nhãn* của mỗi điểm dữ liệu được xác định bởi $y_i = 1$ (class 1) hoặc $y_i = -1$ (class 2).

Giả sử các điểm vuông xanh thuộc class 1, các điểm tròn đỏ thuộc class (-1) và mặt $\mathbf{w}^T \mathbf{x} + b = w_1 x_1 + w_2 x_2 + b = 0$ là mặt phân chia giữa 2 class. Class 1 nằm về phía dương, class -1 nằm về phía âm của mặt phân chia.

Như vậy, khoảng cách từ một điểm bất kỳ đến mặt phân chia là:

$$\frac{y_n(\mathbf{w}^T \mathbf{x}_n + b)}{\|\mathbf{w}\|_2}$$

Cùng với giả sử y_n luôn cùng dấu với phía của \mathbf{x}_n . Với mặt phân chia như trên, *margin* được tính là khoảng cách gần nhất từ 1 điểm tới mặt đó (bất kể điểm nào trong hai class):

$$margin = \min \frac{y_n(\mathbf{w}^T \mathbf{x}_n + b)}{\|\mathbf{w}\|_2}$$

Bài toán tối ưu trong SVM chính là bài toán tìm \mathbf{w} và b sao cho margin này đạt giá trị lớn nhất:

$$\begin{aligned} (\mathbf{w}, b) &= \arg \max_{\mathbf{w}, b} \left\{ \min_n \frac{y_n(\mathbf{w}^T \mathbf{x}_n + b)}{\|\mathbf{w}\|_2} \right\} \\ &= \arg \max_{\mathbf{w}, b} \left\{ \frac{1}{\|\mathbf{w}\|_2} \min_n y_n(\mathbf{w}^T \mathbf{x}_n + b) \right\} \quad (1) \end{aligned}$$

Nhận xét quan trọng nhất là nếu thay vector hệ số \mathbf{w} bởi $k\mathbf{w}$ và b bởi kb trong đó k là một hằng số dương thì mặt phân chia không thay đổi, tức khoảng cách từ từng điểm đến mặt phân chia không đổi, tức *margin* không đổi. Dựa trên tính chất này, giả sử:

$$y_n(\mathbf{w}^T \mathbf{x}_n + b) = 1$$

Như vậy với mọi n : $y_n(\mathbf{w}^T \mathbf{x}_n + b) \geq 1$

Vậy bài toán tối ưu (1) có thể đưa về bài toán tối ưu có ràng buộc sau đây:

$$(\mathbf{w}, b) = \arg \max_{\mathbf{w}, b} \left\{ \frac{1}{\|\mathbf{w}\|_2} \right\}$$

Điều kiện: $y_n(\mathbf{w}^T \mathbf{x}_n + b) \geq 1, \forall n = 1, 2, \dots, N$ (2)

Biến đổi biểu thức trên, thu được:

$$(\mathbf{w}, b) = \arg \min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|_2$$

Điều kiện: $1 - y_n(\mathbf{w}^T \mathbf{x}_n + b) \leq 0, \forall n = 1, 2, \dots, N$ (3)

Trong bài toán (3), hàm mục tiêu là một norm, nên là một hàm lồi. Các hàm bất đẳng thức ràng buộc là các hàm tuyến tính theo \mathbf{w} và b , nên chúng cũng là các hàm lồi. Vậy bài toán tối ưu (3) có hàm mục tiêu là lồi, và các hàm ràng buộc cũng là lồi, nên nó là một bài toán lồi. Hơn nữa, nó là một Quadratic Programming. Thậm chí, hàm mục tiêu là strictly convex vì $\|\mathbf{w}\|_2^2 = \mathbf{w}^T \mathbf{I} \mathbf{w}$ với \mathbf{I} là ma trận đơn vị - là một ma trận xác định dương. Từ đây có thể suy ra nghiệm cho SVM là duy nhất [3].

Bài toán này thường được giải bài toán đối ngẫu. Thứ nhất, bài toán đối ngẫu có những tính chất đặc biệt khiến nó được giải hiệu quả hơn. Thứ hai, trong quá trình xây dựng bài toán đối ngẫu, có thể thấy rằng SVM có thể được áp dụng cho những bài toán mà dữ liệu không linearly separable, tức các đường phân chia không phải là một mặt phẳng mà có thể là các mặt có hình thù phức tạp hơn.

Xác định lớp cho một điểm dữ liệu mới: Sau khi tìm được mặt phân cách $\mathbf{w}^T \mathbf{x} + b = 0$, class của một điểm nào đó sẽ được xác định bằng cách:

$$\text{class}(\mathbf{x}) = \text{sgn}(\mathbf{w}^T \mathbf{x} + b)$$

Trong đó hàm sgn là hàm xác định dấu, nhận giá trị 1 nếu đối số là không âm và -1 nếu ngược lại.

2.1.2. Ưu điểm

Hiệu suất tốt trong không gian nhiều chiều: SVM hoạt động tốt trong không gian nhiều chiều (high-dimensional space), giúp phân loại dữ liệu có số chiều lớn, chẳng hạn trong xử lý ảnh và xử lý ngôn ngữ tự nhiên.

Hiệu suất tốt với dữ liệu ít: SVM thường hoạt động tốt khi có ít dữ liệu huấn luyện. Điều này là do thuật toán tập trung vào các điểm dữ liệu quan trọng (support vectors) để xác định siêu phẳng tốt nhất, do đó tránh được overfitting.

Khả năng xử lý dữ liệu phi tuyến tính: SVM có thể xử lý dữ liệu không phải là tuyến tính bằng cách sử dụng các hàm nhân (kernel functions) để ánh xạ dữ liệu vào không gian cao hơn, nơi chúng có thể được tách biệt bằng siêu phẳng tuyến tính.

Khả năng kiểm soát overfitting: SVM có các tham số như siêu phẳng margin và siêu phẳng miền mục tiêu (C) cho phép kiểm soát overfitting và tạo ra mô hình tổng quan tốt hơn.

2.1.3. Nhược điểm

Nhạy cảm với nhiễu: SVM có thể nhạy cảm với dữ liệu nhiễu và điểm ngoại lai (outliers), đặc biệt là khi sử dụng margin lớn.

Khó tính toán với dữ liệu lớn: Thuật toán SVM có thể trở nên khó tính toán và tốn thời gian với dữ liệu lớn. Điều này đặc biệt đúng khi sử dụng các hàm nhân phức tạp và cần tối ưu hóa siêu phẳng trong không gian cao chiều.

Lựa chọn hàm nhân và tham số: Lựa chọn hàm nhân và điều chỉnh các tham số của SVM có thể là một nhiệm vụ khó khăn và đòi hỏi kiến thức chuyên sâu về mô hình.

Không đưa ra xác suất trực tiếp: SVM ban đầu không cung cấp xác suất trực tiếp cho việc phân loại, và bạn cần phải thực hiện một phần mềm để tính xác suất hoặc sử dụng một biến thể như SVM xếp hạng (SVM-Rank).

2.2. Neural Network

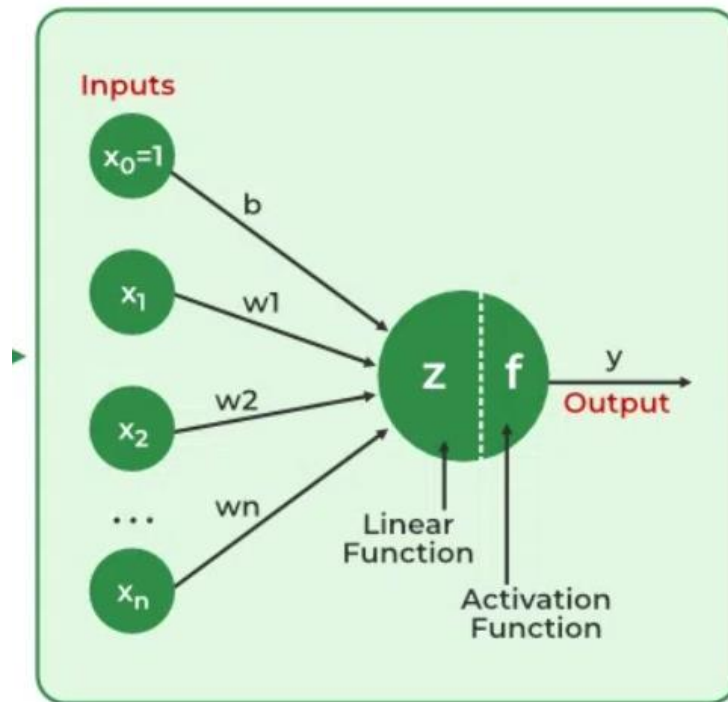
2.2.1. Tổng quan

Mạng nơ-ron là các mô hình tính toán mô phỏng các chức năng phức tạp của bộ não con người. Mạng nơ-ron bao gồm các nút hoặc nơ-ron được kết nối với nhau để xử lý và học hỏi từ dữ liệu, cho phép thực hiện các tác vụ như nhận dạng mẫu và ra quyết định trong học máy.

Mạng nơ-ron trích xuất các đặc điểm nhận dạng từ dữ liệu mà thiếu sự hiểu biết được lập trình sẵn. Các thành phần mạng bao gồm tế bào thần kinh, kết nối, trọng số, độ lệch, hàm truyền và quy tắc học. Các nơ-ron nhận đầu vào, được điều chỉnh bởi các ngưỡng và chức năng kích hoạt. Các kết nối liên quan đến trọng số và độ lệch điều chỉnh việc truyền thông tin. Quá trình học, điều chỉnh trọng số và độ lệch, diễn ra trong ba giai đoạn: tính toán đầu vào, tạo đầu ra và sàng lọc lặp lại, nâng cao khả năng thành thạo của mạng trong các nhiệm vụ đa dạng.

Những nhiệm vụ này bao gồm:

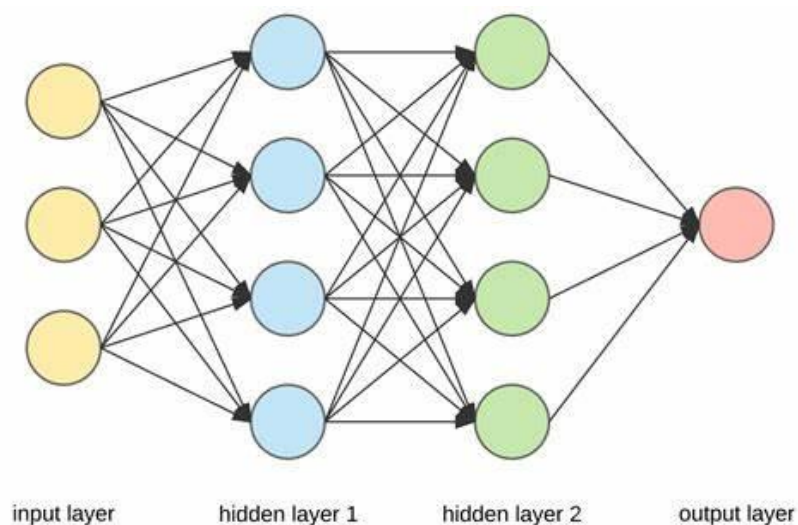
- Mạng nơ-ron được mô phỏng bởi một môi trường mới.
- Sau đó, các tham số tự do của mạng nơ-ron được thay đổi do kết quả của việc mô phỏng này.
- Mạng nơ-ron sau đó phản hồi theo một cách mới đối với môi trường do sự thay đổi trong các tham số tự do của nó.



Hình 2.3: Sơ đồ của một nơ-ron trong mạng nơ-ron nhân tạo [18]

2.2.2. Cách thức hoạt động của mạng nơ-ron

Mạng nơ-ron là những hệ thống phức tạp mô phỏng một số đặc điểm của cách hoạt động của não người. Nó bao gồm một lớp đầu vào, một hoặc nhiều lớp ẩn, và một lớp đầu ra, được tạo thành từ các lớp nơ-ron nhân tạo được kết nối với nhau. Hai giai đoạn của quy trình cơ bản được gọi là lan truyền ngược và lan truyền xuôi.



Hình 2.4: Sơ đồ một mạng nơ-ron nhân tạo [18]

❖ **Lan truyền xuôi (Forward propagation):**

- **Lớp đầu vào:** Mỗi đặc trưng trong lớp đầu vào được đại diện bởi một nút trên mạng, nhận dữ liệu đầu vào.
- **Trọng số và Kết nối:** Trọng số của mỗi kết nối nơ-ron biểu thị mức độ mạnh của kết nối đó. Trong suốt quá trình huấn luyện, các trọng số này được thay đổi.
- **Các lớp ẩn:** Mỗi nơ-ron trong lớp ẩn xử lý các đầu vào bằng cách nhân chúng với trọng số, cộng chúng lại, và sau đó truyền qua một hàm kích hoạt. Bằng cách này, tính phi tuyến được giới thiệu, cho phép mạng nhận ra các mẫu phức tạp.
- **Đầu ra:** Kết quả cuối cùng được tạo ra bằng cách lặp lại quá trình này cho đến khi đạt đến lớp đầu ra.

❖ **Lan truyền ngược (Backpropagation):**

- **Tính toán mất mát:** Đầu ra của mạng được đánh giá so với các giá trị mục tiêu thực tế, và một hàm mất mát được sử dụng để tính toán sự khác biệt. Đối với bài toán hồi quy, một trong những hàm mất mát phổ biến trong hồi quy tuyến tính là Mean Square Error (MSE) thường được sử dụng làm hàm chi phí. Hàm mất mát có công thức sau:

$$MSE = (1/n) * \sum (y_i - \bar{y})^2$$

Trong đó:

- MSE là giá trị mất mát (loss).
 - n là số lượng mẫu dữ liệu.
 - y_i là giá trị thực tế của mẫu thứ i.
 - \bar{y} là giá trị dự đoán tương ứng của mẫu thứ i.
 - \sum là ký hiệu tổng của tất cả các mẫu dữ liệu.
- **Gradient Descent:** Gradient descent sau đó được sử dụng bởi mạng để giảm mất mát. Để giảm thiểu độ sai lệch, các trọng số được thay đổi dựa trên đạo hàm của mất mát đối với mỗi trọng số.

- **Điều chỉnh trọng số:** Các trọng số được điều chỉnh tại mỗi kết nối bằng cách áp dụng quá trình lặp này, hay lan truyền ngược qua mạng.
- **Huấn luyện:** Trong quá trình huấn luyện với các mẫu dữ liệu khác nhau, toàn bộ quá trình lan truyền xuôi, tính toán mất mát và lan truyền ngược được thực hiện lặp đi lặp lại, cho phép mạng thích ứng và học các mẫu từ dữ liệu.
- **Hàm kích hoạt:** Tính phi tuyến của mô hình được giới thiệu bởi các hàm kích hoạt như đơn vị tuyến tính chỉnh lưu (ReLU) hoặc sigmoid. Quyết định liệu có "kích hoạt" một nơ-ron hay không dựa trên tổng đầu vào có trọng số.

2.2.3. Các loại mạng nơ-ron

Mạng truyền thẳng (Feedforward Networks): Một mạng nơ-ron truyền thẳng là một kiến trúc mạng nơ-ron nhân tạo đơn giản, trong đó dữ liệu di chuyển từ đầu vào đến đầu ra theo một hướng duy nhất. Nó có các lớp đầu vào, ẩn, và đầu ra; không có vòng phản hồi. Kiến trúc đơn giản của nó làm cho nó phù hợp với nhiều ứng dụng, chẳng hạn như hồi quy và nhận dạng mẫu. Một trong những ưu điểm chính của mạng truyền thẳng là tính dễ hiểu và dễ triển khai của nó, làm cho nó trở thành lựa chọn lý tưởng cho các vấn đề không quá phức tạp [18].

Mạng Perceptron nhiều lớp (Multilayer Perceptron - MLP): MLP là một loại mạng nơ-ron truyền thẳng với ba hoặc nhiều lớp, bao gồm một lớp đầu vào, một hoặc nhiều lớp ẩn, và một lớp đầu ra. Nó sử dụng các hàm kích hoạt phi tuyến tính như ReLU hoặc sigmoid để giới thiệu tính phi tuyến cho mô hình. Điều này cho phép MLP học các quan hệ phức tạp hơn giữa đầu vào và đầu ra, làm cho nó rất hữu ích trong các tác vụ như phân loại và dự đoán [18].

Mạng nơ-ron tích chập (Convolutional Neural Network - CNN): Một Mạng nơ-ron tích chập (CNN) là một loại mạng nơ-ron nhân tạo chuyên biệt được thiết kế cho xử lý hình ảnh. Nó sử dụng các lớp tích chập để tự động học các đặc trưng phân cấp từ các hình ảnh đầu vào, cho phép nhận dạng và phân

loại hình ảnh hiệu quả. CNN đã cách mạng hóa thị giác máy tính và đóng vai trò quan trọng trong các nhiệm vụ như phát hiện đối tượng và phân tích hình ảnh. Các thành phần quan trọng của CNN bao gồm các lớp tích chập, lớp gộp (pooling), và lớp kết nối đầy đủ, mỗi lớp đóng một vai trò khác nhau trong việc trích xuất và xử lý thông tin hình ảnh [18].

Mạng nơ-ron hồi quy (Recurrent Neural Network - RNN): Mạng nơ-ron hồi quy là một loại mạng nơ-ron nhân tạo được thiết kế để xử lý dữ liệu tuần tự. Nó sử dụng các vòng phản hồi, cho phép thông tin tồn tại trong mạng, giúp nó phù hợp cho các ứng dụng mà các phụ thuộc ngữ cảnh là quan trọng, chẳng hạn như dự đoán chuỗi thời gian và xử lý ngôn ngữ tự nhiên. RNN có khả năng ghi nhớ thông tin trong khoảng thời gian dài, tuy nhiên, nó thường gặp phải vấn đề vanishing gradient, làm giảm khả năng học thông tin từ các chuỗi dài [18].

Bộ nhớ ngắn hạn dài hạn (Long Short-Term Memory - LSTM): LSTM là một loại RNN được thiết kế để khắc phục vấn đề vanishing gradient trong việc huấn luyện RNN. Nó sử dụng các ô nhớ (memory cells) và các cổng (gates) để chọn lọc đọc, ghi và xóa thông tin. LSTM có khả năng học và ghi nhớ các mẫu thông tin trong thời gian dài, làm cho nó rất hữu ích cho các ứng dụng như dịch máy, nhận dạng giọng nói, và phân tích cảm xúc. Bằng cách quản lý thông tin một cách có hệ thống, LSTM cải thiện hiệu suất của mạng nơ-ron trong việc xử lý các chuỗi dữ liệu phức tạp [18].

2.2.4. Ưu điểm

Mạng nơ-ron được sử dụng rộng rãi trong nhiều ứng dụng khác nhau nhờ vào nhiều lợi ích của chúng:

Tính thích ứng: Mạng nơ-ron rất hữu ích cho các hoạt động mà mối liên hệ giữa đầu vào và đầu ra phức tạp hoặc không được xác định rõ ràng vì chúng có thể thích nghi với các tình huống mới và học hỏi từ dữ liệu. Điều này làm cho mạng nơ-ron trở thành một công cụ mạnh mẽ trong các lĩnh vực như dự đoán tài chính, chẩn đoán y khoa, và tối ưu hóa chuỗi cung ứng.

Nhận dạng mẫu: Khả năng nhận dạng mẫu của mạng nơ-ron làm cho chúng hiệu quả trong các nhiệm vụ như nhận dạng âm thanh và hình ảnh, xử lý ngôn ngữ tự nhiên, và các mẫu dữ liệu phức tạp khác. Chúng có thể tự động phân loại email, phát hiện gian lận trong giao dịch tài chính, và thậm chí hiểu ý định của người dùng trong các trợ lý ảo.

Xử lý song song: Bản chất có khả năng xử lý song song của mạng nơ-ron cho phép chúng xử lý nhiều tác vụ cùng một lúc, giúp tăng tốc và cải thiện hiệu quả tính toán. Điều này rất quan trọng trong các ứng dụng yêu cầu xử lý dữ liệu lớn theo thời gian thực, như trong hệ thống giám sát video hoặc xử lý dữ liệu cảm biến trong xe tự lái.

Tính phi tuyến: Mạng nơ-ron có khả năng mô hình hóa và hiểu các mối quan hệ phức tạp trong dữ liệu nhờ vào các hàm kích hoạt phi tuyến được tìm thấy trong các nơ-ron, giúp khắc phục những hạn chế của các mô hình tuyến tính. Điều này cho phép chúng học các biểu diễn dữ liệu phong phú và phức tạp, cần thiết trong các lĩnh vực như thị giác máy tính, học sâu và trí tuệ nhân tạo.

Tổng kết lại, mạng nơ-ron không chỉ linh hoạt và mạnh mẽ trong việc xử lý các vấn đề phức tạp, mà còn có khả năng học hỏi và thích nghi liên tục, làm cho chúng trở thành một công cụ quan trọng trong kỷ nguyên dữ liệu hiện đại.

2.2.5. Nhược điểm

Mạng nơ-ron tuy mạnh mẽ nhưng không phải không có nhược điểm và khó khăn trong tính toán:

Cường độ tính toán: Đào tạo mạng lưới thần kinh lớn có thể là một quá trình tốn nhiều công sức và đòi hỏi tính toán, đòi hỏi nhiều sức mạnh tính toán.

Overfitting: Overfitting là một hiện tượng trong đó mạng lưới thần kinh đưa tài liệu đào tạo vào bộ nhớ thay vì xác định các mẫu trong dữ liệu. Mặc dù các phương pháp chính quy hóa giúp giảm bớt điều này nhưng vấn đề vẫn tồn tại.

Cần các tập dữ liệu lớn: Để đào tạo hiệu quả, mạng lưới thần kinh thường cần các tập dữ liệu có nhãn, có kích thước lớn; nếu không, hiệu suất của chúng có thể bị ảnh hưởng do dữ liệu không đầy đủ hoặc sai lệch.

2.3. YOLO (You only look one)

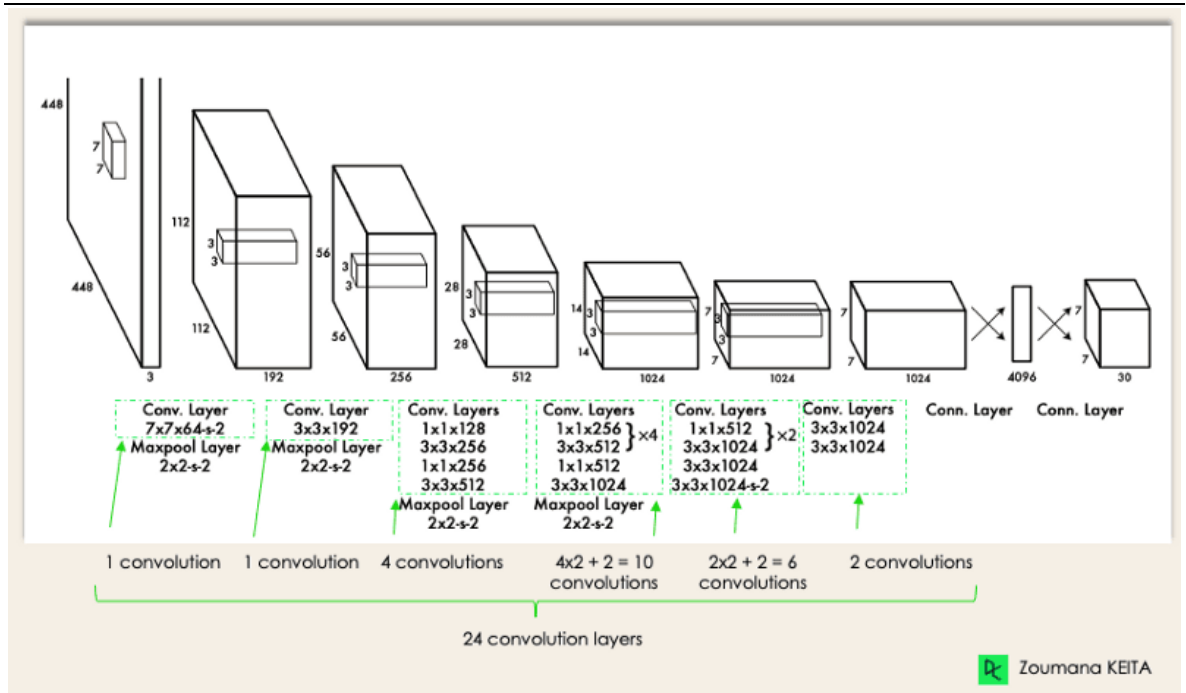
2.3.1. Giới thiệu về YOLO

Phát hiện đối tượng hay phát hiện vật thể là một bài toán quan trọng trong lĩnh vực Thị giác máy tính, các thuật toán phát hiện đối tượng được chia thành 2 nhóm chính:

- Họ các mô hình RCNN (Region-Based Convolutional Neural Networks) để giải quyết các bài toán về định vị và nhận diện vật thể.
- Họ các mô hình về YOLO (You Only Look Once) dùng để nhận diện đối tượng được thiết kế để nhận diện các vật thể real-time.

YOLO là một mô hình mạng CNN cho việc phát hiện, nhận diện, phân loại đối tượng. YOLO được tạo ra từ việc kết hợp giữa các lớp tích chập và các lớp connected. Trong đó các lớp tích chập sẽ trích xuất ra các thuộc tính đặc trưng của ảnh, còn các lớp fully-connected sẽ dự đoán ra xác suất đó và tọa độ của đối tượng.

YOLO là một phương pháp mới trong việc phát hiện đối tượng. Công việc trước đó về phát hiện đối tượng tái sử dụng các bộ phân loại để thực hiện phát hiện. Thay vào đó, hãy xem xét việc phát hiện đối tượng như là một vấn đề hồi quy để xác định các hộp giới hạn được phân tách không gian và xác suất lớp tương ứng. Một mạng nơ-ron đơn dự đoán các hộp giới hạn và xác suất lớp trực tiếp từ hình ảnh đầy đủ trong một lần đánh giá. Vì toàn bộ quy trình phát hiện là một mạng duy nhất, nó có thể được tối ưu hóa từ đầu đến cuối trực tiếp trên hiệu suất phát hiện.



Hình 2.5: Kiến trúc YOLO tổng quan [7]

2.3.1. Cách thức hoạt động

Mô hình YOLO (You Only Look Once) là một mô hình phát hiện đối tượng trong hình ảnh và video. Đầu vào của mô hình YOLO thường là một hình ảnh hoặc một khối video. Cụ thể, đầu vào của mô hình YOLO thường là một tensor hình ảnh với kích thước cố định và các kênh màu được chuẩn hóa. Đối với ảnh màu, thường sử dụng các kênh màu RGB (Red, Green, Blue), trong đó giá trị của mỗi pixel thường nằm trong khoảng từ 0 đến 255. Tuy nhiên, trước khi đưa vào mô hình, hình ảnh thường được chuẩn hóa để có giá trị pixel nằm trong khoảng từ 0 đến 1 hoặc từ -1 đến 1 để tăng hiệu suất của mô hình. Phương pháp hoạt động của YOLO có thể được mô tả như sau:

- **Chia hình ảnh thành lưới ô:** Hình ảnh đầu vào được chia thành một lưới ô có kích thước cố định. Mỗi ô trong lưới này được dự đoán có chứa đối tượng hoặc không chứa đối tượng.
- **Dự đoán bounding box:** Mỗi ô trong lưới dự đoán một hoặc nhiều bounding box để bao quanh đối tượng. Mỗi bounding box được mô tả bằng 5 thông số: tọa độ tâm (x, y), chiều rộng (w), chiều cao (h) và xác suất tồn tại của đối tượng trong bounding box.

- **Dự đoán xác suất lớp của đối tượng:** Ngoài các thông số của bounding box, mỗi ô cũng dự đoán xác suất của các lớp đối tượng có thể xuất hiện trong bounding box đó. Số lượng lớp đối tượng cần dự đoán được xác định trước và thường được biểu diễn bằng số chiều cuối cùng của tensor đầu ra.
- **Kết hợp dự đoán:** Kết hợp thông tin từ tất cả các ô trong lưới để tạo ra dự đoán cuối cùng về vị trí và lớp của các đối tượng trong hình ảnh. Các dự đoán này được thực hiện thông qua các bước như làm mềm xác suất (softmax), loại bỏ các bounding box dự đoán có xác suất thấp và thực hiện non-max suppression để loại bỏ các bounding box trùng lặp.

Kiến trúc YOLO thống nhất các thành phần riêng lẻ của việc phát hiện đối tượng vào một mạng nơ-ron duy nhất. Mạng YOLO được sử dụng các đặc trưng từ toàn bộ hình ảnh để dự đoán mỗi hộp giới hạn. Nó cũng dự đoán tất cả các hộp giới hạn qua tất cả các lớp cho một hình ảnh cùng một lúc. Điều này có nghĩa là mạng YOLO suy luận một cách toàn cầu về toàn bộ hình ảnh và tất cả các đối tượng trong hình ảnh. Thiết kế YOLO cho phép việc huấn luyện từ đầu đến cuối và tốc độ thời gian thực trong khi vẫn duy trì độ chính xác trung bình cao.

Mô hình YOLO chia hình ảnh đầu vào thành một lưới có kích thước $S \times S$. Nếu trung tâm của một đối tượng nằm trong một ô lưới, ô lưới đó sẽ chịu trách nhiệm phát hiện đối tượng đó. Mỗi ô lưới dự đoán B hộp giới hạn và confidence scores cho những hộp đó. Các điểm tự tin này phản ánh mức độ tự tin của mô hình rằng hộp chứa một đối tượng và cũng là mức độ chính xác mà nó nghĩ rằng hộp được dự đoán là đúng. Theo cách hình thức, xác định confident là $\text{Pr}(\text{Object}) * \text{IOU}_{\text{PRED}}^{\text{TRUTH}}$. Nếu không có đối tượng nào tồn tại trong ô đó, các confidence scores sẽ là không. Thực tế, cần confidence score bằng với tổng liên kết (IOU) giữa hộp được dự đoán và thực tế. Mỗi hộp giới hạn bao gồm 5 dự đoán: x, y, w, h và độ tin cậy. Các tọa độ (x, y) đại diện cho trung tâm của hộp so với ranh giới của ô lưới. Chiều rộng và chiều cao được dự đoán

so với toàn bộ hình ảnh. Cuối cùng, dự đoán độ tin cậy đại diện cho IOU giữa hộp được dự đoán và bất kỳ hộp thực tế nào. Mỗi ô lưới cũng dự đoán C xác suất lớp có điều kiện, $\text{Pr}(\text{Class}_i | \text{Object})$. Những xác suất này được điều kiện trên ô lưới chứa một đối tượng. Mô hình chỉ dự đoán một bộ xác suất lớp duy nhất cho mỗi ô lưới, bất kể số lượng hộp B . Trong thời gian kiểm tra, nhân các xác suất lớp có điều kiện và các dự đoán độ tin cậy của từng hộp:

$$\text{Pr}(\text{Class}_i | \text{Object}) * \text{Pr}(\text{Object}) * \text{IOU}_{\text{PRED}}^{\text{TRUTH}} = \text{Pr}(\text{Class}_i) * \text{IOU}_{\text{PRED}}^{\text{TRUTH}}$$

Điều này mang lại các confidence score cụ thể cho mỗi hộp. Các confidence score này mã hóa cả xác suất của lớp đó xuất hiện trong hộp và mức độ phù hợp của hộp được dự đoán với đối tượng.



Hình 2.6: Minh họa việc phát hiện người trong ảnh với YOLO [7]

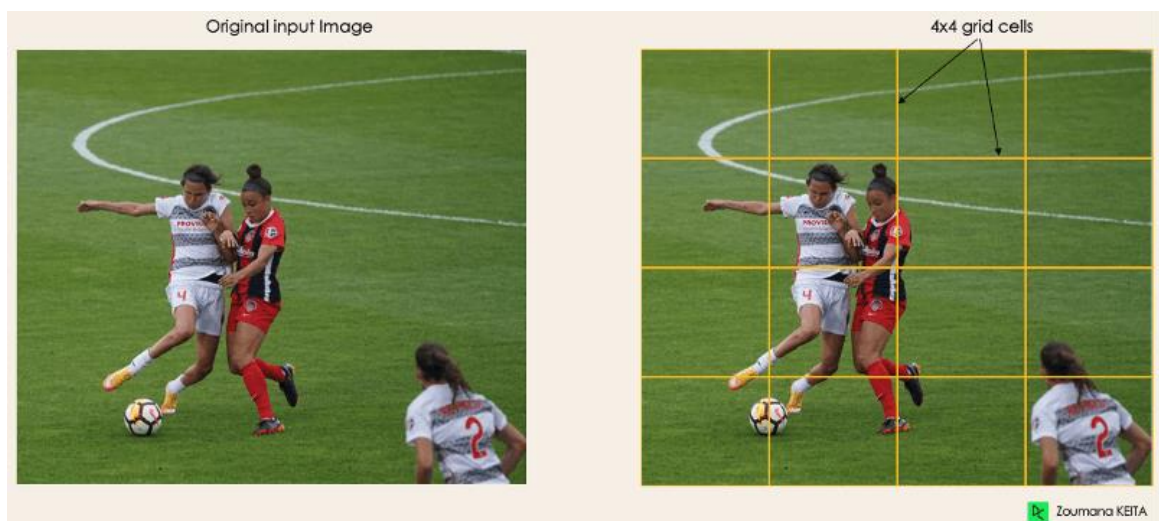
Để hiểu rõ hơn về các thành phần cũng như cách thức hoạt động của YOLO, hãy cùng đi sâu tìm một số thành phần mà một mô hình YOLO cần có dưới đây:

❖ Residual Block:

Residual Block hay có thể hiểu là khối dư, là khối kết nối trượt học các hàm dư thừa với tham chiếu đến các đầu vào lớp, thay vì học các hàm không có tham chiếu. Trong mạng nơ-ron sâu, các khối dư (residual blocks) cho phép mạng học các hàm dư thừa (residual functions) bằng cách thêm hoặc điều chỉnh

các phần nhỏ của đầu vào. Thay vì học toàn bộ hàm mới từ đầu, mạng chỉ cần học các sự thay đổi nhỏ (residual) để biến đổi đầu vào thành đầu ra mong muốn. Điều này giúp mạng học được các biểu diễn sâu hơn và hiệu quả hơn, cũng như giải quyết vấn đề của việc huấn luyện mạng sâu.

Trong kiến trúc YOLO, để phát hiện vật thể trong bức ảnh, ví dụ như phát hiện người trong ảnh, bước đầu tiên trong quá trình này là chia bức tranh ban đầu thành một lưới hình ô có hình dạng bằng nhau, giống như cách bạn chia một bức tranh thành các ô vuông nhỏ. Mỗi ô trong lưới đảm nhận trách nhiệm định vị và dự đoán lớp của đối tượng mà nó bao phủ, cùng với giá trị xác suất/tin cậy. Nói cách khác, mỗi ô sẽ cố gắng xác định xem đối tượng nào (ví dụ: con người, quả bóng) nằm trong phạm vi của nó và dự đoán xác suất rằng đối tượng đó có tồn tại trong ô đó, cùng với mức độ tin cậy (confidence) của dự đoán đó. Điều này giúp "phân vùng" bức tranh và tập trung vào việc xác định các đối tượng cụ thể trong từng phần của bức tranh một cách hiệu quả.



Hình 2.7: Bức ảnh được chia thành 4x4 các ô vuông có kích thước bằng nhau [7]

❖ Bounding box:

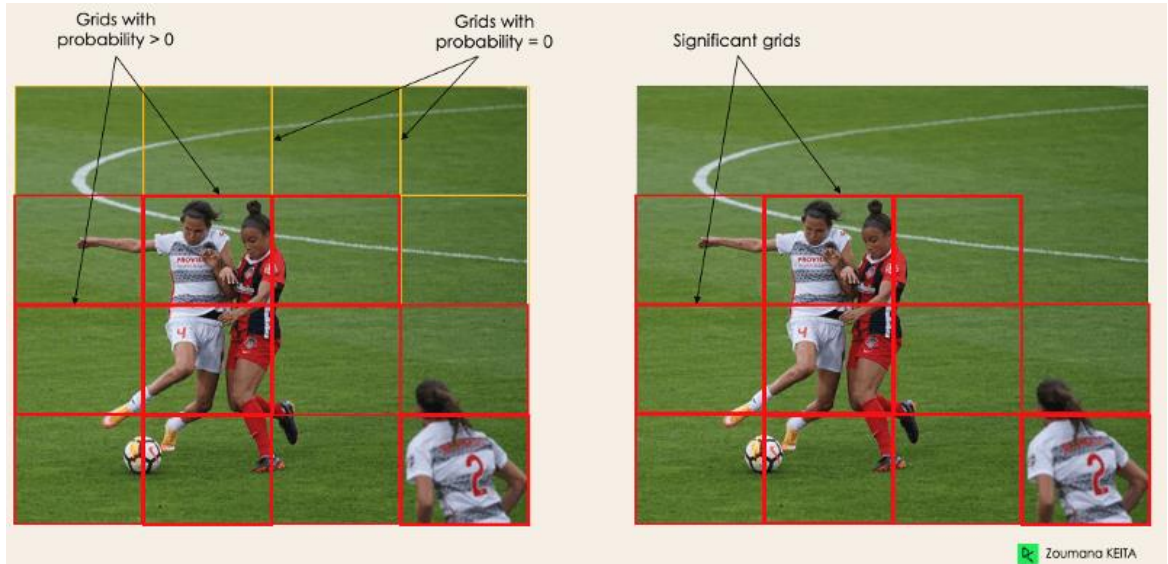
Bước tiếp theo là xác định các hộp giới hạn (bounding boxes) tương ứng với các hình chữ nhật làm nổi bật tất cả các đối tượng trong hình ảnh. Chúng ta có thể có nhiều hộp giới hạn như có bao nhiêu đối tượng trong một hình ảnh cụ thể.

YOLO xác định các thuộc tính của các hộp giới hạn này bằng cách sử dụng một mô-đun hồi quy duy nhất theo định dạng sau, trong đó Y là biểu diễn

vector cuối cùng cho mỗi hộp giới hạn. Điều này đặc biệt quan trọng trong quá trình huấn luyện của mô hình

$$Y = [pc, bx, by, bh, bw, c1, c2]$$

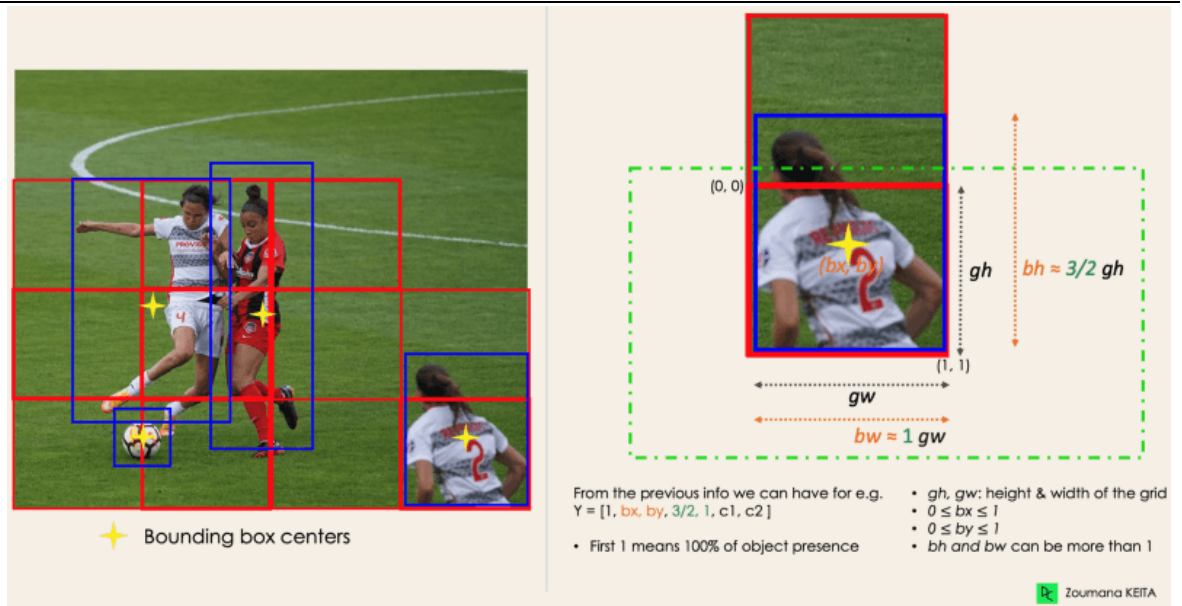
Các thành phần trong Y bao gồm: pc tương ứng với điểm số xác suất của lưới chứa một đối tượng. Ví dụ, tất cả các lưới màu đỏ sẽ có điểm số xác suất lớn hơn không như hình dưới đây.



Hình 2.8: Điểm số xác suất các ô màu đỏ và vàng [7]

Hình ảnh bên phải là phiên bản đơn giản hóa vì xác suất của mỗi ô màu vàng là không (không đáng kể). bx và by là các tọa độ x và y của trung tâm của bounding box liên quan đến ô lưới bao quanh. bh và bw tương ứng với chiều cao và chiều rộng của hộp giới hạn liên quan đến ô lưới bao quanh. $c1$ và $c2$ tương ứng với hai lớp Player và Ball. Chúng ta có thể có nhiều lớp hơn tùy thuộc vào trường hợp sử dụng.

Để hiểu, chúng ta hãy chú ý kỹ hơn đến người chơi ở phía dưới bên phải:



Hình 2.9: Bounding box centers [7]

❖ Intersection Over Unions – IOU:

Giao nhau qua Hiệu (Intersection Over Union - IOU) là một phép đo thước đo sự overlap (giao nhau) giữa hai vùng quan sát. Trong ngữ cảnh của bài toán phát hiện đối tượng, IOU thường được sử dụng để đánh giá sự chính xác của các hộp giới hạn được dự đoán so với hộp giới hạn thực tế.

Công thức tính IOU được biểu diễn bằng tỷ lệ giữa diện tích phần giao và diện tích phần hợp của hai hộp giới hạn:

$$IOU = \frac{\text{Diện tích phần giao nhau}}{\text{Diện tích phần hợp}}$$

Kết quả của IOU nằm trong phạm vi từ 0 đến 1, trong đó giá trị càng gần 1 thì sự overlap giữa hai hộp giới hạn càng lớn, và giá trị càng gần 0 thì hai hộp giới hạn không giao nhau. Trong phát hiện đối tượng, IOU thường được sử dụng để xác định xem một hộp giới hạn dự đoán có chứa một đối tượng thực tế hay không, dựa trên một ngưỡng IOU được xác định trước.

Hầu hết thời gian, một đối tượng duy nhất trong một hình ảnh có thể có nhiều ứng viên hộp lưới để dự đoán, mặc dù không phải tất cả đều có liên quan. Mục tiêu của IOU (một giá trị từ 0 đến 1) là loại bỏ các ô lưới không cần thiết để chỉ giữ lại những ô lưới quan trọng. Dưới đây là logic đằng sau nó:

- Người dùng xác định ngưỡng lựa chọn IOU của mình, ví dụ, 0.5.
- Sau đó, YOLO tính toán IOU của mỗi ô lưới, là diện tích giao chia cho diện tích hợp.
- Cuối cùng, nó bỏ qua dự đoán của các ô lưới có $\text{IOU} \leq \text{ngưỡng}$ và chỉ xem xét những ô lưới có $\text{IOU} > \text{ngưỡng}$.

Dưới đây là một minh họa về việc áp dụng quá trình lựa chọn lưới cho đối tượng ở góc dưới bên trái. Chúng ta có thể quan sát rằng đối tượng ban đầu có hai ứng viên lưới, sau đó chỉ "Lưới 2" được chọn cuối cùng.



Hình 2.10: Minh họa quá trình lựa chọn lưới [7]

❖ Non-Maximum Suppression (NMS):

Thiết lập một ngưỡng cho IOU không phải lúc nào cũng đủ vì một đối tượng có thể có nhiều hộp với IOU vượt quá ngưỡng đó, và để lại tất cả những hộp đó có thể gây nhiễu. Đây là nơi chúng ta có thể sử dụng NMS để chỉ giữ lại các hộp có điểm số xác suất phát hiện cao nhất.

NMS giúp chúng ta giữ lại chỉ các hộp tốt nhất bằng cách thực hiện các bước sau:

- Sắp xếp tất cả các hộp theo điểm số xác suất của phát hiện theo thứ tự giảm dần.

- Chọn hộp có điểm số xác suất cao nhất làm hộp chính thức và đưa vào danh sách cuối cùng.
- Loại bỏ tất cả các hộp có IOU với hộp chính thức lớn hơn hoặc bằng một ngưỡng được xác định trước.
- Lặp lại quy trình trên các hộp còn lại cho đến khi không còn hộp nào có thể được loại bỏ.

Bằng cách này, NMS giúp giữ lại chỉ các hộp có xác suất phát hiện cao nhất, loại bỏ nhiễu và đảm bảo rằng chỉ có một hộp tốt nhất được giữ lại cho mỗi đối tượng.

❖ Loss function:

Hàm mất mát hay hàm lỗi trong YOLO được tính trên việc dự đoán nhãn mô hình để tính. Cụ thể, nó là tổng độ lỗi của 3 thành phần con:

- Classification loss: Độ lỗi của việc dự đoán loại nhãn của đối tượng.
- Localization loss: Độ lỗi của dự đoán tọa độ tâm, chiều dài, chiều rộng của hộp giới hạn (bounding box).
- Confidence loss: Độ lỗi của việc dự đoán hộp giới hạn đó chứa đối tượng so với nhãn thực tế tại ô vuông đó.

Hãy cùng đi sâu vào tìm hiểu về 3 thành phần con trên:

a. Classification loss:

Classification loss - độ lỗi của việc dự đoán loại nhãn của object, hàm lỗi này chỉ tính trên những ô vuông có xuất hiện object, còn những ô vuông khác ta không quan tâm. Classification loss được tính bằng công thức sau:

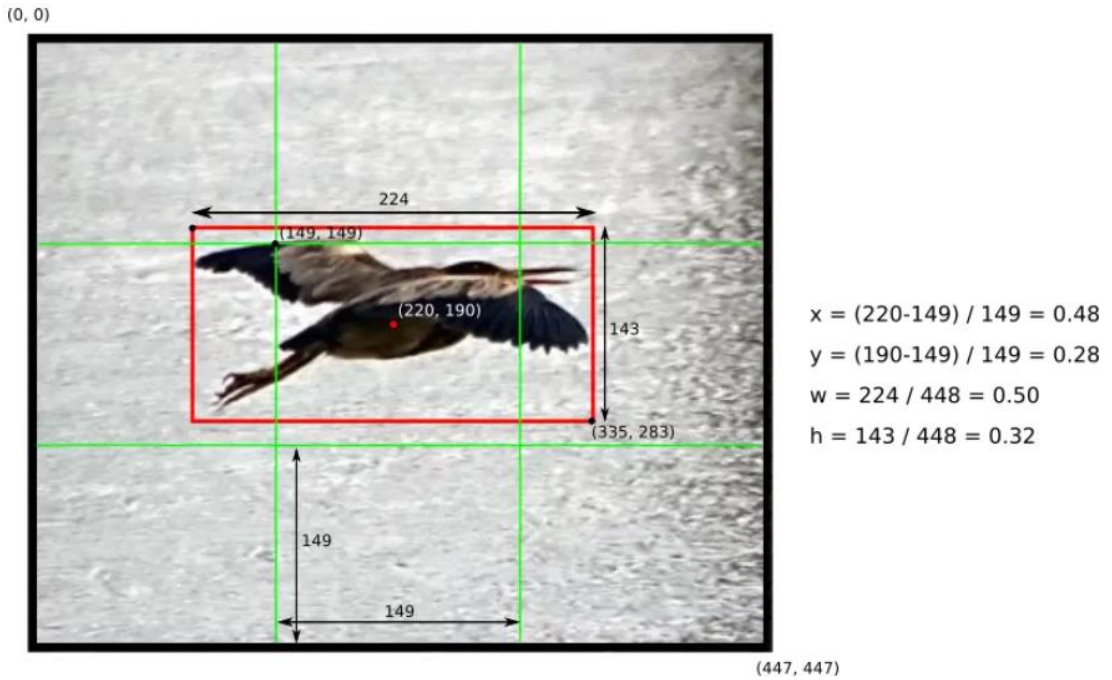
$$L_{\text{classification}} = \sum_{i=0}^{S^2} \mathbb{I}_i^{\text{obj}} \sum_{c \in \text{class}} (p_i(c) - \hat{p}_i(c))^2$$

Trong đó: $\mathbb{I}_i^{\text{obj}}$: Bằng 1 nếu ô vuông đang xét có obj, ngược lại bằng 0

$\hat{p}_i(c)$: Là xác suất có điều kiện của lớp c tại ô vuông tương ứng mà mô hình dự đoán.

b. Localization loss:

Localization loss là hàm lỗi dùng để tính giá trị lỗi cho boundary box được dự đoán bao gồm tọa độ tâm, chiều rộng, chiều cao của so với vị trí thực tế từ dữ liệu huấn luyện của mô hình. Lưu ý rằng không nên tính giá trị hàm lỗi này trực tiếp từ kích thước ảnh thực tế mà cần phải chuẩn hóa về $[0, 1]$ so với tâm của bounding box. Việc chuẩn hóa kích thước này giúp cho mô hình dự đoán nhanh hơn và chính xác hơn so với để giá trị mặc định của ảnh, ví dụ:



Hình 2.11: Location loss [7]

Giá trị hàm Localization loss được tính trên tổng giá trị lỗi dự đoán tọa độ tâm (x, y) và (w, h) của predicted bounding box với ground-truth bounding box. Tại mỗi ô có chứa object, ta chọn 1 boundary box có IOU (Intersection over union) tốt nhất, rồi sau đó tính độ lỗi theo các boundary box này. Giá trị hàm lỗi dự đoán tọa độ tâm (x, y) của predicted bounding box và (\hat{x}, \hat{y}) là tọa độ tâm của truth bounding box được tính như sau:

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2$$

Giá trị hàm lỗi dự đoán (w, h) của predicted bounding box so với truth bounding box được tính như sau:

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2$$

Với ví dụ trên thì $S=7$, $B=2$, còn λ_{coord} là trọng số thành phần trong paper gốc tác giả lấy giá trị là 5.

c. Confident loss:

Confidence loss là độ lỗi giữa dự đoán boundary box đó chứa object so với nhãn thực tế tại ô vuông đó. Độ lỗi này tính trên cả những ô vuông chứa object và không chứa object.

$$L_{\text{confidence}} = \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 + \lambda_{\text{noobject}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{noobj}} (C_i - \hat{C}_i)^2$$

Với ví dụ trên thì $S=7$, $B=2$, còn $\lambda_{\text{noobject}}$ là trọng số thành phần trong paper gốc tác giả lấy giá trị là 0.5. Đối với các hộp j của ô thứ i nếu xuất hiện object thì $C_i=1$ và ngược lại.

d. Tổng loss:

Ta có hàm lỗi là tổng của 3 hàm lỗi phía trên:

$$L_{\text{total}} = L_{\text{Classification}} + L_{\text{Localization}} + L_{\text{Confident}}$$

2.3.2. YOLOv5

YOLOv5 là một thuật toán phát hiện đối tượng mạnh mẽ được phát triển bởi Ultralytics. Bài viết này đi sâu vào tìm hiểu về kiến trúc YOLOv5, chiến lược tăng cường dữ liệu, phương pháp đào tạo và kỹ thuật tính toán tối ưu. Sự hiểu biết toàn diện này sẽ giúp cải thiện ứng dụng thực tế của bạn về phát hiện đối tượng trong các lĩnh vực khác nhau, bao gồm giám sát, xe tự hành và nhận dạng hình ảnh. YOLOv5 có kiến trúc của nó bao gồm ba phần chính:

- **Xương sống:** Đây là cơ quan chính của mạng. Trong YOLOv5, xương sống được thiết kế bằng cách sử dụng New CSP-Darknet53 cấu trúc, một sửa đổi của kiến trúc Darknet được sử dụng trong các phiên bản trước.
- **Cổ:** Phần này kết nối xương sống và đầu. Trong YOLOv5, kiến trúc SPPF và New CSP-PAN được sử dụng.

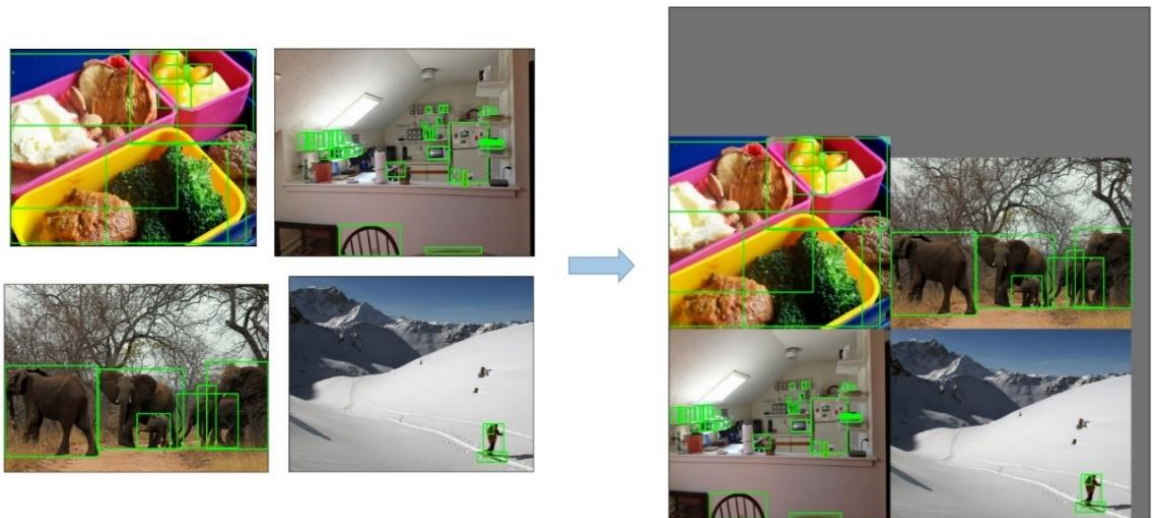
- Đầu: Phần này chịu trách nhiệm tạo ra đầu ra cuối cùng. YOLOv5 sử dụng YOLOv3 Head cho mục đích này.

Tiếp theo chúng ta cùng đi sâu vào một số khía cạnh đặc biệt của YOLOv5 giúp nó trở nên tối ưu và hiệu quả hơn so với các phiên bản tiền nhiệm:

❖ Kỹ thuật tăng cường dữ liệu:

YOLOv5 Sử dụng các kỹ thuật tăng cường dữ liệu khác nhau để cải thiện khả năng khái quát hóa và giảm overfitting của mô hình. Những kỹ thuật này bao gồm:

- **Mosaic Augmentation:** Một kỹ thuật xử lý hình ảnh kết hợp bốn hình ảnh đào tạo thành một theo cách khuyến khích các mô hình phát hiện đối tượng xử lý tốt hơn các quy mô và bản dịch đối tượng khác nhau.



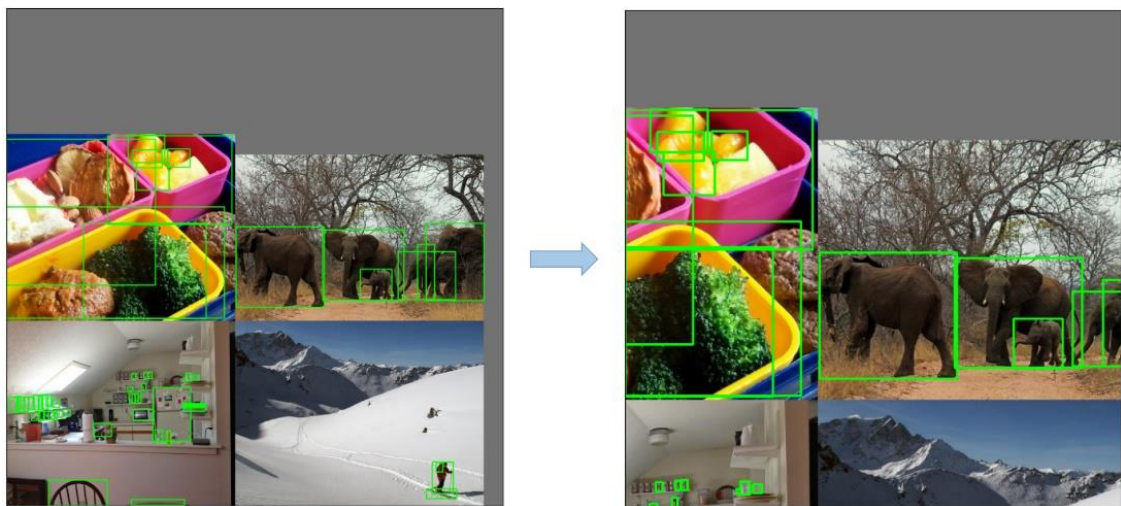
Hình 2.12: Minh họa kỹ thuật Mosaic Augmentation [8]

- **Tăng cường sao chép-dán:** Một phương pháp tăng cường dữ liệu sáng tạo sao chép các bản vá ngẫu nhiên từ một hình ảnh và dán chúng vào một hình ảnh được chọn ngẫu nhiên khác, tạo ra một mẫu đào tạo mới một cách hiệu quả.



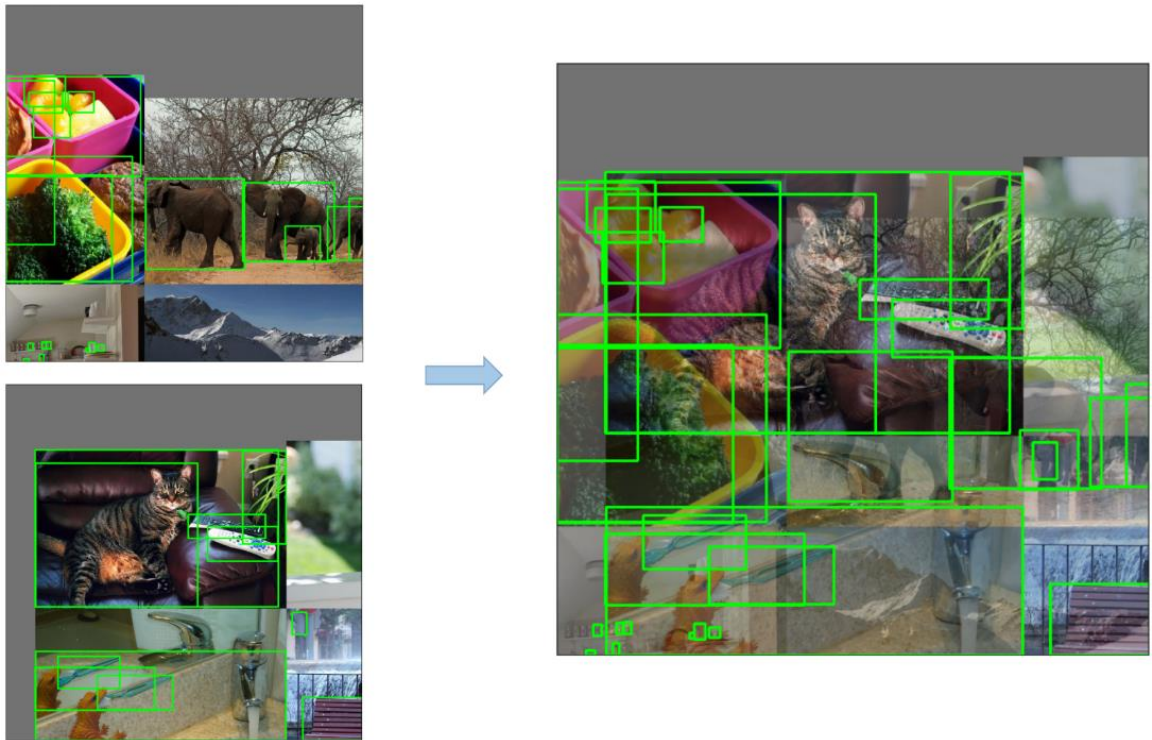
Hình 2.13: Mô tả phương pháp tăng cường sao chép-dán [8]

- **Biến đổi affine ngẫu nhiên:** Điều này bao gồm xoay ngẫu nhiên, chia tỷ lệ, dịch và cắt hình ảnh.



Hình 2.14: Mô tả kỹ thuật biến đổi ngẫu nhiên [8]

- **MixUp Augmentation:** Một phương pháp tạo ra hình ảnh tổng hợp bằng cách kết hợp tuyến tính của hai hình ảnh và các nhãn liên quan của chúng



Hình 2.15: Minh họa kỹ thuật MixUp Augmentation [8]

❖ Chiến lược đào tạo:

YOLOv5 Áp dụng một số chiến lược đào tạo tinh vi để nâng cao hiệu suất của mô hình, bao gồm:

- **Đào tạo đa cấp độ:** Các hình ảnh đầu vào được thay đổi tỷ lệ ngẫu nhiên trong phạm vi từ 0,5 đến 1,5 lần kích thước ban đầu của chúng trong quá trình đào tạo.
- **AutoAnchor:** Chiến lược này tối ưu hóa các hộp neo trước đó để phù hợp với các đặc điểm thống kê của các hộp sự thật cơ bản trong dữ liệu tùy chỉnh của bạn.
- **Khởi động và lập lịch Cosine LR:** Một phương pháp để điều chỉnh tốc độ học tập để nâng cao hiệu suất mô hình.
- **Đường trung bình động hàm mũ (EMA):** Một chiến lược sử dụng trung bình của các tham số trong các bước trước đây để ổn định quá trình đào tạo và giảm lỗi khái quát hóa.

- **Đào tạo chính xác hỗn hợp:** Một phương pháp để thực hiện các hoạt động ở định dạng nửa chính xác, giảm sử dụng bộ nhớ và tăng cường tốc độ tính toán.
- **Hyperparameter Evolution:** Một chiến lược để tự động điều chỉnh các siêu tham số để đạt được hiệu suất tối ưu.

❖ **Các tính năng bổ sung:**

- **Compute loss:**

Hàm mất mát trong YOLOv5 được tính là sự kết hợp của ba thành phần tổn thất riêng lẻ:

- **Class Loss (BCE Loss):** Binary Cross-Entropy loss, đo sai số cho nhiệm vụ phân loại.
- **Objectness loss (BCE Loss):** Một Binary Cross-Entropy loss khác, tính toán lỗi trong việc phát hiện xem một đối tượng có mặt trong một ô lưới cụ thể hay không.
- **Location Loss (CIoU Loss):** Completed IoU loss, đo lỗi trong việc bản địa hóa đối tượng trong ô lưới.

Hàm mất mát tổng thể được mô tả bởi công thức sau:

$$Loss = \lambda_1 L_{cls} + \lambda_2 L_{obj} + \lambda_3 L_{loc}$$

- **Balance loss:**

Các hàm mất mát về đối tượng của ba lớp dự đoán (P3, P4, P5) được đặt trọng số khác nhau. Các trọng số cân bằng lần lượt là [4.0, 1.0, 0.4]. Phương pháp này đảm bảo rằng các dự đoán ở các tỷ lệ khác nhau đóng góp một cách phù hợp vào tổng hàm mất mát.

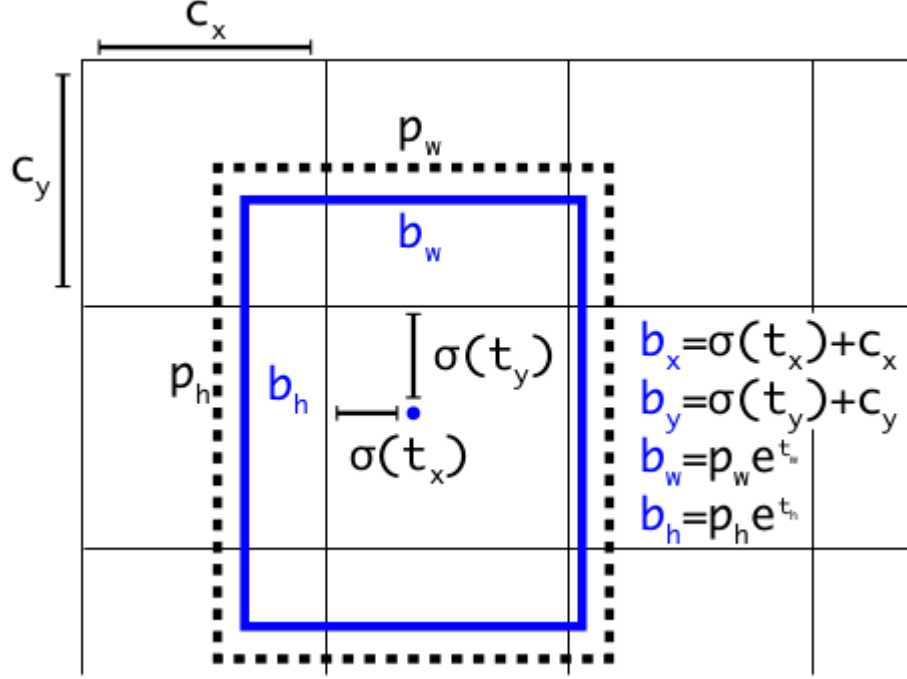
$$L_{obj} = 4.0 \cdot L_{obj}^{small} + 1.0 \cdot L_{obj}^{medium} + 0.4 \cdot L_{obj}^{large}$$

- **Loại bỏ độ nhạy cảm của lưới:**

Kiến trúc YOLOv5 thực hiện một số thay đổi quan trọng đối với chiến lược dự đoán bounding box so với các phiên bản YOLO trước đó. Trong

YOLOv2 và YOLOv3, các tọa độ của hộp được dự đoán trực tiếp bằng cách sử dụng kích hoạt của lớp cuối cùng.

$$b_x = \sigma(t_x) + c_x b_y = \sigma(t_y) + c_y b_w = p_w \cdot e^{t_w} b_h = p_h \cdot e^{t_h}$$

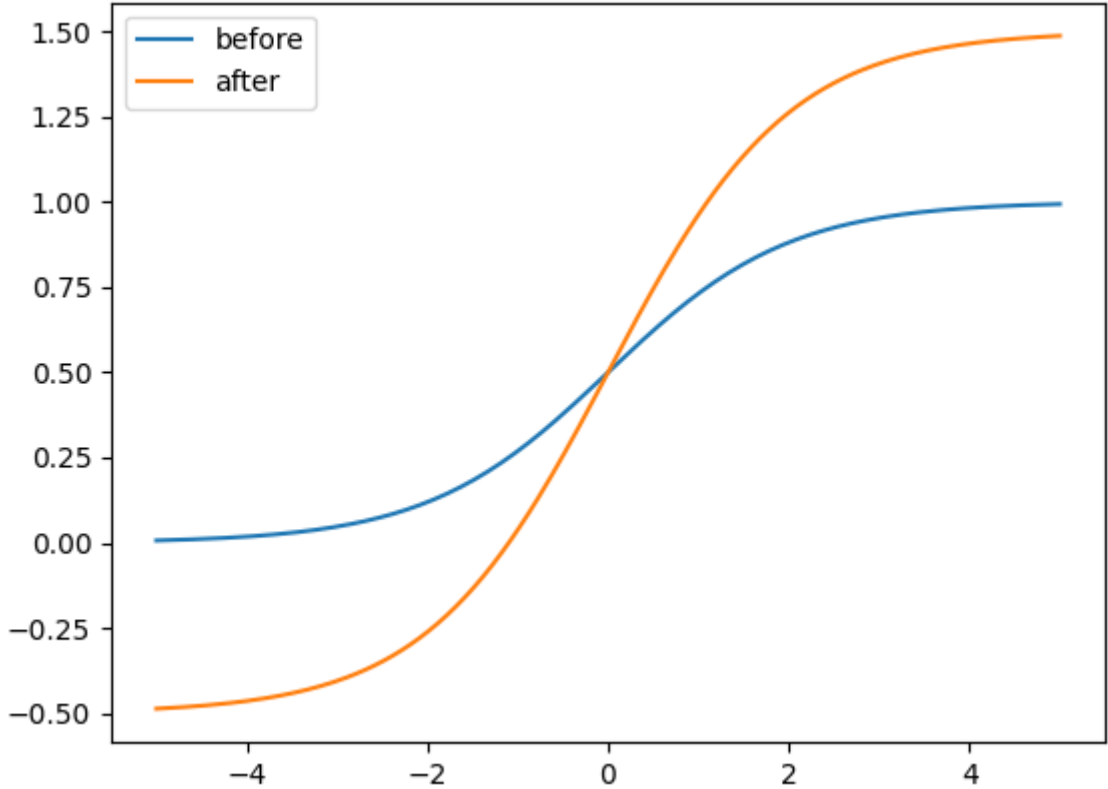


Hình 2.16: Cơ chế định vị hộp giới hạn trong YOLO [15]

Trong YOLOv5, công thức để dự đoán tọa độ của hộp đã được cập nhật để giảm độ nhạy cảm của lưới và ngăn mô hình dự đoán các kích thước hộp không giới hạn. Các công thức được điều chỉnh để tính toán hộp giới hạn được dự đoán như sau:

$$\begin{aligned} b_x &= (2 \cdot \sigma(t_x) - 0.5) + c_x b_y = (2 \cdot \sigma(t_y) - 0.5) + c_y b_w \\ &= p_w \cdot (2 \cdot \sigma(t_w))^2 b_h = p_h \cdot (2 \cdot \sigma(t_h))^2 \end{aligned}$$

So sánh sự thay đổi về phân dịch chuyển điểm trung tâm trước và sau khi co giãn. Phạm vi phân dịch chuyển điểm trung tâm được điều chỉnh từ (0, 1) thành (-0.5, 1.5). Do đó, phân dịch chuyển có thể dễ dàng đạt được giá trị 0 hoặc 1.



Hình 2.17: Đồ thị Loss Function trước và sau [8]

- **Xây dựng hàm mục tiêu:**

Quá trình xây dựng mục tiêu trong YOLOv5 rất quan trọng đối với hiệu quả đào tạo và độ chính xác của mô hình. Nó liên quan đến việc gán các hộp sự thật mặt đất cho các ô lưới thích hợp trong bản đồ đầu ra và khớp chúng với các hộp neo thích hợp.

Quá trình này tuân theo các bước sau:

- Tính tỷ lệ kích thước Ground truth box và kích thước của mỗi mẫu Anchor box.

$$r_w = w_{gt}/w_{at}$$

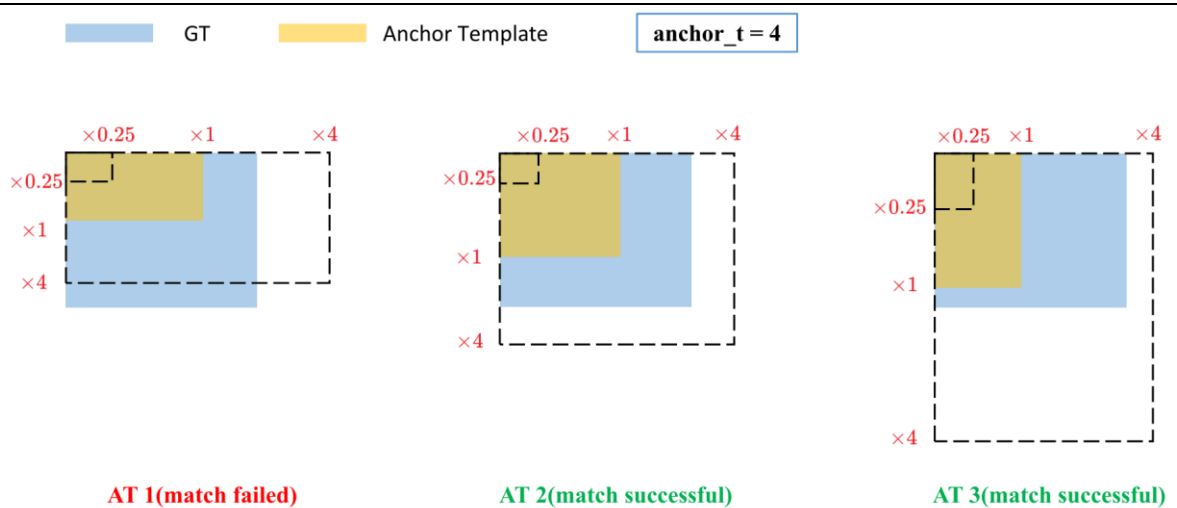
$$r_h = h_{gt}/h_{at}$$

$$r_w^{\max} = \max(r_w, 1/r_w)$$

$$r_h^{\max} = \max(r_h, 1/r_h)$$

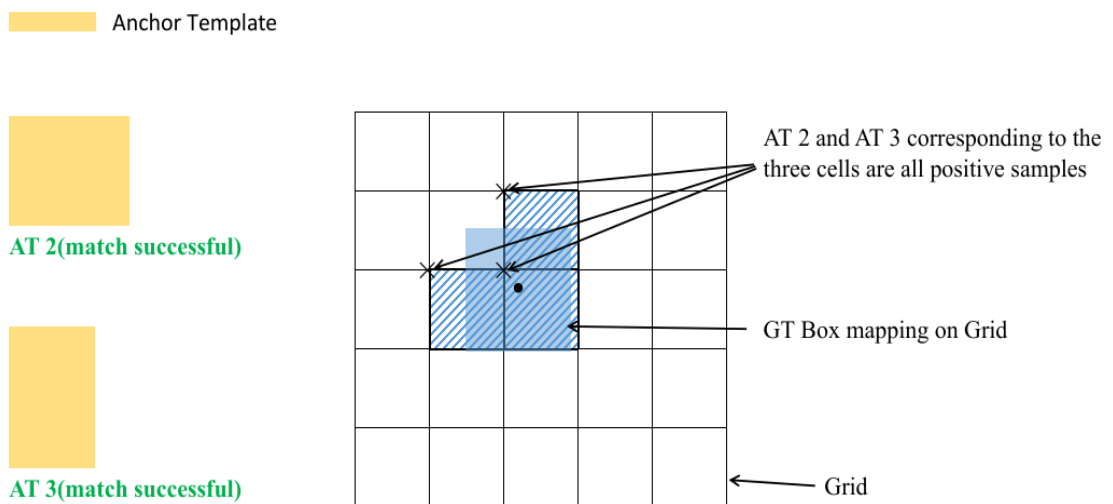
$$r^{\max} = \max(r_w^{\max}, r_h^{\max})$$

$$r^{\max} < \text{anchor}_t$$



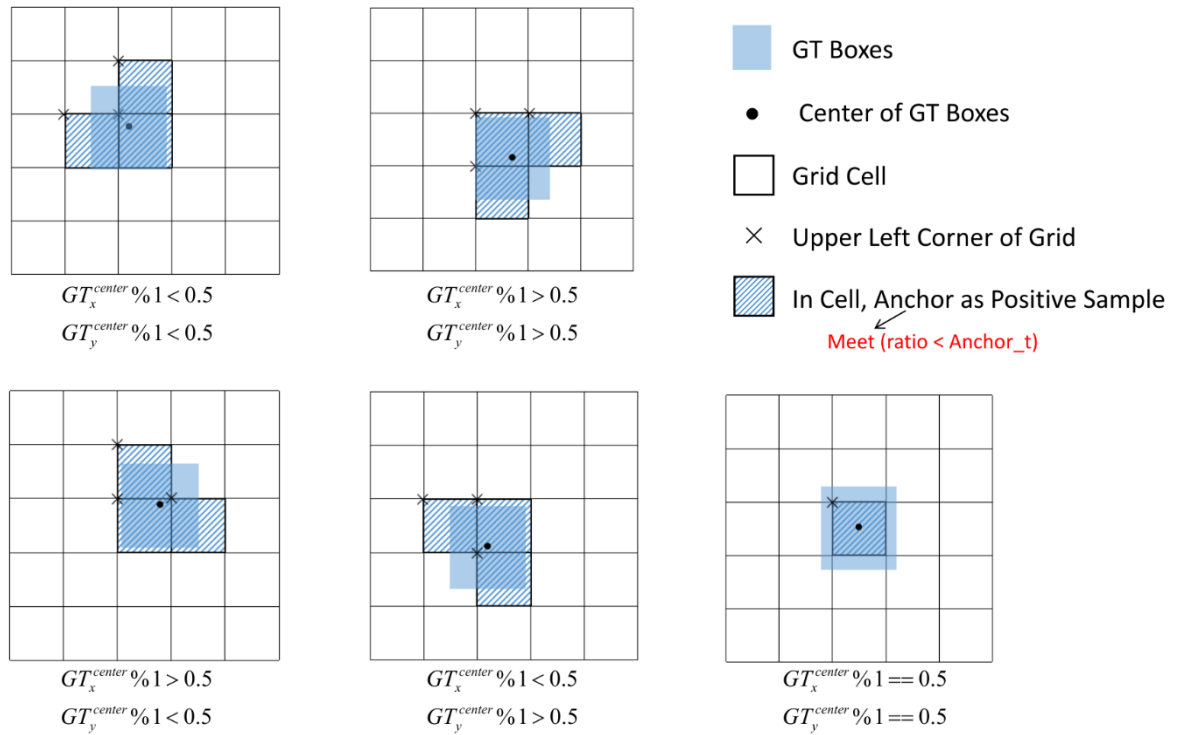
Hình 2.18: Tính tỉ lệ Ground truth box [8]

- Nếu tỷ lệ tính toán nằm trong ngưỡng, kết hợp hộp ground truth với Anchor box.



Hình 2.19: Kết hợp Ground truth box và Anchor box [8]

- Gán anchor box phù hợp cho các ô tương ứng, nhớ rằng do phân dịch chuyển điểm trung tâm được điều chỉnh lại, một hộp ground truth có thể được gán cho nhiều hơn một anchor. Do phạm vi phân dịch chuyển điểm trung tâm được điều chỉnh từ (0, 1) thành (-0.5, 1.5), hộp Ground Truth có thể được gán cho nhiều anchor.



Hình 2.20: Gán các Anchor box phù hợp vào ô tương ứng [8]

Như vậy, quá trình xây dựng hàm mục tiêu đảm bảo rằng mỗi đối tượng thực tế được gán và kết hợp một cách chính xác trong quá trình huấn luyện, giúp cho YOLOv5 học nhiệm vụ phát hiện đối tượng một cách hiệu quả hơn.

2.3.3. Ưu điểm

Hiệu suất cao: YOLO được thiết kế để thực hiện phát hiện đối tượng nhanh chóng trên toàn bộ hình ảnh một cách hiệu quả. Bằng cách thực hiện dự đoán đồng thời trên toàn bộ hình ảnh, YOLO có thể đạt được tốc độ phát hiện cao mà vẫn đảm bảo độ chính xác.

Dự đoán đồng thời: Mô hình YOLO thực hiện dự đoán đồng thời cho tất cả các bounding box và lớp đối tượng trên một lưới ô được chia nhỏ từ hình ảnh đầu vào. Điều này cho phép YOLO tránh được các bước chia nhỏ và trích xuất tính năng riêng lẻ, giúp giảm thiểu lỗi và tăng hiệu suất.

Độ chính xác tốt: Dù có tốc độ nhanh, YOLO vẫn đạt được độ chính xác tốt trong việc phát hiện đối tượng. Các phiên bản mới của YOLO như YOLOv4 và YOLOv5 đã được cải tiến để cải thiện độ chính xác của mô hình.

Khả năng áp dụng thời gian thực: Với tốc độ nhanh và độ chính xác cao, YOLO thích hợp cho các ứng dụng yêu cầu phát hiện đối tượng trong thời gian thực như xe tự lái, giám sát an ninh, nhận diện biển số, và nhiều ứng dụng khác.

Học cập nhật nhanh: YOLO là một mô hình dựa trên mạng neural convolutional, có thể được huấn luyện và cập nhật một cách nhanh chóng trên các bộ dữ liệu lớn. Điều này giúp YOLO có thể được tinh chỉnh và cải thiện hiệu suất một cách dễ dàng cho các ứng dụng cụ thể.

Ứng dụng đa dạng: YOLO có thể được áp dụng trong nhiều lĩnh vực khác nhau bao gồm xe tự lái, giám sát an ninh, quản lý giao thông, nhận diện hành vi con người, y học và nhiều lĩnh vực khác, nhờ vào khả năng phát hiện đa dạng loại đối tượng và độ chính xác của nó.

2.3.1. Nhược điểm

Hạn chế trong việc xử lý các vật thể mờ: YOLO có thể gặp khó khăn trong việc nhận diện các vật thể mờ hoặc có độ tương phản thấp. Điều này có thể làm giảm độ chính xác của mô hình đặc biệt khi sử dụng trong điều kiện ánh sáng kém hoặc hình ảnh nhiễu.

Khả năng nhận diện đối tượng chính xác phụ thuộc vào kích thước lưới: YOLO chia hình ảnh thành lưới ô có kích thước cố định, do đó khả năng nhận diện đối tượng chính xác phụ thuộc vào việc lựa chọn kích thước lưới ô. Việc lựa chọn kích thước lưới ô không phù hợp có thể dẫn đến việc mô hình không nhận diện được các đối tượng nhỏ hoặc chi tiết trong hình ảnh.

Yêu cầu phần cứng mạnh mẽ: YOLO yêu cầu một phần cứng mạnh mẽ để thực hiện dự đoán đồng thời trên toàn bộ hình ảnh, đặc biệt là với các phiên bản mới như YOLOv4 và YOLOv5. Điều này có thể là một hạn chế đối với các ứng dụng có tài nguyên hạn chế hoặc yêu cầu tính di động cao.

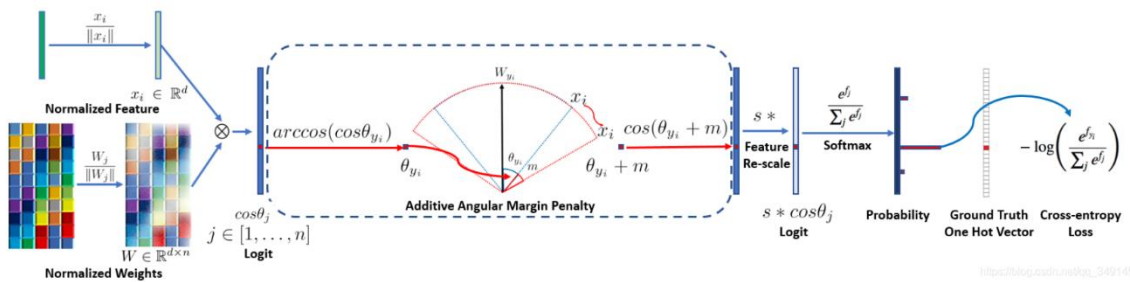
2.4. ArcFace

2.4.1. Tổng quan

Trong thời đại công nghệ hiện đại, việc nhận diện khuôn mặt sâu đang trở thành một lĩnh vực nghiên cứu động lực với nhiều ứng dụng tiềm năng. Từ việc bảo mật thông tin cá nhân đến công nghệ an ninh, từ giao tiếp trực tuyến đến ứng dụng trong thương mại điện tử, nhận diện khuôn mặt sâu đóng vai trò quan trọng trong việc cung cấp các giải pháp hiệu quả cho nhiều ngành công nghiệp khác nhau. Tuy nhiên, mặc dù đã có sự tiến bộ đáng kể trong việc phát triển các mô hình nhận diện khuôn mặt dựa trên mạng neural, việc tạo ra các biểu diễn khuôn mặt chính xác và phân biệt giữa các đối tượng vẫn là một thách thức.

Trong bối cảnh đó, phương pháp ArcFace ra đời như một cải tiến quan trọng trong lĩnh vực nhận diện khuôn mặt sâu. ArcFace không chỉ kế thừa tính hiệu quả của các phương pháp trước đó mà còn mang lại những cải tiến đáng kể trong việc tạo ra các biểu diễn khuôn mặt có độ phân biệt cao.

Phương pháp ArcFace là một sự kết hợp tinh tế giữa nhiều thành phần chính trong lĩnh vực nhận diện khuôn mặt sâu, nhằm mục tiêu tối ưu hóa hiệu suất của các mô hình nhận diện khuôn mặt. Điểm mạnh của ArcFace nằm ở khả năng kết hợp tính biểu diễn mạnh mẽ của softmax loss và khả năng phân biệt chính xác của Angular margin loss, tạo ra một phương pháp huấn luyện mạnh mẽ và linh hoạt.



Hình 2.21: Huấn luyện mô hình nhận diện khuôn mặt sâu bằng ArcFace Loss ($K=1$) và Sub-center ArcFace Loss ($K=3$) [9]

Tiếp đến chúng ta cùng tìm hiểu sâu hơn về một số điểm đặc biệt của ArcFace giúp nó trở thành sự lựa chọn tốt cho bài toán nhận diện khuôn mặt.

2.4.2. Additive Angular Margin Loss (ALLM) trong ArcFace

Trước hết, chúng ta cùng tìm hiểu về Softmax loss trong bài toán tối ưu hàm chi phí. Softmax loss là một phương pháp chính trong việc đào tạo mô hình nhận diện khuôn mặt sâu. Phương pháp này tính toán xác suất của mỗi lớp và sử dụng hàm cross-entropy loss để đo lường sự khác biệt giữa dự đoán và nhãn thực tế. Cụ thể, với mỗi mẫu huấn luyện (x_i, y_i) trong tập dữ liệu, với x_i là đặc trưng của mẫu và y_i là nhãn thực tế, softmax loss được tính bằng công thức:

$$L_{\text{softmax}}(x_i, y_i) = -\log \left(\frac{\exp(W_{y_i}^T x_i + b_{y_i})}{\sum_{j=1}^N \exp(W_j^T x_i + b_j)} \right)$$

Trong đó:

- N là số lớp (số người trong bộ dữ liệu huấn luyện).
- W_j là trọng số của lớp thứ j trong tập dữ liệu.
- b_j là điều chỉnh độ lệch của lớp thứ j .
- W_{y_i} và b_{y_i} là trọng số và điều chỉnh độ lệch tương ứng của lớp được gán nhãn cho mẫu x_i .

Mục tiêu của softmax loss là tối thiểu hóa giá trị loss trung bình trên toàn bộ tập dữ liệu huấn luyện:

$$L_{\text{softmax}} = \frac{1}{N} \sum_{i=1}^N L_{\text{softmax}}(x_i, y_i)$$

ArcFace đề xuất một cải tiến đáng chú ý bằng cách sử dụng một phương pháp được gọi là "Angular margin loss" trong quá trình đào tạo mô hình nhận diện khuôn mặt sâu. Angular margin loss là một kỹ thuật tiên tiến trong lĩnh vực này, nhằm tối ưu hóa việc phân biệt giữa các lớp thông qua việc ánh xạ các đặc trưng vào không gian góc.

Ý tưởng chính của Angular margin loss là tạo ra các biểu diễn đặc trưng sao cho các lớp khác nhau được phân tách rõ ràng trong không gian góc, giúp mô hình dễ dàng phân loại các đối tượng một cách chính xác và hiệu quả. Thay

vì chỉ xem xét khoảng cách Euclidean giữa các đặc trưng và các trọng số lớp như trong softmax loss, Angular margin loss cố gắng tối ưu hóa các góc giữa chúng.

Trong ArcFace, mỗi lớp được biểu diễn bằng một góc phụ, được tính dựa trên trọng số thực tế của lớp và một góc margin m . Công thức tính toán cho góc giữa mỗi đặc trưng và trọng số lớp tương ứng là:

$$\cos(\theta_i) = \frac{W_i^T x_i}{\|W_i\| \|x_i\|}$$

Trong đó:

- W_i là vector trọng số của lớp thứ i .
- x_i là đặc trưng của mẫu i .
- θ_i là góc giữa W_i và x_i .
- $\|W_i\|$ và $\|x_i\|$ là độ dài của các vector tương ứng.

ArcFace thêm một margin góc m vào mô hình Angular margin loss, tạo ra một biến thể cải tiến được gọi là Additive Angular Margin Loss (AAML). Additive Angular Margin Loss (AAML) là một hàm mất mát chuyên biệt được thiết kế để tăng cường khả năng phân biệt giữa các lớp và độ phân loại trong các mô hình nhận diện khuôn mặt sâu. Nó đóng vai trò là một thành phần chính trong khung ArcFace, góp phần vào hiệu suất vượt trội so với các phương pháp truyền thống.

AAML thêm một góc phụ vào hàm softmax loss, nhằm mục đích tối ưu hóa không gian góc giữa biểu diễn đặc trưng và trọng số của các lớp. Bằng cách tích hợp margin này, AAML khuyến khích mô hình học các đặc trưng phân biệt hơn, hiệu quả phân tách các lớp trong không gian góc.

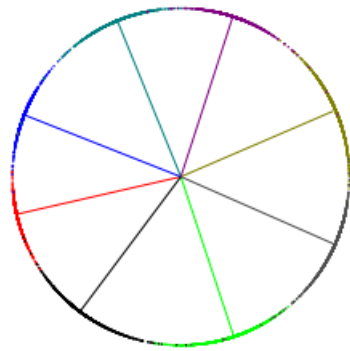
Trong toán học, AAML được biểu diễn như sau:

$$L_{\text{AAML}}(x_i, y_i) = -\log \left(\frac{\exp(s_{y_i} \cdot \cos(m\theta_{y_i}))}{\sum_{j=1}^N \exp(s_j \cdot \cos(m\theta_j))} \right)$$

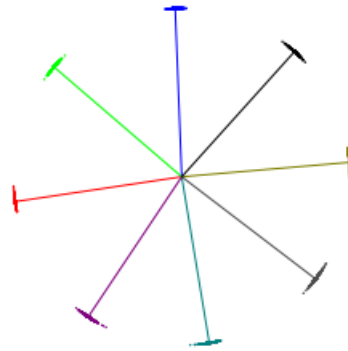
Trong đó:

- $L_{AAML}(x_i, y_i)$ là hàm mất mát AAML cho mẫu i với nhãn y_i .
- s_{y_i} là điểm số của lớp thực sự y_i .
- m là hằng số margin, được xác định trước.
- θ_{y_i} biểu thị góc giữa w_i và x_i cho lớp thực sự y_i .
- N là tổng số lớp.

Hàm mất mát AAML khuyến khích mô hình tối thiểu hóa khoảng cách góc giữa các đặc trưng và trọng số lớp tương ứng của chúng, hiệu quả tăng cường sức mạnh phân biệt của các biểu diễn đã học. Bằng cách tối ưu hóa hàm mất mát chuyên biệt này, ArcFace đạt được hiệu suất vượt trội trong các nhiệm vụ nhận diện khuôn mặt, đặc biệt là trong việc phân biệt giữa các khuôn mặt tương đồng và cải thiện tỷ lệ phân loại tổng thể.



(a) Norm-Softmax



(b) ArcFace

Hình 2.22: So sánh hàm Softmax thông thường và AAM Loss trong ArcFace [9]

Như hình trên đây cho thấy, hàm softmax (a) cung cấp khả năng nhúng đặc trưng gần như có thể tách rời nhưng tạo ra sự mơ hồ đáng kể trong ranh giới quyết định, trong khi đó hàm ArcFace (b) được đề xuất rõ ràng có thể tạo ra khoảng cách rõ ràng hơn giữa các lớp gần nhất.

Hàm mất mát chuyên biệt này đóng vai trò quan trọng trong hiệu suất của ArcFace, góp phần vào việc nâng cao hiệu suất của mô hình trong việc nhận diện khuôn mặt sâu.

2.4.3. Ưu và nhược điểm

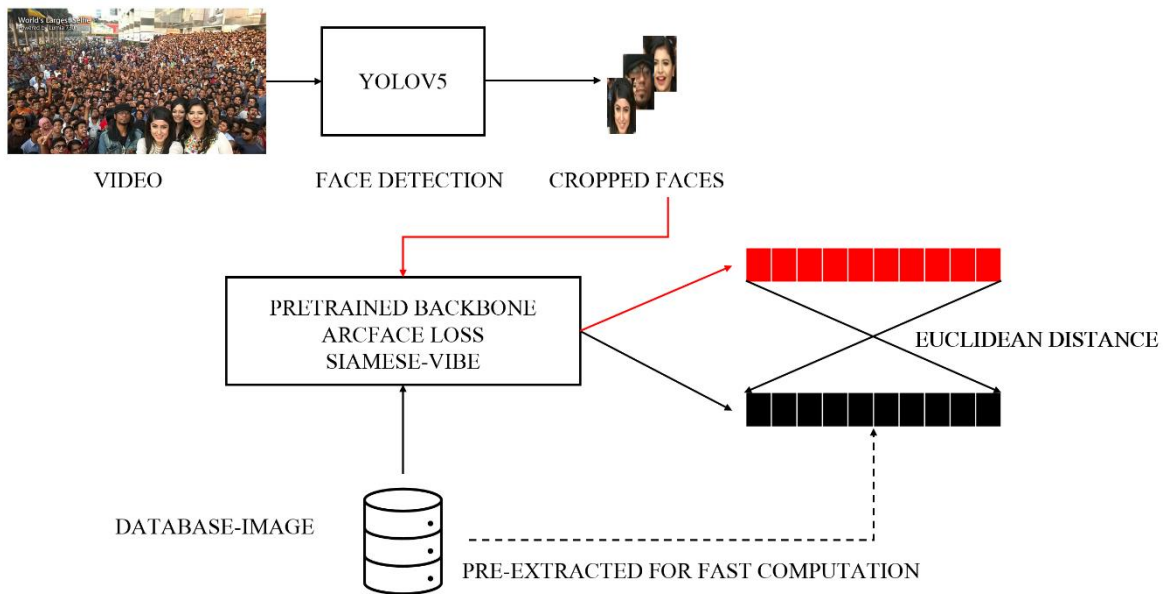
Về điểm mạnh, ArcFace có khả năng phân biệt tốt kể cả khi có các biến thể lớn trên diện mạo khuôn mặt. Hiện tại, theo những kết quả benchmark của nhóm tác giả, ArcFace vẫn giữ được những vị trí cao so với các phương pháp đương thời khác.

Về điểm yếu, nếu hình ảnh khuôn mặt trong dữ liệu huấn luyện là mẫu nhiễu thì nó không thuộc lớp dương tương ứng. Trong ArcFace, mẫu nhiễu này tạo ra giá trị mất mát lớn nhưng sai, làm ảnh hưởng đến quá trình huấn luyện mô hình. Vì thế, nhóm tác giả đã giới thiệu sub-center ArcFace. Phương pháp này khiến ràng buộc intra-class buộc mẫu huấn luyện phải gần với một trong nhiều trung tâm phụ dương nhưng không phải tất cả chúng. Điểm nhiễu có khả năng hình thành một lớp con không chiếm ưu thế và sẽ không được buộc vào lớp con chiếm ưu thế. Do đó, sub-center ArcFace có khả năng chống nhiễu tốt hơn.

Như vậy với những ưu và nhược điểm trên, em quyết định sử dụng mô hình ArcFace là phương pháp chính để giải quyết bài toán biểu diễn và nhận diện khuôn mặt.

Chương 3. MỘT SỐ KẾT QUẢ THỰC NGHIỆM

Trong chương trước, em đã giới thiệu các nghiên cứu liên quan đến các kỹ thuật giải quyết từng bài toán con cho bài toán nhận diện khuôn mặt theo thời gian thực, và đưa ra các phương pháp chính sử dụng trong đề tài này là sử dụng mô hình YOLOv5 để phát hiện khuôn mặt và ArcFace để nhận diện khuôn mặt. Thông qua các kết quả nghiên cứu trên, em sẽ trình bày quá trình thực nghiệm bao gồm thu tập và xử lý dữ liệu, huấn luyện mô hình YOLOv5 để phát hiện vùng khuôn mặt từ video và sử dụng thuật toán ArcFace để nhận diện khuôn mặt, từ đó ứng dụng cho việc tìm kiếm người trong đám đông.



Hình 3.1: Các giai đoạn trong mô hình xử lý nhận diện khuôn mặt theo thời gian thực

Như ảnh minh họa quá trình thực nghiệm ở trên, quy trình thực hiện sẽ sử dụng video trực tiếp lấy từ webcam, video được xử lý thông qua model YOLOv5 để phát hiện khuôn mặt và tách ảnh khuôn mặt. Ảnh khuôn mặt sau khi tách ra được đưa vào model ArcFace, thông qua pre-train model Resnet để trích xuất đặc trưng khuôn mặt, tiếp đó thì biểu diễn vector hóa đặc trưng khuôn mặt đó nhờ ArcFace rồi thực hiện so sánh sự tương đồng của chúng với đặc trưng khuôn mặt gốc có trong cơ sở dữ liệu bằng cosine similarity, từ đó đưa ra kết quả, với giá trị cosine similarity càng gần 1 thì xác suất trùng khớp với

khuôn mặt càng lớn. Để hiểu rõ hơn về quy trình thực nghiệm, tiếp đến em sẽ trình bày chi tiết hơn dưới đây.

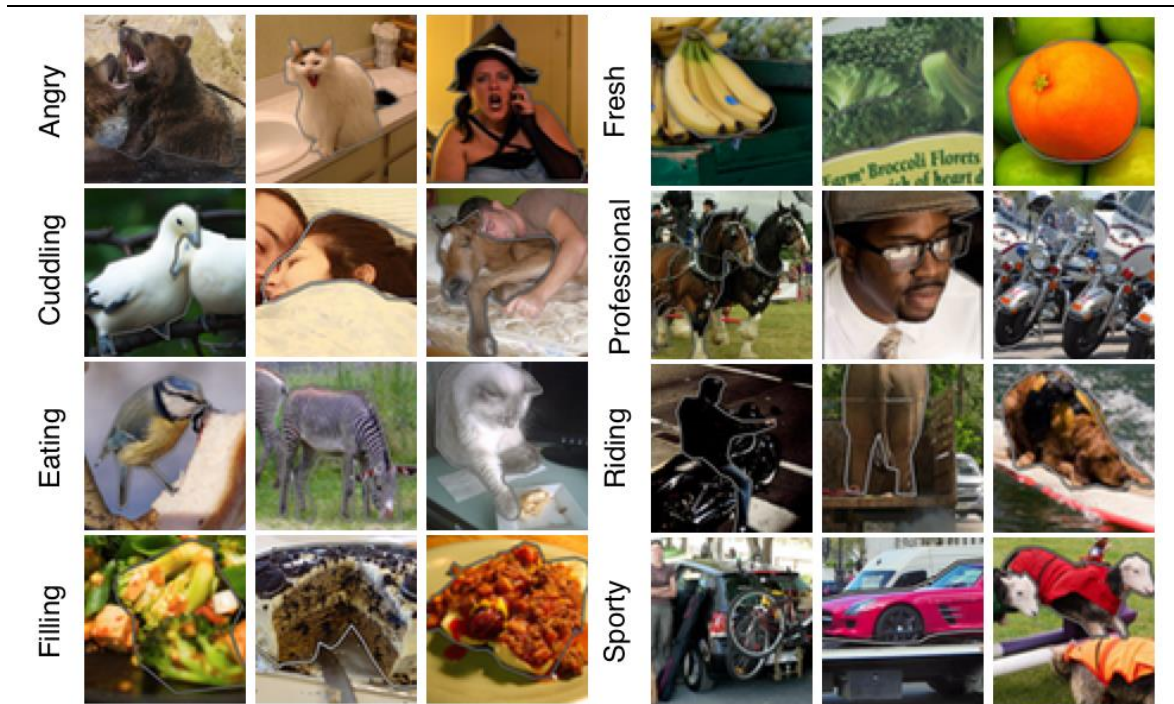
3.1. Thu thập và tiền xử lý dữ liệu

3.1.1. Thu thập dữ liệu

Bộ dữ liệu COCO, viết tắt của "Common Objects in Context," là một trong những bộ dữ liệu phổ biến nhất được sử dụng cho việc huấn luyện và kiểm định các mô hình nhận diện vật thể. Bộ dữ liệu này được tạo ra bởi Microsoft và được sử dụng để đánh giá hiệu suất của các thuật toán nhận diện vật thể trên một loạt các đối tượng phổ biến trong ngữ cảnh thực tế.

COCO bao gồm hơn 200.000 hình ảnh, mỗi hình ảnh được gắn nhãn cho đến 80 loại đối tượng khác nhau, bao gồm con người, động vật, vật thể và nền. Mỗi hình ảnh được chú thích bằng cách định vị và đặt tên cho các đối tượng trong đó, giúp cho các thuật toán có thể học được cách nhận diện và định vị các đối tượng trong ảnh. COCO cung cấp các số liệu đánh giá được tiêu chuẩn hóa như Độ chính xác trung bình (mAP) để phát hiện đối tượng và Thu hồi trung bình trung bình (mAR) cho các tác vụ phân đoạn, làm cho nó phù hợp để so sánh hiệu suất mô hình tập dữ liệu COCO được chia thành ba tập hợp con

- Train2017: Chứa 118K hình ảnh để đào tạo các mô hình phát hiện, phân đoạn và tạo phụ đề.
- Val2017: Có hình ảnh 5K được sử dụng cho mục đích xác thực trong quá trình đào tạo mô hình.
- Test2017: Bao gồm 20K hình ảnh được sử dụng để kiểm tra và đo điểm chuẩn cho các mô hình được đào tạo.



Hình 3.2: Một số lớp trong bộ dữ liệu COCO

Bộ dữ liệu COCO được sử dụng rộng rãi để đào tạo và đánh giá các mô hình deep learning trong phát hiện đối tượng (chẳng hạn như YOLO, Faster R-CNN và SSD), phân đoạn phiên bản (chẳng hạn như Mask R-CNN) và phát hiện điểm chính (chẳng hạn như OpenPose). Tập hợp các danh mục đối tượng đa dạng, số lượng lớn hình ảnh có chú thích và số liệu đánh giá được tiêu chuẩn hóa của tập dữ liệu khiến nó trở thành tài nguyên thiết yếu cho các nhà nghiên cứu và thực hành thị giác máy tính.

Để cải thiện khả năng phát hiện khuôn mặt cho mô hình, cần huấn luyện mô hình YOLO với bộ dữ liệu WIDER FACE, bộ dữ liệu WIDER bao gồm hơn 30k hình ảnh với hơn 390k khuôn mặt, mỗi khuôn mặt có hộp bounding và các định dạng nhãn khác nhau. Bộ dữ liệu WIDER FACE là bộ dữ liệu chuẩn nhận diện khuôn mặt, trong đó các hình ảnh được chọn từ bộ dữ liệu WIDER có sẵn công khai. Tác giả chọn 32.203 hình ảnh và gắn nhãn cho 393.703 khuôn mặt với mức độ thay đổi cao về tỷ lệ, tư thế và độ che phủ như được mô tả trong ảnh mẫu. Bộ dữ liệu WIDER FACE được tổ chức dựa trên 61 lớp sự kiện. Đối với mỗi lớp sự kiện, tác giả chọn ngẫu nhiên 40%/10%/50% dữ liệu làm tập

huấn luyện, xác thực và kiểm tra. Tác giả áp dụng thước đo đánh giá tương tự được sử dụng trong bộ dữ liệu PASCAL VOC.



Hình 3.3: Bộ dữ liệu WIDER FACE

3.1.2. Tiền xử lý dữ liệu

COCO Dataset: COCO là một trong những bộ dữ liệu lớn và phổ biến trong lĩnh vực thị giác máy tính, với hơn 200.000 hình ảnh và 80 loại đối tượng khác nhau. Dữ liệu này đã được tiền xử lý kỹ lưỡng bởi tác giả, bao gồm việc gán nhãn cho các đối tượng, bối cảnh, vị trí, và các thông tin khác. Do đó, không cần phải thực hiện bất kỳ tiền xử lý nào khác trước khi sử dụng.

WIDER FACE Dataset: Bộ dữ liệu WIDER FACE được sử dụng để huấn luyện mô hình phát hiện khuôn mặt. Dữ liệu này cũng đã được chuẩn bị và tiền xử lý trước đó bởi tác giả để phù hợp với mô hình và phương pháp huấn luyện sử dụng. Thông thường, dữ liệu đã được chuẩn bị để chứa các hình ảnh khuôn mặt, với các nhãn tương ứng cho mỗi khuôn mặt. Vì vậy không cần phải tiền xử lý dữ liệu trên trước khi sử dụng.

3.2. Huấn luyện mô hình và đánh giá độ chính xác

3.2.1. YOLOv5

Bước đầu trong việc xây dựng chương trình là huấn luyện một mô hình có thể phát hiện vùng được khuôn mặt từ camera. Em lựa chọn sử dụng lại và huấn luyện lại mô hình YOLOv5 đã được pre-trained trước đó. Mô hình YOLOv5 được sử dụng này là phiên bản mô hình đã được tiền huấn luyện với quy mô lớn gồm 300 epoch thông qua bộ dữ liệu COCO với 7.6 triệu tham số,

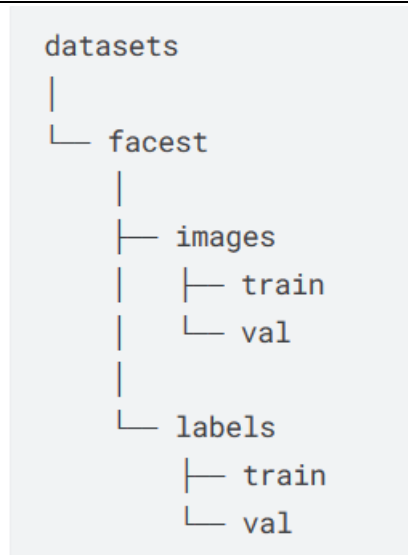
đặc biệt dành cho việc xử lý phát hiện vật thể. Do tính ưu việt về bộ xử lý tính toán của kaggle so với máy tính cá nhân, em lựa chọn công cụ kaggle cho việc huấn luyện. Cụ thể phần cứng được sử dụng trên kaggle được thể hiện trong bảng sau:

Bảng 3.1: Cấu hình phần cứng trên Kaggle

Phần cứng xử lý	Số Core	Dung lượng	Giờ/Tuần
NVIDIA Tesla P1000 GPU	2584 CUDA core	16 GB	30 giờ
NVIDIA T4 x 2 GPU	2560 CUDA core	16 GB	30 giờ
Google TPU v3-8	8 TPU v3 core	128 GB	20 giờ
Intel Xeon 2.20 Ghz CPU	4 vCPU core	32 GB	

Quá trình huấn luyện thực hiện trên Kaggle với bộ dữ liệu WIDER FACE, các bước thực hiện huấn luyện được mô tả sau đây:

Tập dữ liệu được sử dụng để đào tạo và đánh giá mô hình đã được chỉ định trong tệp cấu hình YAML, được cung cấp dưới dạng đối số cho tập lệnh. Tệp này chứa các đường dẫn đến tập hợp hình ảnh huấn luyện và xác thực, cũng như số lượng lớp đối tượng có trong tập dữ liệu. Lớp LoadImagesAndLabels từ module utils.datasets chịu trách nhiệm tải hình ảnh và nhãn sự thật cơ bản tương ứng của chúng từ các đường dẫn đã chỉ định.

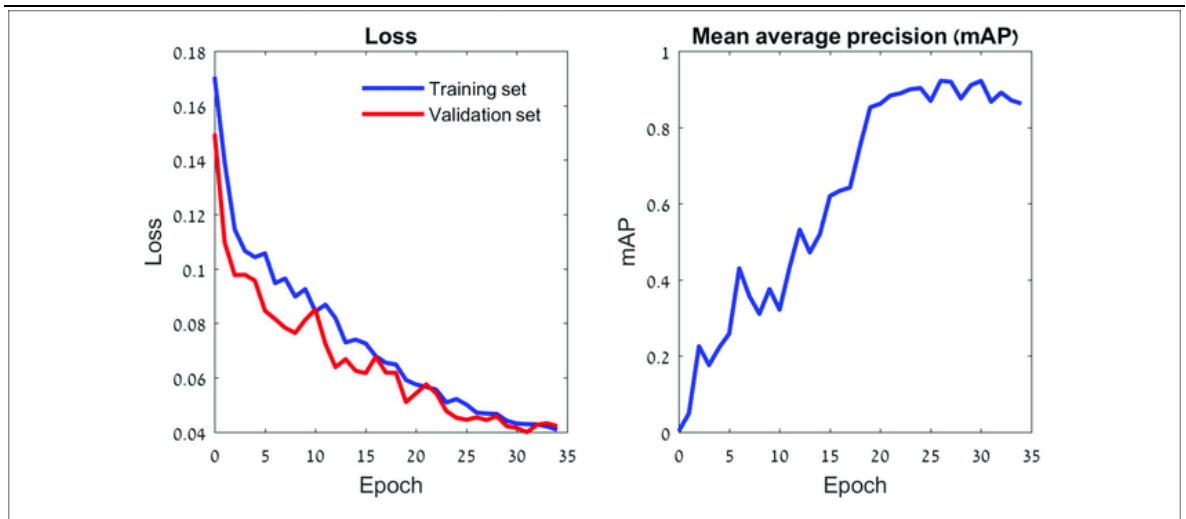


Hình 3.4: Cấu trúc bộ dữ liệu

Quá trình đào tạo được thực hiện qua nhiều epoch, mỗi epoch bao gồm nhiều lần lặp, trong mỗi lần lặp thực hiện các việc sau:

- Clone mã nguồn YOLOv5 từ GitHub của nhà phát triển ultralytics bao gồm tệp model yolov5s.pt đã được training trên tập dữ liệu COCO.
- Cấu hình optimizer và scheduler để tối ưu hóa mô hình.
- Tạo tệp cấu hình dataset YAML: Tạo một tệp YAML để cấu hình dataset cho việc huấn luyện. Tệp này chứa đường dẫn tới dữ liệu huấn luyện và kiểm tra, số lượng lớp (nc), và tên của các lớp.
- Load dữ liệu và huấn luyện mô hình bằng transfer learning trên model yolov5s.pt với tập dữ liệu WIDER FACE trong 12 epoch.
- Ngoài ra, các kỹ thuật tăng cường dữ liệu khác nhau, chẳng hạn như thay đổi kích thước hình ảnh, xoay và chuyển đổi không gian màu, đã được áp dụng cho dữ liệu huấn luyện để cải thiện tính khái quát của mô hình.

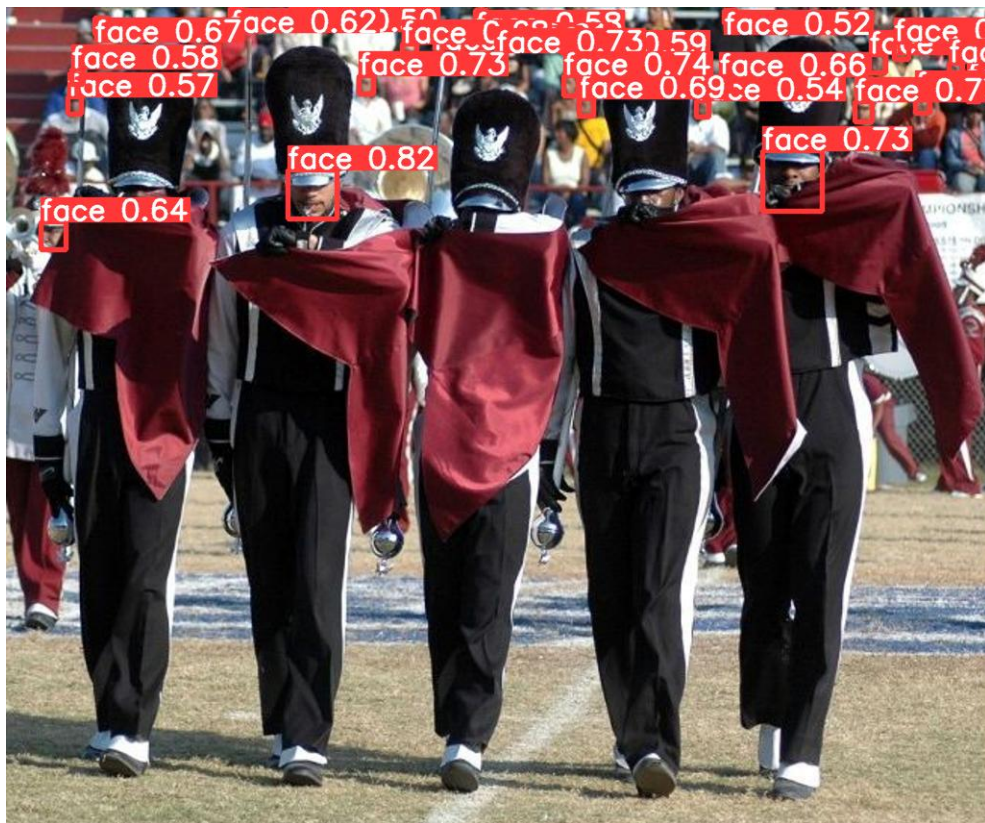
Quá trình huấn luyện được tiếp tục cho tới khi đạt số epoch chỉ định. Sau mỗi giai đoạn đào tạo, hiệu suất của mô hình được đánh giá trên tập validation và các số liệu như độ chính xác, mức thu hồi, Độ chính xác trung bình (mAP) loss score được tính toán. Các số liệu này đã được ghi lại và sử dụng để theo dõi tiến trình của mô hình trong quá trình đào tạo.



Hình 3.5: Kết quả đánh giá mô hình sau khi training

Như kết quả đánh giá trên có thể thấy mô hình có Loss score giảm dần về 0 và mAP tăng dần về 1 và đạt cao nhất 92,6%, như vậy ta có thể đánh giá mô hình đã đạt được độ chính xác đủ tốt cho việc phát hiện khuôn mặt và ứng dụng cho các bước sau.

Kết quả trên ảnh Test với mô hình YOLOv5 dùng phát hiện khuôn mặt:



Hình 3.6: Ảnh kết quả test phát hiện khuôn mặt với mô hình YOLOv5

3.2.2. ArcFace

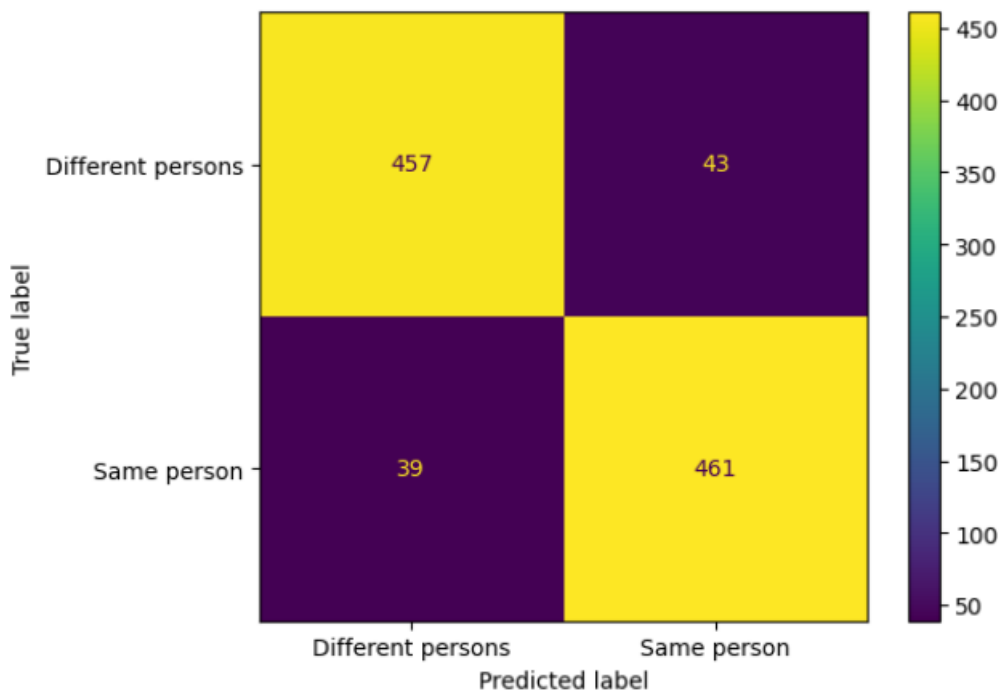
Để có thể trích xuất đặc trưng khuôn mặt và vector hóa các đặc trưng khuôn mặt dùng so sánh và nhận diện, ArcFace sử dụng một phiên bản của ResNet để làm backbone (xương sống) cho việc trích xuất đặc trưng khuôn mặt. Pre-train model được sử dụng ở đây sẽ là model resnet50 được lưu dưới dạng file backbone.pth để sử dụng trích xuất đặc trưng khuôn mặt và đưa vào mạng ArcFace để nhận diện khuôn mặt:

Sử dụng ResNet để trích xuất đặc trưng: Trước tiên, model ResNet được sử dụng để trích xuất các đặc trưng từ ảnh khuôn mặt. ResNet là một mạng nơ-ron sâu với khả năng học được các biểu diễn phức tạp của hình ảnh. Khi ảnh khuôn mặt được đưa qua mạng ResNet, mạng này sẽ tạo ra một vector đặc trưng (embedding) cho mỗi ảnh.

Chuẩn hóa đặc trưng: Trước khi đưa các vector đặc trưng vào quá trình so sánh, chúng thường sẽ được chuẩn hóa để đảm bảo rằng chúng có cùng tổng quát và không bị biến dạng do sự khác biệt về biên độ. mỗi vector đặc trưng được chuẩn hóa bằng cách chia cho norm của nó (độ dài của vector). Sau đó, tích vô hướng (dot product) giữa các vector này được tính để đo lường sự tương đồng giữa chúng trong không gian đặc trưng. ArcFace thêm một biên góc (angular margin) vào quá trình normalization này. Biên góc này giúp tạo ra một khoảng cách lớn hơn giữa các vector đặc trưng của các khuôn mặt khác nhau. Điều này thúc đẩy việc phân biệt giữa các lớp (các cá nhân) trong không gian đặc trưng.

So sánh đặc trưng: Sau khi chuẩn hóa, các vector đặc trưng của các khuôn mặt được so sánh với nhau để xác định mức độ tương đồng giữa chúng. Cách được sử dụng là sử dụng độ đo cosine similarity để tính toán khoảng cách giữa các vector đặc trưng. Ngưỡng được em sử dụng ở đây là 0,5, Nếu khoảng cách cosine lớn hơn ngưỡng, thì tên của tệp ảnh đó được gán cho nhãn của khuôn mặt được nhận diện.

Model ArcFace mà em sử dụng được test và có độ chính xác khá cao với đa phần giá trị dự đoán khuôn mặt đều khá giống với giá trị thực tế với 457/500 trường hợp nhãn “Different persons” và 461/500 trường hợp nhãn “Same person” được dự đoán đúng. Kết quả nhận diện cho độ chính xác và hiệu quả cao như ảnh dưới đây.



Hình 3.7: So sánh giá trị dự đoán với thực tế

3.3. Kết hợp các model sau khi được huấn luyện vào hệ thống

3.3.1. Phát hiện khuôn mặt từ camera và trích xuất đặc trưng

Sau khi có model phát hiện khuôn mặt YOLOv5, em sẽ thực hiện việc xây dựng hàm tính toán xử lý việc tách ảnh khuôn mặt từ vùng khuôn mặt phát hiện được để nhận diện. Cách thức xử lý như sau:

Phát hiện khuôn mặt và cắt ảnh khuôn mặt: Trước tiên, sử dụng mô hình YOLOv5 đã thu được lúc trước dự đoán các hộp giới hạn cho các khuôn mặt trong từng khung hình thu được qua camera. Sau khi có các hộp giới hạn, sử dụng tọa độ của hộp giới hạn để cắt ra các khuôn mặt từ khung hình gốc. Điều này được thực hiện bằng cách sử dụng các tọa độ xác định của hộp giới hạn để cắt ra một phần của hình ảnh ban đầu, chứa khuôn mặt của người.



Hình 3.8: Ảnh vùng khuôn mặt được cắt ra với bounding box bao quanh trên một khung hình

Tiền xử lý ảnh khuôn mặt được cắt ra: Khi đã có các khuôn mặt được cắt ra từ frame, chúng được chuẩn bị để phù hợp với mô hình nhận diện khuôn mặt (trong trường hợp này là ArcFace). Các bước chuẩn bị này bao gồm:

- Chuyển đổi kích thước: Các khuôn mặt được điều chỉnh kích thước sao cho phù hợp với kích thước mà mô hình nhận diện khuôn mặt yêu cầu (trong trường hợp này là 128x128 pixels).
- Chuyển đổi định dạng: Dữ liệu hình ảnh được chuyển đổi sang định dạng và kiểu dữ liệu phù hợp để đưa vào mô hình.
- Tạo một phiên bản lật theo chiều ngang của ảnh để làm tăng tính đa dạng của dữ liệu.
- Chuẩn hóa giá trị pixel của ảnh để giá trị nằm trong khoảng $[-1, 1]$.

Khuôn mặt được phát hiện từ các bước trước đó sau khi được chuẩn bị sẽ được đưa vào mô hình ArcFace để vector hóa:

Đưa dữ liệu vào mô hình: Ảnh khuôn mặt sau khi tiền xử lý được chuyển đổi thành tensor và đưa vào mô hình ArcFace. Mô hình sẽ nhận ảnh đầu vào và tiến hành phân tích để trích xuất các đặc trưng nổi bật.

Lấy các vector đặc trưng: Mô hình ArcFace tạo ra các vector đặc trưng từ ảnh khuôn mặt đầu vào. Mỗi vector đại diện cho một tập hợp các đặc điểm nổi bật của khuôn mặt từ các góc nhìn khác nhau.

Ghép các vector đặc trưng: Các vector đặc trưng được ghép lại thành một vector duy nhất bằng cách nối chúng lại với nhau (concatenate). Vector kết quả này có kích thước lớn hơn và chứa đầy đủ thông tin đặc trưng của khuôn mặt.

Vector đặc trưng sau khi được trích xuất sẽ được sử dụng để so sánh và nhận diện khuôn mặt ở bước xử lý sau.

3.3.2. So sánh đặc trưng khuôn mặt và nhận diện.

Để chương trình có thể nhận diện được khuôn mặt, chúng ta cần có cơ sở dữ liệu gồm các ảnh khuôn mặt gốc để trích xuất đặc trưng và đưa vào so sánh nhận diện với khuôn mặt thu được từ camera. Quá trình này được thực hiện như sau:

Duyệt qua các hình ảnh: Tải hình ảnh khuôn mặt của những người muốn nhận diện vào một thư mục và duyệt qua các hình ảnh trong thư mục được chỉ định để trích xuất đặc trưng.

Tiền xử lý hình ảnh cho việc xử lý: Mỗi hình ảnh được đọc bằng OpenCV và được chuẩn bị để phù hợp với mô hình. Hình ảnh được chuyển đổi sang kích thước phù hợp và các phép biến đổi khác có thể được thực hiện.

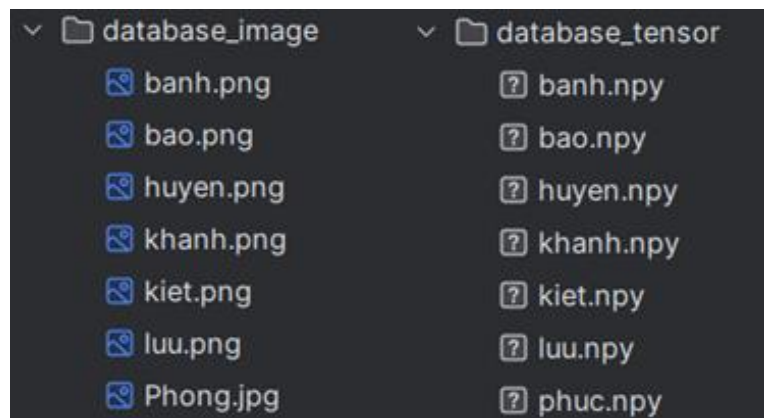
Chuyển đổi hình ảnh thành tensor và đưa vào mô hình: Hình ảnh được chuyển đổi thành tensor để có thể đưa vào mô hình PyTorch. Tensor này sau đó được chuyển đến thiết bị tính toán (CPU hoặc GPU) được chỉ định cho việc tính toán.

Trích xuất đặc trưng thông qua mô hình ArcFace: Hình ảnh được đưa qua mô hình ArcFace để trích xuất đặc trưng khuôn mặt. Quá trình này thực hiện sau khi tiền xử lý hình ảnh và sau đó đưa vào mô hình để tính toán đặc trưng.

Chuyển đổi đặc trưng và lưu trữ: Kết quả đặc trưng được trả về dưới dạng mảng numpy và lưu vào một từ điển. Key của từ điển là tên của hình ảnh (hoặc một định danh duy nhất), và giá trị là đặc trưng khuôn mặt tương ứng với hình ảnh đó.

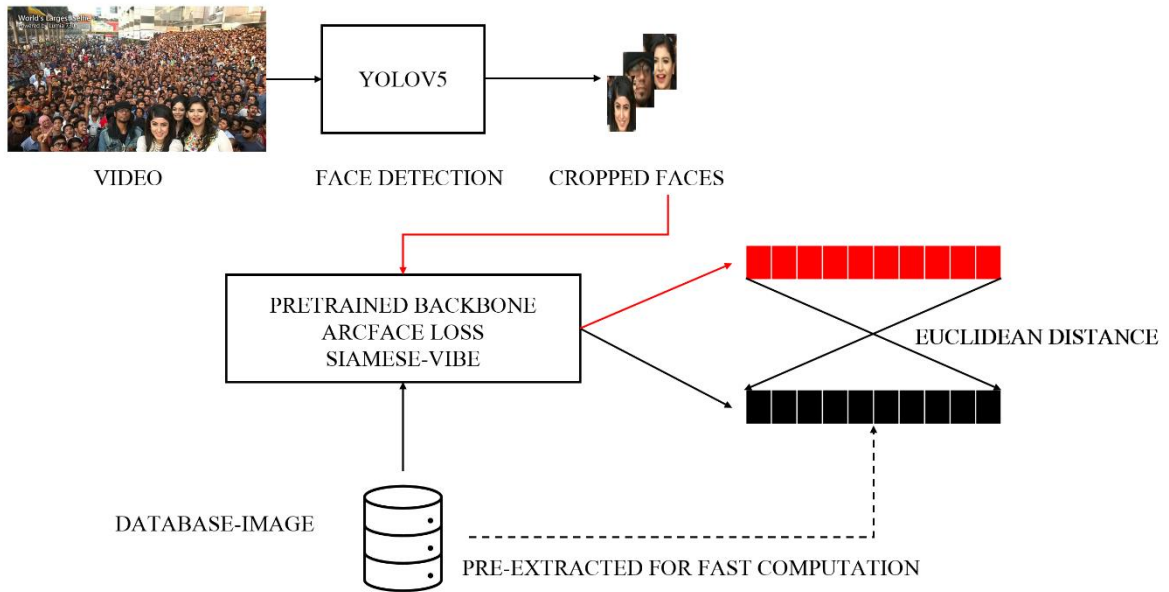
Kết quả cuối cùng của quá trình này là một từ điển chứa các đặc trưng khuôn mặt được trích xuất từ các hình ảnh trong thư mục. Điều này cho phép sử dụng các đặc trưng này cho các nhiệm vụ như nhận diện hoặc so sánh khuôn mặt sau này.

Các đặc trưng của hình ảnh được lưu dưới dạng file .npy trong thư mục tensor và được sử dụng để nhận diện sau này.



Hình 3.9: Thư mục chứa ảnh gốc và vector đặc trưng sau khi trích xuất

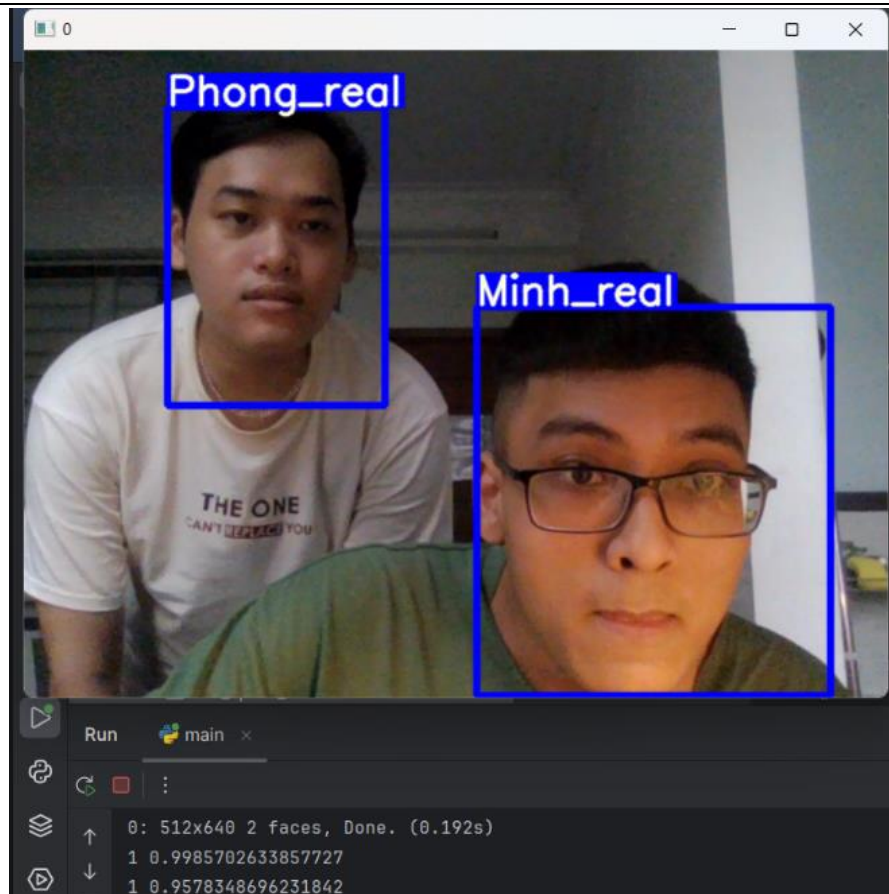
Cuối cùng, sau khi đã có mô hình phát hiện khuôn mặt YOLOv5 và mô hình nhận diện khuôn mặt ArcFace, cùng với bộ dữ liệu đặc trưng khuôn mặt của ảnh gốc, em sẽ kết hợp và tạo nên chương trình nhận diện khuôn mặt theo thời gian thực với video từ camera hoặc ảnh đầu vào để nhận diện khuôn mặt trong đám đông. Cách thức hoạt động của hệ thống khi kết hợp các model như sau:



Hình 3.10: Luồng hoạt động của hệ thống

- Với đầu vào là camera theo thời gian thực, hệ thống duyệt qua từng khung hình và phát hiện khuôn mặt với YOLOv5 model.
- Áp dụng Non-Maximum Suppression (NMS) để loại bỏ các bounding box trùng lặp và giữ lại các bounding box có confidence cao.
- Duyệt qua từng bounding box đã nhận diện được.
- Tiến hành trích xuất khuôn mặt từ bounding box và thực hiện các bước tiền xử lý để chuẩn bị dữ liệu cho model ArcFace.
- Sử dụng model ArcFace để trích xuất đặc trưng của khuôn mặt từ mỗi bounding box.
- So sánh đặc trưng của khuôn mặt được trích xuất với các đặc trưng đã biết từ cơ sở dữ liệu (sử dụng cosine similarity) để xác định danh tính của người đó.
- Đưa ra video theo thời gian thực bao gồm các hình ảnh khuôn mặt được đánh nhãn trực tiếp trên video.

Kết quả của chương trình nhận diện khuôn mặt theo thời gian thực được trình bày trong ảnh dưới đây:



Hình 3.11: Kết quả nhận diện khuôn mặt

Từ kết quả test cho thấy độ chính xác trung bình của mô hình YOLOv5 trong phát hiện khuôn mặt đạt 0.85%, giá trị cosine similarity đạt trung bình 0.55 khi so sánh đặc trưng khuôn mặt. Đây là các kết quả rất tốt trong thực tế và cho thấy chương trình đã đạt được những kì vọng trước đó.

Chương 4. XÂY DỰNG HỆ THỐNG WEBSITE NHẬN DIỆN KHUÔN MẶT THEO THỜI GIAN THỰC

Trong chương này, em sẽ trình bày về hệ thống website tích hợp cho chương trình nhận diện khuôn mặt theo thời gian thực ứng dụng tìm kiếm người trong đám đông.

4.1. Giới thiệu framework Streamlit

Streamlit là một framework mã nguồn mở được sử dụng để xây dựng ứng dụng web nhanh chóng và trực quan. Nó được thiết kế để tạo ra các ứng dụng trực tuyến tương tác với dữ liệu và mô hình máy học.

Streamlit có rất nhiều điểm mạnh như:

- Ngôn ngữ lập trình Python: Streamlit sử dụng Python làm ngôn ngữ chính để xây dựng ứng dụng. Điều này giúp cho việc học và sử dụng Streamlit trở nên dễ dàng đối với những người đã quen thuộc với Python.
- Tương tác nhanh chóng với dữ liệu: Streamlit cho phép người dùng tương tác trực tiếp với dữ liệu một cách nhanh chóng. Người dùng có thể thay đổi các tham số và xem kết quả ngay lập tức trong ứng dụng, giúp cho quá trình phân tích dữ liệu và kiểm tra mô hình trở nên dễ dàng.
- Hiển thị dữ liệu trực quan: Streamlit cung cấp các API đơn giản để hiển thị dữ liệu một cách trực quan. Người dùng có thể tạo các biểu đồ, bảng, hình ảnh và video để trình bày dữ liệu một cách rõ ràng và dễ hiểu.
- Tích hợp dễ dàng: Streamlit có thể dễ dàng tích hợp với các thư viện và công cụ phổ biến khác trong cộng đồng khoa học dữ liệu và máy học như Pandas, NumPy và Scikit-learn. Điều này giúp cho việc phân tích dữ liệu và xây dựng mô hình trở nên thuận tiện và linh hoạt.
- Chia sẻ ứng dụng dễ dàng: Streamlit cung cấp tính năng chia sẻ dễ dàng, cho phép người dùng chia sẻ ứng dụng của mình trực tuyến với người

khác thông qua các liên kết chia sẻ hoặc tích hợp trực tiếp vào các nền tảng như Heroku hoặc AWS.

- Tính linh hoạt và mở rộng: Streamlit cho phép người dùng tùy chỉnh và mở rộng ứng dụng của mình theo ý muốn. Thông qua các API và thành phần giao diện người dùng linh hoạt, người dùng có thể tạo ra các ứng dụng độc đáo và phù hợp với nhu cầu của mình.

Với những đặc tính này, Streamlit là một lựa chọn phù hợp để em xây dựng một trang web trực tuyến, trực quan và dễ tương tác cho đề tài của mình.

4.2. Các bước xây dựng hệ thống tích hợp

4.2.1. Phân tích yêu cầu

Website được tạo ra với mục đích tích hợp chương trình nhận diện khuôn mặt theo thời gian thực vào một giao diện người dùng đơn giản nhằm giúp cho người dùng dễ dàng thao tác sử dụng. Website cho phép người dùng tải ảnh người cần nhận diện lên và tùy chỉnh các tham số mô hình cho phù hợp với mong muốn. Tiếp đó, website cho phép người dùng sử dụng nguồn video từ webcam để nhận diện đối tượng cần tìm kiếm từ camera theo thời gian thực, hệ thống có ứng dụng thực tế rất lớn trong việc hỗ trợ tìm kiếm trong trung tâm thương mại từ camera giám sát hay truy tìm tội phạm lẫn trốn trong đám đông. Thông qua hình ảnh phân tích từ camera, hệ thống có thể giúp người dùng phát hiện và đánh dấu đối tượng trực tiếp trên màn hình hiển thị, giúp cho người dùng dễ dàng xác định vị trí của đối tượng tìm kiếm dù là trong đám đông nhiều người.

4.2.2. Thiết kế giao diện người dùng và logic ứng dụng

Giao diện người dùng phải đáp ứng được một số yêu cầu sau:

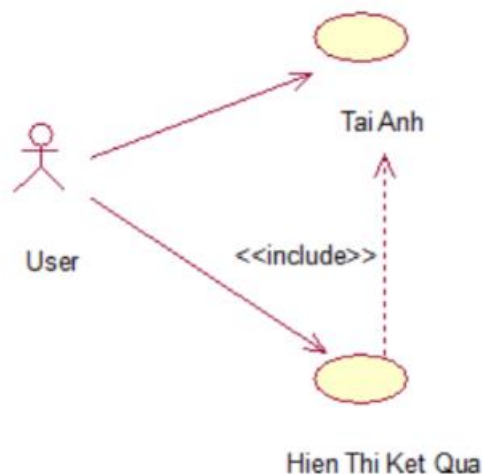
- Tính Trực Quan và Dễ Sử Dụng: Giao diện cần dễ hiểu, trực quan, và dễ điều hướng.
- Tính Nhất Quán: Màu sắc, font chữ, và layout nên nhất quán trên toàn bộ ứng dụng.

- Phản Hồi Nhanh: Hệ thống nên cung cấp phản hồi nhanh chóng và rõ ràng đối với mọi hành động của người dùng.
- Tối Ưu Hóa cho Các Thiết Bị: Giao diện nên phù hợp với nhiều loại màn hình và thiết bị.
- Thân Thiện với Người Dùng: Cung cấp các hướng dẫn sử dụng, thông báo lỗi rõ ràng và trợ giúp khi cần thiết.

Giao diện người dùng được liên kết logic ứng dụng thông qua streamlit đảm bảo logic truyền tải dữ liệu ảnh từ giao diện người dùng vào mô hình và xử lý kết quả trả về.

4.3. Phân tích thiết kế hệ thống tích hợp

4.3.1. Vẽ biểu đồ use case



Hình 4.1: Biểu đồ use case

4.3.2. Mô tả chi tiết use case

Bảng 4.1: Use case tải ảnh và tùy chỉnh tham số

Use case Tải ảnh và tùy chỉnh các tham số:

- 1: Tên use case: Tải ảnh người cần nhận diện và tùy chỉnh các tham số.
- 2: Mô tả vắn tắt: Use case này cho phép người dùng tải ảnh người cần nhận diện và tùy chỉnh các tham số để hệ thống thực hiện nhận diện qua camera.

3: Các luồng sự kiện:

3.1: Luồng cơ bản

- 1) Người dùng tải hình ảnh khuôn mặt người cần nhận diện lên hệ thống.
- 2) Người dùng tùy chỉnh các tham số hệ thống theo mong muốn.
- 3) Hệ thống lưu trữ ảnh và các tham số, tiến hành xử lý ảnh.

Use case kết thúc

3.2: Luồng rẽ nhánh: Không có

- 4: Các yêu cầu đặc biệt: Hệ thống cần truy cập vào bộ nhớ thiết bị cũng như phần cứng để xử lý.
- 5: Tiền điều kiện: Người dùng đã chọn chức năng tải ảnh người cần nhận diện.
- 6: Hậu điều kiện: Ảnh đã được tải lên và có sẵn để sử dụng.
- 7: Điểm mở rộng: Hệ thống có thể hỗ trợ nhiều định dạng tệp khác nhau cho việc tải ảnh.

Bảng 4.2: Use case hiển thị Video kết quả nhận diện

Use Case: Hiển Thị Video kết quả nhận diện

- 1: Tên use case: Hiển Thị Video kết quả nhận diện
- 2: Mô tả vắn tắt: Use case này cho phép người dùng xem video hiển thị kết quả nhận diện khuôn mặt theo thời gian thực.
- 3: Các luồng sự kiện:
 - 3.1. Luồng cơ bản:
 - 1) Người dùng nhấn nút ‘Start Detection’ trên màn hình giao diện.
 - 2) Hệ thống thực hiện xử lý nhận diện khuôn mặt trên video và hiển thị kết quả theo thời gian thực.

4) Người dùng có thể dừng lại hoặc tiếp tục xem video.

Use case kết thúc.

3.2. Luồng rẽ nhánh: Không có

4: Các yêu cầu đặc biệt: Hệ thống cần truy cập vào camera của thiết bị và có khả năng xử lý video trực tiếp.

5: Tiền điều kiện: Người dùng đã mở chức năng xem video từ camera.

6: Hậu điều kiện: Người dùng có thể xem video và nhận diện khuôn mặt trên video.

7: Điểm mở rộng: Hệ thống có thể cung cấp các tính năng bổ sung như ghi lại video hoặc thêm hiệu ứng đồ họa vào video.

4.4. Kiểm thử website nhận diện khuôn mặt theo thời gian thực

4.4.1. Phân tích thiết kế kiểm thử

❖ **Test case:** Kiểm tra nhận diện khuôn mặt từ webcam trên website:

1) **Tóm tắt test case:** Xác minh hệ thống có thể nhận diện khuôn mặt từ webcam và hiển thị kết quả trên giao diện Streamlit.

2) **Xem xét cơ sở test:**

- Yêu cầu: Hệ thống phải nhận diện khuôn mặt từ webcam theo thời gian thực và hiển thị kết quả trên giao diện Streamlit.
- Đặc điểm thiết kế: Sử dụng mô hình YOLOv5 để phát hiện khuôn mặt và mô hình ArcFace để nhận diện.
- Phân tích rủi ro: Khả năng nhận diện không chính xác, hiệu suất không đáp ứng thời gian thực, vấn đề kết nối với webcam.
- Kiến trúc: Hệ thống được triển khai qua Streamlit, sử dụng các mô hình học sâu đã được huấn luyện.
- Giao diện: Giao diện Streamlit với các tùy chọn thiết lập và hiển thị video đầu ra.

3) **Xác định điều kiện test:**

-
- Webcam phải hoạt động tốt và được kết nối.
 - Các file trọng số (weights) cần thiết đã được tải lên và đúng vị trí.
 - Cấu hình hệ thống phù hợp (CUDA hoặc CPU).

4.4.2. Thực hiện kiểm thử

❖ Điều kiện tiên quyết:

- Cấu hình hệ thống đã được thiết lập đầy đủ.
- Trình duyệt có quyền truy cập vào webcam.
- Các file trọng số (weights) cần thiết đã có sẵn trong dự án.

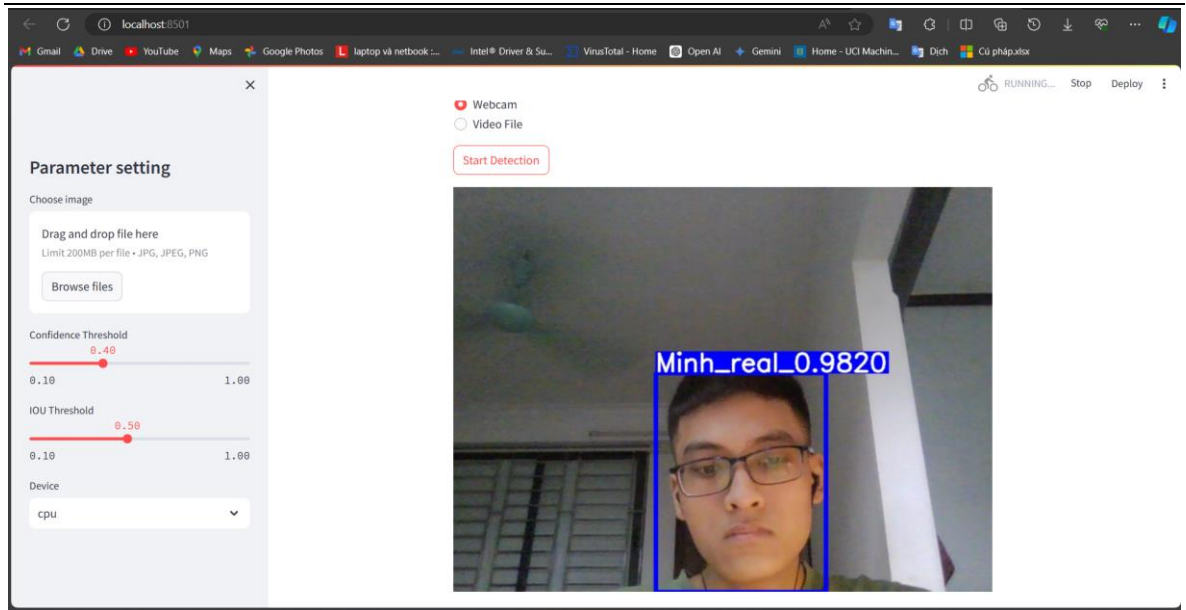
❖ Các bước thực hiện:

- Bước 1: Khởi chạy ứng dụng Streamlit
- Bước 2: Chọn 'Webcam' trong mục Select Source
- Bước 3: Điều chỉnh Confidence Threshold tới 0.5 và ngưỡng IOU tới 0.4
- Bước 4: Chọn CPU làm thiết bị phần cứng
- Nhấn nút 'Start Detection'

❖ Kết quả mong đợi:

- Hệ thống khởi động webcam và bắt đầu nhận diện khuôn mặt theo thời gian thực.
- Kết quả nhận diện được hiển thị trong cửa sổ video trên giao diện Streamlit.
- Mỗi khuôn mặt được nhận diện sẽ hiển thị một nhãn với tên hoặc thông tin đã nhận dạng.

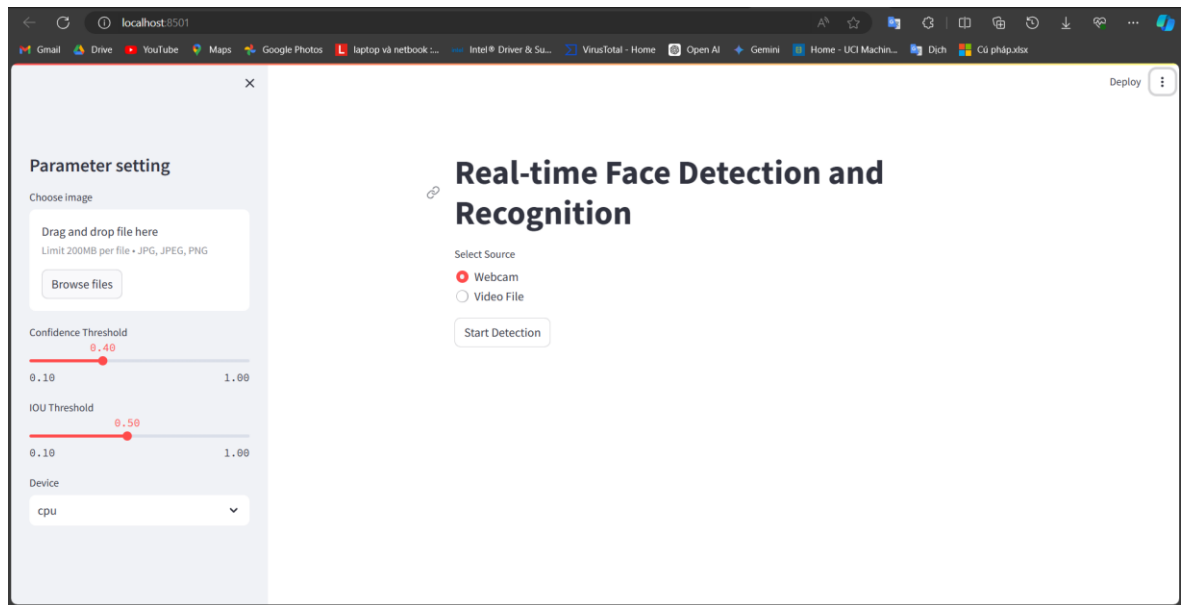
❖ Kết quả thực tế: Các kết quả mong đợi đều được đáp ứng và website hoạt động bình thường, đưa ra kết quả nhận diện chính xác.



Hình 4.2: Kết quả kiểm thử thực tế với website

4.5. Các kết quả đạt được và nhận xét

Chương trình website tích hợp hệ thống sẽ được em trình bày dưới đây:

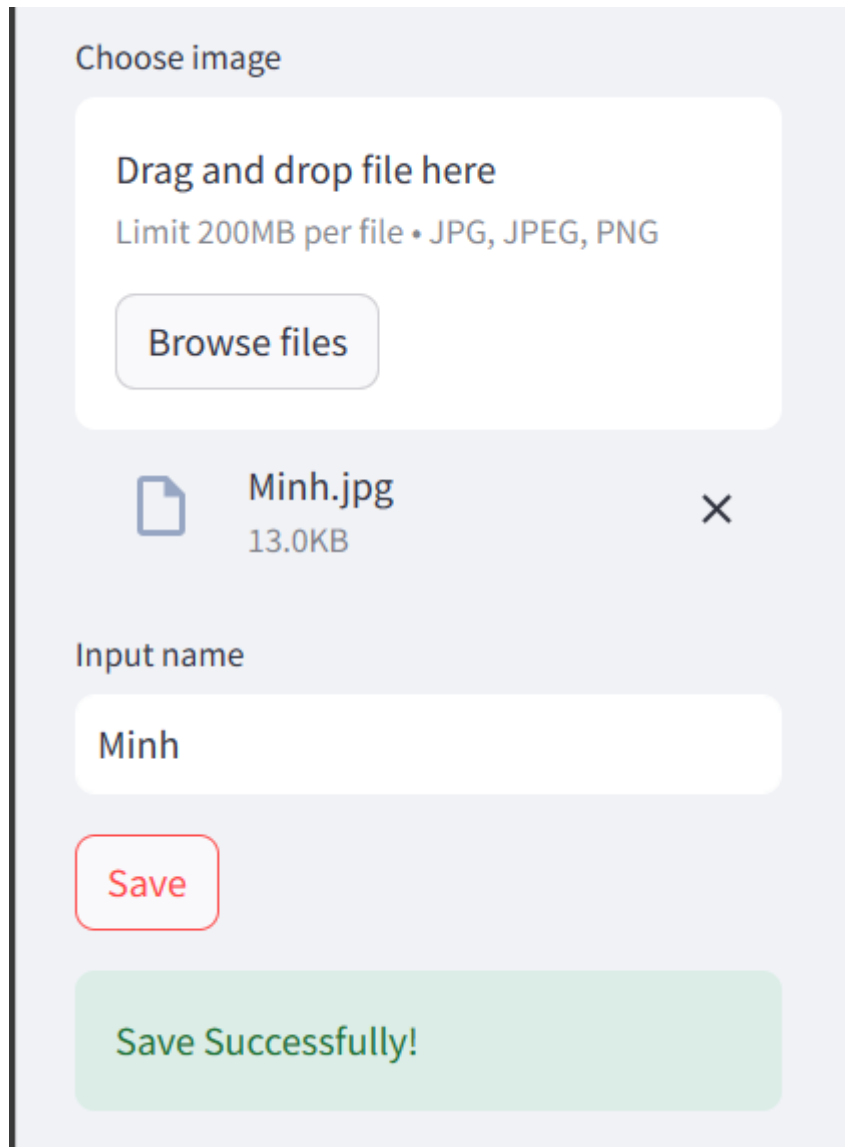


Hình 4.3: Giao diện màn hình trang chủ của website

Trên đây là giao diện màn hình trang chủ của website, màn hình giao diện với thanh bên trái là nơi người dùng lựa chọn cấu hình cho hệ thống và bên phải là nơi hiển thị các lựa chọn người dùng cũng như hiển thị video nhận diện.

Để sử dụng các chức năng của website, người dùng cần tùy chỉnh các tham số ở sidebar bên trái và upload ảnh người cần nhận diện. Các tham số bên trái người dùng cần lựa chọn bao gồm:

- Hình ảnh gốc khuôn mặt cần nhận diện: Đây là nơi upload ảnh khuôn mặt của người cần nhận diện, người dùng chọn nút Browse files, một cửa sổ File explore hiện lên và người dùng lựa chọn và lựa chọn một bức ảnh khuôn mặt cho một người. Tiếp đến người dùng nhập tên cho người cần nhận diện và nhấn nút Save. Một thông báo hiện ra thông báo cho người dùng biết bức ảnh đã upload thành công như ảnh dưới đây.



Choose image

Drag and drop file here

Limit 200MB per file • JPG, JPEG, PNG

Browse files

Minh.jpg
13.0KB

Input name

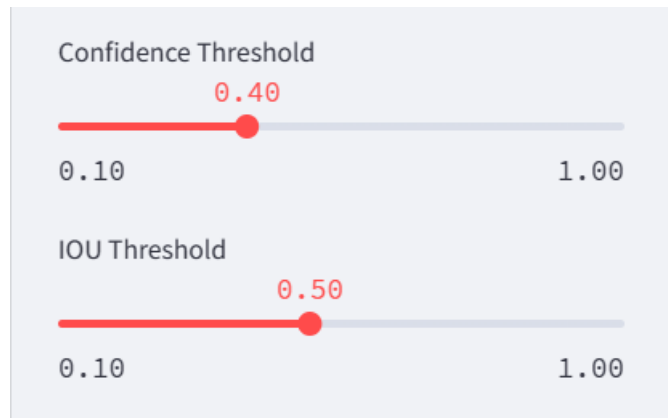
Minh

Save

Save Successfully!

Hình 4.4: Thông báo upload ảnh thành công

- Tiếp đến người dùng lựa chọn các tham số cho mô hình nhận diện như Confidence Threshold, IOU Threshold. Các tham số lựa chọn nằm trong khoảng từ 0 đến 1, lựa chọn càng cao thì yêu cầu độ chính xác nhận diện càng lớn. Mặc định các tham số sẽ được đặt là Confidence Threshold = 0,4 và IOU Threshold = 0.5.

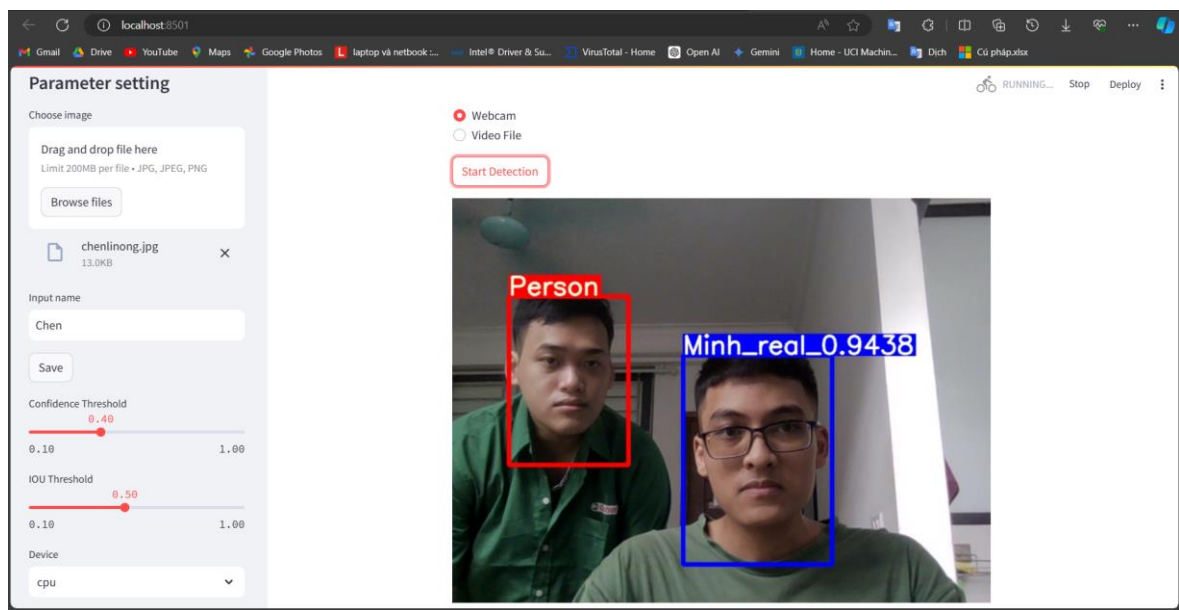


Hình 4.5: Confidence Threshold và IOU Threshold

- Tiếp đến là lựa chọn thiết bị phần cứng cho hệ thống, tùy thuộc vào máy chủ sử dụng thiết bị phần cứng là CPU hoặc GPU mà người dùng có thể lựa chọn. Với thiết bị GPU sẽ cho ra hiệu suất nhận diện tốt hơn so với CPU, mặc định hệ thống sẽ sử dụng CPU làm thiết bị phần cứng.
- Cuối cùng là lựa chọn nguồn video nhận diện, người dùng lựa chọn nguồn video là từ một video cũ được lưu trữ hoặc nguồn video theo thời

gian thực từ camera, mặc định hệ thống sẽ lựa chọn nguồn video là từ camera theo thời gian thực.

- Cuối cùng quá trình nhận diện được thực hiện khi người dùng nhấn nút Start Detection, màn hình giao diện chính sẽ hiển thị video nhận diện theo thời gian thực để người dùng theo dõi.



Hình 4.6: Kết quả nhận diện khuôn mặt theo thời gian thực trên Website

Thông qua kết quả nhận diện có thể thấy, chương trình đã thành công nhận diện người thông qua camera theo thời gian thực. Với người có cơ sở dữ liệu trong hệ thống, video nhận diện sẽ hiển thị với xử lý khoanh vùng khuôn mặt người đó và vẽ khung nhận diện màu xanh có tên người đó và độ chính xác ở trên khung. Với người không có trong cơ sở dữ liệu, video nhận diện hiển thị xử lý với khung màu đỏ và chữ Person bao quanh khuôn mặt người lạ. Qua kết quả nhận diện có thể kết luận chương trình website tích hợp hoạt động ổn định và đạt kết quả mong đợi.

4.6. Hạn chế và hướng phát triển

4.6.1. Hạn chế

Độ chính xác: Mặc dù YOLOv5 có thể nhanh chóng phát hiện đối tượng trên hình ảnh hoặc video, nhưng độ chính xác của việc phát hiện khuôn mặt chưa thực sự cao như các phương pháp chuyên biệt khác. Điều này có thể dẫn

đến việc nhận diện khuôn mặt không chính xác hoặc nhận diện nhầm. Ngoài ra do còn hạn chế về mặt phần cứng của thiết bị khi chỉ sử dụng được cpu thay vì gpu, cũng như việc camera có độ phân giải chưa thực sự tốt đã khiến kết quả nhận diện còn có sai lệch cũng như hiệu suất còn chưa tốt.

Hiệu suất: Sử dụng YOLOv5 và ArcFace đồng thời có thể yêu cầu một lượng lớn tài nguyên tính toán, đặc biệt là khi thực hiện trong thời gian thực. Điều này có thể tạo ra tải tài nguyên cao trên hệ thống, đặc biệt là trên các thiết bị có tài nguyên hạn chế như các thiết bị nhúng.

Khả năng triển khai: Việc triển khai một hệ thống nhận diện khuôn mặt theo thời gian thực sử dụng YOLOv5 và ArcFace có thể phức tạp và đòi hỏi kiến thức sâu rộng về cả hai mô hình. Điều này có thể làm tăng khó khăn trong việc xây dựng và duy trì hệ thống.

Tính năng đa dạng: Hệ thống website tích hợp vẫn còn sơ sài và chưa có nhiều tính năng nâng cao dành cho người dùng, đồng thời vẫn chưa phát triển tốt cho việc tích hợp vào hệ thống nhúng như camera giám sát.

4.6.2. Hướng phát triển

Tối ưu hóa mô hình: Tối ưu hóa YOLOv5 và ArcFace để tăng hiệu suất tính toán và độ chính xác của hệ thống. Điều này có thể bao gồm việc điều chỉnh siêu tham số, sử dụng các kỹ thuật tối ưu hóa mã nguồn mở hoặc tối ưu hóa phần cứng.

Học sâu hơn về dữ liệu: Thu thập và sử dụng một bộ dữ liệu lớn hơn và đa dạng để huấn luyện mô hình nhận diện khuôn mặt và nhận diện đúng, giúp cải thiện độ chính xác của hệ thống.

Triển khai trên nền tảng nhúng: Phát triển và tối ưu hóa hệ thống để có thể chạy trên các thiết bị nhúng có tài nguyên hạn chế, như điện thoại di động hoặc camera thông minh.

Sử dụng kỹ thuật song hành: Kết hợp các kỹ thuật kiểm tra như face anti-spoofing để đảm bảo rằng hệ thống không bị đánh lừa bởi ảnh chụp hoặc video của người dùng, từ đó nâng cao tính an toàn và đáng tin cậy của hệ thống.

Nâng cấp phần cứng: Để chương trình có thể nhận diện chính xác hơn cũng như có hiệu suất tốt hơn, cần nâng cấp phần cứng về gpu và camera có độ phân giải cao.

KẾT LUẬN

Trong dự án này, em đã phát triển một hệ thống nhận diện khuôn mặt theo thời gian thực sử dụng mô hình YOLOv5 và ArcFace đã được huấn luyện trước. Hệ thống cũng đã được thực hiện tích hợp các hàm liên quan để đảm bảo hệ thống hoạt động một cách chính xác và hiệu quả.

Trong quá trình hoàn thiện dự án đã đạt được một số thành tựu đáng kể. Đầu tiên là việc thành công trong việc huấn luyện mô hình phát hiện và nhận diện khuôn mặt từ đám đông, mô hình cho độ chính xác cao và có thể ứng dụng. Tiếp đến là xây dựng một chương trình ứng dụng các mô hình đã được huấn luyện ở trên để sử dụng nhận diện khuôn mặt theo thời gian thực, tích hợp chương trình vào website nhằm để cho người dùng dễ dàng sử dụng.

Mặc dù đã đạt được một số thành tựu kể trên nhưng vẫn còn một số điều chưa hoàn thiện. Vì đây là mô hình deep learning với độ phức tạp tính toán lớn, vì vậy khi sử dụng trên các thiết bị cũ và không trang bị GPU rời sẽ cho ra hiệu suất chưa thực sự tốt. Hai là đối với trường hợp cơ sở dữ liệu lớn, toàn bộ dữ liệu ảnh được lưu trữ không được nén và độ phức tạp thời gian cho truy vấn thông tin là $O(N)$, nên có thể tốn rất nhiều không gian phần cứng và khiến tốc độ truy vấn thông tin nhận diện giảm đi rất nhiều.

Trong tương lai, em sẽ cố gắng phát triển dự án để nâng cao tính năng, cải thiện độ ổn định và hiệu suất hệ thống, phát triển giao diện người dùng hiện đại và dễ dùng hơn cũng như thêm các tính năng mới.

Tổng kết lại, dự án “Nhận diện khuôn mặt theo thời gian thực ứng dụng tìm kiếm người trong đám đông” đã đạt được những kết quả đáng kể và mở ra tiềm năng phát triển trong tương lai. Em thực sự hài lòng với những thành tựu đã đạt được và sẽ tiếp tục nỗ lực để nâng cao hiệu suất và tính năng của hệ thống, nhằm đáp ứng tốt hơn nhu cầu và yêu cầu của người dùng.

Tài liệu tham khảo

Tài liệu tiếng Việt:

- [1] Phùng Đức Hòa (Chủ biên), Hoàng Quang Huy, Hoàng Văn Hoàn, Nguyễn Đức Lưu, Trịnh Bá Quý, (2019). "Giáo trình Nhập môn công nghệ phần mềm".
- [2] Nguyễn Thanh Tuấn, (2020). "Deep Learning Cơ Bản".
- [3] Vũ Hữu Tiếp (2020). "Machine Learning Cơ Bản".

Tài liệu tiếng Anh:

- [4] Kevin P. Murphy, (2012). "Machine Learning: A Probabilistic Perspective".
- [5] Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2019). ArcFace: Additive Angular Margin Loss for Deep Face Recognition. arXiv preprint arXiv:1801.07698

Website:

- [6] Redmon, J., & Farhadi, A. (2016). YOLO: Real-Time Object Detection GeeksforGeeks.

URL:<https://www.geeksforgeeks.org/yolo-you-only-look-once-real-time-object-detection/>

Lần truy cập gần nhất ngày: 1/5/2024.

- [7] Saito, K. (2019). YOLO: Object Detection Explained. DataCamp.

URL:<https://www.datacamp.com/blog/yolo-object-detection-explained>

Lần truy cập gần nhất ngày: 3/5/2024

- [8] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv5: In-Depth Description of Architecture. Ultralytics Documentation.

URL:https://docs.ultralytics.com/yolov5/tutorials/architecture_description/,

Lần truy cập gần nhất ngày: 3/5/2024

[9] Deng, Jiankang and Guo, Jia and Xue, Niannan and Zafeiriou, Stefanos (2019). ArcFace: Additive Angular Margin Loss for Deep Face Recognition

URL: [ArcFace | InsightFace: an open source 2D&3D deep face analysis library](#)

Lần truy cập gần nhất ngày: 5/5/2024

[10] Joseph Redmon , Santosh Divvala, Ross Girshick, Ali Farhadi University of Washington, Allen Institute for AI. You Only Look Once: Unified, Real-Time Object Detection.

Lần truy cập gần nhất ngày: 3/5/2024

URL: <https://arxiv.org/pdf/1506.02640>

[11] Deepti Chamoli. Deep learning-based Live-Streaming Face Recognition

URL: <https://medium.com/analytics-vidhya/deep-learning-based-live-streaming-face-recognition-31e9b005ffb>

Lần truy cập gần nhất ngày: 6/5/2024

[12] GeekforGeeks. Introduction to Convolution Neural Network.

URL: [Introduction to Convolution Neural Network - GeeksforGeeks](#)

[13] Nguyễn Long Trần. Understanding Convolutional Neural Networks for Natural Language processing.

Lần truy cập gần nhất ngày: 25/4/2024

URL: <https://viblo.asia/p/understanding-convolutional-neural-networks-for-natural-language-processing-bJzKmW0B19N>

[14] Issac kondreddy. Chapter 2: Convolutional Neural Networks — Unveiling Hidden Patterns in Images.

Lần truy cập gần nhất ngày: 25/4/2024

URL: <https://medium.com/@issackondreddy/chapter-2-convolutional-neural-networks-unveiling-hidden-patterns-in-images-b4574d34f556>

Lần truy cập gần nhất ngày: 25/4/2024

[15] Elizabeth Van Campen. Overview of loss functions for Machine Learning.

URL: <https://medium.com/analytics-vidhya/overview-of-loss-functions-for-machine-learning-61829095fa8a>

Lần truy cập gần nhất ngày: 28/4/2024

[16] Việt Anh. Tìm hiểu mô hình YOLO cho phát hiện vật - Từ YOLOv1 đến YOLOv3.

URL: <https://vn.nrl.ai/blog/2020-10-11-tim-hieu-mo-hinh-yolo>

Lần truy cập gần nhất ngày: 3/5/2024

[17] geeksforgeek. Swish Activation Function.

URL: <https://www.geeksforgeeks.org/swish-activation-function/>

Lần truy cập gần nhất ngày: 25/5/2024

[18] geekforgeek. What is a neural network?.

URL: [What is a neural network? - GeeksforGeeks](#)

Lần truy cập gần nhất ngày: 3/5/2024

[19] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In European conference on computer vision (pp. 740-755). Springer, Cham.

URL: [COCO - Common Objects in Context \(cocodataset.org\)](https://cocodataset.org)

Lần truy cập gần nhất ngày: 14/5/2024

[20] Yang, S., Luo, P., Loy, C. C., & Tang, X. (2016). WIDER FACE: A face detection benchmark. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 5525-5533).

URL: [WIDER FACE: A Face Detection Benchmark | Papers With Code](#)

Lần truy cập gần nhất ngày: 10/5/2024