BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

ĐỒ ÁN TỐT NGHIỆP

NGÀNH : CÔNG NGHỆ THÔNG TIN

Sinh viên thực hiện : Cù Văn Tú

Giáo Viên Hướng Dẫn : TS. Hồ Thị Hương Thơm

HẢI PHÒNG – 2025

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

XÂY DỰNG MÔ HÌNH PHÁT HIỆN NGƯỜI LÁI XE Ô TÔ CÓ HÀNH VI SỬ DỤNG ĐIỆN THOẠI DI ĐỘNG KHI LÁI XE

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

NGÀNH: Công nghệ thông tin

Sinh viên thwucj hiện : Cù Văn Tú

Giáo Viên Hướng Dẫn : TS. Hồ Thị Hương Thơm

HẢI PHÒNG – 2025

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên: **Cù Văn Tú** **Mã SV**: **2112111041**

Lớp : CT2501C

Ngành : Công nghệ thông tin

Tên đề tài: ***"Xây dựng mô hình phát hiện người lái xe ô tô có hành vi xử dụng điện thoại di động khi lái xe"***

NHIỆM VỤ ĐỀ TÀI

1. Nội dung và các yêu cầu cần giải quyết trong nhiệm vụ đề tài tốt nghiệp

Mô tả tóm tắt đề tài

- Mô hình hệ thống AI được thiết kế để phát hiện người lái xe ô tô có hành vi sử dụng điện thoại di động khi đang lái xe
- Cung cấp khả năng phát hiện đối tượng theo thời gian thực với độ chính xác cao, giúp đưa ra cảnh báo kịp thời đảm bảo an toàn khi tham gia giao thông.

2. Nội dung hướng dẫn

- Nghiên cứu lý thuyết nhận dạng và phân loại đối tượng.
- Vận dụng lý thuyết xây dụng mô hình AI để phát hiện hành vi sử dụng điện thoại di động khi đang lái xe có thể gây mất tập trung không đảm bảo an toàn khi lưu thông trên đường.
- Thử nghiệm và đánh giá mô hình

3. Kết quả cần đạt được

Xây dụng mô hình AI để phát hiện người lái xe ô tô có hành vi sử dụng điện thoại di động khi đang lái xe ô tô để đưa ra cảnh báo kịp thời đảm bảo an toàn giao thông

4. Địa điểm thực tập tốt nghiệp

Trường Đại học Quản lý và Công nghệ Hải Phòng

LỜI CẢM ƠN

Đồ án tốt nghiệp được hoàn thành tại Trường Đại Học Quản Lý và Công Nghệ Hải Phòng. Trong thời gian học tập và làm đề tài đồ án " Xây dụng mô hình phát hiện người lái xe ô tô có hành vi sử dụng điện thoại khi lái xe" em đã nhận được rất nhiều sự hướng dẫn chỉ bảo để hoàn thiện đề tài.

Điều đầu tiên, em xin gửi lời cảm ơn trân thành đến có TS. Hồ Thị Hương Thơm. Cô đã hướng dẫn, chỉ bảo, truyền đạt kiến thức tâm huyết để em có thể hoàn thành đề tài đồ án này.

Tiếp đến em xin chân thành cảm ơn đến Trường Đại Học Quản Lý và Công Nghệ Hải Phòng nơi em đã học tập, các học viên nhóm lớp Công Nghệ Thông Tin khóa 2021 – 2025 đã tạo điều kiện, giúp đỡ, cổ vũ về tinh thần cho em trong suốt quá trình học tập, hoàn thiện đồ án.

Trân trọng

Hải Phòng,ngày tháng năm 2025

Người viết

Cù Văn Tú

LỜI CAM ĐOAN

Sinh viên xin cam đoan đề tài " Xây dụng mô hình phát hiện người lái xe ô tô có hành vi sử dụng điện thoại khi lái xe" là công trình nghiên cứu độc lập dưới sự hướng dẫn của TS. Hồ Thị Hương Thơm – Giảng viên Khoa Công nghệ thông tin Trường Đại Học Hàng Hải Việt Nam. Đề tài là sự cố gắng, quyết tâm của cá nhân sinh viên trong việc tiếp cận kiến thức chuyên môn mới để hoàn thành đạt mục tiêu của đề tài. Các dữ liệu đề nghiên cứu thực nghiệm là trung thức có nguồn gốc trích dẫn rõ ràng.

Sinh viên xin hoàn toàn chịu trách nghiệm với lời cam đoan này

Người cam đoan

Cù Văn Tú

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

DANH MỤC BẢNG VIẾT TẮT

STT	Từ viết tắt	Nghĩa tiếng Anh	Nghĩa tiếng Việt
1	AI	Artificial Intelligence	Trí tuệ nhân tạo
2	CNN	Convolutional Neural Network	Mạng nơ-ron tích chập
3	R-CNN	Region-based Convolutional Neural Network	Mạng CNN dựa trên vùng
4	Fast R-CNN	Fast Region-based CNN	Mạng CNN nhanh dựa trên vùng
5	Faster R-CNN	Faster Region-based CNN	Mạng CNN nhanh hơn dựa trên vùng
6	YOLO	You Only Look Once	Thuật toán phát hiện đối tượng YOLO
7	SSD	Single Shot MultiBox Detector	Bộ phát hiện đa hộp một lần
8	HOG	Histogram of Oriented Gradient	Biểu đồ Gradien hướng
9	GPU	Graphics Processing Unit	Bộ xử lý đồ họa
10	TPU	Tensor Processing Unit	Bộ xử lý Tensor
11	CPU	Central Processing Unit	Bộ xử lý trung tâm
12	IDE	Integrated Development Environment	Môi trường phát triển tích hợp
13	IT	Information Technology	Công nghệ thông tin

14	HTML	HyperText Markup Language	Ngôn ngữ đánh dấu siêu văn bản
15	CSS	Cascading Style Sheets	Ngôn ngữ định kiểu trang
16	JS	JavaScript	Ngôn ngữ lập trình JavaScript
17	C++	C++	Ngôn ngữ lập trình C++
18	C#	C Sharp	Ngôn ngữ lập trình C#
19	F#	F Sharp	Ngôn ngữ lập trình F#
20	JSON	JavaScript Object Notation	Định dạng dữ liệu JSON
21	VS Code	Visual Studio Code	Trình soạn thảo mã nguồn VS Code
22	API	Application Programming Interface	Giao diện lập trình ứng dụng
23	Colab	Google Colaboratory	Công cụ lập trình Google Colab

CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN

Chương 1 trình bày những thông tin mở đầu của đề tài, bao gồm bối cảnh hình thành vấn đề, lý do lựa chọn đề tài và mục tiêu nghiên cứu. Chương này cũng nêu rõ phạm vi, đối tượng nghiên cứu và phương pháp tiếp cận được sử dụng. Cuối cùng, chương giới thiệu bố cục toàn bộ báo cáo, giúp người đọc có cái nhìn tổng quan trước khi đi vào các nội dung chi tiết của các chương tiếp theo.

1.1. Đặt vấn đề

Trong những năm gần đây, tai nạn giao thông đang trở thành một vấn nạn nghiêm trọng tại nhiều quốc gia, đặc biệt là ở các đô thị lớn. Một trong những nguyên nhân phổ biến và nguy hiểm dẫn đến tai nạn là hành vi sử dụng điện thoại khi đang lái xe. Theo nhiều nghiên cứu, việc sử dụng điện thoại – dù để nghe, gọi, nhắn tin hay lướt mạng xã hội – đều làm giảm khả năng phản xạ và mức độ tập trung của người điều khiển phương tiện. Điều này dẫn đến nguy cơ mất kiểm soát và gây tai nạn cao gấp nhiều lần so với lái xe bình thường.

Trên thế giới, nhiều giải pháp đã được nghiên cứu nhằm hỗ trợ giám sát tình trạng tỉnh táo của tài xế, từ cảm biến sinh học (nhịp tim, sóng não) cho đến hệ thống camera giám sát hành vi. Trong đó, việc ứng dụng trí tuệ nhân tạo [1] và thị giác máy tính [2] để tự động phát hiện dấu hiệu sử dụng điện thoại khi lái xe từ hình ảnh/video khuôn mặt, mắt và tư thế lái xe đang được đánh giá là hướng đi hiệu quả, khả thi.

Đồ án này tập trung vào việc áp dụng một trong những công nghệ tiên tiến nhất trong lĩnh vực nhận diện đối tượng, đó là YOLO11 [8] (You Only Look Once version 11), để phát hiện và cảnh báo về nguy hiểm khi sử dụng điện thoại khi đang lái xe từ hình ảnh. YOLO11 là một mô hình nhận diện vật thể mạnh mẽ, có khả năng phát hiện vật thể nhanh chóng và chính xác trên cả ảnh và video.

Trong đồ án này, em đã tiến hành thu thập dữ liệu, huấn luyện mô hình và phát triển một hệ thống phát hiện người lái xe ô tô sử dụng điện thoại khi đang lái xe từ hình ảnh sử dụng YOLO11. Em hy vọng rằng công việc của em sẽ đóng

góp vào việc tăng cường khả năng phát hiện và phản ứng sớm trước nguy cơ gây tai nạn, từ đó giảm thiểu thiệt hại và nguy cơ cho cộng đồng.

Hình 1.1. Hình ảnh người lái xe sử dụng điện thoại di động khi lái xe

1.2. Khái niệm trong ảnh

Đối tượng trong hình ảnh là các yếu tố, vật thể, hoặc phần tử cụ thể mà máy ảnh hoặc người xem hình ảnh có thể nhận diện được. Khái niệm này không chỉ áp dụng cho nhiếp ảnh mà còn cho các lĩnh vực khác như xử lý ảnh, trí tuệ nhân tạo, và thị giác máy tính. Một đối tượng trong hình ảnh có thể là bất kỳ ai, đàn ông, đàn bà, v.v. Trong phân tích hình ảnh, việc nhận diện và phân loại đối tượng là một trong những thách thức quan trọng.

Hình 1.2. Hình ảnh người đàn ông đang sử dụng điện thọa đang lái xe

1.3. Phát hiện đối tượng

Phát hiện đối tượng là mội lĩnh vực vô cùng quan trọng trong lĩnh vực xử lý ảnh và thị giác máy tính [2].Phát hiện đối tượng liên quan đến việc xác định trường hợp cụ thể của đối tượng trong hình ảnh hoặc video. Các mô hình phát hiện đối tượng thường được dùng để xác định đối tượng trên hình ảnh hoặc video camera trong thời gian thực.

Đây là một hướng xử lý rất mạnh mẽ nếu muốn hệ thống có khả năng xác định rõ vị trí các đối tượng như điện thoại, tay, khuôn mặt trong ảnh hoặc video, từ đó suy luận hành vi sử dụng điện thoại khi lái xe.

Các phương pháp phổ biến cho việc phát hiện đối tượng bao gồm sử dụng các mô hình học sâu như các mạng nơ-ron convolutional (CNN) và các kỹ thuật như R-CNN, Fast R CNN, Faster R-CNN, YOLO (You Only Look Once), SSD (Single Shot MultiBox Detector), và RetinaNet.

1.4. Nhận dạng đối tượng

Sau khi đối tượng được phát hiện, bước tiếp theo là nhận dạng đối tượng, tức là xác định chính xác loại đối tượng đó là gì. Điều này thường liên quan đến việc so sánh các đặc trưng của đối tượng đã được rút trích từ hình ảnh với các đặc trưng của các đối tượng đã biết trước trong cơ sở dữ liệu. Các phương pháp cho việc nhận dạng đối tượng có thể dựa trên học máy, học sâu, hoặc các kỹ thuật xử lý ảnh truyền thống. Cả phát hiện và nhận dạng đối tượng đều là những thách thức quan trọng trong lĩnh vực trí tuệ nhân tạo và có ứng dụng rộng rãi trong nhiều lĩnh vực như nhận dạng khuôn mặt, giám sát an ninh, tự động hóa công nghiệp, xe tự lái, và nhiều ứng dụng khác

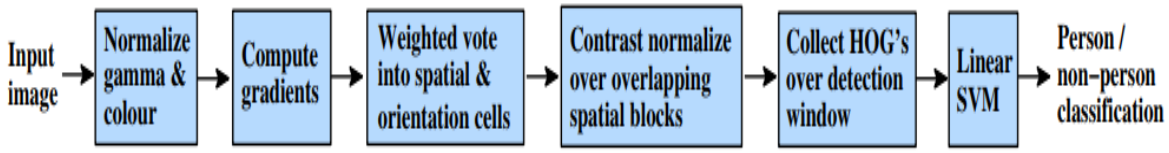
Hình 1.3. Phát hiện đối tượng

1.5 . Các phương pháp phát hiện phổ biến hiện nay

1.5.1. Phương pháp mô tả đặc trưng (HOG)

Phương pháp mô tả đặc trưng (Histogram of Oriented Gradient – HOG) được xem là một trong những kỹ thuật phát hiện đối tượng cổ điển và được sử dụng từ khá sớm. Ý tưởng ban đầu được giới thiệu vào năm 1986, tuy nhiên phải đến năm 2005 phương pháp này mới thực sự phổ biến và được ứng dụng rộng rãi trong các bài toán thị giác máy tính [2]. HOG đóng vai trò là một công cụ trích xuất đặc trưng, hỗ trợ nhận dạng và xác định vị trí đối tượng trong hình ảnh.

Phương pháp mô tả đặc trưng (HOG) là một cách tiếp cận trong các lĩnh vực như học máy (machine learning), thị giác máy tính (computer vision), và xử lý dữ liệu, nhằm trích xuất và mô tả các đặc trưng (features) quan trọng từ dữ liệu (ví dụ: hình ảnh, văn bản, âm thanh...) để phục vụ cho việc phân tích, phân loại hoặc nhận dạng.

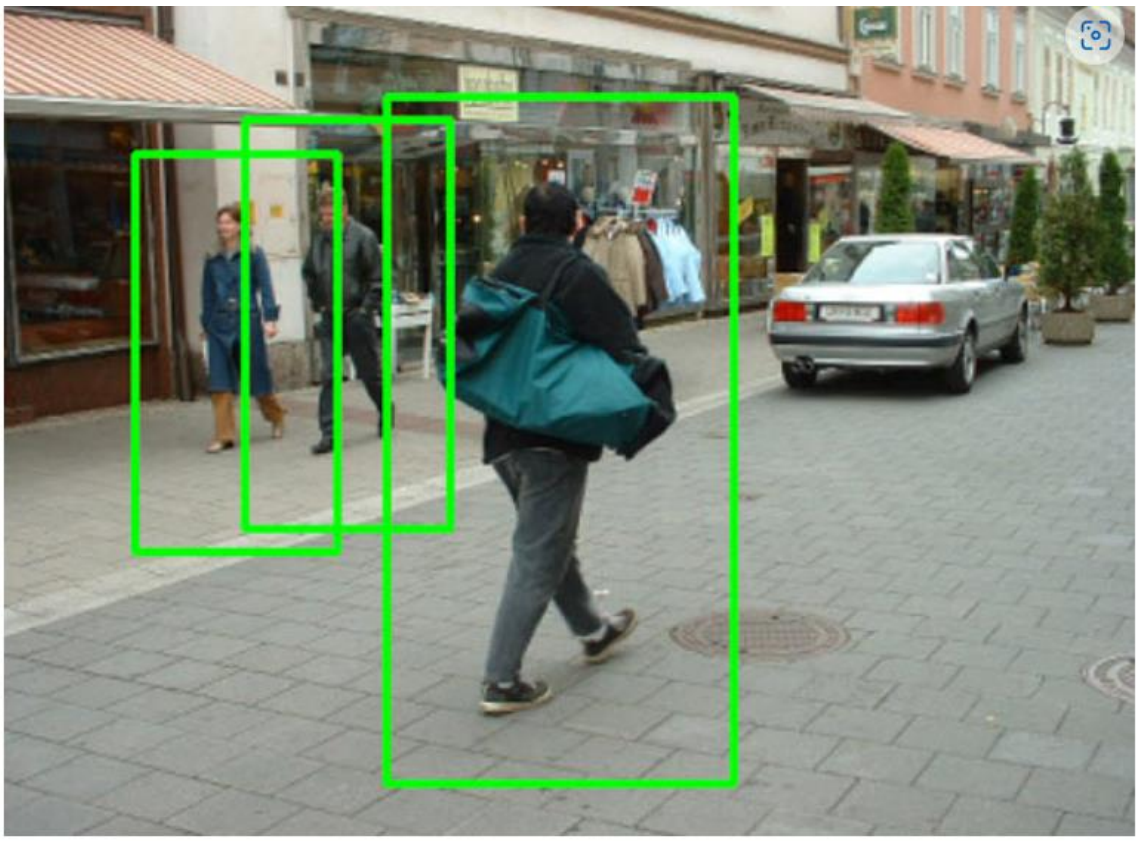


Hình 1.4. Sơ đồ hệ thống HOG để phát hiện đối tượng

Điểm chính trong nguyên lý hoạt động của HOG là mô tả hình dạng của một vật thể cục bộ thông qua hai ma trận: magnitude gradient và **orientation gradient*. Để tạo ra hai ma trận này, ảnh được chia thành một lưới ô vuông, trên mỗi ô vuông, ta tính toán biểu đồ histogram thống kê độ lớn gradient. Mỗi ô vuông thường có kích thước 8x8 pixels và bao gồm nhiều ô cục bộ. HOG descriptor được tạo thành bằng cách nối liền 4 vector histogram từ mỗi ô cục bộ. HOG descriptor được tạo thành bằng cách nối liền 4 vector histogram từ mỗi ô cục bộ thành một vector tổng hợp. Để cải thiện độ chính xác, mỗi giá trị của vector histogram trên mỗi vùng cục bộ được chuẩn hóa theo Norm 2 hoặc Norm 1. Ưu điểm của HOG là nó bao gồm tính bất biến đối với biến đổi hình học và thay đổi độ sáng, cũng như khả năng loại bỏ chuyển động cơ thể trong phát hiện con người nếu họ duy trì tư thế đứng thẳng.

Ứng dụng của HOG:

Nhận diện người (human detection): Lần đầu tiên ứng dụng này được giới thiệu trong bài báo Histograms of Oriented Gradients for Human Detection của Dalal và Trigg. HOG có thể phát hiện được một hoặc nhiều người đi bộ trên cùng một hình ảnh



Hình 1.5.Sử dụng HOG để phát hiện một hoặc nhiều người đi bộ

Nhận diện khuôn mặt (face detection) [4]: Thường chúng ta sẽ nghĩ ngay đến thuật toán Haar Cascde Classifier. Tuy nhiên HOG cũng là một thuật toán rất hiệu quả được áp dụng trong bài toán này. Bởi nó có khả năng biểu diễn các đường nét chính của khuôn mặt dựa trên phương và độ lớn gradient thông qua các véc tơ trên mỗi cell như hình mô tả bên dưới:

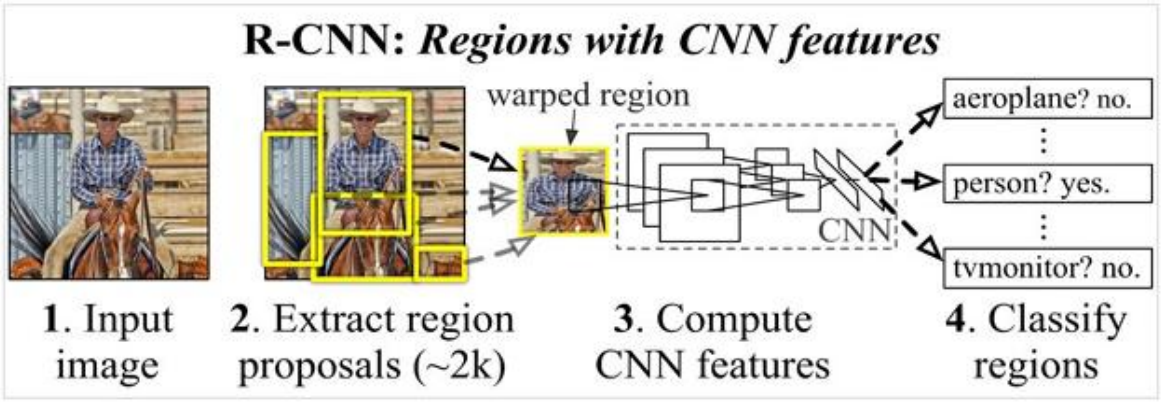
Hình 1.6. NHận diện khuôn mặt bằng phương pháp mô tả đặc trưng

Nhận diện các vật thể khác: Ngoài ra còn rất nhiều các trường hợp nhận diện vật thể trên ảnh tĩnh như phương tiện, tín hiệu giao thông, động vật hoặc thậm chỉ là ảnh động từ video.

1.5.2. Phương pháp học sâu Mạng nơ-ron

Mạng nơ-ron tích chập theo vùng (R-CNN):

Các mạng nơ-ron tích chập dựa trên khu vực là một cải tiến trong quy trình phát hiện đối tượng từ các phương pháp HOG và SIFT (biến đổi đặc trưng không đổi theo tỉ lệ) trước đây. Trong các mô hình R-CNN, chúng ta cố gắng trích xuất các đặc trưng thiết yếu nhất (thường là khoảng 2000 đặc trưng) bằng cách sử dụng các đặc trưng chọn lọc. Quá trình lựa chọn các trích xuất quan trọng nhất có thể được thực hiện với sự trợ giúp của thuật toán tìm kiếm chọn lọc, nhằm đạt được các đề xuất khu vực quan trọng hơn**.**



Hình 1.7: Mô hình R-CNN

Quy trình làm việc của thuật toán tìm kiếm chọn lọc để chọn các đề xuất khu vực quan trọng nhất bao gồm việc tạo nhiều phân đoạn phụ trên một hình ảnh cụ thể và chọn các mục ứng cử viên cho nhiệm vụ của bạn. Thuật toán tham lam sau đó có thể được sử dụng để kết hợp các mục hiệu quả phù hợp cho một quy trình định kỳ, nhằm kết hợp các phân đoạn nhỏ hơn thành các phân đoạn lớn hơn phù hợp.

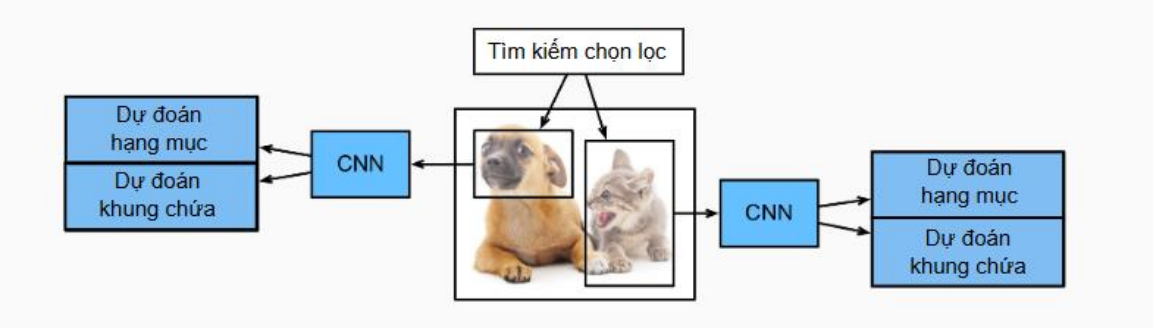
Khi thuật toán tìm kiếm chọn lọc hoàn thành thành công, nhiệm vụ tiếp theo của chúng ta là trích xuất các đặc trưng và đưa ra dự đoán thích hợp. Sau đó, chúng ta có thể đưa ra các đề xuất ứng cử viên cuối cùng và các mạng nơ-ron tích

chập có thể được sử dụng để tạo vectơ đặc trưng n chiều (2048 hoặc 4096) làm đầu ra. Với sự trợ giúp của mạng nơ-ron tích chập được đào tạo trước, chúng ta có thể đạt được nhiệm vụ trích xuất đặc trưng một cách dễ dàng.

Bước cuối cùng của R-CNN là đưa ra các dự đoán thích hợp cho hình ảnh và gắn nhãn hộp giới hạn tương ứng cho phù hợp. Để có được kết quả tốt nhất cho mỗi nhiệm vụ, các dự đoán được thực hiện bằng cách tính toán mô hình phân loại cho từng nhiệm vụ, trong khi mô hình hồi quy được sử dụng để sửa phân loại hộp giới hạn cho các vùng được đề xuất toán tương tự như cách hoạt động của Fast R-CNN.

Mạng nơ-ron tích chập theo vùng, hay các vùng với đặc trưng CNN (R-CNN) là một hướng tiếp cận tiên phong ứng dụng mô hình sâu cho bài toán phát hiện vật thể. Trong phần này, chúng ta sẽ thảo luận về R-CNN và một loạt các cải tiến sau đó: Fast R-CNN, Faster R-CNN, và Mask R-CNN.

Đầu tiên, các mô hình R-CNN sẽ chọn một số vùng đề xuất từ ảnh (ví dụ, các khung neo cũng là một phương pháp lựa chọn) và sau đó gán nhãn hạng mục và khung chứa (ví dụ, các giá trị độ dời) cho các vùng này. Tiếp đến, các mô hình này sử dụng CNN để thực hiện lượt truyền xuôi nhằm trích xuất đặc trưng từ từng vùng đề xuất. Sau đó, ta sử dụng các đặc trưng của từng vùng được đề xuất để dự đoán hạng mục và khung chứa.



Hình 1.8.Mô hình R_CNN

Cụ thể, R-CNN có bốn phần chính sau:

1. Tìm kiếm chọn lọc trên ảnh đầu vào để lựa chọn các vùng đề xuất tiềm năng. Các vùng đề xuất thông thường sẽ có nhiều tỷ lệ với hình dạng và

kích thước khác nhau. Hạng mục và khung chứa nhãn gốc sẽ được gán cho từng vùng đề xuất.

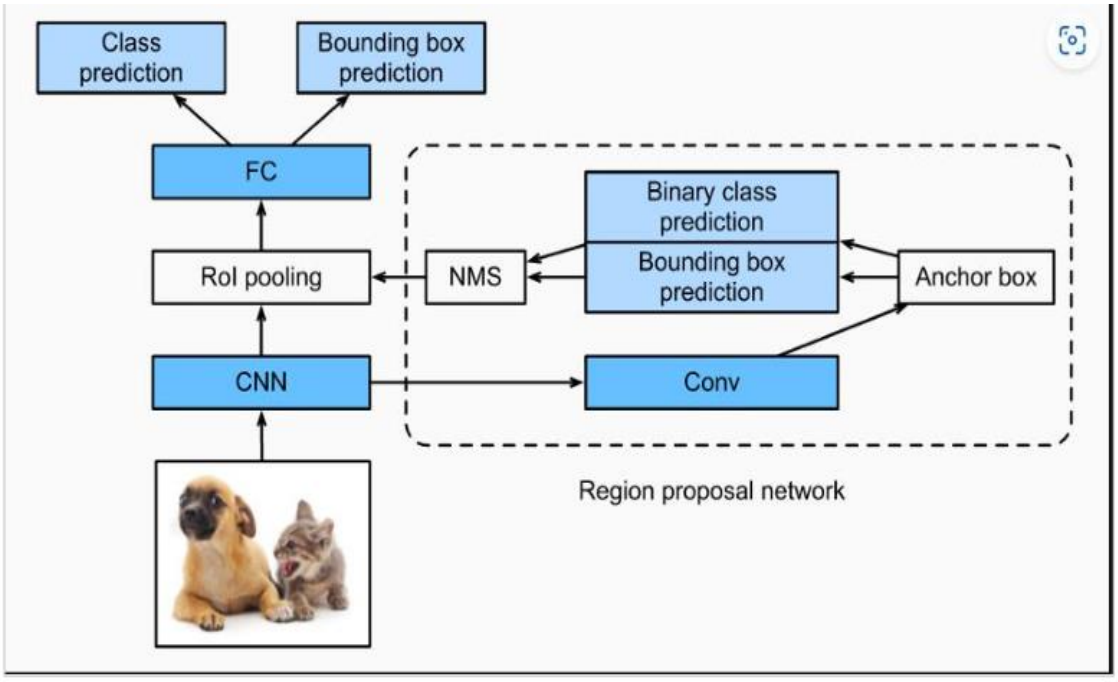
2. Sử dụng một mạng CNN đã qua tiền huấn luyện, ở dạng rút gọn, đặt trước tầng đầu ra. Mạng này biến đổi từng vùng đề xuất thành các đầu vào có chiều phù hợp với mạng và thực hiện các lượt truyền xuôi để trích xuất đặc trưng từ các vùng đề xuất tương ứng.
3. Các đặc trưng và nhãn hạng mục của từng vùng đề xuất được kết hợp thành một mẫu để huấn luyện các máy vector hỗ trợ cho phép phân loại vật thể. Ở đây, mỗi máy vector hỗ trợ được sử dụng để xác định một mẫu có thuộc về một hạng mục nào đó hay không.
4. Các đặc trưng và khung chứa được gán nhãn của mỗi vùng đề xuất được kết hợp thành một mẫu để huấn luyện mô hình hồi quy tuyến tính, để phục vụ dự đoán khung chứa nhãn gốc.

Mặc dù các mô hình R-CNN sử dụng các mạng CNN đã được tiền huấn luyện để trích xuất các đặc trưng ảnh một cách hiệu quả, điểm hạn chế chính yếu đó là tốc độ chậm. Có thể hình dung, với hàng ngàn vùng đề xuất từ một ảnh, ta cần tới hàng ngàn phép tính truyền xuôi từ mạng CNN để phát hiện vật thể. Khối lượng tính toán nặng nề khiến các mô hình R-CNN không được sử dụng rộng rãi trong các ứng dụng thực tế.

Faster R-CNN:

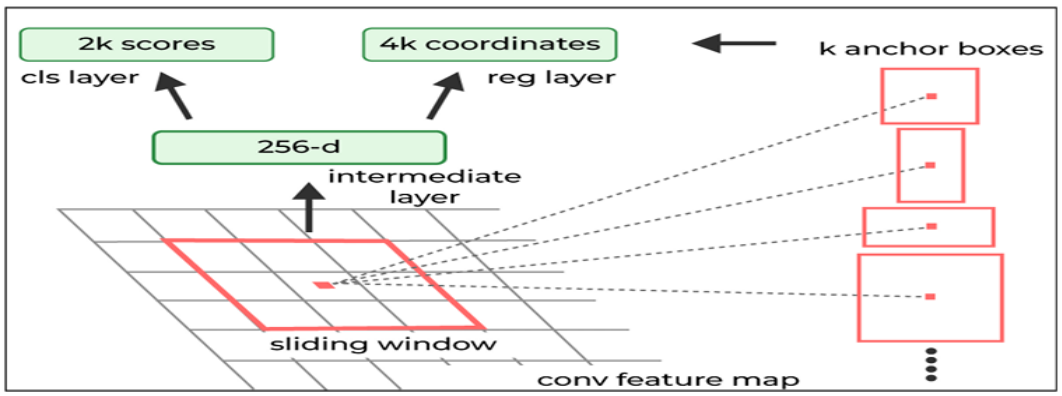
R-CNN, một mô hình phát hiện đối tượng, đã đạt được kết quả đáng chú ý nhưng lại gặp phải vấn đề về tốc độ. Để khắc phục điều này, Fast R-CNN đã được giới thiệu. Trong Fast R-CNN, toàn bộ hình ảnh được truyền qua một mạng nơ-ron

tích chập đã được huấn luyện trước, thay vì chỉ xem xét các phân đoạn con của hình ảnh.



Hình 1.9. Faster R-CNN – Object Detection Algorithm

Tiếp theo, Faster R-CNN, một phiên bản nâng cấp của Fast R-CNN, đã được ra mắt. Faster R-CNN đã thay thế thuật toán tìm kiếm chọn lọc, được sử dụng trong R-CNN và Fast R-CNN để tính toán các đề xuất vùng, bằng một mạng đề xuất vùng vượt trội hơn. Mạng đề xuất vùng (RPN) tính toán hình ảnh từ một phạm vi rộng và các tỷ lệ khác nhau để tạo ra các đầu ra hiệu quả.



Hình 1.10. Mạng đề xuất vùng (RPN)

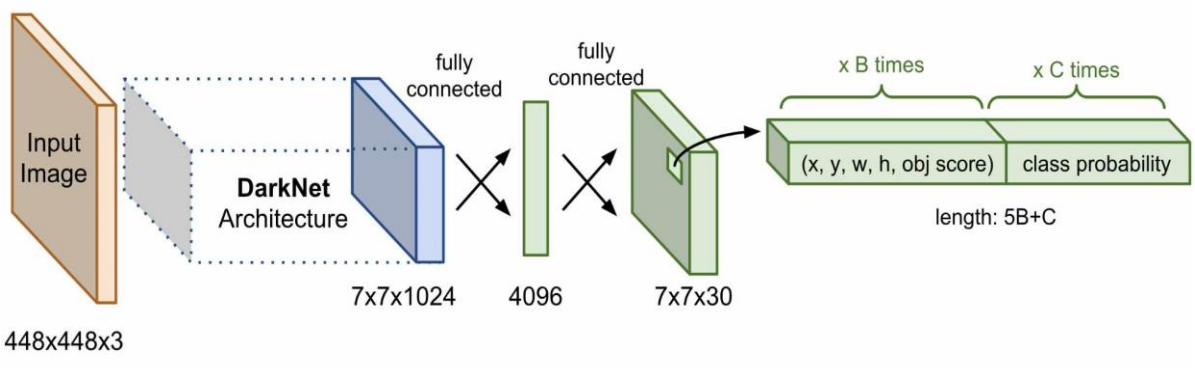
RPN giảm thời gian tính toán biên xuống còn khoảng 10 ms cho mỗi hình ảnh. Mạng này bao gồm một lớp tích chập từ đó chúng ta có thể lấy được các bản đồ đặc trưng cần thiết của mỗi pixel. Đối với mỗi bản đồ đặc trưng, chúng ta có nhiều hộp neo có các tỷ lệ, kích thước và tỷ lệ khía cạnh khác nhau. Đối với mỗi hộp neo, chúng ta dự đoán một lớp nhị phân cụ thể và tạo ra một hộp giới hạn cho nó.

Thông tin sau đó được truyền qua giảm thiểu tối đa để loại bỏ bất kỳ dữ liệu không cần thiết nào vì nhiều chồng chéo được tạo ra khi tạo bản đồ đặc trưng. Đầu ra từ giảm thiểu tối đa được truyền qua vùng quan tâm, và phần còn lại của quá trình và tính

1.5.3. YOLO (You Only Look Once)

YOLO [8] là phương pháp phát hiện đối tượng nhanh chóng và chính xác bằng cách sử dụng một mạng nơ-ron duy nhất để dự đoán các hộp giới hạn và xác suất của các lớp đối tượng trong một lần chạy. YOLO [8] được biết đến với khả năng thời gian thực và hiệu suất cao. Thật vậy, về độ chính xác thì YOLO [8] có thể không phải là thuật toán tốt nhất nhưng nó là thuật toán nhanh nhất trong các lớp mô hình phát hiện đối tượng (object detection). Nó có thể đạt được tốc độ gần như thời gian thực mà độ chính xác không quá giảm so với cái mô hình thuộc top đầu. Kiến trúc mạng YOLO [8] bao gồm mạng cơ sở là các mạng phép tích chập (convolution) làm nhiệm vụ trích xuất đặc trưng. Phần phía sau là những lớp bổ sung(Extra Layes) được áp dụng để phát hiện vật thể trên bản đồ đặc trưng (feature map) của mạng cơ sở (base network).

Sơ đồ kiến trúc mạng của YOLO



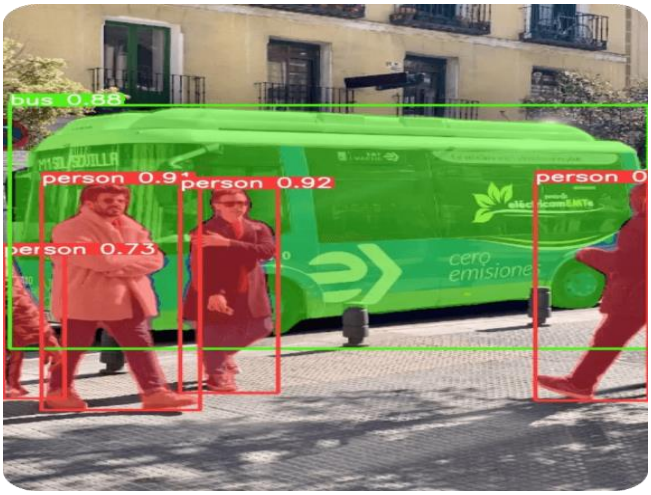
Hình 1.11.Cấu trúc mạng YOLO

Thành phần kiến trúc darknet (Darknet Architechture) được gọi là mạng cơ sở có tác dụng trích suất đặc trưng. Đầu ra của mạng cơ sở là một feature map có kích thước 7x7x1024 sẽ được sử dụng làm đầu vào cho các các lớp bổ sung (Extra

layers) có tác dụng dự đoán nhãn và tọa độ hộp giới hạn (bounding box) của vật thể.

YOLO11 [8] là phiên bản mới nhất trong chuỗi các mô hình YOLO (You Only Look Once), được phát hành vào cuối năm 2024 bởi nhóm phát triển Ultralytics. Đây là mô hình phát hiện đối tượng học sâu tiên tiến, được thiết kế nhằm cải thiện cả tốc độ xử lý, độ chính xác, và khả năng tổng quát hóa trên nhiều tác vụ thị giác máy tính như: phát hiện đối tượng, phân đoạn ảnh, nhận dạng tư thế và phân loại ảnh.

YOLO11 [8] tích hợp nhiều cải tiến mới về kiến trúc, giúp tăng hiệu suất mà không làm tăng đáng kể số lượng tham số



Hình 1.12.Ảnh đươợc sử lý bởi YOLO11

Một số lưu ý khi huấn luyện YOLO:

Khi huấn luyện YOLO [8] sẽ cần phải có RAM dung lượng lớn hơn để lưu được 10647 hộp giới hạn như trong kiến trúc này.

Không thể thiết lập các batch_size quá lớn như trong các mô hình phân loại vì rất dễ hết bộ nhớ. Gói darknet của YOLO [8]đã chia nhỏ một batch thành các phân nhóm cho vừa với RAM. Thời gian xử lý của một bước trên YOLO [8] lâu hơn rất rất nhiều lần so với các mô hình phân loại. Do đó nên thiết lập bước giới hạn huấn luyện cho YOLO nhỏ. Đối với các tác vụ nhận diện dưới 5 lớp, dưới 5000 bước là có thể thu được nghiệm tạm chấp nhận được. Các mô hình có nhiều lớp hơn có thể tăng số lượng bước theo cấp số nhân tùy bạn.

1.5.4.Các phiên bản của YOLO

Các phiên bản của YOLO [8]:

- YOLOv1 [8] (2016): Người phát triển**:** Joseph Redmon.Ý tưởng chính Chia ảnh thành lưới (grid) và dự đoán bounding boxes class cho từng ô.

  Ưu điểm**:** Nhanh (real-time).

  Hạn chế**:** Không chính xác với vật thể nhỏ. Khó phát hiện các vật thể gần nhau

- YOLOv2 [8] (YOLO9000 – 2017) :Cải tiến Dùng Anchor boxes.Kết hợp phân loại và phát hiện (9000 lớp). Sử dụng mô hình Darknet-19.
  - Ưu điểm: Tăng độ chính xác. Phát hiện nhiều đối tượng hơn.
- YOLOv3 [8] (2018) :Cải tiến lớn: Mô hình Darknet-53 sâu hơn. Dự đoán ở nhiều tỉ lệ (multi-scale). Sử dụng ResNet-like skip connections.

  Ưu điểm: Cân bằng tốt giữa tốc độ và độ chính xác. Phát hiện tốt vật thể nhỏ.

- YOLOv4 [8] (2020): Phát triển bởi: Alexey Bochkovskiy (tiếp nối từ Redmon). Cải tiến: Tích hợp nhiều kỹ thuật như: CSPDarknet53, Mish, Mosaic data augmentation, CIoU loss.

  Ưu điểm: Rất tốt cho cả GPU mạnh và yếu. Cực kỳ hiệu quả trong thực tế.

- YOLOv5 [8] (2020, Ultralytics): Lưu ý: Không do nhóm tác giả YOLO gốc phát triển. Viết bằng: PyTorch. Đặc điểm: Dễ sử dụng, cài đặt nhanh. Hỗ trợ xuất sang ONNX, CoreML, TensorRT,... Nhiều kích thước model: YOLOv5s, m, l, x.

  Ưu điểm: Rất phổ biến trong cộng đồng. Dễ tùy chỉnh và triển khai thực tế

- YOLOv6 [8] (2022): Phát triển bởi: Meituan Tối ưu cho: Industrial applications. Cải tiến: Tăng tốc inference. Sử dụng mô hình Anchor-free. Chưa phổ biến rộng rãi như YOLOv5/v8.
- YOLOv7 [8] (2022) Phát triển bởi: Wong Kin-Yiu. Cải tiến: Tối ưu mạnh mẽ về tốc độ & độ chính xác. Hỗ trợ detection, instance segmentation,...

  Ưu điểm: Hiệu năng vượt trội, gần như tốt nhất năm 2022. Có các mô hình nhẹ như YOLOv7-tiny.

- YOLOv8 [8] (2023) Phát triển bởi: Ultralytics (tác giả YOLOv5).

  Tính năng nổi bật: Hỗ trợ cả: object detection, segmentation, pose estimation, tracking Tích hợp huấn luyện, kiểm thử, triển khai trong một API. Mô hình nhẹ hơn, dễ dùng hơn. Là phiên bản YOLO chính thống nhất hiện nay.

- YOLO11 [8] (2024) :Tính năng nổi bật: Phát hiện chính xác các đối tượng trong ảnh với tốc độ cao. Tối ưu hiệu quả trên cả thiết bị mạnh (GPU) và thiết bị nhúng (Edge AI). Hỗ trợ nhiều tác vụ: nhận diện, phân đoạn, phân loại, ước lượng tư thế, định hướng đối tượng.

  Ưu điểm: Tốc độ suy luận nhanh hơn so với YOLOv8, YOLOv10. Độ chính xác cao hơn với ít tham số hơn.Kiến trúc mới như C3k2, SPPF, C2PSA giúp tối ưu hoá việc trích xuất đặc trưng. Hỗ trợ đa nhiệm: Detection, Segmentation, Pose Estimation, OBB, Classification

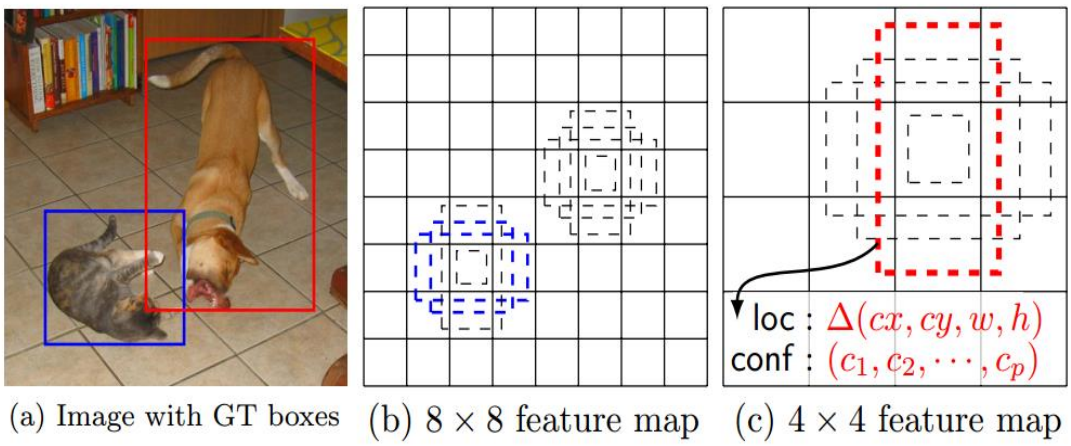
1.5.4. SSD (Single Shot MultiBox Detector)

SSD (Single Shot MultiBox Detector) là một kiến trúc mạng học sâu dùng để phát hiện đối tượng (object detection) trong ảnh hoặc video, được giới thiệu lần đầu tiên bởi nhóm nghiên cứu của Google vào năm 2016.

SSD (Single Shot MultiBox Detector) là một mô hình object detection một bước (single-shot), nghĩa là nó phát hiện và phân loại các vật thể trong ảnh chỉ

trong một lần duyệt qua mạng (forward pass), trái ngược với các phương pháp hai bước như R-CNN hay Faster R-CNN.

SSD có thể đạt được hiệu suất cao và làm việc trực tiếp trên toàn bức ảnh một các hiệu quả.



Hình 1.13. Cách thức phân chia feature map để nhận diện các ảnh với kích thước khác nhau

Trong quá trình training SSD chỉ cần ảnh đầu vào và các hộp dữ liệu thực tế cho mỗi sự vật.

Sau khi đi qua một số lớp Convolution để trích xuất đặc trưng, chúng ta nhận được được một số bản đồ đặc trưng. Ví dụ ở trên chúng ta nhận được bản đồ đặc trưng với kích thước hoặc (chưa tính channels). Convolutional layer 3x3 được áp dụng lên các bản đồ đặc trưng này để đưa ra dự đoán.8x84x4.

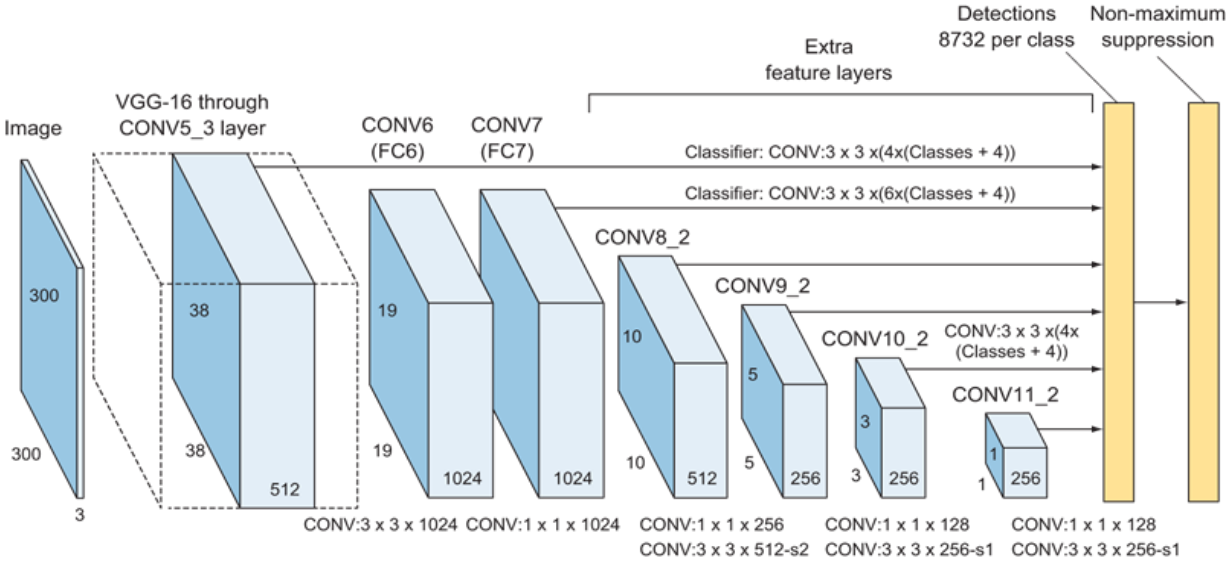
Ứng với mỗi vị trí trên bản đồ đặc trưng chúng ta có hộp mặc định. Hộp mặc định này có kích thước và tỉ lệ khác nhau. Việc chọn số lượng hộp mặc định này dựa trên các sự vật trong bộ tập dữ liệu, có thể dùng K-Means để phân cụm các hộp sữ liệu thực tế để có lựa chọn phù hợp.

Kiến trúc của SSD mô hình bao gồm 3 thành phần chính:

Mạng cơ sở để trích xuất bản đồ đặc trưng. SSD được xây dựng dựa trên mô hình cơ sở VVG-16 có loại bỏ các lớp các lớp được kết nối đầy đủ. Các mô hình cơ sở khác được sử dụng như ResNet… có thể cho kết quả tốt hơn.

Các lớp tính năng đa tỷ lệ: bộ lọc tích chập chuỗi được thêm vào sau mạng cơ sở, bắt đầu từ CONV6 như hình bên dưới. Những lớp này sẽ giảm kích thước để cho phép dự đoán sự vật với nhiều quy mô khác nhau và giảm kích thước đầu vào ở các lớp chuyển đổi tiếp theo.

Non-maximum suppression - NMS được dùng để loại bỏ các hình chồng lên nhau.



Hình 1.14 .Sơ đồ kiến trúc của mạng SSD

1.6. Bài toán "Xây dựng mô hình phát hiện người lái xe ô tô có hành vi sử dụng điện thoại di động khi lái xe"

Bài toán : Xây dựng mô hình phát hiện người lái xe ô tô có hành vi sử dụng điện thoại khi lái xe:

- Phát hiện sớm và phản ứng nhanh chóng: Bằng cách sử dụng YOLO11 [8] để nhận diện người lái xe ô tô sử dụng điện thoại từ hình ảnh, hệ thống có thể phát hiện và cảnh báo về sự nguy hiểm một cách nhanh chóng, giúp giảm thiểu thời gian phản ứng và giảm thiệt hại.
- Bảo vệ an toàn con người: Hệ thống cảnh báo người lái xe ô tô sử dụng điện thoại từ hình ảnh có thể giúp bảo vệ tính mạng và an toàn của con người bằng cách cung cấp cảnh báo kịp thời để người lái xe kịp xử lý các tình huống có thể sảy ra.

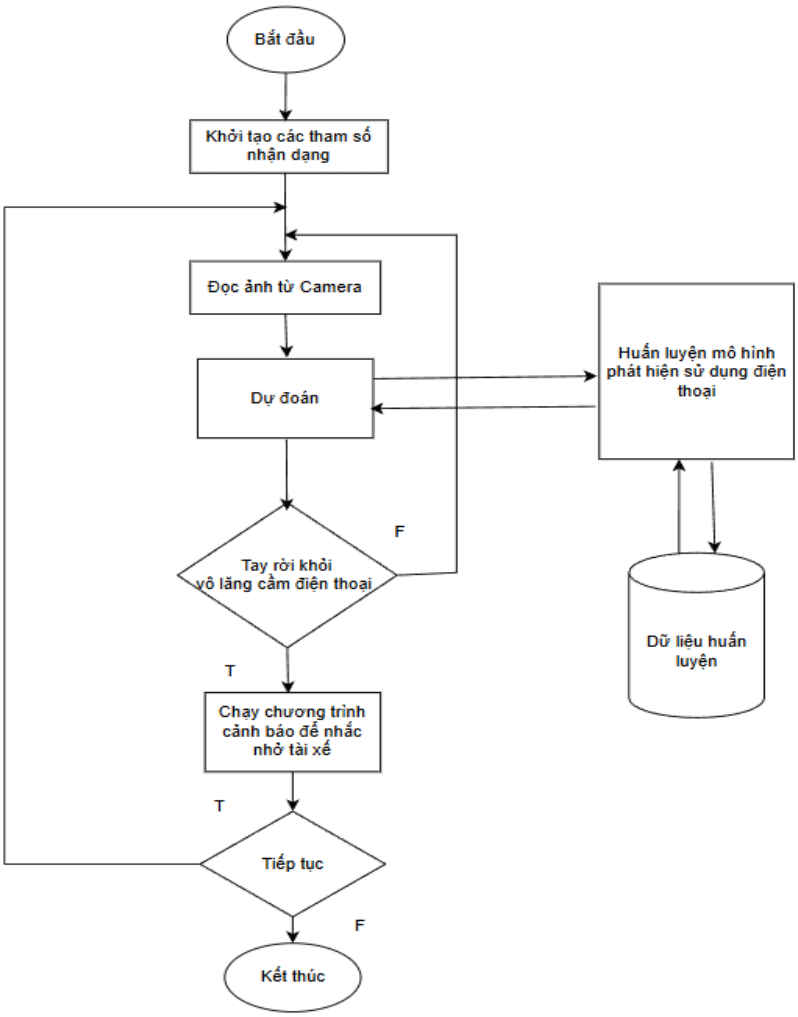
- Bảo vệ tài sản và môi trường: Phát hiện người lái xe sử dụng điện thoại sớm cũng giúp giảm thiểu thiệt hại về tài sản và môi trường, từ việc ngăn chặn tai nạn giao thông đến việc triển khai các biện pháp khắc phục sớm.
- Giảm thiểu rủi ro và nâng cao chất lượng sống: Sự hiện diện của hệ thống cảnh báo người lái xe ô tô sử dụng điện thoại từ hình ảnh giúp giảm thiểu rủi ro và tạo ra một môi trường sống và làm việc an toàn hơn cho cộng đồng.

CHƯƠNG 2. MÔ HÌNH PHÁT HIỆN NGƯỜI LÁI XE ÔTÔ CÓ HÀNH VI SỬ DỤNG ĐIỆN THOẠI DI ĐỘNG KHI LÁI XE

Chương 2 trình bày cơ sở lý thuyết và mô hình được sử dụng để phát hiện hành vi sử dụng điện thoại khi lái xe. Nội dung bao gồm giới thiệu các thuật toán thị giác máy tính, đặc biệt là các mô hình YOLO trong nhận dạng đối tượng. Chương này mô tả quy trình xây dựng mô hình, từ thu thập dữ liệu, gán nhãn, tiền xử lý, đến huấn luyện và đánh giá hiệu suất. Đồng thời, chương cũng phân tích các tiêu chí nhận diện hành vi vi phạm và cách mô hình xác định người lái xe có sử dụng điện thoại trong hình ảnh hoặc video.

2.1. Sơ đồ hoạt động của mô hình phát hiện người lái xe ô tô sử dụng điện thoại di động khi lái xe

Sơ đồ hoạt động của mô hình:



Hình 2.1.Sơ đồ hoạt động của mô hình phát hiện người lái xe ô tô sử dụng điện thoại di động khi lái xe

Mô hình cho phép thực hiện nhận dạng với tùy chọn dữ liệu đầu vào bao gồm file ảnh, file video hoặc hình ảnh trực tiếp từ camera. Với dữ liệu đầu vào là hình ảnh cần cung cấp đường dẫn tuyệt đối của file ảnh, kết quả nhận dạng là hình ảnh được lưu lại trên đó chỉ ra vị trí ngọn lửa và độ tin cậy của dự đoán.

Các thành phần chính của mô hình:

- Khởi tạo tham số: là quá trình gán giá trị ban đầu (giá trị mặc định) cho các tham số. Khi tham số đã được khởi tạo, nếu người dùng không truyền giá trị tương ứng khi gọi hàm thì chương trình sẽ sử dụng giá trị mặc định này.
- Đọc hình ảnh từ camera: Thu nhận dữ liệu ảnh trực tiếp từ thiết bị camera (webcam hoặc camera rời). Phục vụ các bài toán như: nhận diện khuôn mặt, quét mã QR, theo dõi chuyển động, v.v.
- Dự đoán: là quá trình ước lượng trước một kết quả hoặc sự kiện chưa xảy ra, dựa trên dữ liệu hiện tại, kiến thức có sẵn, hoặc các mô hình phân tích.
- Lưu và hiển thị kết quả: Lưu và hiển thị kết quả là hai bước quan trọng trong quá trình xử lý dữ liệu hoặc thực hiện thuật toán:
    + Lưu kết quả: Ghi lại kết quả xử lý (dữ liệu, hình ảnh, văn bản, mô hình,...) vào bộ nhớ tạm hoặc lưu trữ lâu dài như tệp, cơ sở dữ liệu, v.v.
    + Hiển thị kết quả: Trình bày kết quả cho người dùng thấy qua màn hình, bảng biểu, hình ảnh hoặc giao diện người dùng.
- Chạy chương trình**:** là quá trình máy tính thực thi các lệnh và câu lệnh

2.2. Phương pháp phát hiện đối tượng bằng YOLO11

Phát hiện đối tượng là một bài toán thị giác máy tính liên quan đến việc xác định và định vị các đối tượng trong hình ảnh hoặc video. Nó là một phần quan trọng của nhiều ứng dụng, chẳng hạn như camera giám sát thông minh, ô tô tự lái hoặc người máy

2.2.1 Mô hình YOLO (You Only Look Once)

YOLO [8] là một mô hình học sâu nhận diện đối tượng trong hình ảnh một cách nhanh chóng và hiệu quả . Mô hình này được thiết kế để xử lý hình ảnh trong thời gian thực, đầy là điểm mạnh đặc biệt hữu ích cho các bài toán giao thông.

Phiên bản YOLO11 [3], được điều chỉnh và nâng cấp từ các phiên bản trước đó, đảm bảo tính chính xác cao hơn trong những trường hợp hình ảnphức tạp như các vết nứt nhỏ trên bề mặt đường.

2.2.2. YOLO11 là gì?

Sự phát triển của thuật toán YOLO [8] đạt đến tầm cao mới với sự ra đời của YOLO11 [8] [3], đại diện cho một bước tiến đáng kể trong công nghệ phát hiện đối tượng theo thời gian thực. Phiên bản mới nhất này xây dựng dựa trên thế mạnh của phiên bản trước đó đồng thời giới thiệu các khả năng mới giúp mở rộng tiện ích của nó trên nhiều ứng dụng CV khác nhau.

YOLO11 [8] [3] nổi bật nhờ khả năng thích ứng được cải tiến, hỗ trợ phạm vi mở rộng các tác vụ CV ngoài việc phát hiện đối tượng truyền thống.

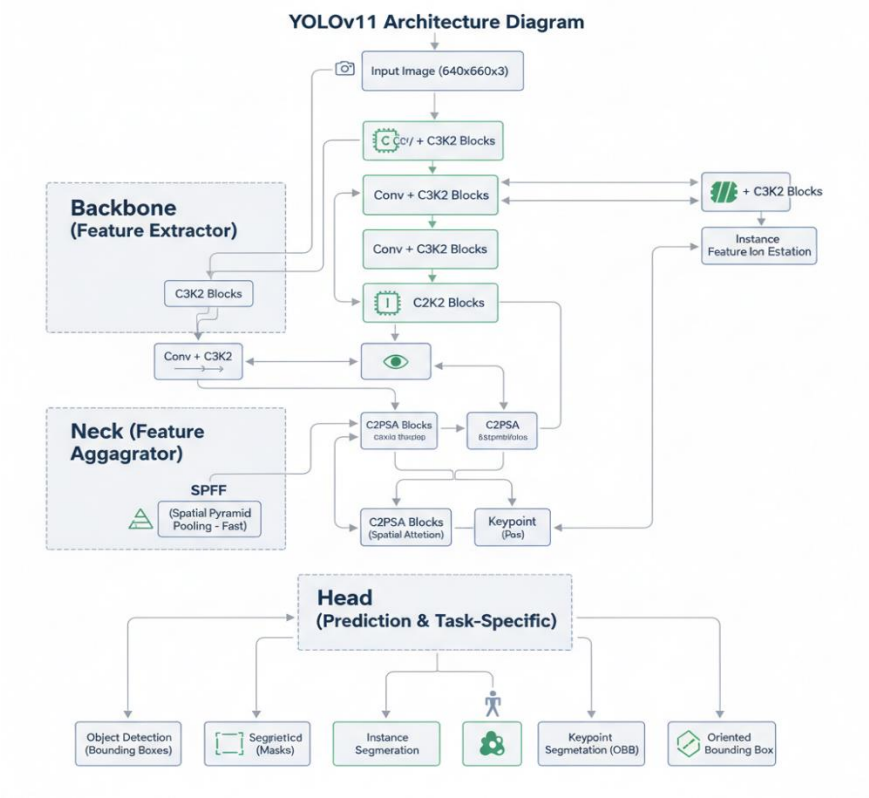
Đángchú ý trong số này là ước tính tư thế và phân đoạn trường hợp, mở rộng khả năng ứng dụng của mô hình trong nhiều lĩnh vực khác nhau. Thiết kế củaYOLO11 [8] [3] tập trung vào việc cân bằng sức mạnh và tính thực tiễn, nhằm giải quyết các thách thức cụ thể trong nhiều ngành công nghiệp khác nhau với độ chính xác và hiệu quả cao hơn.

Mô hình mới nhất này chứng minh sự phát triển liên tục của công nghệ phát hiện đối tượng theo thời gian thực, mở rộng ranh giới của những gì có thể trong các ứng dụng CV. Tính linh hoạt và cải tiến hiệu suất của nó định vị YOLO11 [8] [3] là một tiến bộ đáng kể trong lĩnh vực này, có khả năng mở ranhững con đường mới cho việc triển khai thực tế trên nhiều lĩnh vực khác nhau

Dựa trên kiến trúc đã được thiết lập này, YOLO11 [8] [3] mở rộng và nâng cao nền tảng được đặt ra bởi YOLOv8, giới thiệu những cải tiến về kiến trúc và

tối ưu hóa tham số để đạt được hiệu suất phát hiện vượt trội như minh họa trong hình dưới đây.

Các phần sau đây trình bày chi tiết các sửa đổi kiến trúc chính được triển khai trong YOLO11 [8]:



Hình 2.2.Các mô-đun chính trong YOLO11

2.2.2.1. Backbone — trích xuất đặc trưng

Mục tiêu của module này là trích xuất biểu diễn đa tỉ lệ (multi-scale) chứa cả thông tin cấu trúc (edges, corners) và ngữ nghĩa (khuôn mặt, mắt). YOLO11 giữ cấu trúc backbone truyền thống nhưng thay thế/tiếp thêm các khối có attention để cải thiện bắt đặc trưng nhỏ.

Khối SPFF và C2PSA:

YOLO11 vẫn giữ nguyên khối Spatial Pyramid Pooling - Fast (SPPF) từ các phiên bản trước nhưng giới thiệu một khối Cross Stage Partial with

Spatial Attention (C2PSA) mới. Khối C2PSA là một bổ sung đáng chú ý giúp tăng cường sự chú ý không gian trong các bản đồ đặc điểm. Cơ chếchú ý không gian

này cho phép mô hình tập trung hiệu quả hơn vào các vùng quan trọng trong hình ảnh. Bằng cách gộp các đặc điểm theo không gian,khối C2PSA cho phép YOLO [9]11 tập trung vào các khu vực quan tâm cụ thể, có khả năng cải thiện độ chính xác phát hiện đối với các đối tượng có kích thước và vị trí khác nhau.

C2PSA (Cross Stage Partial with Spatial Attention) là thành phần quan trọng nhất trong Backbone của YOLO11, đóng vai trò tăng cường khả năng biểu diễn đặc trưng đa không gian. Khối này bao gồm hai nhánh xử lý song song theo cơ chế CSP và một nhánh chú ý không gian PSA.

Nhánh trích xuất đặc trưng (Feature Extraction Branch):

Nhánh này tiếp nhận một phần của tensor đầu vào và thực hiện chuỗi các lớp tích chập 1×1 và 3×3.

- Tích chập 1×1 có nhiệm vụ điều chỉnh số kênh đặc trưng.
- Các tích chập 3×3 được sử dụng để học đặc trưng không gian sâu hơn. Sự kết hợp này cho phép khối học được các mẫu phức tạp trong ảnh như đường viền khuôn mặt, cấu trúc mắt và vùng chuyển động nhỏ.

Nhánh chú ý không gian (Spatial Attention Branch – PSA):

Nhánh PSA xử lý phần còn lại của đặc trưng đầu vào và tạo ra bản đồ chú ý nhằm làm nổi bật các vùng quan trọng trong ảnh. Quá trình này bao gồm các bước:

- Giảm số kênh đầu vào bằng tích chập 1×1 để tối ưu chi phí tính toán;
- Trích xuất ngữ cảnh không gian bằng các phép pooling kích thước 5×5 cùng với tích chập giãn cách (dilated convolution).
- Tổng hợp thông tin từ nhiều nguồn đặc trưng để tạo ra bản đồ chú ý không gian.
- Chuẩn hóa bản đồ chú ý bằng hàm kích hoạt sigmoid và nhân trực tiếp vào tensor đặc trưng.

Nhánh PSA giúp mô hình tập trung vào các vùng đặc biệt quan trọng như mắt, miệng và đầu người – những chi tiết rất nhỏ nhưng đóng vai trò then chốt trong bài toán phát hiện trạng thái buồn ngủ.

Hợp nhất hai nhánh:

Sau khi hai nhánh xử lý xong, tensor đặc trưng được ghép nối và đưa qua một lớp tích chập 1×1 để kết hợp thông tin và ổn định số kênh. Việc hợp nhất này giúp mô hình vừa giữ được đặc trưng chi tiết cục bộ vừa tập trung được vào các vùng quan trọng nhờ cơ chế chú ý.

Các lớp giảm kích thước đặc trưng:

Giữa các tầng backbone, YOLO11 sử dụng các lớp tích chập 3×3 với bước nhảy 2 để giảm kích thước đặc trưng theo từng mức. Việc giảm kích thước này giúp mô hình trích xuất được các đặc trưng ngữ nghĩa ở mức cao hơn, đồng thời giảm chi phí tính toán ở các tầng sâu.

Khối SPPF – Spatial Pyramid Pooling Fast:

SPPF được đặt tại cuối Backbone nhằm mở rộng vùng cảm thụ (receptive field) của mô hình. Khối này sử dụng kỹ thuật stacking các lớp pooling 5×5 liên tiếp để mô phỏng pooling 9×9 và 13×13 mà không cần sử dụng kernel lớn.

Cấu trúc SPPF bao gồm:

- Một lớp tích chập 1×1 để nén thông tin ban đầu. ba tầng pooling nối tiếp nhau.
- Ghép nối kết quả của bốn tầng đặc trưng;
- Một lớp tích chập 1×1 cuối để hợp nhất thông tin.

Khối SPPF giúp Backbone hiểu được ngữ cảnh rộng của khuôn mặt và hình thể, đặc biệt hiệu quả khi đối tượng có nhiều biến dạng về tư thế.

2.2.2.2 Cấu tạo nội bộ của Neck

Neck của YOLO11 đảm nhiệm nhiệm vụ tổng hợp đặc trưng từ nhiều mức độ khác nhau để tăng cường khả năng phát hiện vật thể nhỏ, trung bình và lớn. Cấu trúc Neck bao gồm hai luồng chính: FPN (Feature Pyramid Network) và PAN (Path Aggregation Network), cùng với khối C3k2 được tối ưu hóa.

Luồng FPN (top–down):

Luồng FPN truyền thông tin từ tầng đặc trưng sâu xuống các tầng nông hơn. Quá trình này bao gồm:

- Phóng to đặc trưng bằng upsampling;
- Ghép nối với đặc trưng có cùng kích thước từ Backbone;
- Trộn đặc trưng bằng khối C3k2.

Nhờ cơ chế này, các tầng có độ phân giải cao hơn được bổ sung thông tin ngữ nghĩa, giúp phát hiện các đối tượng nhỏ (như mắt hoặc mí mắt) hiệu quả hơn.

Luồng PAN (bottom–up) :

Luồng PAN thực hiện chiều ngược lại:

- Giảm kích thước đặc trưng bằng lớp tích chập stride 2;
- Ghép nối với đặc trưng ở mức sâu hơn;
- Xử lý bằng khối C3k2.

PAN giúp tăng cường thông tin chi tiết cho các tầng sâu, cải thiện khả năng mô hình phân biệt giữa các trạng thái tương tự (như chớp mắt và ngủ gật).

Khối C3k2 :

YOLO11 giới thiệu một thay đổi đáng kể bằng cách thay thế khối C2f trong cổ bằng khối C3k2. Khối C3k2 được thiết kế để nhanh hơn và hiệu

quả hơn, nâng cao hiệu suất tổng thể của quy trình tổng hợp tính năng. Sau khi lấy mẫu và nối, cổ trong YOLO11 kết hợp khối cải tiến này, dẫn đến tốc độ và hiệu suất được nâng cao

Các đặc điểm chính của khối C3k2:

- Xử lý nhanh hơn: Việc sử dụng hai phép tích chập nhỏ hơn giúp giảm chi phí tính toán so với một phép tích chập lớn, giúp trích xuất tính năng nhanh hơn.
- Hiệu quả tham số: C3k2 là phiên bản nhỏ gọn hơn của nút thắt CSP, giúp kiến trúc hiệu quả hơn về số lượng tham số có thể đào tạo

2.2.2.3. Cấu tạo nội bộ của Detection Head

Detection Head là thành phần cuối của mô hình, chịu trách nhiệm dự đoán hộp giới hạn và nhãn lớp trên ba mức đặc trưng đầu ra

Kiến trúc anchor-free :

Khác với các phiên bản YOLO cũ sử dụng anchor cố định, YOLO11 dự đoán trực tiếp tọa độ hộp bao thông qua các điểm đặc trưng (feature points). Điều này giúp mô hình:

- Giảm lỗi dự đoán do anchor không phù hợp;
- Tăng độ chính xác đối với các đối tượng nhỏ;
- Đơn giản hóa quá trình huấn luyện và tính toán.

Các đầu ra của một điểm đặc trưng:

Mỗi điểm đặc trưng dự đoán các giá trị sau:

- Vị trí tương đối của tâm hộp;
- Chiều rộng và chiều cao của hộp;
- Độ tin cậy (objectness);
- Xác suất lớp;

- Giá trị chất lượng (quality) nhằm cải thiện quá trình NMS.

Detection Head xử lý đặc trưng qua một số lớp tích chập nhẹ, sau đó chia thành các nhánh dự đoán riêng biệt.

Cơ chế Loss và tối ưu hóa nội bộ :

YOLO11 sử dụng tổ hợp các hàm Loss nâng cao:

- SIoU, EIoU hoặc CIoU cho dự đoán hộp bao;
- Varifocal Loss để khớp nhãn mềm và giá trị chất lượng;
- Distribution Focal Loss giúp dự đoán biên hộp chính xác hơn.

Những hàm mất mát này giúp YOLO11 đạt độ chính xác cao hơn, đặc biệt ở các đối tượng nhỏ hoặc bị che phủ một phần.

Tổng kết cấu tạo nội bộ YOLO11:

Cấu trúc nội bộ của YOLO11 được tổ chức theo hướng module hóa, trong đó:

- Backbone chịu trách nhiệm trích xuất đặc trưng đa cấp;
- C2PSA giúp mô hình tập trung vào các vùng quan trọng nhờ cơ chế chú ý không gian;
- SPPF mở rộng vùng nhận thông tin trong khi vẫn giữ tốc độ;
- C3k2 trong Neck tối ưu quá trình tổng hợp đa đặc trưng;
- Detection Head anchor-free đơn giản hóa dự đoán và nâng cao hiệu quả.
- Các hàm Loss hiện đại giúp mô hình hội tụ nhanh và ổn định.

Nhờ kiến trúc tối ưu và chặt chẽ này, YOLO11 trở thành một trong những mô hình hiệu quả nhất trong họ YOLO, đặc biệt phù hợp cho các bài toán nhận dạng thời gian thực như phát hiện tài xế ngủ gật.

2.2.3.Những tính năng chính của YOLO11

Dưới đây là những chức năng chính của YOLO11 [8]:

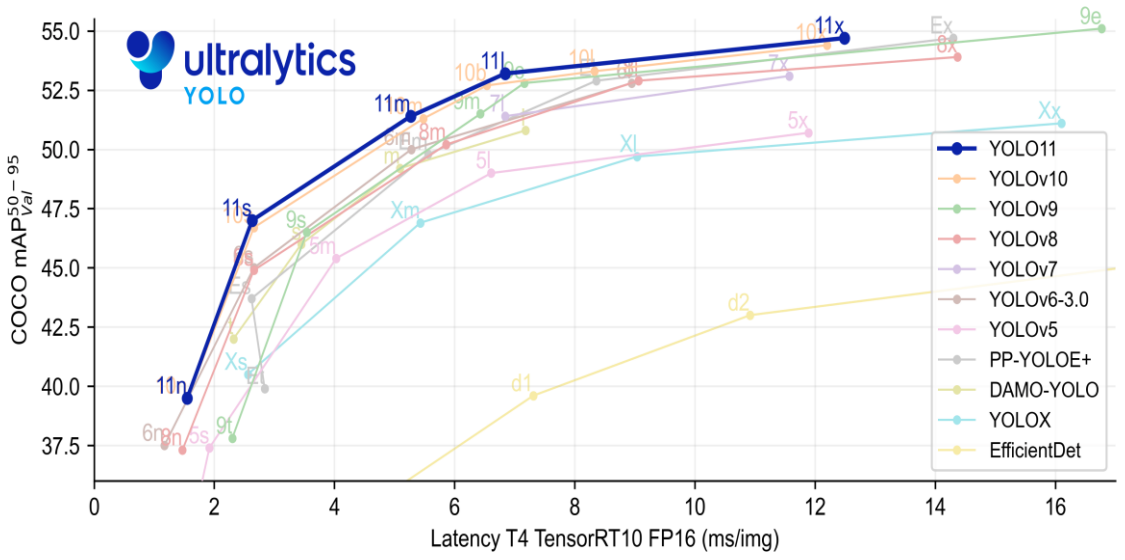
1. Độ chính xác được cải thiện với độ phức tạp được giảm bớt: Biến thể YOLO11 [8] [3] đạt được điểm Độ chính xác trungbình (mAP) vượt trội trên tập dữ liệu COCO trong khi sử dụng ít hơn 22% tham số so với biến thể YOLOv8m, chứng minh hiệuquả tính toán được cải thiện mà không ảnh hưởng đến độ chính xác .
2. Tính linh hoạt trong các tác vụ CV: YOLO11 [8] [3] thể hiện sự thành thạo trong nhiều ứng dụng CV khác nhau, bao gồmước tính tư thế, nhận dạng đối tượng, phân loại hình ảnh, phân đoạn thể hiện và phát hiện hộp giới hạn định hướng (OBB).
3. Tốc độ và hiệu suất được tối ưu hóa: Thông qua các thiết kế kiến trúc tinh tế và quy trình đào tạo hợp lý, YOLO11 [8] [3] đạtđược tốc độ xử lý nhanh hơn trong khi vẫn duy trì sự cân bằng giữa độ chính xác và hiệu quả tính toán.
4. Số lượng tham số được sắp xếp hợp lý: Việc giảm các tham số góp phần làm cho hiệu suất mô hình nhanh hơn mà không ảnh hưởng đáng kể đến độ chính xác tổng thể của YOLO11 [8] [3].
5. Trích xuất tính năng nâng cao: YOLO11 [8] [3] kết hợp các cải tiến trong cả kiến trúc xương sống và cổ, mang lại khả năngtrích xuất tính năng nâng cao và do đó, phát hiện đối tượng chính xác hơn.
6. Khả năng thích ứng theo ngữ cảnh: YOLO11 [8] chứng minh tính linh hoạt trong nhiều tình huống triển khai khác nhau,bao gồm nền tảng đám mây, thiết bị biên và hệ thống được tối ưu hóa cho GPU NVIDIA.

2.3. Điểm mạnh và điểm yếu của YOLO11

Điểm mạnh của YOLO11

- Cân bằng hiệu suất vượt trội: Đạt được sự cân bằng vượt trội giữa tốc độ và độ chính xác, vượt trội hơn các mô hình khác trên nhiều nền tảng phần cứng khác nhau.

- Tính Linh hoạt Vượt trội: Một họ mô hình duy nhất xử lý năm tác vụ AI thị giác chính, đơn giản hóa quá trình phát triển cho các ứng dụng phức tạp.
- Hệ sinh thái được duy trì tốt: Được hỗ trợ bởi sự phát triển tích cực, một cộng đồng lớn mạnh, cập nhật thường xuyên và các tài nguyên toàn diện, đảm bảo độ tin cậy và hỗ trợ.
- Dễ sử dụng: Được thiết kế cho trải nghiệm người dùng được tối ưu hóa, cho phép cả người mới bắt đầu và chuyên gia huấn luyện và triển khai mô hình với mức độ phức tạp tối thiểu.
- Hiệu quả đào tạo và triển khai: Được tối ưu hóa cho thời gian đào tạo nhanh hơn và sử dụng bộ nhớ thấp hơn, lý tưởng cho nhiều loại phần cứng từ thiết bị biên đến máy chủ đám mây.



Hình 2.3. YOLO11 so với các phiên bản trước

Điểm Yếu của YOLO11

- Là một mô hình hiện đại, các biến thể YOLO11 [8] lớn nhất đòi hỏi tài nguyên tính toán đáng kể để đạt được độ chính xác tối đa, mặc dù chúng vẫn rất hiệu quả so với lớp hiệu suất của chúng.
- Khó khăn với vật thể cực nhỏ hoặc cảnh đông đúc
- Yêu cầu phần cứng vẫn tương đối cao với biến thể lớn

- Triển khai cụ thể môi trường/edge có thể gặp trục trặc
- Yêu cầu dữ liệu tốt và huấn luyện kỹ lưỡng
- Không phải lúc nào phiên bản nhỏ nhất là nhanh nhất

2.4. Ứng dụng của YOLO11 vào giải quyết bài toán phát hiện người lái xe ô tô sử dụng điện thoại di động khi lái xe

Mô hình phát hiện người lái xe ô tô sử dụng điện thoại di động khi lái xe bằng YOLO11 [8] là một ứng dụng tiên tiến trong lĩnh vực nhận dạng hành vi lái xe nguy hiểm. Được xây dựng dựa trên YOLO11 [8], phiên bản mới nhất trong dòng mô hình You Only Look Once, mô hình này có khả năng phát hiện chính xác và nhanh chóng hành vi sử dụng điện thoại của tài xế trong thời gian thực.

YOLO11 là một mô hình phát hiện đối tượng mạnh mẽ, nổi bật với các đặc điểm sau:

1. Hiệu suất cao: Xử lý lên đến 60 khung hình mỗi giây (FPS), với độ chính xác mAP đạt 61,5% và số lượng tham số chỉ khoảng 40 triệu, giúp giảm độ trễ và tăng tốc độ xử lý so với các phiên bản trước như YOLOv10
2. Kiến trúc cải tiến: Sử dụng Transformer Backbone, Dynamic Head Design, và các mô-đun như PSA và Dual Label Assignment để nâng cao khả năng phát hiện đối tượng nhỏ, bị che khuất hoặc trong môi trường phức tạp
3. Tối ưu hóa cho thiết bị di động và nhúng: Kích thước mô hình nhỏ gọn (khoảng 2,83 MB sau khi lượng tử hóa), phù hợp cho các ứng dụng trên thiết bị di động và các hệ thống nhúng như điện thoại thông minh và máy tính bảng

Mô hình YOLO11 có thể được huấn luyện để nhận diện hành vi sử dụng điện thoại của tài xế thông qua các bước sau:

1. Thu thập và gán nhãn dữ liệu: Sử dụng các bộ dữ liệu như Roboflow hoặc tạo bộ dữ liệu riêng với các hình ảnh có nhãn "mobile use" để huấn luyện mô hình.

2. Huấn luyện mô hình: Sử dụng các công cụ như Ultralytics YOLO để huấn luyện mô hình với dữ liệu đã chuẩn bị, điều chỉnh số epoch và kích thước ảnh phù hợp.
3. Tích hợp và triển khai: Triển khai mô hình đã huấn luyện lên các hệ thống giám sát giao thông hoặc thiết bị di động để phát hiện hành vi sử dụng điện thoại trong thời gian thực.

CHƯƠNG 3: CÀI ĐẶT, THỬ NGHIỆM VÀ ĐÁNH GIÁ

Chương này trình bày quá trình triển khai mô hình phát hiện hành vi sử dụng điện thoại khi lái xe, bao gồm môi trường cài đặt, cấu hình mô hình, kết quả thử nghiệm và đánh giá hiệu năng. Trước hết, chương mô tả các công cụ và thư viện được sử dụng như Python và YOLO. Tiếp theo, quy trình huấn luyện, kiểm thử và tham số mô hình được trình bày rõ ràng. Cuối cùng, chương đưa ra kết quả thực nghiệm thông qua các chỉ số như độ chính xác (Accuracy), độ nhạy (Recall), độ chính xác dự đoán (Precision), mAP và thời gian xử lý, đồng thời phân tích những điểm mạnh, hạn chế và khả năng ứng dụng thực tế của mô hình**.**

3.1.Môi trường thử nghiệm

3.1.1. Python

3.1.1.1. Khái niệm

Python [5] là một ngôn ngữ lập trình bậc cao cho các mục đích lập trình đanăng, do Guido van Rossum tạo ra và lần đầu ra mắt vào năm 1991. Python được thiết kế với ưu điểm mạnh là dễ đọc, dễ học và dễ nhớ. Python [5] là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình và là ngôn ngữ lập trình dễ học, được dùng rộng rãi trong phát triển trí tuệ nhân tạo. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu. Vào tháng 7 năm 2018, Van Rossum đã từ chức lãnh đạo trong cộng đồng ngôn ngữ Python sau 30 năm làm việc.

Python [5] hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động; do vậy nó tương tự như Perl, Ruby, Scheme, Smalltalk, và Tcl. Python được phát triển trong một dự án mã mở, do tổ chức phi lợi nhuận Python Software Foundation quản lý.

Ban đầu, Python [5] được phát triển để chạy trên nền Unix. Nhưng rồi theo thời gian, Python dần mở rộng sang mọi hệ điều hành từ MS-DOS đến Mac OS, OS/2, Windows, Linux và các hệ điều hành khác thuộc họ Unix. Mặc dù sự phát triển của Python có sự đóng góp của rất nhiều cá nhân, nhưng Guido van

Rossum hiện nay vẫn là tác giả chủ yếu của Python. Ông giữ vai trò chủ chốt trong việc quyết định hướng phát triển của Python.

Python [5] luôn được xếp hạng vào những ngôn ngữ lập trình phổ biến nhất.

3.1.1.2. Đặc tính

Python [5] đang trở nên phổ biến trong cộng đồng lập trình nhờ các đặc tính sau:

- Ngôn ngữ thông dịch: Python [5] được xử lý trong thời gian chạy bởi trình thông dịch Python.
- Ngôn ngữ hướng đối tượng: Python [5] hỗ trợ các đặc trưng và kỹ thuật lập trình hướng đối tượng.
- Ngôn ngữ lập trình tương tác: Người dùng có thể tương tác trực tiếp với trình thông dịch python để viết chương trình.
- Ngôn ngữ dễ học: Python [5] rất dễ học, đặc biệt là cho người mới bắt đầu.
- Cú pháp đơn giản: Việc hình thành cú pháp Python [5] rất đơn giản và dễ hiểu, điều này cũng làm cho nó trở nên phổ biến.
- Mã nguồn mở: Python [5] cho phép người dùng có thể sử dụng, chỉnh sửa và phân phối mã nguồn một cách tự do.
- Di động: Mã Python [5] có thể chạy trên nhiều nền tảng phần cứng có cùng giao diện.
- Có thể mở rộng: Người dùng có thể thêm các mô-đun cấp thấp vào trình thông dịch Python [5].
- Có thể cải tiến: Python [5] cung cấp một cấu trúc cải tiến để hỗ trợ các chương trình lớn sau đó là shell-script.

Hỗ trợ cộng đồng: Python [5] có một cộng đồng lớn với nhiều người dùng và nhà phát triển trên khắp thế giới đóng góp vào việc phát triển cà cải tiến Python. Cộng đồng này cũng cung cấp hỗ trợ và tài liệu để giúp người dùng python giải quyết cá vấn đề một cách nhanh chóng và hiệu quả

3.1.1.3. Các phiên bản của Python

Python [5] đã được Guido van Rossum tạo ra vào những năm 1980 tại Trung tâm Toán học – Tin học (Centrum Wiskunde & Informatica, CWI) ở Hà Lan như là một ngôn ngữ kế tục ngôn ngữ ABC – một ngôn ngữ được lấy cảm hứng từ SETL (ngôn ngữ được phát triển bởi Jacob T. Schwartz và các đồng nghiệp), có khả năng xử lý ngoại lệ và giao tiếp với hệ điều hành Amoeba. Nó bắt đầu được triển khai vào tháng 12 năm 1989.

Python [5] 2.0 được ra mắt vào ngày 16 tháng 10 năm 2000, với nhiều đặc trưng mới mẻ, bao gồm một bộ dọn rác phát hiện theo chu kỳ và khả năng hỗ trợ Unicode.

Python [5] 3.0 được ra mắt vào ngày mùng 3 tháng 12 năm 2008. Đây là một phiên bản lớn của Python không tương thích ngược hoàn toàn. Nhiều đặc trưng lớn của nó đã được chuyển mã ngược (backport) về loạt phiên bản Python 2.6.x và 2.7.x. Các bản phát hành của Python 3 có đi kèm với công cụ 2to3, có tác dụng tự động hoá việc dịch mã Python 2 sang Python 3.

Python [5] 3.9.2 và 3.8.8 được xúc tiến vì tất cả các phiên bản trước của Python (bao gồm cả 2.7) gặp một số vấn đề bảo mật, có thể dẫn đến thực thị mã từ xa và "đầu độc" bộ nhớ đệm.

Trong năm 2022, Python [5] 3.10.4 và 3.9.12 được xúc tiến cùng với 3.8.13 và 3.7.13, nguyên nhân là do một vài vấn đề về bảo mật. Khi Python 3.9.13 được phát hành vào tháng Năm năm 2022, loạt phiên bản 3.9 (cùng với loạt 3.8 và 3.7) được thông báo rằng sẽ chỉ nhận được các bản vá bảo mật trong tương lai. Vào ngày 7 tháng Chín năm 2022, bốn bản cập nhật mới được phát hành do có khả năng xảy ra một cuộc tấn công từ chối dịch vụ: 3.10.7, 3.9.14, 3.8.14 và 3.7.14.

Tính đến tháng 10 năm 2023, Python 3.12 là bản phát hành ổn định mới nhất. Một số thay đổi đáng chú ý từ bản 3.11 bao gồm các thay đổi về ngôn ngữ và thư viện chuẩn.

3.1.2. Visual Studio Code

3.1.2.1. Khái niệm

Visual Studio Code [7] (VS Code) là một trình soạn thảo mã nguồn miễn phí, nhẹ và mạnh mẽ, phát triển bởi Microsoft, hoạt động trên đa nền tảng (Windows, macOS, Linux). Nó hỗ trợ nhiều ngôn ngữ lập trình và cung cấp các tính năng hữu ích như gỡ lỗi (debug), tô sáng cú pháp (syntax highlighting), tự động hoàn thành mã, và tích hợp Git. VS Code được ưa chuộng vì khả năng tùy biến cao và kho tiện ích mở rộng phong phú, phù hợp cho cả người mới bắt đầu và lập trình viên chuyên nghiệp. Nó là đa nền tảng, hoạt động trên Microsoft Windows, macOS và Linux.

3.1.2.2. Các đặc trưng

Visual Studio Code [7] là gì được rất nhiều người tìm hiểu. Đây cũng là một trong các ứng dụng được dân IT "săn đón" và tải về và sử dụng rất nhiều. Visual Studio Code cũng luôn có những cải tiến và tạo ra đa dạng các tiện ích đi kèm từ đó giúp cho các lập trình viên sử dụng dễ dàng hơn. Trong đó có thể kể đến những ưu điểm sau:

Đa dạng ngôn ngữ lập trình giúp người dùng thỏa sức sáng tạo và sử dụng như HTML, CSS, JavaScript, C++,…

- Ngôn ngữ, giao diện tối giản, thân thiện, giúp các lập trình viên dễ dàng định hình nội dung.
- Các tiện ích mở rộng rất đa dạng và phong phú.

Không phải ngẫu nhiên mà Visual Studio Code được các lập trình viên ưa chuộng sử dụng. Visual Studio Code mang rất nhiều ưu điểm vượt trội so với bất kỳ IDE nào khác:

- Hỗ trợ đa nền tảng: Linux, Mac, Windows,..
- Hỗ trợ đa ngôn ngữ: C/C++, C#, F#, JavaScript, JSON, Visual Basic,HTML,CSS,…
- Ít dung lượng.

- Tính năng mạnh mẽ.
- Intellisense chuyên nghiệp.
- Giao diện thân thiện
- Kiến trúc mạnh mẽ và người dùng có thể khai thác mở rộng
- Số lượng người sử dụng lớn tạo nên ộng đồng hỗ trợ rộng rãi

3.1.3. Google Collab

3.1.3.1. Khái niệm

Colaboratory hay còn gọi là Google Colab [6], là một sản phẩm từ Google Re search, nó cho phép chạy các dòng code python thông qua trình duyệt, đặc biệt phù hợp với Data analysis, machine learning và giáo dục. Google Colab [6] không cần yêu cầu cài đặt hay cấu hình máy tính, mọi thứ có thể chạy thông qua trình duyệt, bạn có thể sử dụng tài nguyên máy tính từ CPU tốc độ cao và cả GPUs và cả TPUs đều được cung cấp cho bạn.

Google Colab [6] cung cấp nhiều loại GPU, thường là Nvidia K80s, T4s, P4s and P100s, tuy nhiên người dùng không thể chọn loại GPU trong Colab, GPU trong Colab thay đổi theo thời gian. Vì là dịch vụ miễn phí, nên Colab sẽ có những thứ tự ưu tiên trong việc sử dụng tài nguyên hệ thống, cũng như giới hạn thời gian sử dụng, thời gian sử dụng tối đa lên tới 12 giờ.

3.1.3.2. Những đặc trưng của Google Colab

- Sử dụng Jupyter Notebooks trực tuyến.

Google Colab [6] Google Colab [6] cho phép tạo và chạy các Jupyter Notebooks trực tuyến mà không cần cài đặt môi trường phát triển phức tạp trên máy tính cá nhân.

Giao diện sử dụng tương tự như Jupyter Notebook truyền thống với các cell cho phép thực thi mã Python hoặc viết markdown để tạo nội dung hướng dẫn.

- Khả năng chia sẻ và cộng tác.

Người dùng có thể chia sẻ notebook với những người khác để cùng làm việc trên cùng một notebook, tạo điều kiện thuận lợi cho việc học tập và làm việc nhóm.

Các đặc trưng như bình luận và chế độ chỉnh sửa đồng thời giúp tăng tính tương tác và hiệu quả của quá trình cộng tác.

- Dùng GPU và TPU miễn phí.

Google Colab [6] cung cấp truy cập miễn phí đến GPU và TPU, đặc biệt hữu ích cho các tác vụ tính toán nặng về mặt số học, đặc biệt là trong lĩnh vực học máy và deep learning.

Việc sử dụng các card GPU hoặc TPU có sẵn giúp tăng tốc độ xử lý và huấn luyện mô hình so với việc sử dụng CPU thông thường.

- Lưu trữ dữ liệu trên Google Drive và tích hợp Google Cloud.

Người dùng có thể truy cập và lưu trữ dữ liệu trực tiếp từ Google Drive, tạo điều kiện thuận lợi cho việc làm việc với tập tin dữ liệu lớn.

Tích hợp với điện toán đám mây của Google cũng cho phép sử dụng các dịch vụ như BigQuery, Cloud Storage, và các API khác từ dịch vụ điện toán đám mây trong quá trình làm việc.

3.2. Tạo bộ dữ liệu học để huấn luyện

Roboflow [9] là một framework hỗ trợ các nhà phát triển trong lĩnh vực thị giác máy tính [2] (Computer Vision) thực hiện việc thu thập, xử lý và quản lý dữ liệu hình ảnh một cách hiệu quả. Công cụ này giúp người dùng tạo ra bộ dữ liệu được tổ chức khoa học, thuận tiện cho quá trình huấn luyện mô hình học máy (Machine Learning) và học sâu (Deep Learning [10]).

Roboflow [9] cung cấp sẵn nhiều tập dữ liệu công khai (public datasets) mà người dùng có thể sử dụng trực tiếp, đồng thời cho phép tải lên và tùy chỉnh dữ liệu riêng của mình. Người dùng có thể xử lý dữ liệu theo nhu cầu cá nhân như

điều chỉnh kích thước ảnh, thay đổi định dạng, tăng cường dữ liệu (data augmentation) hoặc lọc bỏ nhiễu để nâng cao chất lượng tập dữ liệu đầu vào.

Để xây dựng một mô hình phát hiện hành vi sử dụng điện thoại khi lái xe, trước tiên cần tạo ra một bộ dữ liệu huấn luyện đủ lớn và đa dạng. Quy trình tạo bộ dữ liệu được thực hiện qua nhiều bước nhằm đảm bảo chất lượng và tính chính xác của dữ liệu.

Dữ liệu được thu thập từ nhiều nguồn khác nhau. Một phần hình ảnh được tự chụp trong môi trường xe ô tô thật với nhiều góc độ và điều kiện ánh sáng khác nhau nhằm tăng tính đa dạng. Một phần khác được thu thập từ internet với các từ khóa liên quan đến hành vi sử dụng điện thoại khi lái xe.

Sau khi hoàn thiện dữ liệu, Roboflow [8] hỗ trợ xuất dữ liệu ở nhiều định dạng chuẩn tương thích với các thư viện và framework phổ biến như TensorFlow, PyTorch, YOLO [9], EfficientDet, MobileNet, v.v.

Hiện nay, Roboflow [8] được ứng dụng rộng rãi trong nhiều lĩnh vực có liên quan đến thị giác máy tính [2], bao gồm:

- Nhận dạng vật thể trong ảnh hoặc video (object detection)
- Phát hiện cháy nổ, khói hoặc người xâm nhập bất hợp pháp
- Theo dõi phương tiện giao thông, quản lý bãi xe
- Dự đoán thiệt hại công trình, mái nhà hoặc cây trồng từ ảnh vệ tinh
- Hỗ trợ xe tự hành và robot trong việc nhận biết môi trường xung quanh

Với những ưu điểm đó, Roboflow [8] trở thành công cụ không thể thiếu trong việc xây dựng và quản lý bộ dữ liệu hình ảnh, góp phần nâng cao độ chính xác và hiệu quả của các mô hình học sâu.

3.3. Quá trình Huấn Luyện

3.3.1 Chuẩn bị ảnh

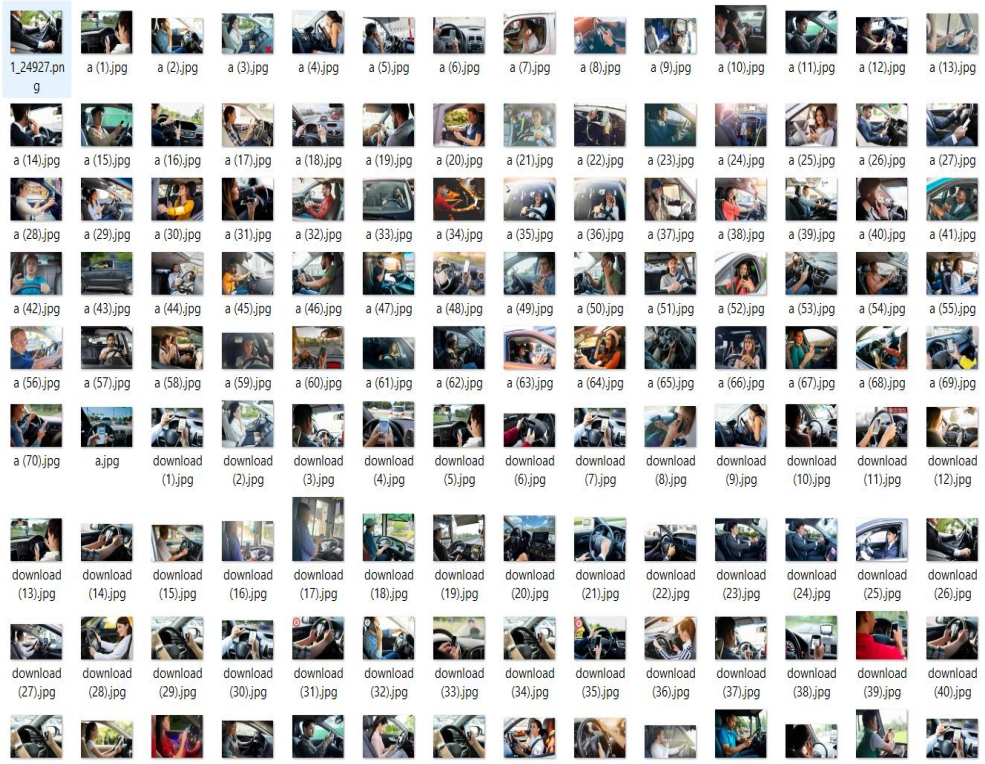
- Chất lượng Ảnh: Ảnh cần có độ phân giải đủ cao để các đối tượng có thể được nhận diện rõ ràng, ngay cả khi chúng nhỏ hoặc ở xa.
- Đa dạng: Bộ dữ liệu nên bao gồm ảnh với các điều kiện ánh sáng khác nhau, góc chụp, và cảnh quan để mô hình có thể học cách phát hiện đối tượng trong nhiều hoàn cảnh.
- Gán nhãn chính xác: Các bounding box cần được vẽ chính xác xung quanh đối tượng, không quá rộng hoặc quá hẹp so với kích thước thực tế của đối tượng.
- Cân bằng dữ liệu: Tránh tình trạng mất cân bằng, nơi một số lớp có nhiều mẫu hơn đáng kể so với các lớp khác, điều này có thể dẫn đến độ chệch trong quá trình huấn luyện.
- Chia tệp dữ liệu: Chia bộ dữ liệu thành các tập train, validation, và test để đánh giá mô hình một cách khách quan.
- Tối ưu hóa kích thước ảnh: Đôi khi cần phải điều chỉnh kích thước ảnh để phù hợp với yêu cầu đầu vào của mô hình, đồng thời giảm bớt gánh nặng tính toán.
- Augmentation: Sử dụng kỹ thuật tăng cường dữ liệu (data augmentation) như xoay, lật, thay đổi màu sắc, để tăng cường khả năng tổng quát hóa của mô hình
- Tổng số lượng ảnh: 216 ảnh

Bảng 3-1: Số Lượng Ảnh của từng đối tượng

Đối tượng	Số lượng ảnh
Người lái xe ô tô sử dụng điện thoại khi lái xe	166
Người lái xe ô tô không sử dụng điện thoại khi lái xe	50

- Tổng số lượng hình ảnh 216 ảnh bao gồm 46 ảnh lấy từ roboflow và 170 ảnh được tạo ra nhờ đoạn code tăng cường hình ảnh và một số ảnh

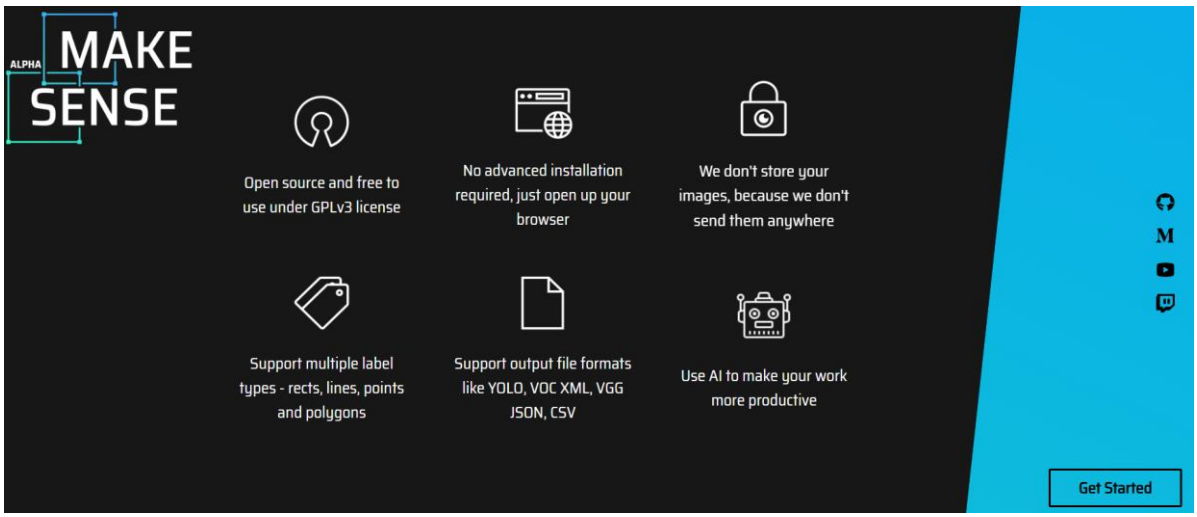
được thu thập từ việc chụp trực tiếp người lái xe ô tô sử dụng điện thoại bên ngoài. Dưới đây là ảnh minh họa địa diện cho bộ dữ liệu:



Hình 3.1: Ảnh minh họa đại diện cho bộ huấn luyện

3.3.2. Đánh nhãn cho đối tượng

Sử dụng công cụ có sẵn trên mạng là makesense để gán nhãn cho đối tượng, trong bài này đối tượng sẽ là tài xế lái xe (https://www.makesense.ai/).

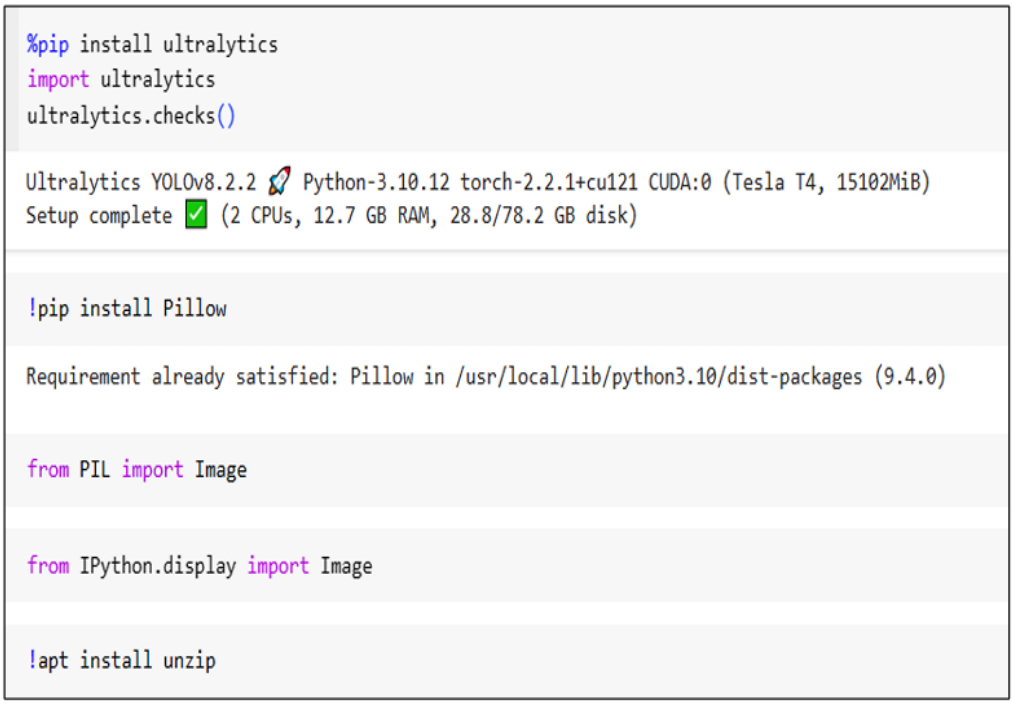


Hình2.2. Hình ảnh về Make sense

3.4. Đào tạo mô hình "Xây dựng mô hình phát hiện người lái xe ô tô có hành vi sử dụng điện thoại khi lái xe"

Để huấn luyện mô hình "Xây dựng mô hình phát hiện người lái xe ô tô có hành vi sử dụng điện thoại khi lái xe" bằng phương pháp học sâu thì em sử dụng một công cụ hỗ trợ ở đây là Google Colab. Google Colab là một dịch vụ máy tính đám mây của Google cho phép người dùng tạo ra các tệp note book Jupyter để thực thi mã Python. Nó cung cấp một môi trường tính toán đám mây miễn phí, với các đặc trưng như GPU miễn phí, RAM miễn phí đến 12GB có thể mở rộng đến 25.5GB nếu có trả phí, lưu trữ đám mây và nhiều đặc trưng khác.

Ngoài ta Google Colab còn liên kết với driver để có thể hộ trợ người dùng trong việc lưu trữ các dữ liệu quan trọng. Google Colab được thiết kế để hỗ trợ việc phát triển và huấn luyện các mô hình học máy và các ứng dụng AI. Nó cung cấp một môi trường tính toán đám mây miễn phí, mạnh mẽ và linh hoạt cho các nhu cầu học tập và nghiên cứu.



```
%pip install ultralytics
import ultralytics
ultralytics.checks()
```

```
Ultralytics YOLOv8.2.2 🚀 Python-3.10.12 torch-2.2.1+cu121 CUDA:0 (Tesla T4, 15102MiB)
Setup complete ✅ (2 CPUs, 12.7 GB RAM, 28.8/78.2 GB disk)
```

```
!pip install Pillow
```

```
Requirement already satisfied: Pillow in /usr/local/lib/python3.10/dist-packages (9.4.0)
```

```
from PIL import Image
```

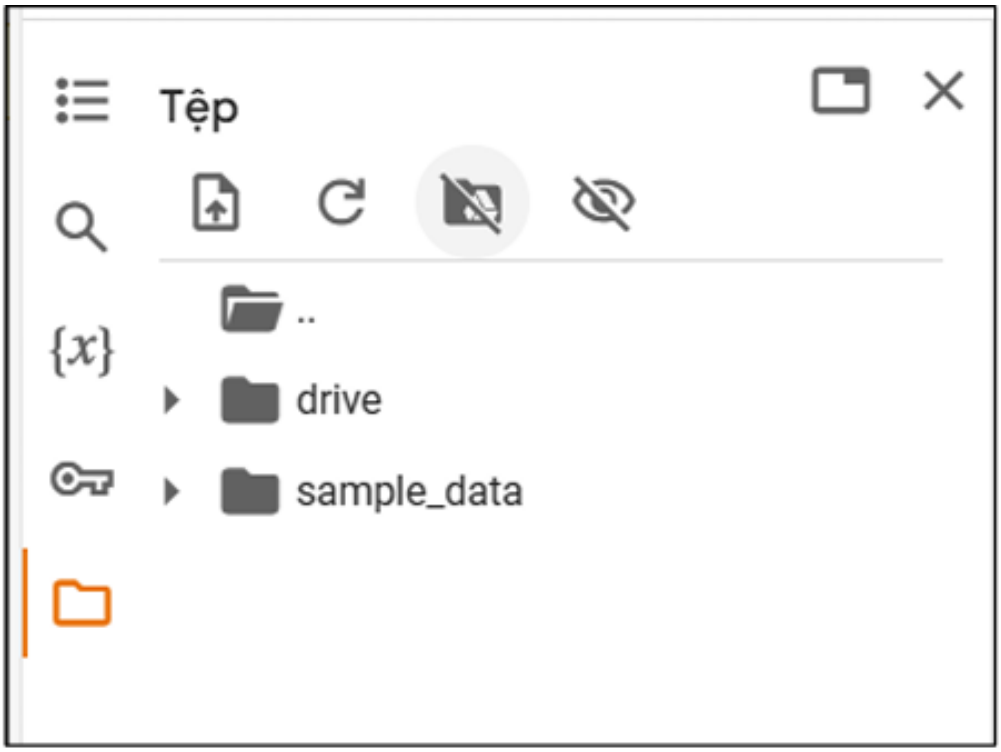
```
from IPython.display import Image
```

```
!apt install unzip
```

Hình 3.2: Cài các thư viện hỗ trợ

Vì Google Colab bản miến phí chỉ cho giới hạn về thời gian chạy nên ta có thể liên kết với drive để lưu trữ để tránh mất dữ liệu sau khi hết phiên.

Ta có thể dễ dàng kết nối với drive với 2 cách:



Hình 3.3: Ấn chọn biểu tượng drive trong phần tệp

```
from google.colab import drive
drive.mount('/content/drive')
```

Hình 3.4: Kết nối qua đoạn mã

Sau khi cài xong một vài đoạn mã ta có thể bắt đầu train với đoạn mã sau

```
%cd {HOME}

yolo task=detect mode=train model=yolov8n.pt data={dataset.location}/data.yaml
epochs=100 imgsz=640
```

Hình 3.5: Đoạn mã để bắt đầu train

Trong đó:

- **%cd**: là di chuyển đến tệp ta sẽ lưu trữ các dữ liệu khi đang train
- **Epochs**: Số lần ảnh được train (số càng lớn dữ liệu train sẽ được cải thiện nhưng đổi lại thì sẽ tốn thời gian hơn)

- **Imgsz**: size ảnh (đối với YOLOv8 nhà phát triển có yêu cầu size ảnh là 640x640).
- **Model**: mô hình được Ultralytics cúng cấp để đào tạo.
- **Data**: đường dẫn đến tệp cấu hình tập dữ liệu (ví dụ: coco8.yaml). Tệp này chứa các tham số dành riêng cho tập dữ liệu, bao gồm đường dẫn đến dữ liệu đào tạo và xác thực, tên lớp và số lượng lớp.

Ta cũng có một số một số thông số khác nhưng ta ta cũng không cần phải ghi hết tất cả để có thể chạy được mọi thông tin ta có thể tìm trên trang train của ultralytics. Ultralytics có đầy đủ các thông số nếu ai có yêu cầu chi tiết trong quá trình đào tạo.

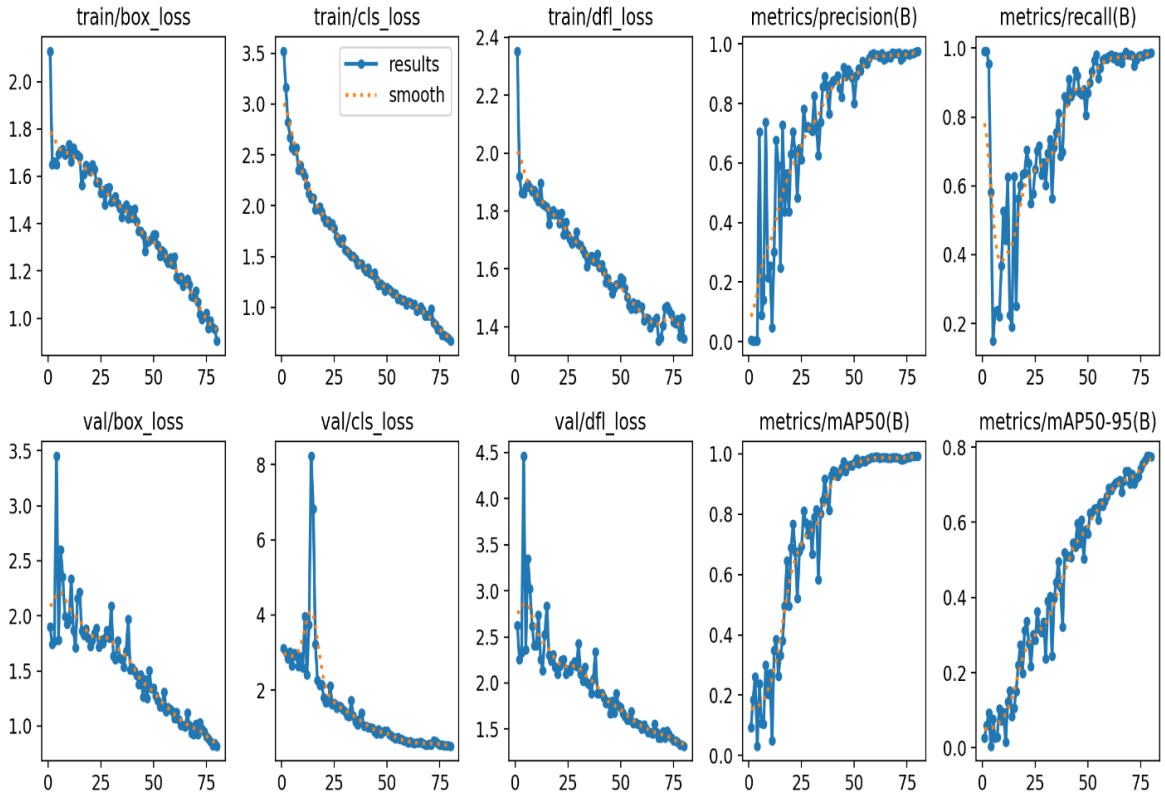
Với từng epochs ta có bảng sau:

Bảng 3-2: Bảng thử nghiệm với các epochs khác nhau

Epochs	ảnh kiểm thử (valid)	Thời gian thực hiện	P	R	mAP	mAP 50-95
80	216	7 phút	0.976	0.987	0.992	0.775
90	216	8 phút	0.983	0.983	0.991	0.805
100	216	9 phút	0.977	0.985	0.993	0.808

Quan sát (bảng 3-2) trên ta có thể nhận thấy kết quả thử nghiệm mô hình với các giá trị của epochs khác nhau nhưng không có nhiều sự chênh lệch. Tuy nhiên với 90 epochs ta được độ chính xác là 0.991 trong thời gian 8 phút khi chạy 20/100 epochs. Khi thay đổi các số epochs ta có thể thấy từ 80 có sự tăng nhẹ trong khi đó lên 100 một vài chỉ số đã bị giảm đi. Từ đó ta có thể kết luận với dữ liệu tầm 216 ảnh đối với mô hình này thì 90 epochs là sự lựa chọn tốt nhất.

3.4.1. Giá trị các biểu đồ

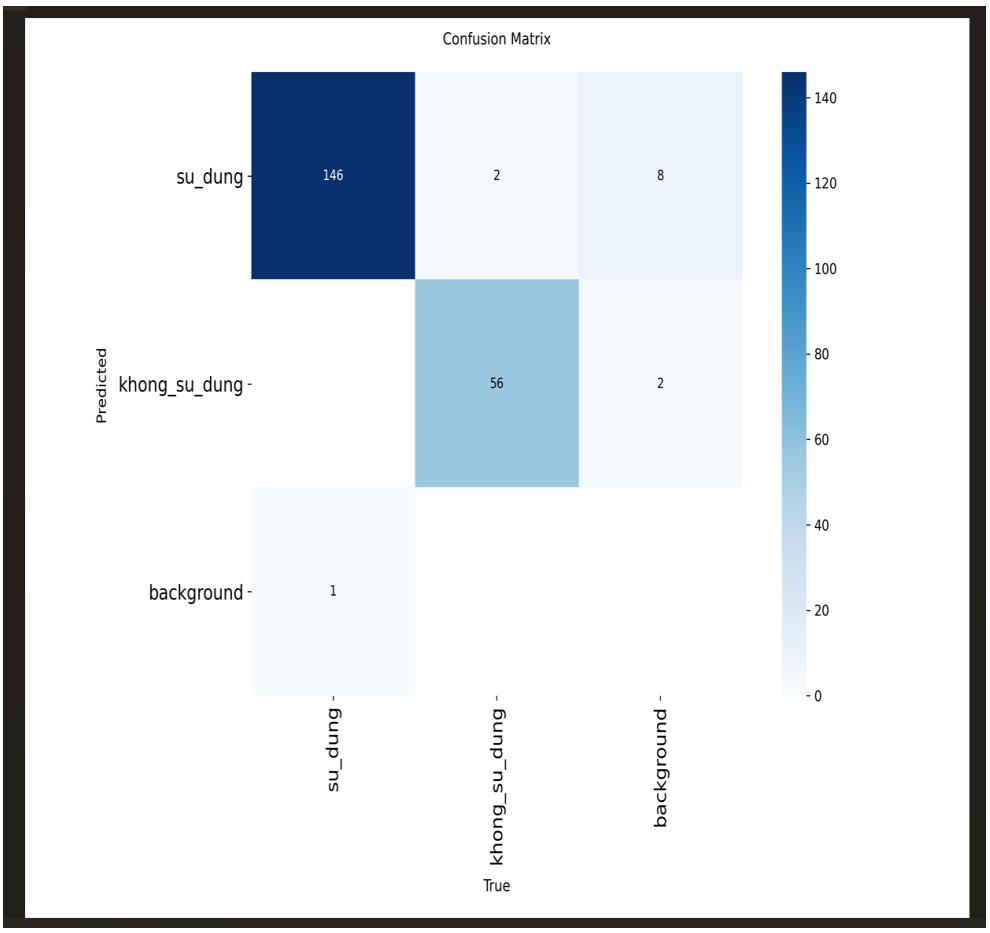


Hình 3.6: Biểu đồ thống kế sau khi đào tạo

Theo như hình ta có thể thấy:

Các biểu đồ mất mát đang giảm dần cho biết được mô hình đang được học tốt hơn.

Biểu đồ độ chính xác: thì lại tăng dần lên, và nhìn từ hình ảnh ta thấy được mô hình có sự tăng trưởng nhanh ở 50 epochs đầu tiên và chậm lại.



Hình 3.7: Ma trận nhầm lẫn (hiệu suất dự đoán của mô hình với epochs=100

Confusion được sử dụng để đánh giá hiệu suất của một mô hình phân loại trong lĩnh vực học máy. Nó thể hiện số lượng các dự đoán chính xác và không chính xác giữa các lớp khác nhau. Cụ thể:

- Các Lớp: Bảng này bao gồm các lớp như ảnh sử dụng điện thoại khi lái xe ô tô và ảnh không sử dụng điện thoại khi lái xe ô tô
- Giá Trị Ô: Các giá trị số trong mỗi ô của ma trận thể hiện số lượng trường hợp cho mỗi sự kết hợp của lớp thực tế và lớp dự đoán.
- Trục X và Y: Trục x với nhãn "True" biểu thị các lớp thực tế, trong khi trục y với nhãn "Predicted" biểu thị các lớp được mô hình dự đoán.

Confusion giúp nhận diện được những lớp mà mô hình phân loại hoạt động tốt và những lớp mà mô hình có thể cải thiện.

Dưới đây là đoạn mã xuất file sang tflite để chạy trên di động

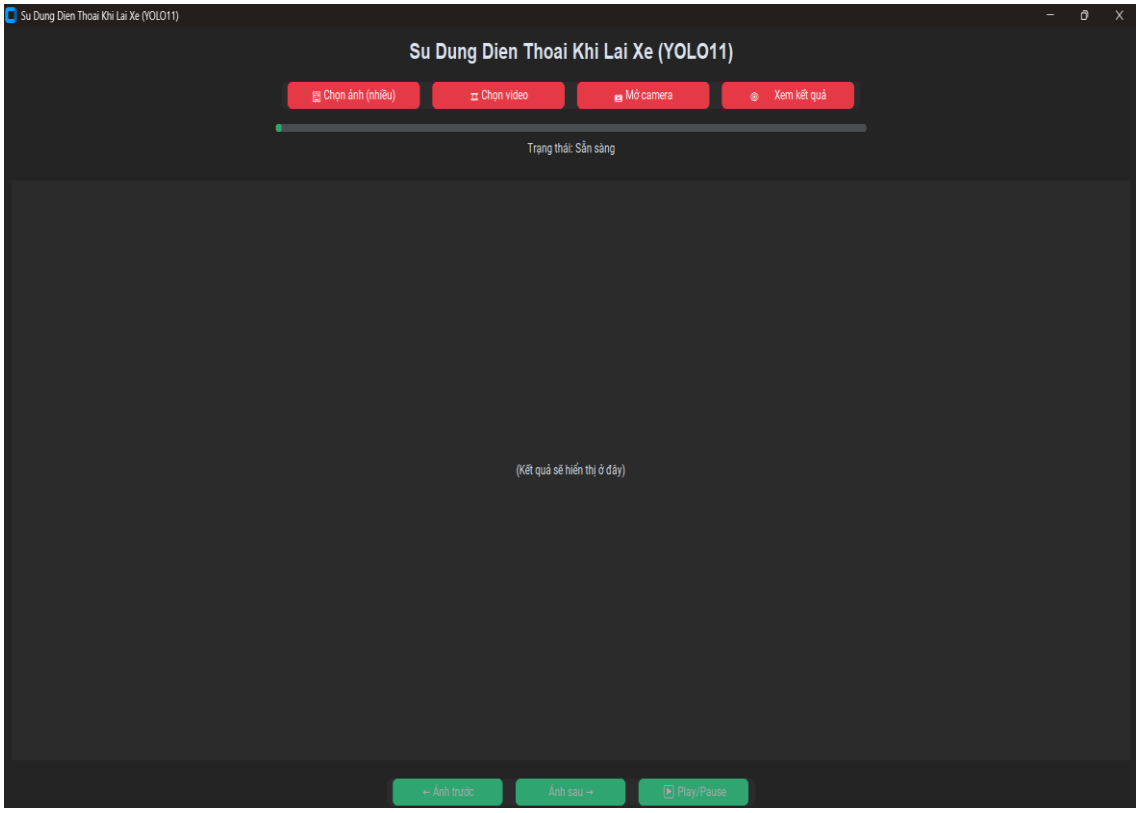
```
from ultralytics import YOLO

# Load mô hình PyTorch
model = YOLO("best.pt")

# Xuất sang định dạng TensorFlow Lite
model.export(format="tflite")
```

Hình 3.8: Đoạn mã xuất sang file tflite để chạy trên di động

3.4.2. Giao diện ứng dụng phát hiện người lái xe có hành vi sử dụng điện thoại khi lái xe



Hình 3.9: Giao diện chương trình

Giao diện có các chức năng cơ bản gồm:

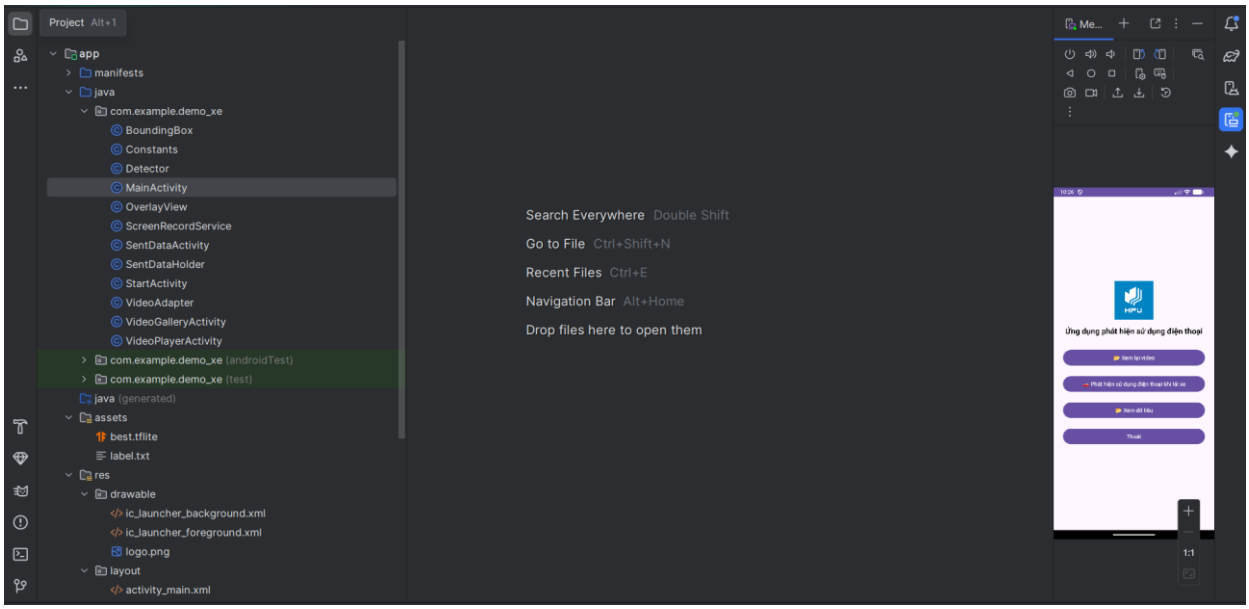
- Chọn ảnh: Dùng model đã train trên Google Colab để xử lý nhận diện ảnh và lưu ảnh về định dạng .JPG và lưu vào 1 tệp chỉ định.

- Chọn Video: Dùng model đã train trên Google Colab để xử lý nhận diện vật trái cây trong video và xuất ra định dạng .mp4 và lưu vào 1 tệp chỉ định.
- Mở camera: dùng webcam của máy để nhận diện đối tượng (sử dụng điện thoại)

3.4.3. Cài đặt mô hình trên thiết bị Android (Android Studio)

3.4.3.1.Giao diện chương trình khi đã được cài trên Android Studio

Giao diện trên Android Studio:



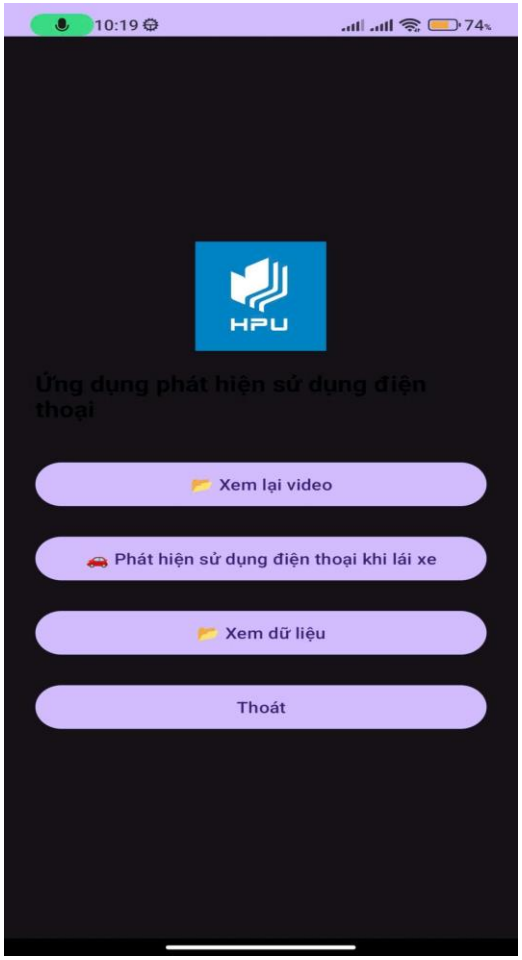
Hình 3.10: Giao diện mô hình trên Android Studio

Trong đó

- Xem lại video: Dùng model đã train trên Google Colab để xử lý nhận diện người lái xe có hành vi sử dụng điện thoại khi lái xe trong video và xuất ra định dạng .mp4 và lưu vào 1 tệp chỉ định.
- Phát hiện sử dụng điện thoại khi lái xe: dùng webcam của máy để nhận diện đối tượng (người lái xe có hành vi sửu dụng điện thoại)
- Xem dữ liệu : dùng để chọn ảnh, video để nhận diện và xem kết quả
- Thoát : thoát chương trình

3.4.3.2.Giao diện trên điện thoại di động Android

Dưới đây là giao diện trên thiết bị Android



Hình 3.11:Giao diện trên thiết bị di động

Trong đó:

- Xem lại video: Dùng model đã train trên Google Colab để xử lý nhận diện người lái xe có hành vi sử dụng điện thoại khi lái xe trong video và xuất ra định dạng .mp4 và lưu vào 1 tệp chỉ định.
- Phát hiện sử dụng điện thoại khi lái xe: dùng webcam của máy để nhận diện đối tượng (người lái xe có hành vi sửu dụng điện thoại)
- Xem dữ liệu : dùng để chọn ảnh, video để nhận diện và xem kết quả
- Thoát : thoát chương trình

3.4.3. Kết quả thử nghiệm trên mô hình đã huấn luyện

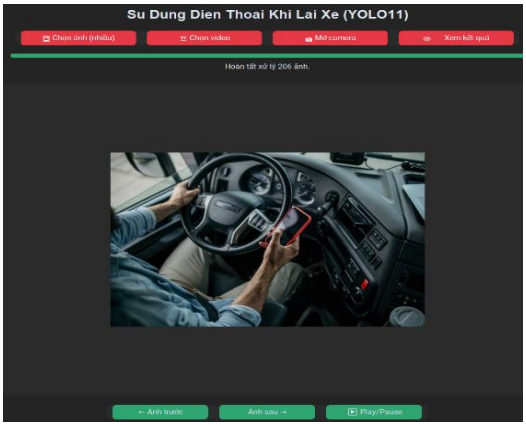
Thực hiện thủ công trên 3 loại dữ liệu đầu vào: ảnh sử dụng điện thoại, ảnh không sử dụng điện thoại và ảnh sử dụng điện thoại khi chạy trên di động

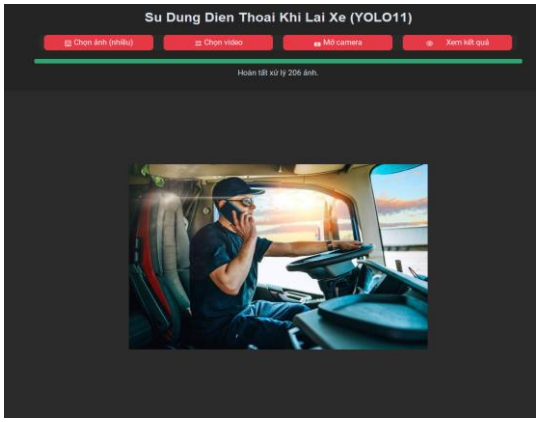
Đối với ảnh sử dụng điện thoại chương trình đem lại độ chính xác trong khoảng 95%.

Bảng 3-3: Bảng thử nghiệm phân loại hình ảnh sử dụng điện thoại

STT	Tên ảnh	Số lượng ảnh	Nhận đúng	Nhận sai	Độ chính xác
1	Ảnh sử dụng điện thoại	50	48	2	95%

Theo như bảng trên ra có thể thấy nhận sai là ở các ảnh:



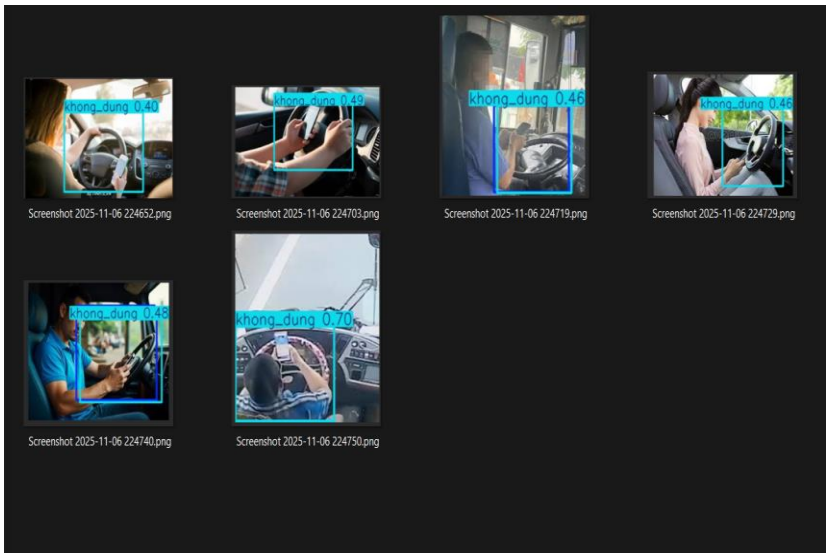


Hình 3.12: Ảnh sử dụng điện thoại bị nhận sai

Đối với ảnh sử dụng điện thoại chương trình đem lại độ chính xác trong khoảng 85%.

Bảng 3-4: Bảng thử nghiệm phân loại hình ảnh không sử dụng điện thoại

STT	Tên ảnh	Số lượng ảnh	Nhận đúng	Nhận sai	Độ chính xác
1	Ảnh không sử dụng điện thoại	50	44	6	85%



Hình 3.13: Ảnh không sử dụng điện thoại bị nhận sai

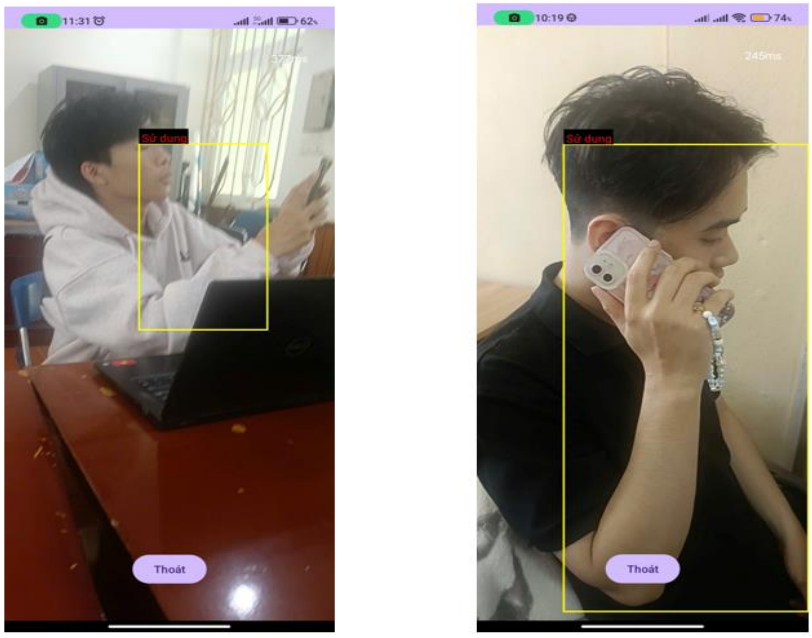
Từ kết quả trên ta có thể thấy được chương trình nhận diện sai nhiều kể cả đối với các loại ảnh không có trong mô hình chương trình cũng nhận diện sai và thông báo độ chắc chắn lên tới ~85%.

Đối với ảnh phát hiện sử dụng điện thoại khi chạy trên di động đem lại độ chính xác trong khoảng 50%.

Bảng 3-5: Bảng thử nghiệm phân loại hình ảnh không sử dụng điện thoại

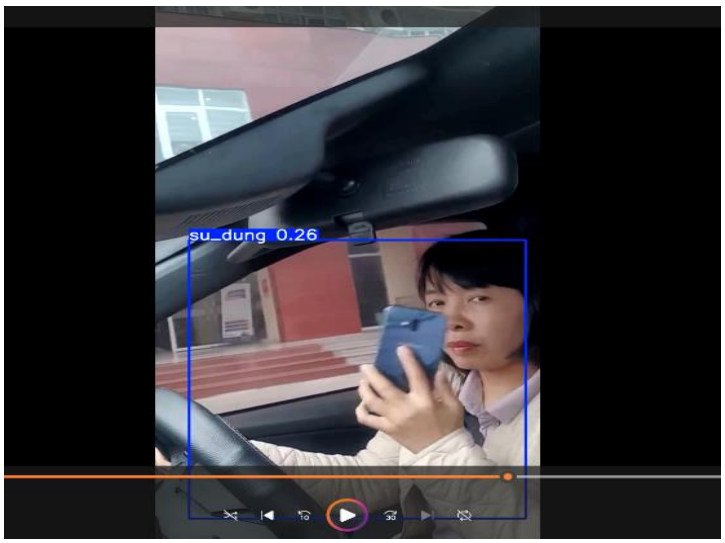
STT	**Tên ảnh**	**Số lượng ảnh**	**Nhận đúng**	**Nhận sai**	**Độ chính xác**
1	Ảnh sử dụng điện thoại chạy trên di động (Android)	20	10	10	50%

Dưới đây là các ảnh minh họa khi chạy mô hình phát hiện sử dụng điện thoại khi đang lái xe ô tô trên di động (Android)



Hình 3.15: Một số ảnh minh họa khi chạy mô hình trên thiết bị di động

Khi sử dụng video để nhận diện chương trình cũng đã thực hoàn thành tốt việc nhận diện những là video có nhiều góc độ khác nhau nên độ chắc chắn của mô hình sẽ có lúc không được quá cao.



Hình 3.15: Nhận diện bằng video

3.5. Đánh giá mô hình sau khi thử nghiệm

Mô hình YOLO11 đã được sử dụng để phân loại người lái xe ô tô có hành vi sử dụng điện thoại khi lái xe trên Google Colab, với bộ dữ liệu gồm khoảng 216 hình ảnh khác nhau. Mô hình đã được huấn luyện trong 100 epochs. Bài báo

cáo này sẽ đánh giá các kết quả đạt được, cũng như nêu rõ các ưu điểm và nhược điểm của mô hình.

3.5.1. Kết quả huấn luyện

Độ chính xác:

- Ở epoch thứ 90, mô hình đạt độ chính xác cao nhất là 75%.
- Khi huấn luyện lên 100 epochs, độ chính xác giảm nhẹ so với tại epoch 90.

Kiểm tra thủ công:

- Khi kiểm tra thủ công trên bảng 3-3, độ chính xác trung bình đạt 95%.
- Khi kiểm tra thủ công trên bảng 3-4, độ chính xác trung bình đạt 85%.
- Khi kiểm tra thủ công trên bảng 3-5, độ chính xác trung bình đạt 50%.
- Bộ ảnh kiểm tra thủ công lấy từ mạng có chất lượng cao và rõ nét, giúp mô hình đạt độ chính xác cao hơn so với bộ ảnh lấy từ webcam.

3.5.2. Thời gian huấn luyện

Thời gian huấn luyện mô hình:

- Mỗi 100 epochs tiêu tốn khoảng 9 phút
- Với bộ dữ liệu 216 ảnh và 100 epochs, thời gian huấn luyện kéo dài khoảng 10 ngày do giới hạn thời gian sử dụng phiên bản miễn phí của Google Colab (9 phút mỗi phiên, phải chờ 1-2 ngày để có phiên mới).

3.5.3. Ưu điểm của mô hình

Ưu điểm của mô hình:

- Hiệu suất và độ chính xác: YOLO11 đã cho thấy hiệu suất khá tốt với độ chính xác cao đối với các hình ảnh tĩnh sử dụng điện thoại và không sử dụng điện thoại.
- Khả năng xử lý: Hệ thống xử lý nhanh và chính xác đối với các hình ảnh tĩnh và những video có người sử dụng điện thoại.

3.5.4. Nhược điểm của mô hình

Những nhược điểm của mô hình:

- Độ chính xác với hình ảnh phức tạp: Với các hình ảnh chứa nhiều loại người khác nhau, độ chính xác chỉ đạt mức trung bình.
- Sai sót trong nhận diện: Mô hình còn gặp khó khăn khi nhận diện đối tượng có màu sắc và hình dáng tương tự nhau, hoặc bị che khuất một phần.

3.5.5. Đề xuất cải tiến

Dưới đây là những đề xuất xải tiến:

- Thêm dữ liệu đào tạo: Bổ sung thêm dữ liệu đa dạng để cải thiện khả năng nhận diện.
- Nâng cấp mô hình: Sử dụng các phiên bản cao cấp hơn của YOLO như YOLO12 mới ra mắt vào ngày 16/2/2025 để tăng độ chính xác.
- Tối ưu hóa tài nguyên: Sử dụng các dịch vụ đám mây khác hoặc phiên bản trả phí của Google Colab để có tài nguyên huấn luyện mạnh mẽ hơn.
- Mô hình YOLO11 đã phần nào đáp ứng được các yêu cầu của đề tài, đặc biệt là đối với các hình ảnh tĩnh. Tuy nhiên, vẫn cần cải thiện thêm để tăng độ chính xác và giảm sai sót trong quá trình nhận người sử dụng điện thoại di động và người không sử dụng điện thoại di động. Việc nâng cấp mô hình và bổ sung dữ liệu sẽ là các bước quan trọng để đạt được kết quả tốt hơn trong tương lai.

KẾT LUẬN

Sau thời gian gần 3 tháng nghiên cứu và thực hiện đồ án tốt nghiệp với sự hướng dẫn của cô TS. Hồ Thị Hương Thơm – Khoa Công Nghệ Thông Tin trường Đại Học Hàng Hải Việt Nam để hoàn thiện được đồ án với đề tài: Xây dụng mô hình phát hiện người lái xe ô tô óc hành vi sử dụng điện thoại khi đang lái xe

Một số mục tiêu mà đồ án đã đạt được:

Hoàn thành đề tài: "Xây dụng mô hình phát hiện người lái xe ô tô có hành vi sử dụng điện thoại di động khi lái xe"

Đào tạo thành công mô hình nhận diện với 216 ảnh khác nhau bằng mô hình YOLO11.

Mô hình phân loại được các loại ảnh với độ chính xác trung bình 75% với 100 epochs nhưng lại chính xác khá cao với ảnh tĩnh với một loại ảnh là 95% và vẫn còn nhiều sai sót với một số ảnh lạ. Tuy nhiên khi đưa vào thực tiễn có thể đáp ứng được phần nào yêu cầu đặt ra nhưng có thể sẽ không được chính xác. Muốn cải thiện độ chính xác thì ảnh trước khi đưa vào phân loại cần phải được xử lý nâng cao chất lượng ảnh, hoặc ảnh được chụp quay từ các thiết bị chuyên dụng có chất lượng cao.

Chúng ta có thể phát triển chương trình này thành một mô-đun tiền xử lý hình ảnh phát hiện đối tượng sử dụng điện thoại khi lái xe, để tích hợp vào hệ thống robot phân loại tự động. Việc tự động hóa này sẽ góp phần hiện đại hóa và giảm thiểu nguy cơ nguy hiểm khi tham gia giao thông

Trong quá trình học tập và nghiên cứu, tìm hiểu một lĩnh vực rất mới và nhiều thử thách đối với bản thân sinh viên cho nên đồ án của sinh viên sẽ còn có những thiếu sót và hạn chế. Sinh viên rất cầu thị quý thầy cô trong khoa Công nghệ thông tin và Viện đào tạo sau đại học có cái nhìn cảm thông và chia sẻ đóng góp những lời nhận xét để sinh viên nâng cao sự hiểu biết về lĩnh vực AI vốn là lĩnh vực mới và hoàn thiện hơn đồ án của mình.

Em xin trân thành cảm ơn!

TÀI LIỆU THAM KHẢO

A. Tài liệu tiếng Việt

1. Nguyễn Văn Vỵ, *Giáo trình Trí tuệ nhân tạo*. Hà Nội: NXB Khoa học và Kỹ thuật, 2023.
2. Phạm Văn Hùng, *Thị giác máy tính và ứng dụng*. Hà Nội: NXB Bách Khoa, 2022.
3. Ultralytics, "Tài liệu hướng dẫn sử dụng YOLOv8–v11," *Trang chủ Ultralytics YOLO*, 2024. Có tại: https://docs.ultralytics.com/vi/models/yolo11/
4. Nguyễn Hữu Tùng, "Ứng dụng học sâu trong nhận diện khuôn mặt," *Tạp chí Công nghệ Thông tin & Truyền thông*, 2024.
5. Python Software Foundation, "Python Documentation," 2024. Có tại: https://docs.python.org/3/
6. Google Research, "Hướng dẫn sử dụng Google Colab," *Google Research Documentation*, 2024. Có tại: https://colab.research.google.com/
7. Microsoft, "Visual Studio Code Documentation," 2024. Có tại: https://code.visualstudio.com/docs/

B. Tài liệu tiếng Anh

8. Ultralytics, "YOLOv11 Model Release and Benchmark Report," 2025. Có tại: https://github.com/ultralytics/YOLOv11
9. Roboflow, "Computer Vision Dataset Management," 2024. Có tại: https://roboflow.com/
10. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016. Có tại : https://www.deeplearningbook.org/