BỘ GIÁO DỤC VÀ ĐÀO TẠO

BÁO CÁO TỔNG KẾT

ĐỀ TÀI THAM GIA XÉT TẶNG GIẢI THƯỞNG KHOA HỌC VÀ CÔNG NGHỆ DÀNH CHO SINH VIÊN TRONG CƠ SỞ GIÁO DỤNG ĐẠI HỌC NĂM 2023

XÂY DỰNG ỨNG DỤNG NHẬN DIỆN BIỂN SỐ XE

Thuộc nhóm ngành khoa học: Khoa học tự nhiên

Chuyên ngành cụ thể của lĩnh vực khoa học và công nghệ:

Khoa học máy tính và thông tin

BỘ GIÁO DỤC VÀ ĐÀO TẠO

BÁO CÁO TỔNG KẾT

ĐỀ TÀI THAM GIA XÉT TẶNG GIẢI THƯỞNG KHOA HỌC VÀ CÔNG NGHỆ DÀNH CHO SINH VIÊN TRONG CƠ SỞ GIÁO DỤNG ĐẠI HỌC NĂM 2023

XÂY DỰNG ỨNG DỤNG NHẬN DIỆN BIỂN SỐ XE

Thuộc nhóm ngành khoa học: Khoa học tự nhiên

Chuyên ngành cụ thể của lĩnh vực khoa học và công nghệ:

Khoa học máy tính và thông tin

MỤC LỤC

DANH MỤC KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT

Từ viết tắt	Từ chuẩn	Diễn giải
	Embedding	Là ma trận được trích xuất từ ảnh đối tượng nhận diện đã được cắt
AI	Artificial Intelligence	Trí tuệ nhân tạo
ANPR	Automatic Number Plate Recogition	Nhận diện biển số xe tự động
API	Application Programming Interface	Giao diện lập trình ứng dụng
ASGI	Asynchronous Server Gateway Interface	Giao diện cổng vào máy chủ không đồng bộ
CNN	Convolutional Neural Network	Mạng nơ ron tích chập
DL	Deep Learning	Học sâu, là một phần của AI
FFARM	Flutter Fast API React Mongodb	Các công nghệ sử được sử dụng trong chương trình
FPS	Frames per second	Số khung hình trên giây
HNSW	Hierarchical Navigable Small World	Là một cấu trúc dữ liệu được sử dụng trong tìm kiếm gần nhất trong không gian nhiều chiều.
HNSWlib	Hierarchical Navigable Small World Library	là một thư viện mã nguồn mở được sử dụng cho cấu trúc dữ liệu không gian mạng HNSW
HTTP	HyperText Transfer Protocol	Giao thức giao tiếp giữa client và server

IOU	Intersection Over Union	Hợp cả 2 vị trí dự đoán và vị trí cố định trong YOLO
KNN	K-Nearest Neighbors	Là một trong những thuật toán học có giám sát được sử dụng nhiều trong khai phá dữ liệu và học máy.
ML	Machine Learning	Máy học, máy tính có khả năng học tập
MUI	Material User Interface	Thư viện của Javascript để thiết kế giao diện người dùng cho website
NPM	Node Package Manager	Là một công cụ tạo và quản lý các thư viện lập trình Javascript cho Node.js
ONNX	Open Neural Network Exchange	Là một hệ sinh thái trí tuệ nhân tạo mã nguồn mở
RTSP	Real Time Streaming Protocol	Giao thức phát trực tuyến trong thời gian thực
SOA	Service-oriented architecture	Kiến trúc hướng dịch vụ
YOLO	You Only Look One	Là một thuật toán nhận diện vật thể

MỞ ĐẦU

1. Tổng quan tình hình nghiên cứu

Ngày nay, với sự phát triển của khoa học công nghệ và các kỹ thuật tiên tiến, hiện đại, việc áp dụng các kỹ thuật đó vào đời sống của con người là thật sự cần thiết, bởi những việc đó không chỉ giúp cho cuộc sống của con người được cải thiện mà còn làm cho xã hội phát triển nhanh chóng, góp phần thúc đẩy sự tiến bộ của khoa học, kỹ thuật cũng như bắt kịp với xu hướng của thời đại.

Một trong những lĩnh vực hiện đại đang dẫn đầu xu hướng hiện nay trong công cuộc phát triển của thời đại 4.0 chính là nghiên cứu về trí tuệ nhân tạo. Trí tuệ nhân tạo là lĩnh vực đang được phát triển nhất trên thế giới, bằng việc sử dụng các phương pháp machine learning hay deep learning,… và một số phương pháp khác đã giúp cho lĩnh vực nghiên cứu về trí tuệ nhân tạo đang dẫn đầu xu hướng không chỉ ở Việt Nam mà còn ra bên ngoài thế giới. Do vậy, trong lĩnh vực nghiên cứu về trí tuệ nhân tạo, tính đến thời điểm hiện tại, đã có nhiều ứng dụng phục vụ đời sống của con người như nhận diện khuôn mặt, phát hiện làn đường, trợ lý ảo, nhận dạng chữ viết tay, nhận diện biển số xe,...

Đặc biệt, cùng với sự phát triển của dân số trên thế giới thì nhu cầu di chuyển hoặc sử dụng các phương tiện cũng được tăng cao, do có quá nhiều phương tiện di chuyển, tham gia giao thông nên việc quản lý cũng rất khó khăn. Trong báo cáo này, sẽ xây dựng một ứng dụng được phát triển trên nhiều nền tảng dùng để nhận diện biển số xe và mô hình liên quan đến việc quản lý các phương tiện tại các bãi đậu xe.

Việc nhận diện biển số xe với các phương pháp máy học đã được công bố ở một số công trình. Nhóm các tác giả Rayson Laroca, Evair Severo, Luiz A. Zanlorensi [1] đã trình bày hệ thống ALPR (automatic license plate recognition) dựa trên mô hình YOLOv2, nhóm tác giả đã thử nghiệm tập dữ liệu SSIG bao gồm 2000 khung hình trong 101 video xe cộ, hệ thống real-time (47 FPS) của họ đạt được độ chính xác 93.53%, để đánh giá độ chính xác trong các trường hợp thực tế, nhóm tác giả thử

nghiệm tập dữ liệu UFPR-ALPR (dữ liệu bao gồm nhiều loại: xe hơi, xe cơ giới, xe máy,...) bao gồm 150 video và 4500 khung hình được ghi lại bởi camera và camera hành trình của xe cộ, hệ thống real-time (35 FPS) của họ nhận diện được có độ chính xác 78.33%.

Trong một bài báo tương tự sử dụng mô hình YOLOv2 trong thời gian thực để phát hiện người đội mũ bảo hiểm và biển số xe của họ từ tác giả Yonten Jamtsho, Panomkhawn Riyamongkol và Rattapoom Waranusast [2]. Trong công trình này, các tác giả đã sử dụng một mạng nơ-ron phức hợp để tự động phát hiện biển số xe của một người lái xe mô tô trong video. Tỷ lệ phát hiện biển số tổng thể trong công trình này là 98,52%.

Nhóm của tác giả Lê Sách Thanh, Trần Nhân Văn, Kazuhiko Hamamoto [3] trình bày phương pháp nhận diện biển số bằng phương pháp nhận diện cạnh Candy và kết hợp nhị phân Sauvola với CCA (phân tích thành phần kết nối) và sự giãn nở cạnh (edge dilation). Thông qua đề xuất phương pháp đó nhóm tác giả kiểm tra độ chính xác bằng cách kiểm tra 994 hình ảnh thì đúng được 921 trên 994 hình ảnh (97.56%).

YOLO (You only look once: Bạn chỉ nhìn một lần) là một thuật toán phát hiện đối tượng chia hình ảnh thành một hệ thống lưới. Mỗi ô trong lưới chịu trách nhiệm phát hiện các đối tượng trong chính nó. YOLO là một trong những mô hình học sâu hiện đại nhất hiện nay, YOLO nổi tiếng do tốc độ và độ chính xác của nó. YOLOv5 là một mô hình Object Detection thuộc họ mô hình YOLO. Khác với những phiên bản tiền nhiệm, YOLOv5 được phát triển dựa trên PyTorch thay vì DarkNet. Đây là một điểm cộng rất lớn cho YOLOv5 vì PyTorch phổ biến nhiều, đồng nghĩa là có nhiều tài liệu cho chúng ta tham khảo về mô hình này. Chính vì vậy, nhóm đã chọn mô hình YOLOv5 để áp dụng vào đề tài nghiên cứu.

2. Mục tiêu đề tài

Xây dựng ứng dụng nhận diện biển số tự động qua hình ảnh thu được từ camera nhằm có thể đáp ứng được nhu cầu quản lý phương tiện tại các bãi đậu xe nói chung và

quản lý các phương tiện giao thông ra vào trường Đại học Đà Lạt nói riêng. Trong quá trình đó, để tăng sự an ninh, chúng tôi nghiên cứu thêm phương pháp nhận diện khuôn mặt để tích hợp hệ thống để định danh xe và chủ xe để vào khu vực.

- Tìm hiểu và sử dụng mô hình YOLO trong việc huấn luyện mô hình
- Tìm hiểu FaceNet, MTCNN, HNSWlib.
- Xây dựng một mô hình học máy cho phép nhận dạng biển số xe thông qua các camera được thiết lập tại trường
- Áp dụng mô hình nhận diện khuôn mặt để định danh chủ xe.
- Áp dụng được các framework UI, Mobile để xây dựng giao diện ứng dụng.
- Vận dụng được các kỹ thuật để xây dựng ứng dụng web, mobile.
- Đánh giá kết quả và đề xuất phương hướng phát triển.
- Xây dựng được mô hình quản lý bãi đậu xe thông minh sau khi nhận diện được biển số.

3. Phương pháp nghiên cứu

Ứng dụng các mô hình học máy kết hợp với một số phương pháp cải tiến khác để xây dựng được ứng dụng có thể áp dụng vào thực tiễn, đặc biệt là có thể áp dụng trong việc quản lý phương tiện trong các phải đậu xe.

4. Phạm vi nghiên cứu

Đề tài chỉ tập trung xây dựng phương pháp cho bài toán nhận diện biển số thông qua hình ảnh thu được từ camera. Từ đó, có thể phát triển thêm các tính năng phù hợp với nhu cầu quản lý phương tiện tại các bãi đậu xe.

5. Nội dung ứng dụng của công trình / đề tài

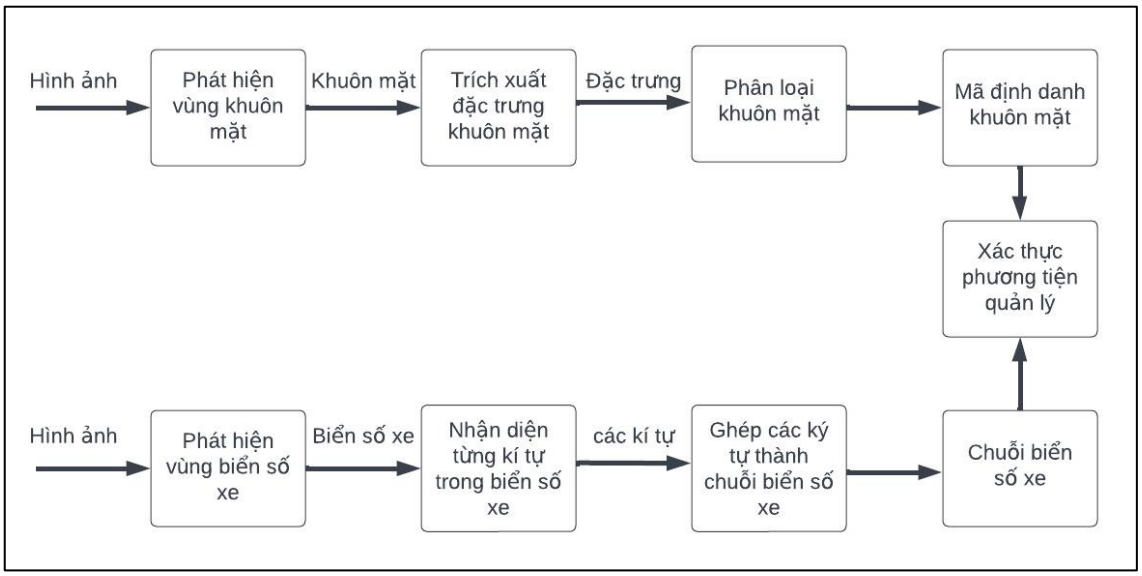
Đề tài tập trung nghiên cứu hệ thống nhận diện biển số thông qua một số kỹ thuật xử lý ảnh, nhận dạng ảnh, học sâu (deep learning) dựa trên mô hình YOLO. Kết quả đạt được trong quá trình nghiên cứu đã cho thấy ứng dụng nhận diện biển số trong ảnh và

cả video trong thời gian thực có độ chính xác trên 90% và có thể áp dụng để được triển khai thành hệ thống để ứng dụng cho quản lý xe thông minh.

Chương I. TỔNG QUAN

1. Hệ thống bãi đậu xe thông minh

Để lên kế hoạch thực hiện được hệ thống quản lý bãi đậu xe thông minh, cần tìm hiểu về các hệ thống quản lý bãi đậu xe hiện nay đã được áp dụng vào thực tế như thế nào và hiện đang là hệ thống hoạt động tối ưu nhất trong việc quản lý các phương tiện tại các bãi đậu xe. Sau quá trình tìm hiểu một số mô hình về hệ thống quản lý bãi đậu xe thông minh, trong báo cáo này, sẽ triển khai một hệ thống với các bước như hình sau:



Hình 1.1: Các bước xác thực phương tiện quản lý trong hệ thống quản lý bãi đậu xe thông minh

Các hình ảnh ở đầu vào được trích xuất từ các camera tại các bãi đậu xe. Thông qua từng khung hình trong camera sẽ có các hình ảnh tương ứng với mỗi khung hình. Các hình ảnh này tùy vào mục đích của hệ thống, có thể được chuyển sang các bước tương ứng.

Với hình ảnh là khuôn mặt của chủ phương tiện, sẽ thực hiện các bước để hoàn thành quá trình nhận diện khuôn mặt của chủ phương tiện bao gồm quá trình trích xuất các đặc trưng của khuôn mặt và phân loại để xác định được mã định danh của khuôn

mặt trong trường hợp đã đăng ký chủ sỡ hữu của phương tiện trong hệ thống. Để có thể nhận diện được khuôn mặt của chủ phương tiện tại các bãi đỗ, cần áp dụng mô hình để thực hiện việc nhận diện khuôn mặt. Trong báo cáo này, sẽ áp dụng mô hình FaceNet và MTCNN để thực hiện bước nhận diện khuôn mặt này.

Với hình ảnh là biển số của phương tiện giao thông, sẽ thực hiện các bước nhận diện biển số với hai quy trình là nhận diện vùng chứa biển số và nhận diện các ký tự có trong biển số đó sau khi nhận diện được vùng chứa. Để có thể thực hiện được quá trình nhận diện biển số này trong hệ thống, cần áp dụng mô hình YOLO để nhận diện đối tượng, cụ thể đối tượng ở đây chính là biển số của phương tiện.

Sau khi nhận diện được hai hình ảnh gồm biển số xe và hình ảnh khuôn mặt của chủ phương tiện, hệ thống sẽ tiến hành vào bước xác thực phương tiện, điều này được áp dụng đối với các trường hợp vào gửi phương tiện tại bãi đỗ hoặc lấy phương tiện ra khỏi bãi đỗ. Tùy vào từng trường hợp sẽ có các chức năng tương ứng để hỗ trợ cho quá trình xác thực này.

Ngoài ra, để có thể triển khai được hệ thống này, cần triển khai một kiến trúc để áp dụng vào hệ thống. Trong báo cáo này, sẽ áp dụng kiến trúc Microservice vào quá trình triển khai hệ thống và các chương trình ứng dụng hỗ trợ cho việc quản lý các phương tiện tại các bãi đậu xe. Các ứng dụng này bao gồm ứng dụng di động và ứng dụng web.

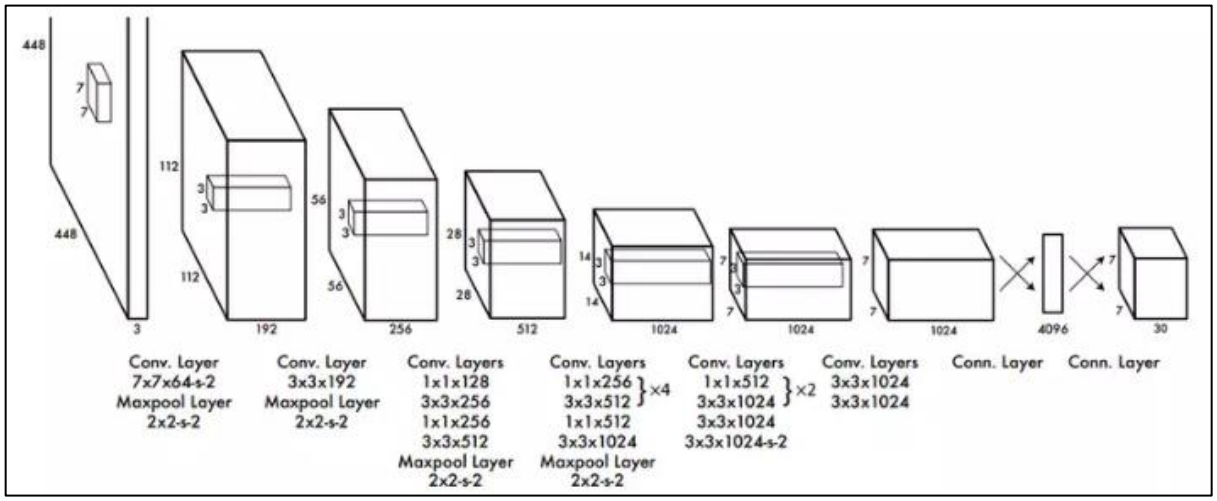
Cụ thể chi tiết việc tìm hiểu các mô hình và kiến trúc áp dụng được triển khai trong hệ thống có nội dung như các phần sau.

1.1. Mô hình YOLO

1.1.1. Giới thiệu

YOLO, 1 từ viết tắt cho ‘You only look once’ (Bạn chỉ nhìn một lần) là một thuật toán phát hiện đối tượng chia hình ảnh thành một hệ thống lưới. Mỗi ô trong lưới chịu trách nhiệm phát hiện các đối tượng trong chính nó. YOLO là một trong những thuật toán phát hiện đối tượng nổi tiếng nhất do tốc độ và độ chính xác của nó.

YOLO là một mô hình mạng CNN cho việc phát hiện, nhận dạng, phân loại đối tượng. YOLO được tạo ra từ việc kết hợp giữa các convolutional layers và connected layers. Trong đó các convolutional layers sẽ trích xuất ra các feature của ảnh, còn full-connected layers sẽ dự đoán ra xác suất đó và tọa độ của đối tượng.



Hình 1.2: Mô hình mạng CNN được sử dụng trong YOLO

1.1.2. Lịch sử, cách thức học máy

Lịch sử của YOLO

YOLOv1 đã được Joseph Redmon phát hành dưới dạng nghiên cứu. Tờ báo có tiêu đề là ‘You Only Look Once: Unified, Real-Time Object Detection’ được phát hành vào ngày 8 tháng 6 năm 2015.

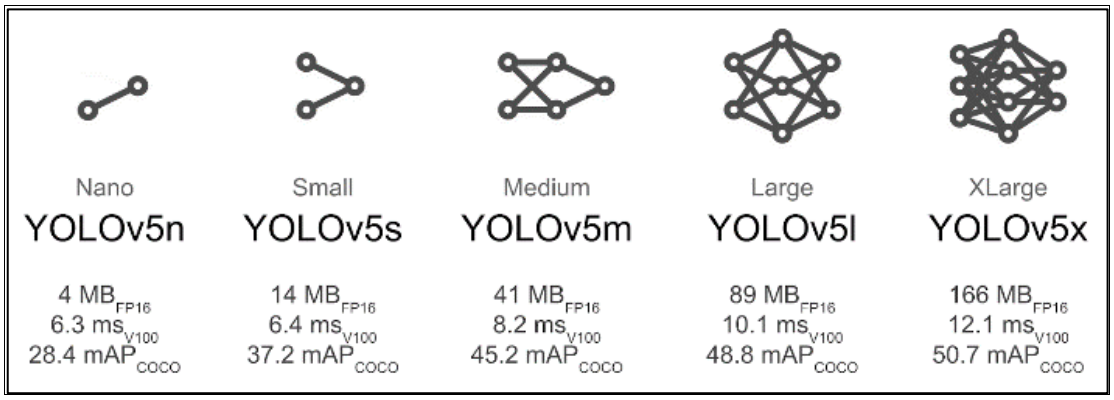
YOLOv2 đã được phát hành bởi Joseph Redmon - tác giả ban đầu của YOLO và Ali Farhadi, cùng nhau xuất bản ‘YOLO9000: Better, Faster, Stronger’ vào ngày 25 tháng 12 năm 2016.

YOLOv3 được cải tiến dựa trên YOLOv2 bởi 2 tác giả ban đầu YOLOv2, họ cùng nhau xuất bản ‘YOLOv3: An incremental Improvement’ vào ngày 8 tháng 4 năm 2018.

YOLOv4 được phát triển bởi Alexey Bochoknovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao vì tác giả ban đầu của YOLO đã bế tắc khi phát hành phiên bản mới,

tờ báo có tiêu đề ‘YOLOv4: Optimal Speed and Accuracy of Object Detection’ vào ngày 23 tháng 4 năm 2020.

YOLOv5 được phát hành ngay sau khi YOLOv4 được phát triển dựa vào framework Pytorch bởi tác giả Glenn Jocher, mã nguồn YOLOv5 có sẵn trên GitHub được phát hành ngày 18 tháng 5 năm 2020.



Hình 1.3: Sự khác biệt của các phiên bản của YOLOv5

Cách thức học máy của mô hình YOLO

Hình ảnh đầu vào sẽ được Thuật toán YOLO chia thành SxS thường thì sẽ 3x3, 7x7, 9x9,...

Với dữ liệu đầu vào là 1 ảnh, đầu ra mô hình sẽ là một ma trận 3 chiều có kích thước SxSx(5xB+C) với số lượng tham số mỗi ô là (5xB+C) với B là số lượng Bounding Box, C là lớp mà mỗi ô cần dự đoán.

Ta có bounding box gồm 5 thành phần dự đoán (x,y,w,h, confidence) với (x,y) là tọa độ tâm bouding box so với giới bound của ô lưới, (w,h) là chiều rộng, chiều cao được dự đoán so với toàn bộ hình ảnh,và cuối cùng dự đoán confidence đại diện cho IOU (Intersection over Union (tạm dịch: hợp cả 2) giữa vị trí được dự đoán và vị trí cố định cho trước (predicted box, ground truth box).

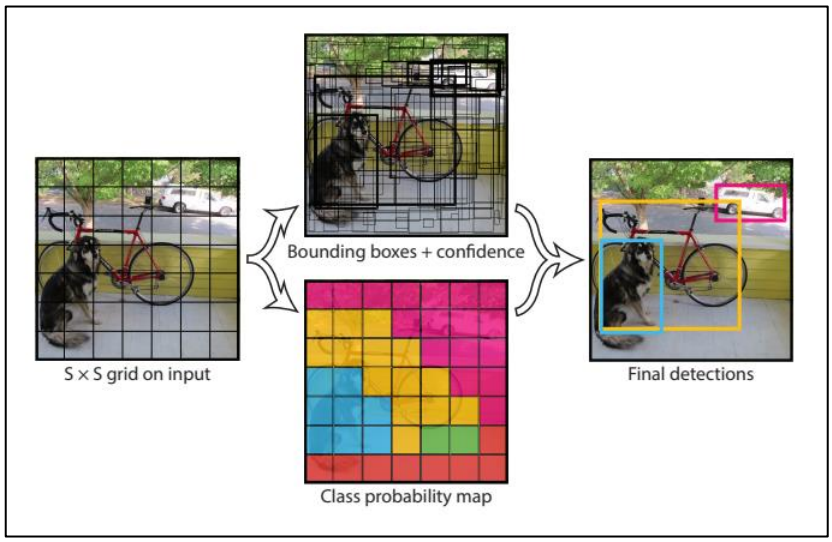
Từng ô lưới dự đoán xác suất lớp (class): Pr(Classi|Object)

Confidence score: nếu không có đối tượng trong ô, thì điểm sẽ là 0, còn không thì bằng Pr(Object)*IOU

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

Hình 1.4: Công thức tính độ chính xác phát hiện ra đối tượng

Theo như công thức trên ta sẽ ra xác suất final detection sẽ dự đoán và bao các đối tượng trong hình ảnh đó như hình minh họa bên dưới:



Hình 1.5: Phát hiện nhận diện đối tượng theo công thức

1.1.3. Ưu và nhược điểm

Ưu điểm

- Một trong những ưu điểm mà YOLO đem lại đó là chỉ sử dụng thông tin toàn bộ bức ảnh một lần và dự đoán toàn bộ object box chứa các đối tượng, mô hình được xây dựng theo kiểu end-to-end nên được huấn luyện hoàn toàn bằng gradient descen
- Xử lý khung hình ở tốc độ 45 FPS đến 150 FPS tốt hơn so với thời gian thực.
- Khả năng tổng quát hình ảnh tốt hơn.

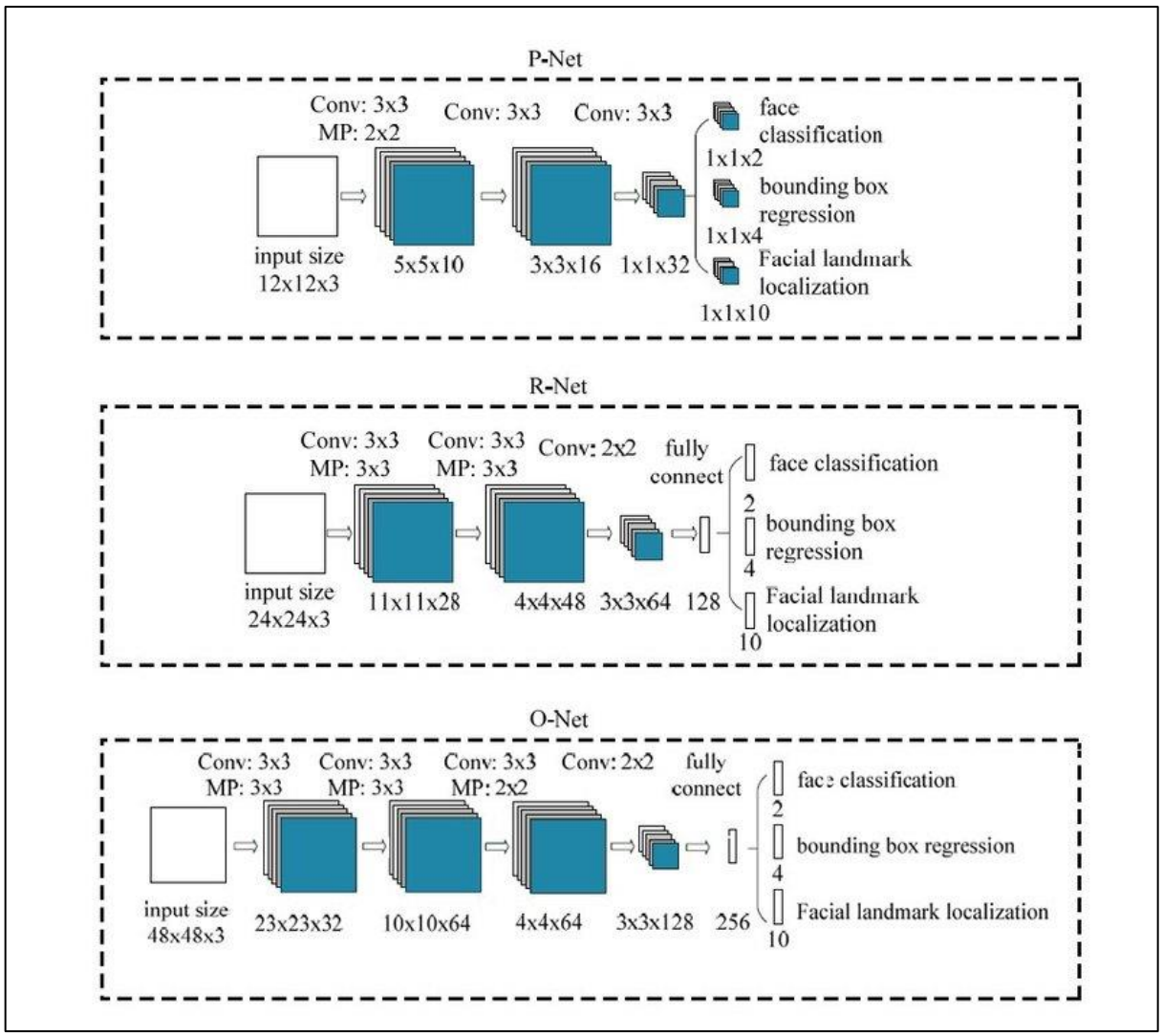
Nhược điểm

- Hiệu suất không cao trong việc phát hiện các đối tượng có kích thước nhỏ hoặc đối tượng cần phát hiện nhỏ hơn 1 ô lưới được phân cắt theo mô hình.
- Việc phát hiện một nhóm các đối tượng ở chung một ô lưới được phân cắt theo mô hình cũng trở nên khó khăn.
- Bị giới hạn về tính linh hoạt trong việc phát hiện các vật thể do mô hình chỉ phát hiện được các vật thể dựa trên dữ liệu đã được học từ trước (được phân vùng theo một kích thước cụ thể) nên khi có một vật thể với tỉ lệ kích thước khác thường dễ dàng dẫn đến việc phát hiện sai vật thể hoặc làm giảm tỷ lệ chính xác.

1.2. Mô hình MTCNN, FaceNet

1.2.1. Giới thiệu mô hình MTCNN

Mô hình MTCNN (Multi-Task Cascaded Convolutional Networks) là một mô hình sử dụng trong việc phát hiện khuôn mặt. MTCNN bao gồm ba giai đoạn và được thiết kế để phát hiện khuôn mặt trong các bức ảnh với độ chính xác cao.



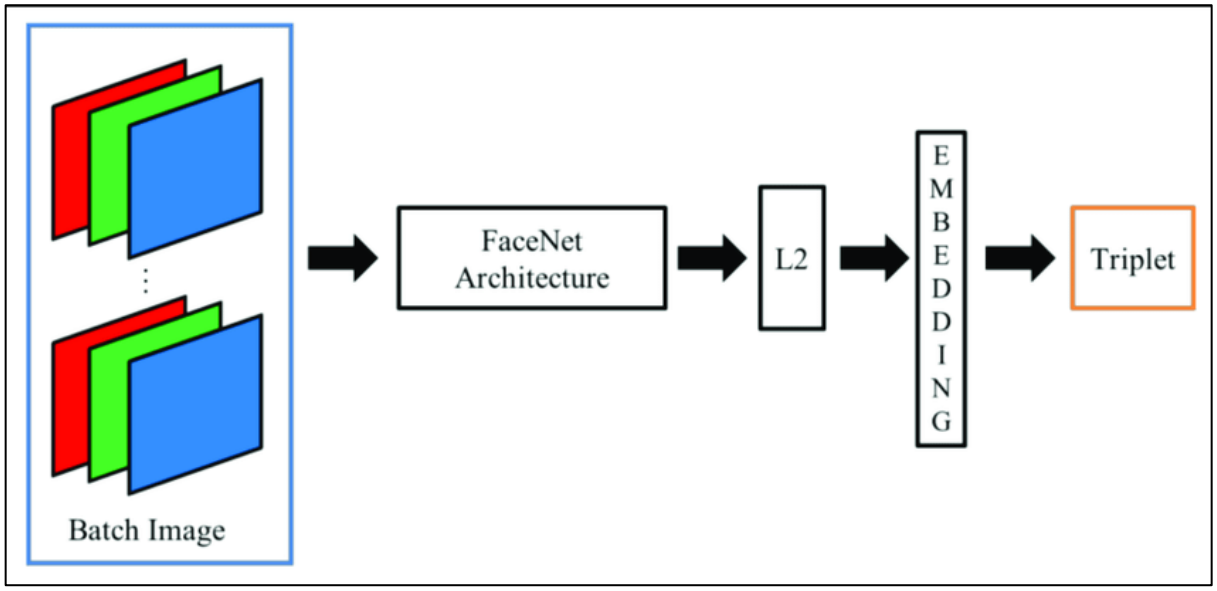
Hình 1.6: Kiến trúc mô hình MTCNN

MTCNN sử dụng một kiến trúc mạng neural gồm ba mạng con, mỗi mạng con đóng vai trò trong việc phát hiện khuôn mặt và điểm đặc trưng trên khuôn mặt với độ chính xác khác nhau. Cụ thể, các mạng con của MTCNN bao gồm:

- P-Net (Proposal Network): Đây là mạng con đầu tiên trong quá trình phát hiện khuôn mặt. P-Net có nhiệm vụ đưa ra các đề xuất khuôn mặt có khả năng xuất hiện trong ảnh đầu vào. P-Net sử dụng một kiến trúc mạng neural convolutional để phát hiện khuôn mặt và các điểm đặc trưng trên khuôn mặt.

- R-Net (Refinement Network): Sau khi các đề xuất khuôn mặt được đưa ra bởi P-Net, R-Net sẽ tiếp tục phát hiện khuôn mặt và các điểm đặc trưng trên các đề xuất này. R-Net sử dụng một kiến trúc mạng neural convolutional phức tạp hơn P-Net và đưa ra các đề xuất khuôn mặt với độ chính xác cao hơn.
- O-Net (Output Network): Mạng cuối cùng trong quá trình phát hiện khuôn mặt là O-Net, nó sẽ xác định chính xác các điểm đặc trưng trên khuôn mặt đã được phát hiện bởi R-Net. O-Net sử dụng một kiến trúc mạng neural convolutional tương tự như R-Net và đưa ra các đề xuất khuôn mặt cuối cùng với độ chính xác cao nhất.

1.2.2. Giới thiệu mô hình FaceNet



Hình 1.7: Mô hình FaceNet

Mô hình FaceNet là một mô hình sử dụng để nhận diện khuôn mặt. FaceNet sử dụng một mạng CNN để rút trích đặc trưng của khuôn mặt và biểu diễn chúng dưới dạng một vectơ số. Mô hình này sử dụng một kỹ thuật gọi là Triplet Loss để đào tạo mô hình. Triplet Loss so sánh khoảng cách giữa các vectơ biểu diễn khuôn mặt để đảm bảo rằng các biểu diễn của cùng một khuôn mặt gần nhau hơn các biểu diễn của các khuôn

mặt khác nhau. Sau khi được đào tạo, mô hình FaceNet có thể được sử dụng để nhận diện khuôn mặt trong các bức ảnh hoặc video.

Theo như [4] độ chính xác của FaceNet hơn 99% trong bộ dữ liệu LFW, và một số kết quả ở các bộ data khác:

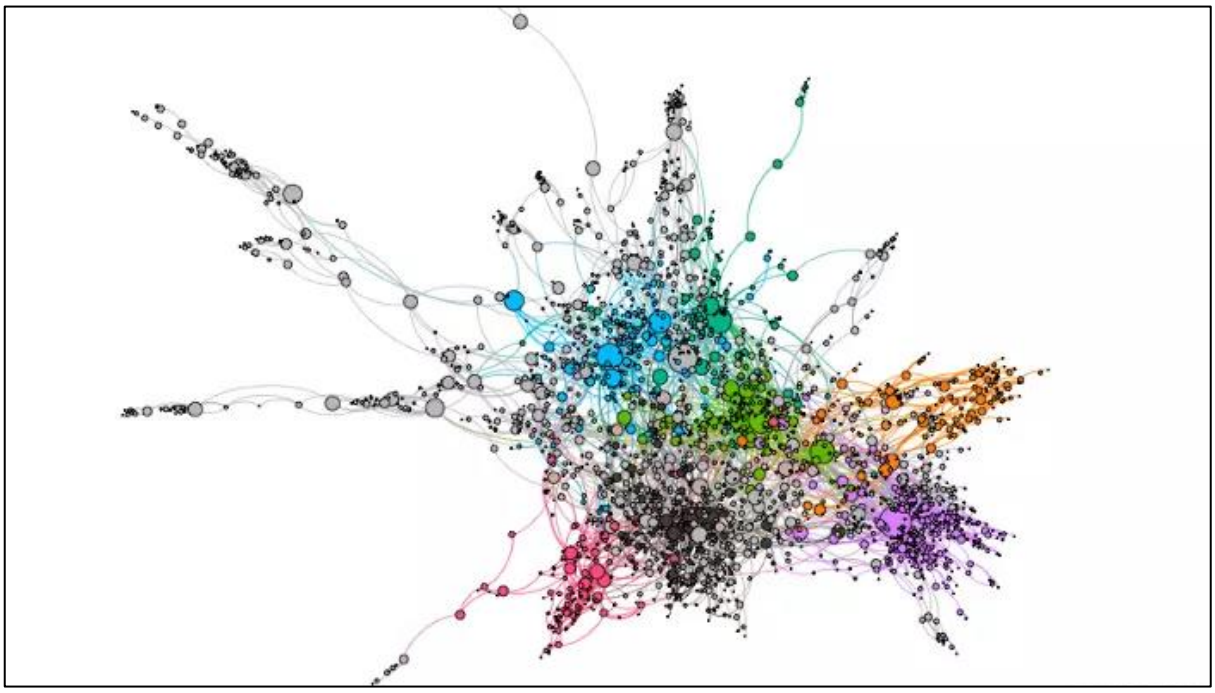
Labeled Faces in the Wild (LFW): 99.63%

YouTube Faces DB: 95.12%

CASIA-Webface: 99.46%

Vì thế chúng tôi chọn FaceNet để có thể tăng độ chính xác khi nhận diện khuôn mặt.

1.2.3. Giới thiệu HNSWlib



Hình 1.8: Đồ thị HNSWlib

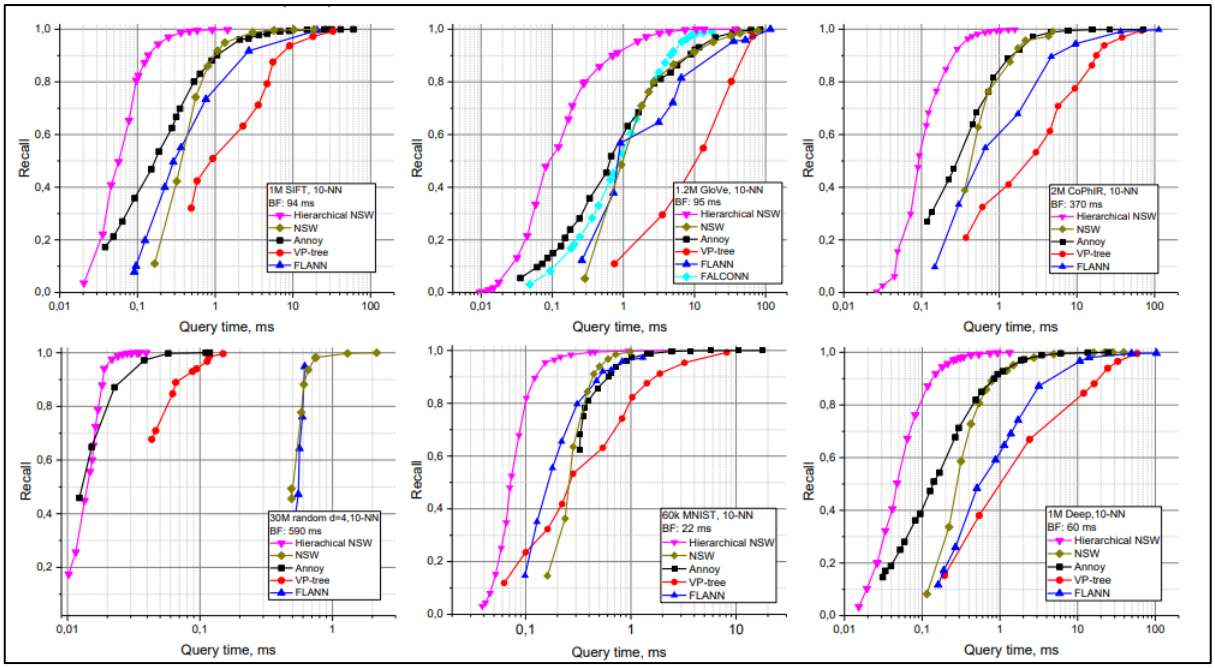
HNSWlib là một thư viện mã nguồn mở được sử dụng cho cấu trúc dữ liệu không gian mạng HNSW (Hierarchical Navigable Small World). HNSW là một cấu trúc dữ liệu được sử dụng trong tìm kiếm gần nhất trong không gian nhiều chiều.

Thư viện HNSWlib cung cấp các công cụ để xây dựng và quản lý cấu trúc dữ liệu HNSW. Nó được viết bằng C++ và cung cấp các giao diện và liên kết với các ngôn ngữ khác như Python, Java, và C#.

HNSWlib cung cấp một số tính năng quan trọng, bao gồm:

- Xây dựng cấu trúc dữ liệu HNSW: Thư viện cho phép tạo ra một cấu trúc dữ liệu HNSW từ tập dữ liệu đầu vào. Quá trình xây dựng này đảm bảo rằng các điểm dữ liệu gần nhau trong không gian nhiều chiều sẽ được kết nối lại với nhau.
- Tìm kiếm gần nhất: HNSWlib cung cấp các phương pháp tìm kiếm gần nhất cho các điểm dữ liệu trong không gian nhiều chiều. Bằng cách sử dụng cấu trúc dữ liệu HNSW, nó có thể tìm kiếm các hàng xóm gần nhất cho một điểm dữ liệu đã cho một cách hiệu quả và nhanh chóng.
- Tùy chỉnh và cấu hình: Thư viện cho phép tùy chỉnh và cấu hình các tham số của cấu trúc dữ liệu HNSW để phù hợp với yêu cầu của bài toán cụ thể. Có thể điều chỉnh số KNNvà các tham số khác để đạt được hiệu suất và độ chính xác mong muốn.

So sánh các thuật toán trong không gian Eulcid:



Hình 1.9: So sánh HNSW (Herachical NSW) với các thuật toán khác

Như hình trên, tác giả đã dùng bộ data K-ANNS [5] benchmark ann-benchmark để kiểm tra. Như kết quả trên, kết quả cho dữ liệu vector đối với tập dữ liệu SIFT, GloVE, DEEP và CoPhIR, cho thấy thời gian truy vấn (Query Time) của HNSW ít hơn so với các thuật toán còn lại nhưng Recall (là thước đo xem có bao nhiêu KNN được xác đinh chính xác) thì cao, nhưng theo dữ liệu ít chiều (d=4) thì HNSW nhanh hơn 1 chút so với ANNOY. Theo kết quả đó chúng tôi chọn HNSW và HNSW có thể tùy chỉnh và cấu hình, và lưu theo dạng key-value, thì có thể tìm kiếm theo dạng KNN tìm gần điểm đặt trưng nhất và có thể trích xuất key ra để định danh người dùng.

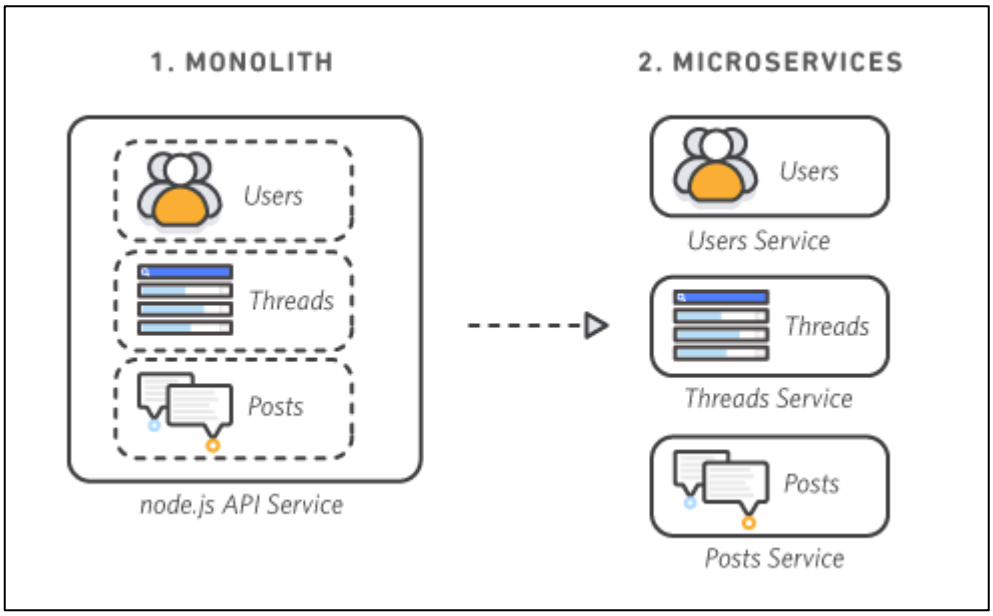
1.3. Kiến trúc Microservice

1.3.1. Giới thiệu kiến trúc

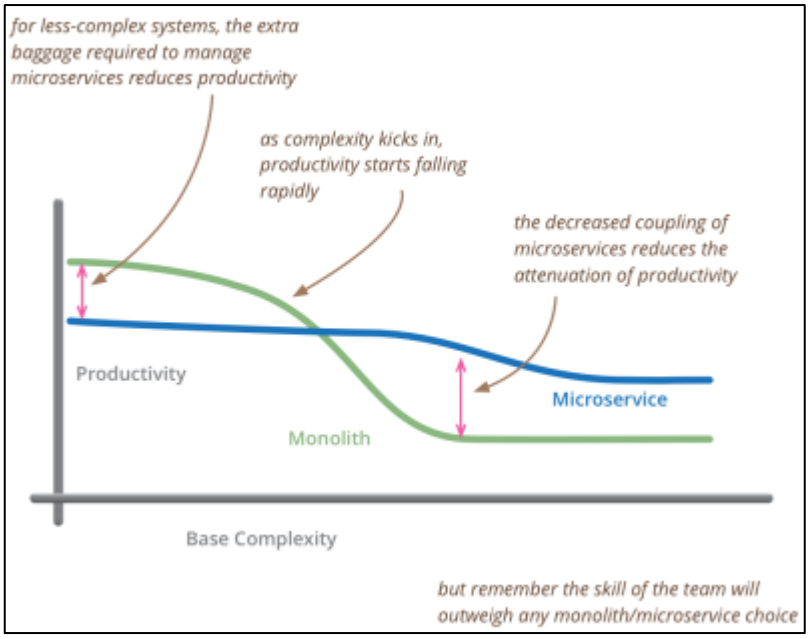
Trong tiếng anh, micro có nghĩa là nhỏ, vi mô. Vậy Microservice, như tên của nó, đó chính là chia một khối phần mềm thành các service nhỏ hơn, có thể triển khai trên các server khác nhau. Mỗi service sẽ xử lý từng phần công việc và được kết nối với nhau thông qua các các giao thức khác nhau, như http, SOA, socket, Message queue (Active MQ, Kafka)... để truyền tải dữ liệu.

Trước khi Microservices xuất hiện, các ứng dụng thường phát triển theo mô hình Monolithic architecture (Kiến trúc một khối). Có nghĩa là tất cả các module (view, business, database) đều được gộp trong một project, một ứng dụng được phát triển theo mô hình kiến trúc một khối thường được phân chia làm nhiều module. Nhưng khi được đóng gói và cài đặt sẽ thành một khối (monolithic). Lợi ích của mô hình kiến trúc một khối đó là dễ dàng phát triển và triển khai. Nhưng bên cạnh đó nó cũng có nhiều hạn chế ví dụ như khó khăn trong việc bảo trì, tính linh hoạt và khả năng mở rộng kém, đặc biệt với những ứng dụng doanh nghiệp có quy mô lớn. Đó chính là lí do ra đời của kiến trúc Microservices.

Microservice là một 1 cách tiếp cận kiến trúc và tổ chức để phát triển phần mềm mà phần mềm được chia ra thành các thành phần nhỏ hơn và các thành phần đó có thể giao tiếp thông qua API.



Hình 1.10: Phân rã kiến trúc monolith sang microservices



Hình 1.11: So sánh độ phức tạp lúc phát triển của kiến trúc Monolith và Microservices

So sánh 2 mô hình phổ biến Monolith và Microservices:

Monolith	**Microservice**
Với kiến trúc Monolith, tất cả các quy trình được gộp lại và chạy trên 1 service. Có nghĩa là nếu 1 quá trình của ứng dụng được nhiều người sử dụng, thì nhóm phát triển phải mở rộng phần mềm của mình. Thêm vào hoặc cải thiện lại tính năng ứng dụng sử dụng kiến trúc Monolith sẽ trở nên phức tạp hơn. Sự phức tạp này hạn chế thử nghiệm và gây ra khó khăn cho việc thực hiện các ý tưởng mới và rủi ro nếu lỗi một phần chương trình thì cả hệ	Với kiến trúc Microservice, 1 ứng dụng được xây dựng mà không phụ thuộc các thành phần mà chạy từng quy trình ứng dụng dưới dạng service. Các service này sẽ giao tiếp thông qua API. Service được xây dựng cho các khả năng công việc và từng service này thực hiện một chức năng duy nhất. Bởi vì các service chạy độc lập nên dễ dàng cập nhật, triển khai và mở rộng để đáp ứng nhu cầu cho các chức năng cụ thể cho

thống sẽ lỗi.	ứng dụng.

1.3.2. Đặc điểm

- **Decoupling:** Các service trong một hệ thống phần lớn được tách rời. Vì vậy, toàn bộ ứng dụng có thể dễ dàng được xây dựng, thay đổi và thu nhỏ.
- **Componentization:** Microservices được coi là các thành phần độc lập có thể dễ dàng thay thế và nâng cấp.
- **Business Capabilities:** mỗi một thành phần trong kiến trúc microservice rất đơn giản và tập trung vào một nhiệm vụ duy nhất.
- **Autonomy:** các lập trình viên hay các nhóm có thể làm việc độc lập với nhau trong quá trình phát triển.
- **Continous Delivery:** Cho phép phát hành phần mềm thường xuyên, liên tục.
- **Responsibility**
- **Decentralized Governance:** không có mẫu chuẩn hóa hoặc bất kỳ mẫu công nghệ nào. Được tự do lựa chọn các công cụ hữu ích tốt nhất để có thể giải quyết vấn đề.
- **Agility:** microservice hỗ trợ phát triển theo mô hình Agile.

1.3.3. Các đặc trưng

Micro-service: Đặc trưng này được thể hiện ngay ở tên của kiến trúc.Từ microservice được sử dụng để người thiết kế có cách tiếp cận đúng đắn. Một ứng dụng lớn cần chia nhỏ ra nhiều thành phần, các thành phần đó cần tách biệt về mặt dữ liệu (database) và đủ nhỏ cả mặt kích cỡ và độ ảnh hưởng của nó trong hệ thống, khi thêm 1 microservice phải đảm bảo nó đủ nhỏ để dễ dàng tháo gỡ khỏi hệ thống mà không ảnh hưởng tới các thành phần khác.

Tính độc lập: Các microservice hoạt động tách biệt nhau trong hệ thống, việc build 1 microservice cũng độc lập với việc build các microservice khác. Do tính độc lập mà mỗi microservice đều dễ dàng thay thế, mở rộng và phát triển các micro linh động hơn. Các microservice có thể được phát triển bởi các team khác nhau, dùng ngôn ngữ phát triển khác nhau và tiến độ dự án cũng nhanh hơn do không có sự phụ thuộc giữa các team, các team có thể chủ động quản lý phần việc của mình.

Tính chuyên biệt: mỗi microservice là một dịch vụ chuyên biệt, có thể hoạt động độc lập. Mỗi microservice sẽ đại diện cho một tính năng của công ty cung cấp tới khách hàng vì vậy người thiết kế hệ thống microservice phải hiểu rõ hoạt động kinh doanh của công ty và định nghĩa rõ ràng từng microservice.

Phòng chống lỗi: kiến trúc microservice sinh ra để dành cho các hệ thống từ lớn tới vô cùng lớn. Nó sử dụng phương pháp chia để trị, phương pháp này giúp việc giám sát, phòng chống lỗi phần mềm, hệ thống hiệu quả. Khi một thành phần trong hệ thống bị lỗi, nó sẽ được thay thế bởi một thành phần dự bị hay sẽ bị loại bỏ, các thành phần khác vẫn hoạt động bình thường, do vậy hoạt động của toàn hệ thống sẽ không hoặc ít bị ảnh hưởng, gián đoạn.

1.3.4. Ưu và nhược điểm

Ưu điểm

- Dễ dàng phân phối và triển khai các ứng dụng lớn và phức tạp.
- Có thể cải thiện khả năng bảo trì nhờ các service có đặc điểm tương đối nhỏ, dễ hiểu và dễ thay đổi.
- Kiểm thử dễ dàng, phát hiện bug sớm khi các services có quy mô nhỏ.
- Có thể triển khai tốt hơn: các services thường rất dễ cho việc triển khai độc lập.
- Cho phép các services được phát triển nhanh chóng bởi những team khác nhau. Khi đó, mỗi team đều sẽ được phát triển và thử nghiệm để triển khai cũng như mở

rộng được quy mô của dịch vụ của mình một cách độc lập nhất với tất cả các team.

- Nếu như có lỗi xảy ra trong một service thì chỉ có service đó bị ảnh hưởng và các service khác sẽ thực hiện xử lý các yêu cầu cần thiết. Trong khi đó, thì mỗi một thành phần nếu như hoạt động sai của kiến trúc một khối thì nó sẽ làm ảnh hưởng đến toàn bộ hệ thống.
- Lập trình viên có thể thay đổi dễ dàng bằng cách sử dụng công nghệ mới khi triển khai các service. Tương tự như khi có thay đổi lớn thì các service đều có thể thực hiện và bạn dễ dàng thay đổi được công nghệ hơn.

Nhược điểm

- Nhà phát triển thường xuyên phải đối phó với sự phức tạp khi tạo ra một hệ thống phân tán.
- Cần phải triển khai việc tương tác giữa các inter-services
- Bắt lỗi rất phức tạp bởi vì luồng xử lý cần phải đi qua nhiều service khác nhau.
- Khi thực hiện các requests trải rộng trên nhiều service cần đòi hỏi sự phối hợp giữa các team.
- Khó khăn trong việc đảm bảo toàn vẹn cho cơ sở dữ liệu nếu như triển khai theo các cấu trúc dạng phân vùng.
- Việc triển khai và quản lý microservices nếu như làm thủ công theo cách làm với ứng dụng thì sẽ rất phức tạp.
- Lập trình viên cần phải xử lý các sự cố kết nối chậm, lỗi nếu như thông điệp không được gửi hoặc nếu như thông điệp được gửi đến nhiều đích đến vào các thời điểm khác nhau.

1.3.5. Thiết kế phần mềm

Để phát triển một phần mềm theo mô hình kiến trúc Microservice, lập trình viên cần đảm bảo một số yếu tố chính như sau:

- Xây dựng hệ cơ sở dữ liệu (database) độc lập.
- Xác định kích thước service phù hợp.
- Đề ra vai trò, chức năng cụ thể, riêng biệt cho từng service.

Bên cạnh đó, cần tuân thủ 6 nguyên tắc sau:

- Single Responsibility Principle (SRP): Nguyên tắc của một service là có phạm vi và chức năng giới hạn, tập trung vào một nhiệm vụ để quá trình phát triển và triển khai dịch vụ trở nên nhanh chóng hơn.
- Trong quá trình thiết kế, nên xác định và giới hạn các services theo chức năng nghiệp vụ thực tế.
- Đảm bảo microservices có thể phát triển và triển khai độc lập thành từng module.
- Mục tiêu của thiết kế của microservices sẽ phục vụ một nghiệp vụ chứ không chỉ đơn giản làm các dịch vụ nhỏ hơn.
- Kích thước hợp lý của một service là kích thước đủ để đáp ứng yêu cầu của một chức năng trong hệ thống.
- Một microservice không nên có quá nhiều hàm hay chức năng hỗ trợ xung quanh.

Chương II. QUY TRÌNH NHẬN DIỆN BIỂN SỐ

Để có thể tiến hành nhận diện biển số, ta có thể chia giai đoạn nhận diện biển số thành hai giai đoạn chính, đó là:

- Giai đoạn 1: Nhận diện vùng chứa biển số xe.
- Giai đoạn 2: Nhận diện các ký tự có trong biển số.

Các giai đoạn trên được thực hiện dựa vào mô hình YOLOv5, là phiên bản cao hơn mô hình YOLO đã giới thiệu trước đó, như là dễ dàng cài đặt hơn (sử dụng PyTorch) và mô hình tối ưu dẫn đến tốc độ nhận diện và độ chính xác cao hơn. Dựa vào kết quả sau khi training tập dữ liệu sẽ cho ra kết quả nhận diện biển số thông qua các giai đoạn có độ chính xác cao.

1. Nhận diện vùng chứa biển số xe

Trong báo cáo này, tiến hành lấy dữ liệu của 9702 hình ảnh chứa biển số xe, trong đó 7762 hình ảnh dùng để học máy, 970 hình ảnh để làm giá trị và 970 hình ảnh để kiểm tra. Với mỗi hình ảnh đã được đánh dấu biển số sẽ tạo ra một tập tin .txt là nhãn của các biển số tương ứng. Trong tập tin txt này chứa thông tin để nhận dạng biển số xe của hình ảnh đó bao gồm lớp, tọa độ x, y, chiều dài và chiều cao.

Hình 2.1: Gán nhãn cho đối tượng

Sau khi đã thu thập đủ dữ liệu nhãn cho các hình ảnh. Cuối cùng, sẽ thực hiện huấn luyện dữ liệu cho việc học máy để đảm bảo rằng với mỗi dữ liệu đầu vào là hình ảnh bất kỳ có chứa biển số xe máy đều có thể nhận dạng được. Cụ thể việc huấn luyện dữ liệu được thực hiện như sau:

Tất cả quá trình học máy được thực hiện trên google colab. Đầu tiên, tiến hành tải mã nguồn của YOLOv5 trên github. Đây là mã nguồn mở nên có thể sử dụng cho việc huấn luyện dữ liệu cho phần học máy với các tập dữ liệu đã được thu thập từ trước cùng với một số chỉnh sửa để phù hợp với quá trình nghiên cứu cũng như trong việc nhận dạng vùng có biển số xe.

Chỉnh sửa tập tin "coco.yaml" cùng với số lớp được khai báo là 1, vì ở đây để nhận dạng được biển số xe thì chỉ có duy nhất một đối tượng là biển số xe, trong mảng được khai báo, đặt tên phần tử là "plate", cấu hình đường dẫn đến thư mục chứa tập tin dữ liệu để bắt đầu huấn luyện. Đó là công việc cần thiết để có thể huấn luyện dữ liệu dựa vào tập dữ liệu đã thu thập trước đó.

coco128.yaml ×

```
1 train: /content/coco128  # train images (relative to 'path') 128 images
2 val: /content/coco128  # val images (relative to 'path') 128 images
3 test:  # test images (optional)
4
5 # Classes
6 nc: 1  # number of classes
7 names: ['plate']  # class names
```

Hình 2.2: Nội dung tập tin "coco.yaml"

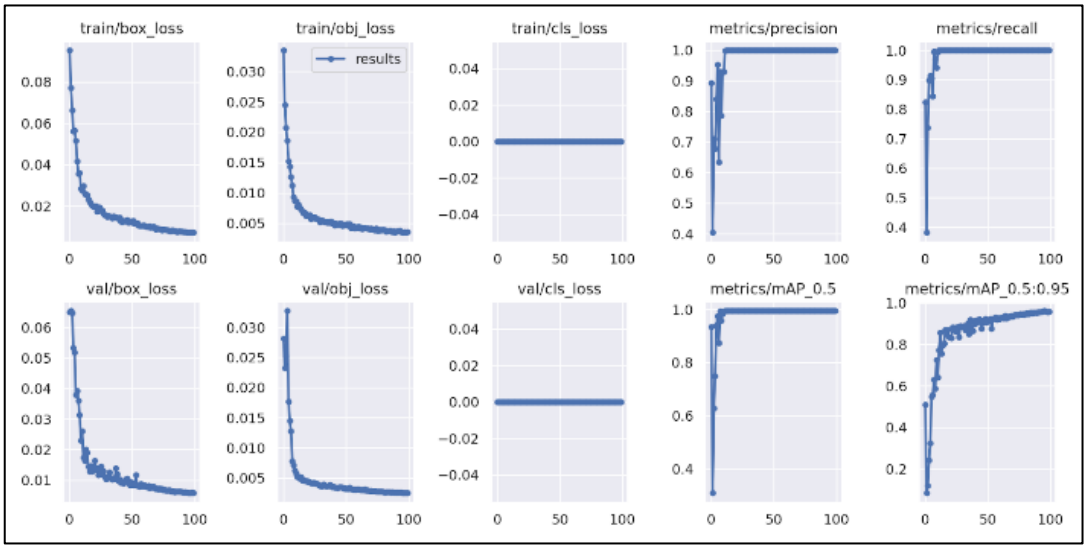
Sau khi chỉnh sửa các nội dung cần thiết trong tập tin, bắt đầu cho quá trình học máy để nhận dạng biển số được diễn ra, để quá trình học máy được thực hiện cho ra kết quả với độ chính xác cao, thiết lập một số thông số cho quá trình học máy như sau: vì đây là quá trình học máy dựa vào tập dữ liệu gồm có hình ảnh và nhãn của từng hình ảnh tương ứng nên thiết lập thông số của "img" là 640 (do tất cả các hình ảnh trong tập dữ liệu có kích thước không lớn hơn 640px), "batch" là 24 (trong quá trình huấn luyện, số

lượng ảnh huấn luyện được cùng một lúc là 24), "epochs" là 100 (số lần lặp lại quá trình huấn luyện là 100), với các thông số được thiết lập ở trên đảm bảo cho quá trình huấn luyện dữ liệu diễn ra cho kết quả với độ chính xác là cao nhất, quá trình học máy được thể hiện như hình:

```
Epoch   gpu_mem       box       obj       cls    labels  img_size
 0/19     5.45G   0.09533   0.03348         0        42       640: 100% 21/21 [00:12<00:00,  1.71it/s]
           Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 11/11 [00:07<00:00,  1.53it/s]
             all        500        500      0.893      0.824      0.935       0.51

Epoch   gpu_mem       box       obj       cls    labels  img_size
 1/19     5.8G   0.07691   0.02448         0        38       640: 100% 21/21 [00:09<00:00,  2.11it/s]
           Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 11/11 [00:05<00:00,  1.90it/s]
             all        500        500      0.624       0.61      0.541       0.104

Epoch   gpu_mem       box       obj       cls    labels  img_size
 2/19     5.8G   0.06467   0.02066         0        44       640: 100% 21/21 [00:09<00:00,  2.14it/s]
           Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 11/11 [00:05<00:00,  1.99it/s]
             all        500        500      0.269      0.706      0.264       0.101

Epoch   gpu_mem       box       obj       cls    labels  img_size
 3/19     5.8G   0.05361   0.01888         0        46       640: 100% 21/21 [00:10<00:00,  2.09it/s]
           Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 11/11 [00:05<00:00,  2.00it/s]
             all        500        500      0.871      0.914      0.953       0.309
```

Hình 2.3: Quá trình huấn luyện dữ liệu

Sau khi huấn luyện học máy với việc xác định biển số xe, kết quả được thể hiện như sau:



Hình 2.4: Biểu đồ đánh giá kết quả huấn luyện dữ liệu

Theo hình trên, nhận thấy rằng mức độ mất mát dữ liệu giảm dần tiệm cận về 0, nghĩa là trong quá trình lặp huấn luyện dữ liệu thì giá trị mất mát giảm dần do trong

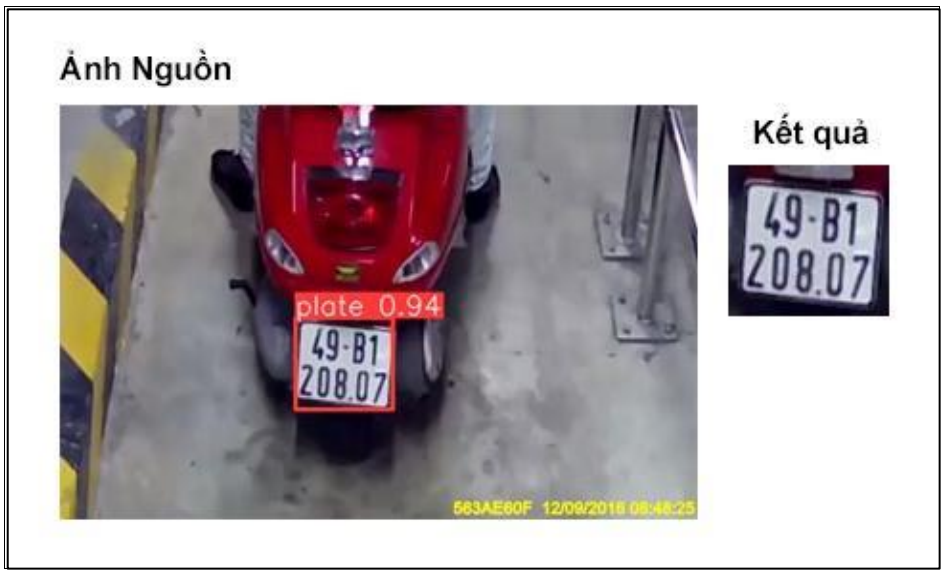
quá trình huấn luyện đã dần nhận dạng được biển thông qua các lần lặp lại trong quá trình huấn luyện dữ liệu, tương tự với giá trị của đối tượng trong biểu đồ mất mát dữ liệu. Do trong quá trình huấn luyện chỉ có một lớp là biển số nên số lớp bị mất mát bằng 0.

Sau khi đã nhận dạng được biển số xe trong một hình ảnh, tiếp tục xử lý vấn đề tách biển số đó khỏi bức ảnh hay nói cách khác là cắt hình có chứa biển số đã được nhận dạng thành một hình ảnh chỉ chứa duy nhất hình ảnh của biển số đó bằng cách sử dụng tọa độ đã cắt có sẵn (coordinates = detect.xyxy[0 trong python).

```
img = cv2.imread(image_src)
detect = self.model_detect(img)
coordinates = detect.xyxy[0][0]
coordinate = [int(coordinates[i]) for i in range(4)]
LpRegion = img[coordinate[1]:coordinate[3],coordinate[0]:coordinate[2]]
return LpRegion
```

Hình 2.5: Mã lệnh cắt hình ảnh biển số

Thep hình, coordinates gồm có x_min,y_min, x_max, y_max, theo đó, tiến hành cắt ảnh theo trục tung (y) từ [y_min,y_max], theo trục hoành (x) từ [x_min,x_max] nên sẽ có hình biển số đã được cắt từ hình ảnh nguồn.



Hình 2.6: Kết quả cắt biển số xe đã được nhận dạng

Với việc sử dụng cách cắt biển số như trên, gần như đã có thể tách được biển số xe khỏi một hình ảnh lớn và chỉ tập trung vào biển số xe máy đó mà thôi. Đây là giai đoạn thiết yếu, cần thiết nhất cho bài toán nhận diện biển số xe. Bởi nếu không có giai đoạn trên thì có thể dẫn đến sự sai lệch về nhận diện các ký tự có trong biển số xe do khung nhìn quá lớn, cũng như các ký tự được nhận dạng cũng có thể không chính xác làm ảnh hưởng đến kết quả nhận dạng các ký tự của biển số xe. Nếu một ký tự được nhận dạng sai hoặc ký tự đó lại không nằm trong vùng của biển số sẽ dẫn đến kết quả nhận dạng không đúng. Điều này là không nên xảy ra, để tránh được vấn đề này và cũng như giải quyết được việc nhận diện các ký tự của biển số xe chỉ nằm trong vùng của biển số thì đây là giai đoạn cực kỳ quan trọng.

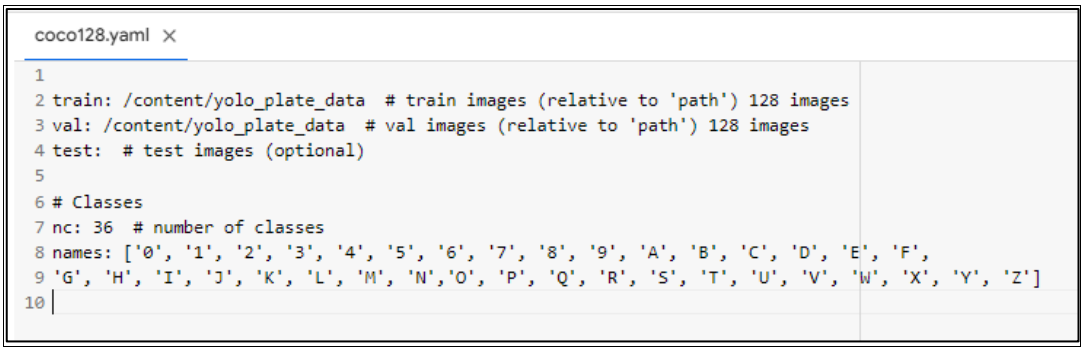
2. Nhận diện các ký tự có trong biển số

Tương tự như việc thực hiện quá trình học máy cho giai đoạn xác định vùng biển số xe ở trên, tiến hành thu thập dữ liệu của 3188 hình ảnh chứa nhãn của các hình ảnh tương ứng (đối với từng hình ảnh đều có nhãn để nhận diện hoặc khoanh vùng các ký tự có trong biển số), để có thể phân biệt được các ký tự có trong biển số xe, chia các ký tự thuộc về các lớp, mỗi lớp là một ký tự khác nhau dùng để phân biệt, tổng số lớp trong tập dữ liệu các ký tự biển số xe cho học máy là 36 lớp bao gồm các ký tự chữ và số (đối với ký tự chữ có 26 ký tự từ a đến z và đối với ký tự số có 10 ký tự từ 0 đến 9).

Sau khi đã thu thập được tập dữ liệu các ký tự có trong biển số xe cho việc học máy, tiến hành huấn luyện bằng cách sử dụng mô hình YOLOv5, dựa vào tập dữ liệu nhãn của các hình ảnh tương ứng, chứa các thông tin là lớp, tọa độ x,y nhỏ nhất và tọa độ x,y lớn nhất là tập dữ liệu phù hợp nhất cho việc huấn luyện học máy sử dụng mô hình YOLOv5. Việc huấn luyện được thực hiện cụ thể như sau:

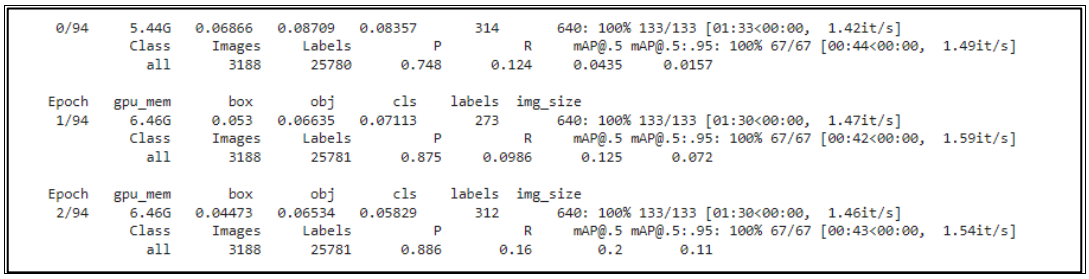
Dựa vào tập dữ liệu đã thu thập từ trước và mã nguồn của mô hình YOLOv5 đã được tải về, tiến hành huấn luyện dữ liệu để có thể tách các ký tự có trong biển số. Để có thể tách được các ký tự có trong biển số với độ chính xác cao, cũng thiết lập một số thông số để có thể huấn luyện dữ liệu cho việc học máy.

Tiến hành chỉnh sửa nội dung tập tin “coco.yaml” để phù hợp với quá trình huấn luyện dữ liệu, các thông tin chỉnh sửa ở đây gồm có đường dẫn đến tập tin dữ liệu dùng để huấn luyện và khai báo các lớp chứa các nhãn của các ký tự, tổng cộng có số lớp là 36, trong mảng các lớp chứa các ký tự chữ từ A đến Z và các ký tự số từ 0 đến 9 như hình sau.



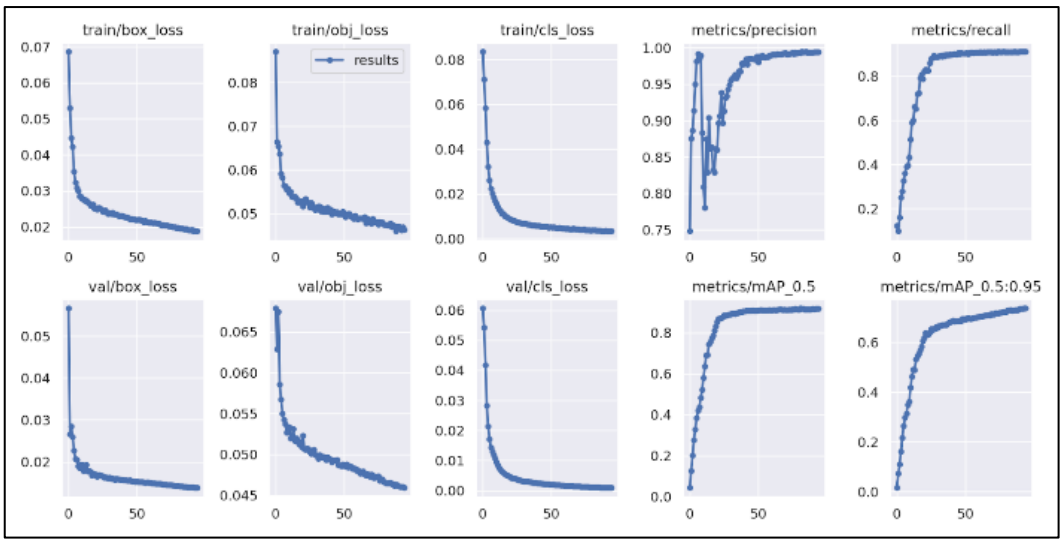
Hình 2.7: Nội dung tập tin “coco.yaml”

Sau khi chỉnh sửa một số thông tin trong nội dung tập tin “coco.yaml” bắt đầu cho quá trình huấn luyện dữ liệu được diễn ra, để có thể cho ra kết quả chính xác nhất về việc nhận dạng các ký tự, thiết lập các thông số cho quá trình huấn luyện cụ thể “img” là 640, batch là 24 và epochs là 95, quá trình huấn luyện diễn ra theo hình:



Hình 2.8: Quá trình huấn luyện dữ liệu nhận diện ký tự

Sau khi huấn luyện học máy thành công, đạt được kết quả thống kê như sau, mức độ mất mát cũng giảm dần tiệm cận về 0 lẫn cả mất mát về các lớp như hình:



Hình 2.9: Biểu đồ đánh giá quá trình huấn luyện nhận dạng ký tự

Class	Images	Labels	P
all	381	3064	0.891
0	381	185	0.989
1	381	409	0.977
2	381	242	0.981
3	381	199	0.966
4	381	203	0.978
5	381	403	0.986
6	381	232	0.977
7	381	229	0.952
8	381	190	0.944
9	381	299	0.968
A	381	35	0.948
B	381	91	0.945
C	381	23	0.794
D	381	10	0.774
E	381	9	0.599
F	381	43	0.795
G	381	33	0.835
H	381	19	0.502
I	381	3	1
J	381	5	1
K	381	18	1
L	381	24	0.889
M	381	8	1
N	381	8	0.531
P	381	18	0.578
R	381	8	1
S	381	23	0.789
T	381	23	0.949
U	381	11	0.835
V	381	13	0.985
X	381	10	1
Y	381	7	1
Z	381	31	0.943

Hình 2.10: Đánh giá độ chính xác từng ký tự

Kết quả nhận phân tách các kí tự như hình dưới đây:

Hình 2.11: Kết quả nhận phân tách các ký tự

Hình 2.12: Kết quả kết hợp nhận diện vùng biển số và phân tách các ký tự

Kết quả thu được từ 2 quá trình nhận diện cho thấy răng khả năng học máy của mô hình YOLO có độ chính xác khá cao (trên 90%). Nó chứng tỏ rằng YOLO là một mô hình học máy nhận diện vật thể mạnh mẽ nhất hiện tại mã nguồn mở mà ta có thể sử

dụng. Từ đó có thể áp dụng vào trong thực tiễn trong việc quản lý hệ thống xe trong thực tế ở các chung cư, tòa nhà, các khu trung tâm tổng hợp và trường học.

Chương III. CÁC GIAI ĐOẠN NHẬN DIỆN KHUÔN MẶT

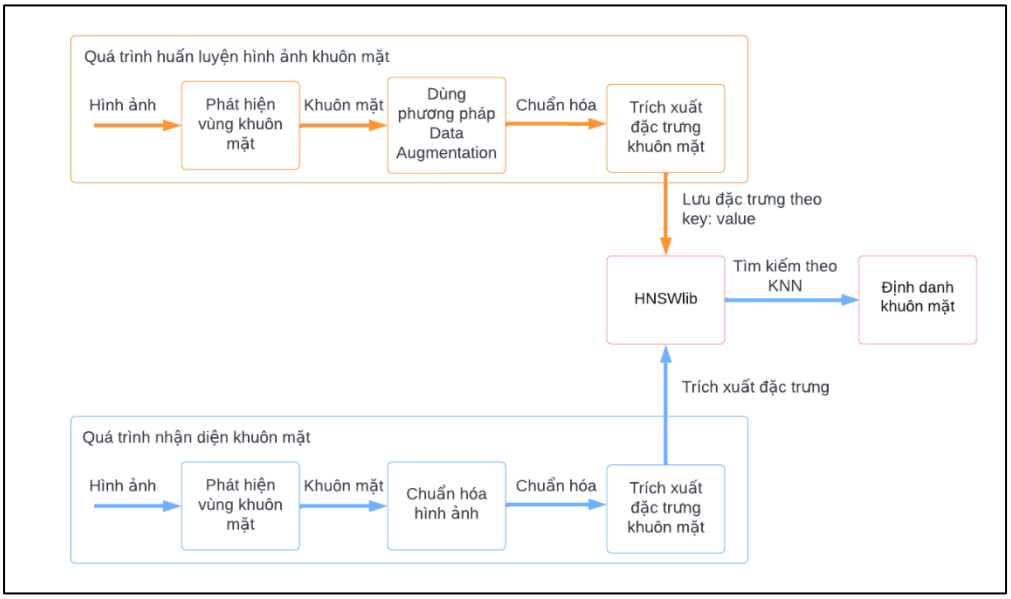
1. Nhận diện vùng chứa khuôn mặt

Nhận diện vùng chứa mặt là quá trình tìm kiếm và xác định vị trí của khuôn mặt trong các bức ảnh hoặc video. Quá trình này thường được thực hiện bằng cách sử dụng các mô hình deep learning như MTCNN. Một số thuật toán khác như Haar Cascades, Viola-Jones cũng được sử dụng trong phát hiện khuôn mặt.

2. Nhận diện khuôn mặt

Nhận diện khuôn mặt là quá trình nhận dạng và phân loại các khuôn mặt đã được phát hiện trong các bức ảnh hoặc video. Quá trình này thường được thực hiện bằng cách sử dụng các mô hình deep learning như FaceNet. Các mô hình khác như VGG-Face, DeepFace cũng được sử dụng trong nhận diện khuôn mặt.

Một số giai đoạn chính trong quá trình nhận diện khuôn mặt bao gồm:

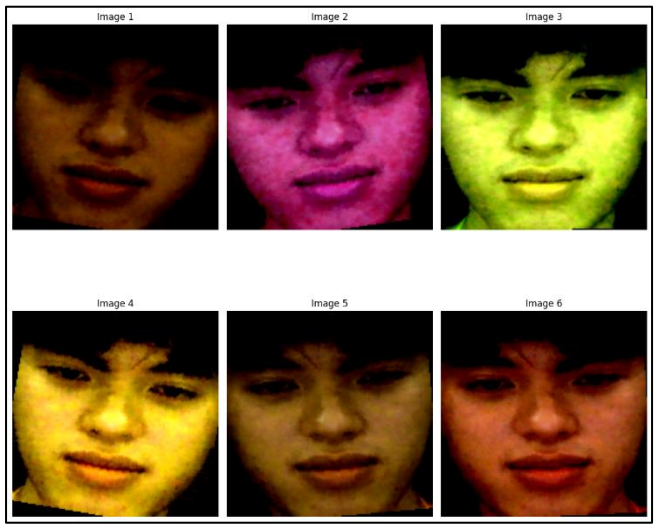


Hình 3.1: Quá trình huấn luyện và nhận diện khuôn mặt

Như hình trên, quá trình huấn luyện và nhận diện khuôn mặt gồm các bước sau:

2.1. Quá trình huấn luyện hình ảnh khuôn mặt:

Phát hiện vùng khuôn mặt thì sử dụng MTCNN để phát hiện khuôn mặt và dùng data augmentation để có thể chuẩn hóa nhiều dạng ảnh khuôn mặt đã được cắt như hình sau:



Hình 3.2: Phương pháp Data Augmentation đa dạng hóa quá trình huấn luyện

Trong quá trình dùng phương pháp Data Augmentation, đã chỉnh các thông số như hình sau:

```
Resize (160, 160)

RandomHorizontalFlip
Probability: 0.5

RandomRotation
Degrees: [-10.0, 10.0]

ColorJitter
Brightness: (0.8, 1.2)
Contrast: (0.8, 1.2)
Saturation: (0.8, 1.2)
Hue: (-0.2, 0.2)

ToTensor

Normalize
```

Hình 3.3: Thông số chuẩn hóa hình ảnh và thay đổi hình ảnh

Trong quá trình tiền xử lý ảnh, sau khi cắt ảnh từ MTCNN sẽ được chuẩn hóa kích thước và tăng cường dữ liệu. Đầu tiên, sẽ sử dụng Resize để điều chỉnh kích thước ảnh về 160x160 pixel. Việc này giúp đảm bảo sự ổn định và hiệu quả trong việc huấn luyện mô hình, vì tất cả các ảnh có kích thước chuẩn 160x160 pixel

Sau đó sẽ dùng RandomHorizontalFip sẽ lật ảnh theo phương ngang với tỉ lệ 50%, trong quá trình chuẩn hóa sẽ có 50% khả năng sẽ lật ngang. Phép biến đổi này giúp mô hình học cách xử lý các biến thể về hướng nhìn và tăng cường khả năng tổng quát hóa của mô hình.

Tiếp theo, dùng ColorJitter có các tham số cụ thể để điều chỉnh các thuộc tính màu sắc của ảnh. Dưới đây là giải thích về các tham số cụ thể theo hình trên:

- Brightness (Độ sáng): Tham số này được đặt trong khoảng từ 0.8 đến 1.2. Nó điều chỉnh độ sáng của ảnh. Giá trị nhỏ hơn 1.0 sẽ làm cho ảnh tối hơn, trong khi giá trị lớn hơn 1.0 sẽ làm cho ảnh sáng hơn. Khoảng giá trị cho phép tạo ra các biến thể về độ sáng khác nhau trong quá trình huấn luyện.

- Contrast (Độ tương phản): Tham số này cũng được đặt trong khoảng từ 0.8 đến 1.2. Nó điều chỉnh độ tương phản của ảnh. Giá trị nhỏ hơn 1.0 sẽ làm cho tương phản giảm, trong khi giá trị lớn hơn 1.0 sẽ làm tăng tương phản. Điều này giúp tạo ra các biến thể về độ tương phản khác nhau trong quá trình huấn luyện.

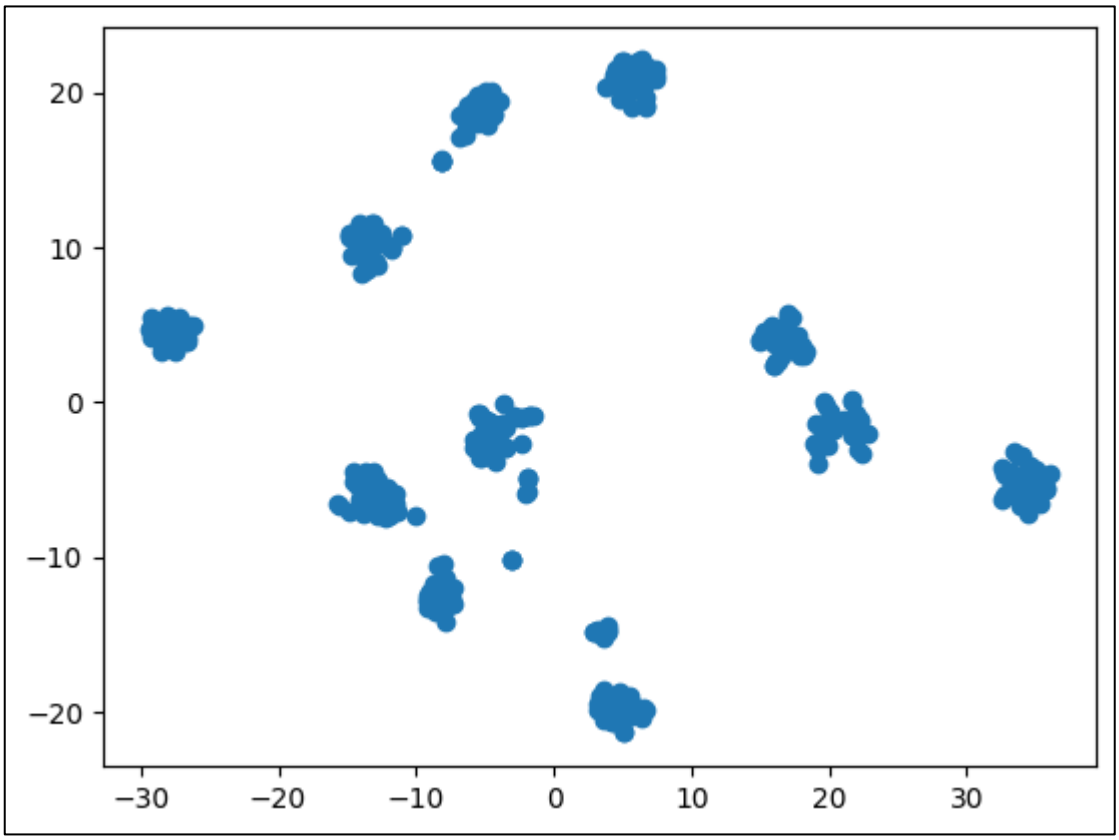
- Saturation (Độ bão hòa): Tham số này cũng nằm trong khoảng từ 0.8 đến 1.2. Nó điều chỉnh độ bão hòa màu sắc của ảnh. Giá trị nhỏ hơn 1.0 sẽ làm mất màu sắc và giảm độ bão hòa, trong khi giá trị lớn hơn 1.0 sẽ làm tăng độ bão hòa. Điều này giúp tạo ra các biến thể về độ bão hòa màu sắc khác nhau trong quá trình huấn luyện.

- Hue (Màu sắc chủ đạo): Tham số này được đặt trong khoảng từ -0.2 đến 0.2. Nó điều chỉnh màu sắc chủ đạo của ảnh. Giá trị dương sẽ làm tăng màu sắc chủ đạo,

trong khi giá trị âm sẽ làm giảm màu sắc chủ đạo. Điều này giúp tạo ra các biến thể về màu sắc chủ đạo khác nhau trong quá trình huấn luyện.

Cuối cùng, dùng Normalize để cho các giá trị pixel 0 tới 1 để có thể cho phép quá trình huấn luyện nhanh hơn

Xong giai đoạn chuẩn hóa thì tới giai đoạn trính xuất đặc trưng, dùng FaceNet để trích xuất đặt trưng của các ảnh đã được chuẩn hóa để ra các embedding và lưu vào trong HNSWlib với key là tên người đã được hash và value là embedding của khuôn mặt đã được trích xuất.



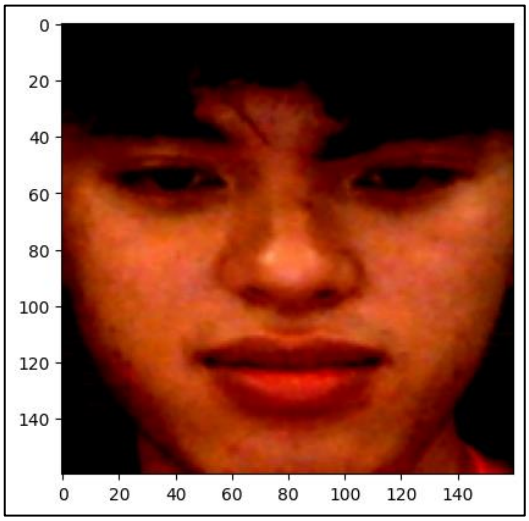
Hình 3.4: Embedding khi được FaceNet rút ra đặc trưng và lưu vào HNSWlib

2.2. Quá trình huấn luyện hình ảnh khuôn mặt

Như bước của quá trình huấn luyện hình ảnh khuôn mặt, Phát hiện vùng khuôn mặt thì sử dụng MTCNN để phát hiện khuôn mặt và chuẩn hóa ảnh khuôn mặt đã được cắt thành 160x160 pixel như thông số sau:

```
Resize (160, 160)

ToTensor

Normalize
```

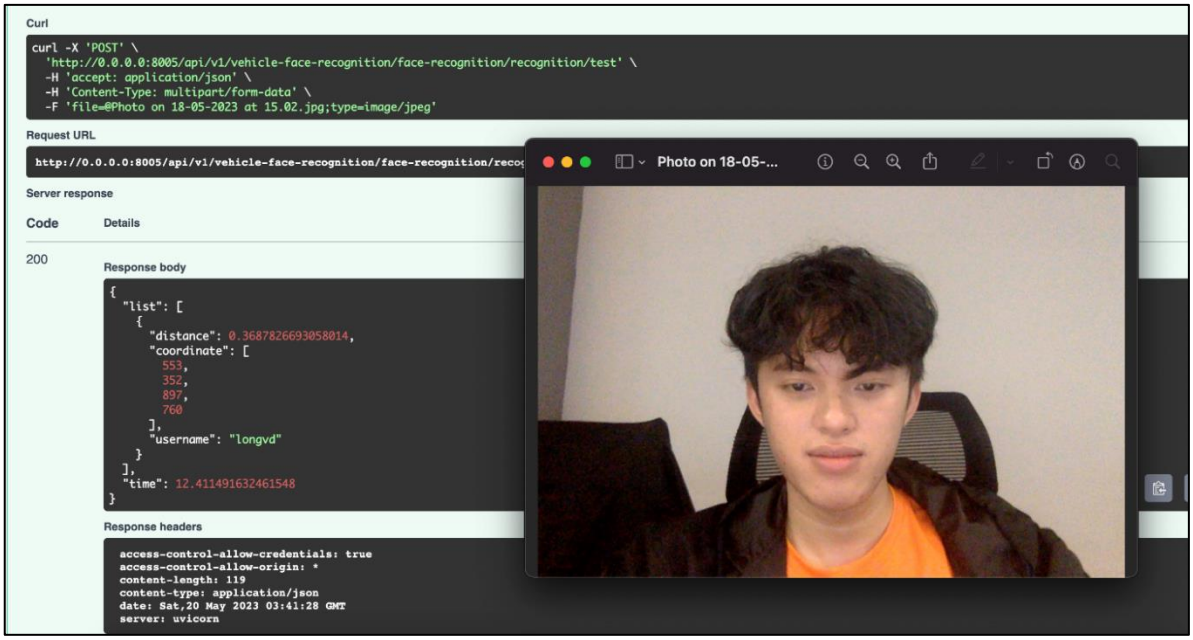
Hình 3.5: Chuẩn hóa hình ảnh và điều chỉnh ảnh về 160x160px



Hình 3.6: Hình ảnh cắt đã được chuẩn hóa

Tiếp theo, dùng FaceNet để trích xuất đặt trưng của các ảnh đã được chuẩn hóa để ra embedding và dùng embedding để truy vấn theo KNN trong HNSWlib để ra key đã được hash và so với cơ sở dữ liệu để hiển thị tên của người dùng.

2.3. Kết quả và đánh giá



Hình 3.7: Kết quả nhận được sau khi nhận diện khuôn mặt

Với kết quả trả về, distance chính là mức độ nhận diện gần đúng của quá trình nhận diện khuôn mặt, với distance càng nhỏ thì độ chính xác về thông tin chủ xe sẽ càng cao. Với các tọa độ, có thể nhận diện được vùng chứa khuôn mặt thông qua hình ảnh thu được từ camera.

Tuy nhiên, việc nhận diện khuôn mặt còn gặp phải nhiều thách thức như ánh sáng môi trường, độ phân giải ảnh, biến dạng khuôn mặt và chính sách bảo mật dữ liệu. Do đó, việc sử dụng các mô hình nhận diện khuôn mặt phải được cân nhắc kỹ lưỡng và tuân thủ các quy định về quyền riêng tư và bảo mật dữ liệu.

Chương IV. XÂY DỰNG HỆ THỐNG

1. Phân tích

Hệ thống quản lý bãi đậu xe thông minh là hệ thống ứng dụng công nghệ thông tin vào hoạt động kiểm soát các phương tiện ra vào bãi đỗ xe. Hệ thống này bao gồm: camera giám sát, cảm biến an toàn, chốt bảo vệ, bảng LED điều khiển, thẻ từ nhận diện.

Hiện nay, việc xây dựng mô hình quản lý các phương tiện giao thông tại các bãi đậu xe đang là vấn đề cấp thiết và cực kỳ quan trọng. Đây là điều kiện tất yếu để có thể giúp quản lý các phương tiện tại các bãi đậu xe dễ dàng hơn, đảm bảo an ninh trong khu vực quản lý, một số yếu tố trong hỗ trợ việc quản lý các phương tiện tại các bãi đỗ xe bao gồm:

- Cảm biến: Hệ thống quản lý bãi đỗ xe thông minh sử dụng các cảm biến để phát hiện và ghi nhận thông tin về việc đỗ xe, bao gồm cả việc xe đỗ và rời khỏi bãi đỗ. Cảm biến này có thể là cảm biến hồng ngoại, cảm biến áp suất, cảm biến từ, hoặc các công nghệ khác để đảm bảo việc thu thập dữ liệu chính xác về tình trạng bãi đỗ xe.
- Mạng kết nối: Hệ thống sử dụng mạng kết nối để truyền tải thông tin từ các cảm biến đến trung tâm quản lý. Mạng kết nối có thể là mạng không dây hoặc mạng có dây để đảm bảo việc truyền thông nhanh chóng và ổn định.
- Trung tâm quản lý: Trung tâm quản lý là nơi tổng hợp và xử lý dữ liệu từ các cảm biến. Nó cung cấp thông tin về tình trạng của bãi đỗ xe, số lượng chỗ trống còn lại và hướng dẫn người dùng tìm kiếm chỗ đỗ xe trống. Trung tâm quản lý cũng có khả năng giám sát và điều khiển các thiết bị trong hệ thống.
- Ứng dụng di động: Hệ thống quản lý bãi đỗ xe thông minh cung cấp ứng dụng di động cho người dùng để tìm kiếm thông tin về bãi đỗ xe gần nhất, xem tình

trạng chỗ trống và thực hiện thanh toán. Ứng dụng di động cũng có thể cung cấp thông báo và hướng dẫn đi đến bãi đỗ xe đã chọn.

- Hệ thống thanh toán: Hệ thống quản lý bãi đỗ xe thông minh có khả năng tích hợp các phương thức thanh toán khác nhau, bao gồm thanh toán tiền mặt, thẻ tín dụng hoặc ví điện tử. Người dùng có thể thanh toán trực tiếp từ ứng dụng di động

Trên thực tế, tại các bãi đỗ xe, để có thể thực hiện được việc quản lý các phương tiện, cần rất nhiều yếu tố kể trên, các yếu tố đó đều đóng vai trò quan trọng và là một phần của hệ thống quản lý bãi đậu xe thông minh. Hầu hết các hệ thống quản lý bãi đậu xe thông minh đều thực hiện phương pháp trên và sử dụng đầy đủ các yếu tố đó.

Một số mô hình tại các hệ thống quản lý bãi đậu xe trên thực tế như sau [6]:

Hình 4.1: Mô hình quản lý bãi đậu xe của VinParking

Với mô hình như hình 4.1 sẽ bao gồm các thiết bị sau:

- Máy vi tính là nơi cài đặt phần mềm, lưu dữ liệu, "giao tiếp" với người sử dụng, hay nói cách khác máy vi tính kết nối, điều khiển, nhận tín hiệu từ các thiết bị, thông báo và xử lý các tác vụ của người dùng.
- Camera dùng để chụp hình người điều khiển xe và hình biển số xe mỗi lượt xe vào - ra, với máy giữ xe thông minh 2 làn xe (1 làn vào + 1 làn ra) sẽ bao gồm 4 camera, trong đó: 2 camera chụp biển số xe và 2 camera chụp người điều khiển xe. Camera thường đặt lắp đặt trụ dưới đất hoặc trên trần, khoảng cách từ camera đến tủ giữ xe vào khoảng 2.5 mét, đây cũng là khoảng cách tốt nhất để camera có thể nhận dạng được biển số xe.
- Đầu đọc thẻ không tiếp xúc sẽ được lắp đặt 2 bên của tủ giữ xe để tiện cho việc người dùng quẹt thẻ, tuy nhiên phụ thuộc vào thực tế địa hình, độ rộng cổng vào ra để chọn vị trí lắp đặt đầu đọc thẻ sao cho thuận tiện việc quẹt thẻ.
- Thẻ giữ xe: là loại thẻ chip, được làm bằng chất liệu mica, có độ bên cao và chống chịu được khi nhiệt độ và thời tiết thay đổi. Với những ưu điểm vật lý đó, thẻ giữ xe có thể tái sử dụng nhiều lần (khách hàng giữ thẻ khi gửi xe và trả lại khi lấy xe) giúp tiết kiệm chi phí so với giữ xe truyền thống (sử dụng vé giấy). Một ưu điểm khác của thẻ giữ xe là thẻ được thiết kế và in nội dung theo yêu cầu của khách hàng, đảm bảo thẩm mỹ và tránh nhầm lẫn với thẻ của các hệ thống khác (thang máy, bãi giữ xe khác,…)
- Tủ giữ xe: Được thiết kế chuyên dụng cho hệ thống máy giữ xe thông minh, tủ giữ xe là nơi lắp đặt các thiết bị như: máy vi tính, màn hình LCD, đầu đọc thẻ, bộ lưu điện và các thiết bị phụ trợ khác. Tủ giữ xe cũng giúp bảo vệ các thiết bị khỏi bụi, nước và tránh các va chạm không mong muốn. Ngoài ra, vỏ tủ được sơn tĩnh điện tránh các sự cố rò điện.

- Phần mềm quản lý bãi xe: Được xem như "bộ não" của hệ thống máy giữ xe thông minh, phần mềm quản lý bãi xe Vinparking tiếp nhận và xử lý các thao tác của người dùng, quản lý dữ liệu, thống kê lập báo cáo,…
- Các thiết bị và vật tư phụ: bao gồm gờ giảm tốc cao su: giúp xe dừng đúng vị trí, bộ lưu điện dự phòng: giúp hệ thống hoạt động ngay cả khi mất điện, trụ lắp đặt camera, hàng rào sắt: giúp phân làn xe vào – ra.

Ngày nay, với sự phát triển của khoa học công nghệ và các kỹ thuật tiên tiến, hiện đại, việc áp dụng các kỹ thuật đó vào đời sống của con người là thật sự cần thiết, bởi những việc đó không chỉ giúp cho cuộc sống của con người được cải thiện mà còn làm cho xã hội phát triển nhanh chóng, góp phần thúc đẩy sự tiến bộ của khoa học, kỹ thuật cũng như bắt kịp với xu hướng của thời đại.

Song song với quá trình xây dựng mô hình bãi đậu xe thông minh, trong báo cáo này sẽ tiến hành xây dựng một ứng dụng với các chức năng cơ bản để áp dụng mô hình này trong việc quản lý các phương tiện tại các bãi đậu xe, đặc biệt là ứng dụng dùng để nhận diện biển số xe và nhận diện khuôn mặt.

Nhận diện biển số xe là một phần quan trọng trong việc quản lý các phương tiện giao thông. Bên cạnh việc sử dụng nhận diện để tiến hành các công việc như thu phí xe ở bãi đậu xe một cách tự động, ứng dụng của nhận diện biển số xe cũng được áp dụng rất nhiều ở nhiều lĩnh vực khác nhau. Đặc biệt là trong ứng dụng giao thông thông minh giúp việc quản lý các phương tiện dễ dàng hơn dựa vào nhận diện biển số chỉ thuộc duy nhất của một phương tiện đó

Những nghiên cứu trước đây cho thấy rằng: sử dụng các phương pháp cho nhận diện biển số xe đã được áp dụng, các phương pháp này tuy có một số đạt độ chính xác không cao nhưng hầu hết đáp ứng được mức độ xử lý của người dùng, đặc biệt là nhận diện biển số xe có độ chính xác trong mức độ cho phép. Các phương pháp ở đây chính là phân tích hình ảnh, phân tích các ký tự chữ và số,... Mỗi phương pháp có một nhiệm

vụ riêng, do yêu cầu của việc nhận diện biển số xe nên cũng có rất nhiều phương pháp kết hợp với nhau để đảm bảo cho quá trình nhận diện biển số một cách chính xác. Cũng như thế, trong bài báo cáo này, đã thực hiện nhận diện biển số xe bằng cách sử dụng hình ảnh được chụp có biển số xe, sử dụng phương pháp phân tích hình ảnh được chụp từ một chiếc xe, phân tích các ký tự có trong biển số đó.

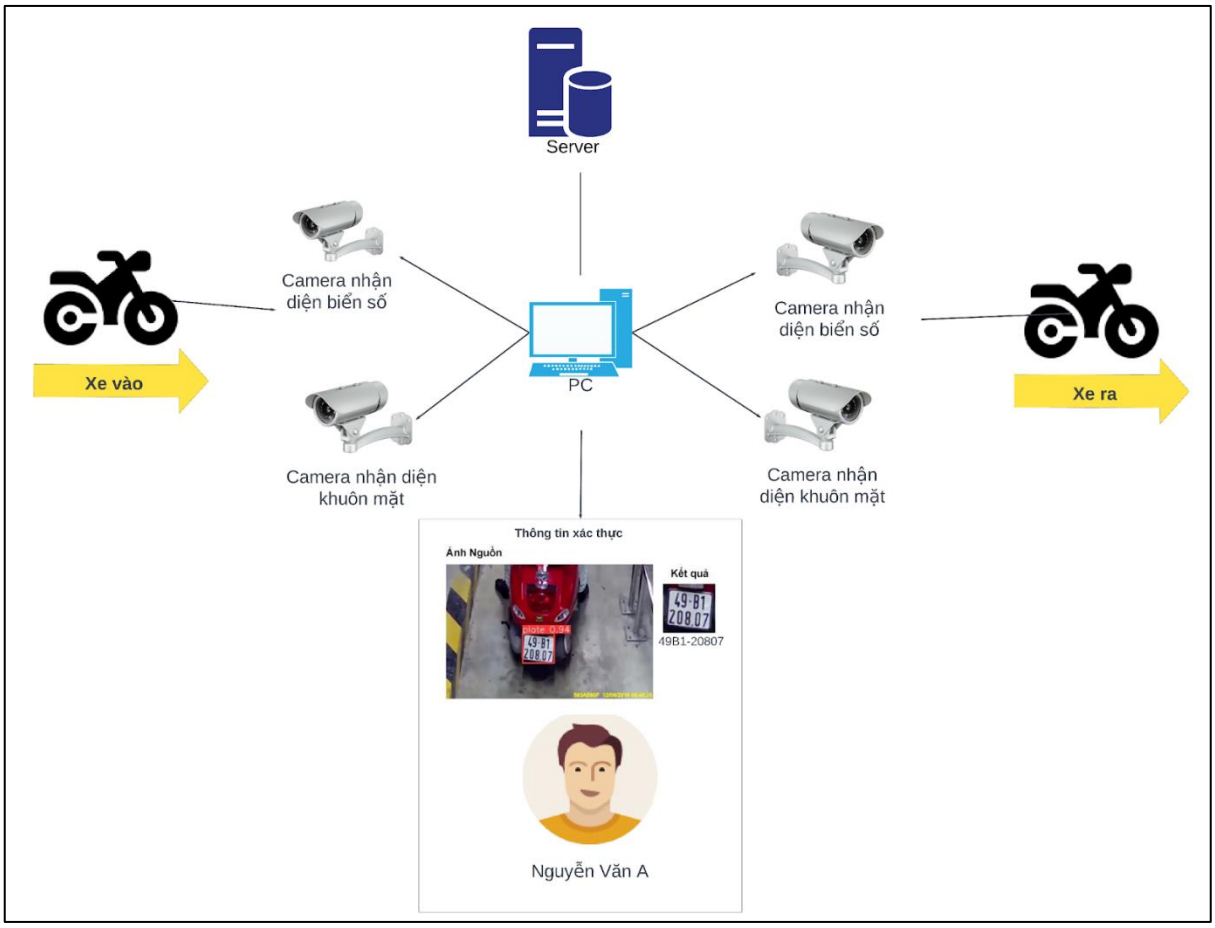
Để thực hiện được các bước nêu ở trên, tiến hành thu thập dữ liệu các ký tự cũng như các chữ số để chuẩn bị cho phần học máy (machine learning), không chỉ có các ký tự chữ và số, mà còn có cả vị trí cũng như kích thước của biển số của từng loại phương tiện, đó là điều kiện tất yếu để có thể thực hiện được việc nhận diện biển số xe thông qua hình ảnh được cung cấp. Áp dụng mô hình khác nhau cho việc nhận diện biển số, cụ thể trong bài báo cáo này, sử dụng mô hình là YOLOv5. Dựa vào các phương pháp đã sử dụng, tiến hành đánh giá kết quả thực hiện cũng như so sánh mức độ chính xác của các phương pháp đó, cuối cùng là đánh giá mô hình được sử dụng cho quá trình nghiên cứu.

2. Mô hình

Trong báo cáo này, tiến hành xây dựng một mô hình quản lý bãi đậu xe thông minh với các yếu tố đơn giản. Các yếu tố được sử dụng trong hệ thống nhận quản lý bãi đậu xe gồm có: camera, phần mềm quản lý, máy vi tính. Với các yếu tố kể trên, chủ yếu thực hiện việc nhận diện biển số xe kết hợp với khuôn mặt để cấp phép và cho phương tiện đó có được phép ra / vào khu vực của bãi đậu xe hay không. Việc triển khai mô hình này tuy không được như các mô hình quản lý bãi đậu xe của các nhà phân phối lớn, nhưng phần nào cũng có thể được áp dụng cho các ứng dụng nhỏ, phù hợp với các cơ sở với mức chi phí không cao. Với mô hình này, có thể giải quyết được các vấn đề như quản lý các phương tiện khi vào khu vực bãi đậu xe sẽ được lưu vào hệ thống và dễ dàng truy xuất dữ liệu. Ngoài ra, việc truy xuất dữ liệu sẽ đa dạng hơn với các lựa chọn như trong một khoảng ngày, khoảng thời gian, khu vực,... Để đảm bảo an ninh trong hệ thống quản lý bãi đậu xe, trong trường hợp cho phép các phương tiện ra khỏi

bãi cũng cần phải đáp ứng được các điều kiện về xác thực bao gồm biển số xe và thông tin khuôn mặt được nhận diện thông qua camera phải trùng khớp với lúc xe vào bãi, thì lúc đó mới đủ điều kiện cho phép phương tiện rời khỏi bãi đậu xe.

Mô hình quản lý bãi đậu xe thông minh



Hình 4.2: Mô hình quản lý bãi đậu xe thông minh

Theo mô hình ở trên, sẽ sử dụng tổng cộng bốn camera để hỗ trợ cho quá trình hoạt động của hệ thống quản lý bãi đậu xe thông minh. Với mỗi camera sẽ có nhiệm vụ riêng để quản lý dữ liệu đầu vào của hệ thống.

Khi phương tiện bắt đầu vào bãi đậu xe, sẽ có hai camera dùng để thực hiện hai công việc, một cái dùng để nhận diện biển số của phương tiện và gửi vào hệ thống, một cái dùng để nhận diện khuôn mặt của chủ phương tiện và sau đó cũng được gửi vào hệ

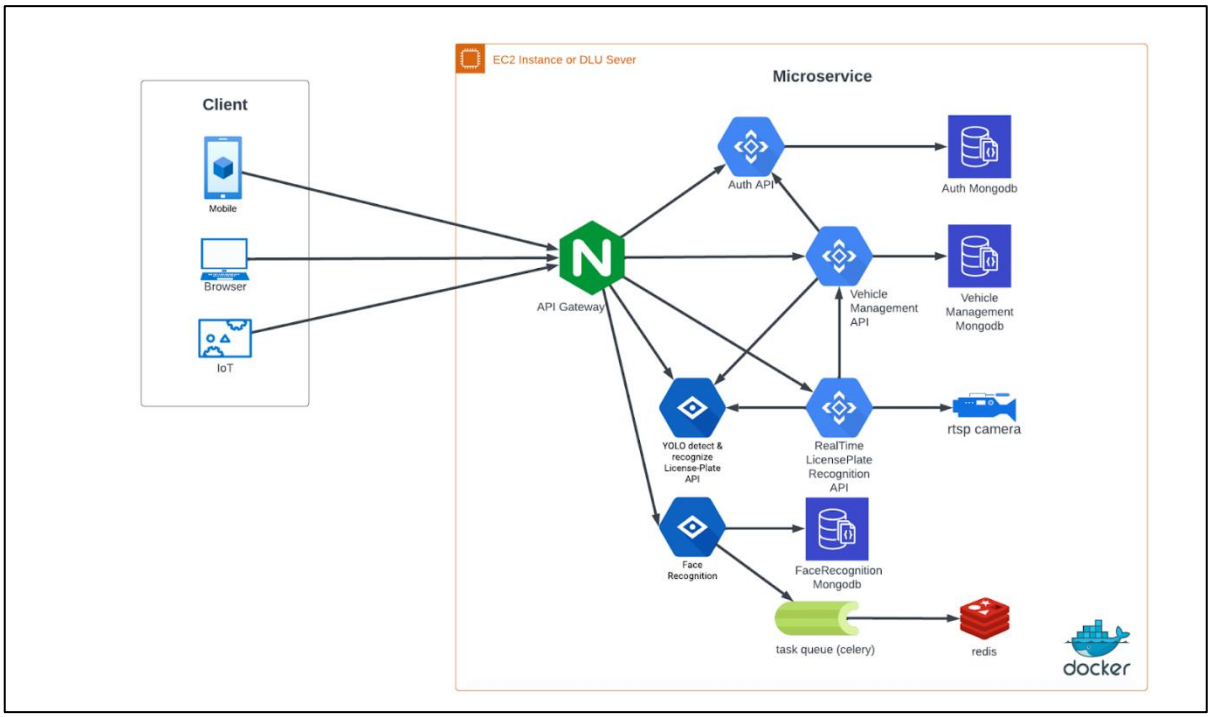
thống. Tại đây, hệ thống sẽ tiến hành xác thực, so sánh giữa hai thông tin được nhận vào từ hai camera thu được và khi dữ liệu trùng khớp, nghĩa là dữ liệu khuôn mặt và dữ liệu biển số xe của chính chủ xe đó, nếu đã có đăng ký chủ xe từ trước, thì sẽ tiến hành cho phép phương tiện đó vào bãi đậu xe. Ngược lại, trong trường hợp dữ liệu đầu vào nhận được từ camera so sánh với nhau và khi có kết quả không trùng khớp sẽ không cho phép phương tiện đó tiến vào bãi đậu xe và có thể cảnh báo cho người quản lý để đảm bảo an ninh. Với mỗi lần phương tiện được phép vào bãi đậu xe hệ thống sẽ ghi nhận là đã vào thành công và lưu thời gian vào hệ thống để phục vụ cho việc thống kê các phương tiện vào / ra trong bãi đậu xe và chi tiết thời gian của, thông tin của phương tiện đó,...

Khi phương tiện bắt đầu rời bãi đậu xe, tại đây sẽ có hai camera, một cái được sử dụng để nhận diện biển số xe lúc muốn rời khỏi bãi đậu xe, một cái được sử dụng để nhận diện khuôn mặt của chủ phương tiện lúc muốn rời khỏi bãi đậu xe, tương tự như lúc xác thực cấp phép xe vào bãi đậu. Tuy nhiệm vụ của hai camera lúc ra bãi đậu xe so với hai camera lúc vào bãi đậu xe là giống nhau, song chức năng xử lý đối với mỗi hình ảnh từ camera trong cặp camera vào hoặc ra lại hoàn toàn khác nhau, vì mỗi camera trong cặp này sẽ đưa ra dữ liệu đầu vào với mục đích là nhận diện biển số và nhận diện khuôn mặt, điều này sẽ giúp cho hệ thống quản lý bãi đậu xe thông minh dễ dàng quản lý và dễ dàng phát triển trong tương lai. Mỗi cặp camera thuộc hai loại khác nhau sẽ thực hiện nhiệm vụ riêng, một cặp dùng để nhận diện phương tiện vào bãi đỗ và một cặp dùng để nhận diện phương tiện ra khỏi bãi đỗ.

Trong báo cáo này, sử dụng một số nền tảng để phát triển phần mềm quản lý hệ thống bãi đậu xe thông minh như: web, mobile. Ứng dụng web dùng để quản lý thông tin người dùng và dùng để thống kê số lượt vào / ra của các phương tiện, được dùng bởi người quản lý hệ thống bãi đậu xe thông minh. Ứng dụng mobile dùng để đăng ký tài khoản và liên kết với phương tiện, thiết lập một số thông tin phục vụ cho quá trình nhận diện, phát hiện chủ sở hữu của phương tiện một cách dễ dàng.

Dựa vào mô hình đã xây dựng, sẽ tiến hành triển khai và áp dụng mô hình này vào quá trình hoàn thiện các chức năng cơ bản của hệ thống quản lý bãi đậu xe. Để từ đó đưa ra nhận xét và đánh giá giúp cải thiện mô hình này trong tương lai.

3. Kiến trúc



Hình 4.3: Kiến trúc Microservice

Trong quá trình phát triển hệ thống, kiến trúc được thiết kế với mục đích phục vụ cho việc quản lý và theo dõi phương tiện giao thông. Các chức năng chính của hệ thống bao gồm:

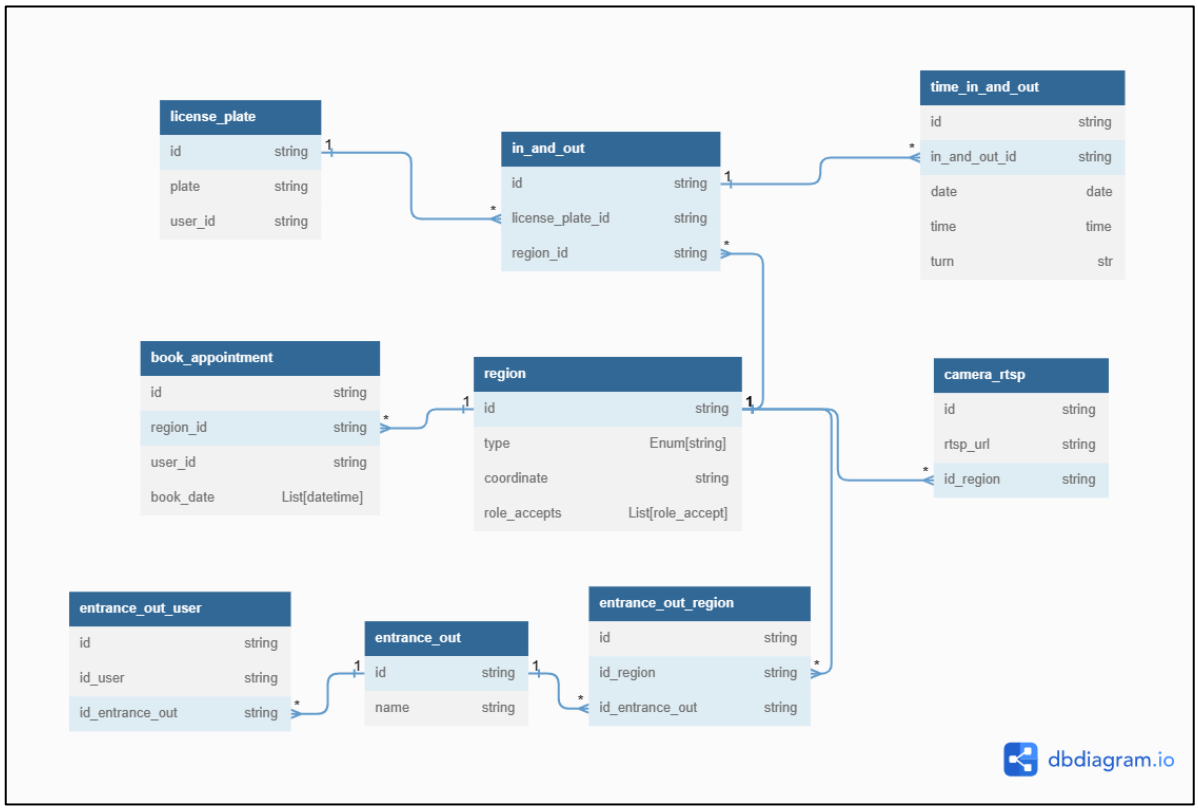
- Phát hiện và nhận dạng biển số xe trong thời gian thực.
- Thống kê xe ra vào trong ngày và thời gian cho trước.
- Đăng kí vào khu vực trong khoảng thời gian cho trước.
- Sử dụng websocket để có thể server gửi về client biển số xe đã nhận diện được trong thời gian thực.

- Sử dụng nhận diện khuôn mặt để xác định có phải chủ xe khi ra khỏi bãi đổ xe.
- Sử dụng giao thức RTSP để truyền tải hình ảnh từ camera qua các client.

Để tối ưu hóa hệ thống, sẽ tiến hành các phương pháp sau. Trước tiên, sẽ thiết lập vị trí của camera sao cho phù hợp nhất với các khu vực cần quan sát. Kế đến, sẽ tối ưu phương pháp các bước ứng dụng phát hiện và nhận diện biển số xe để đảm bảo hiệu suất cao và chính xác. Cuối cùng, sẽ tối ưu hóa cơ sở dữ liệu để có thể ghi lại các phương tiện đi ra vào một cách chính xác và có hiệu quả.

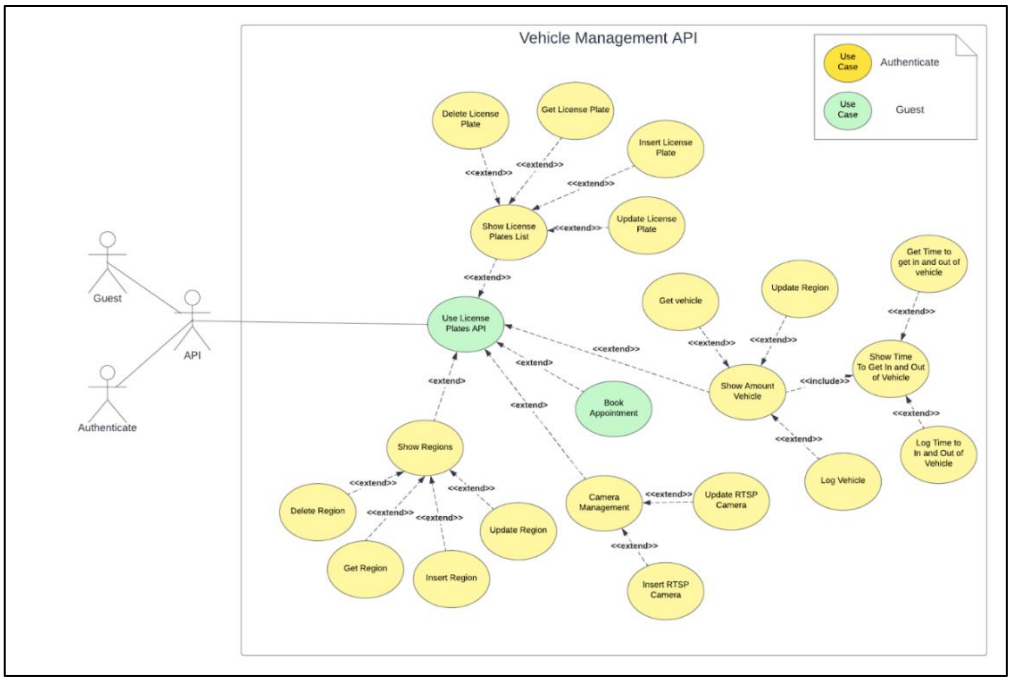
Với kiến trúc và các phương pháp tối ưu trên, tin rằng hệ thống sẽ đem lại giá trị lớn cho quản lý và theo dõi phương tiện giao thông, đồng thời cũng đáp ứng được nhu cầu của người dùng.

Cơ sở dữ liệu về người dùng:



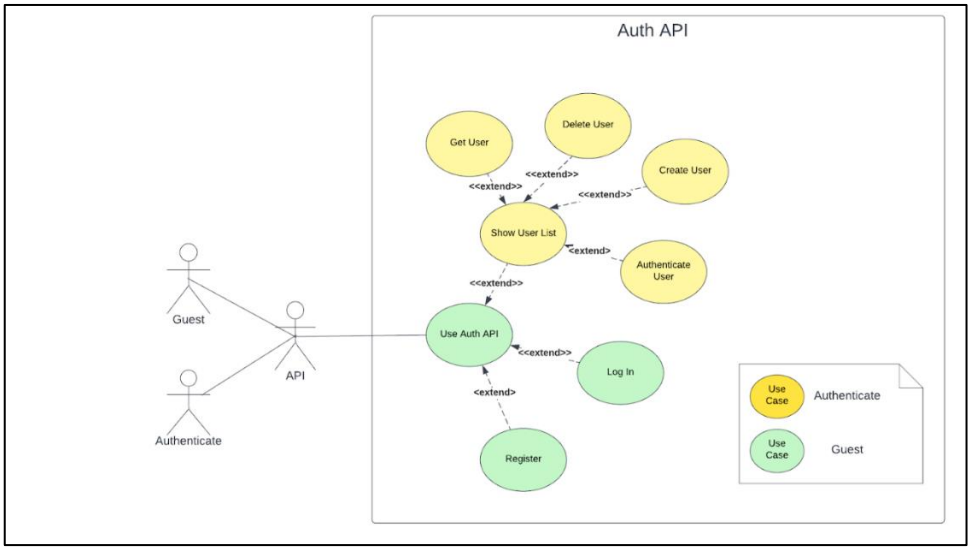
Hình 4.4: Cơ sở dữ liệu của hệ thống quản lý bãi đậu xe thông minh

Một số sơ đồ use case được áp dụng vào phát triển API, ứng dụng di động và ứng dụng web:



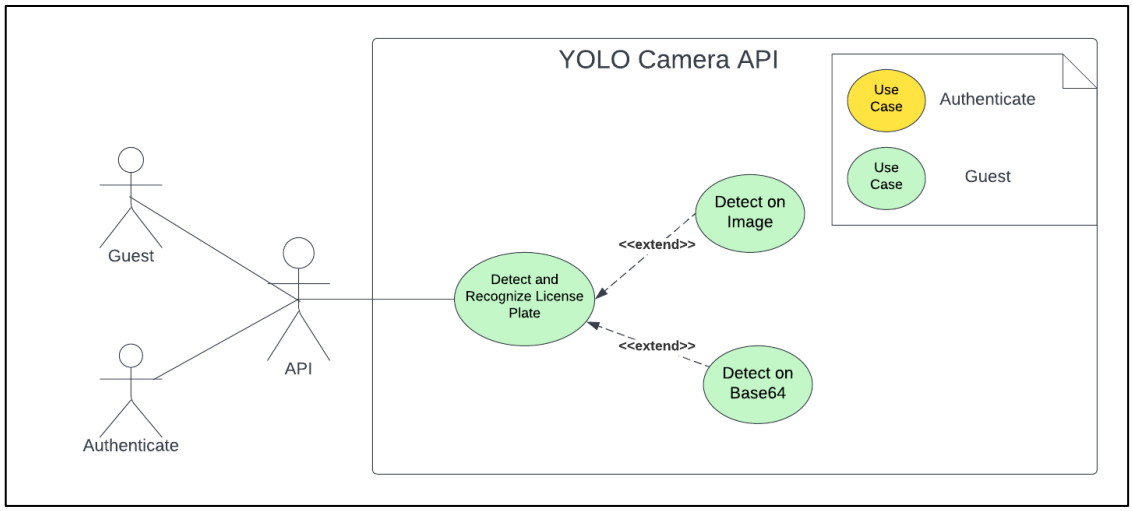
Hình 4.5: Use case quản lý xe

API bao gồm các chức năng như: quản lý khu vực, quản lý camera, chi tiết số lượng phương tiện vào / ra, quản lý danh sách các biển số xe.



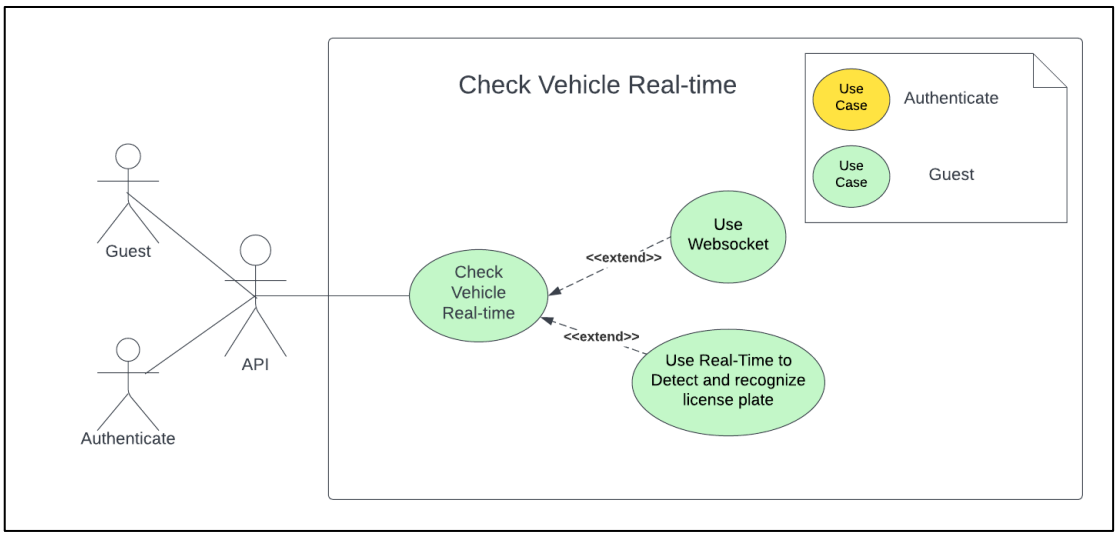
Hình 4.6: Use case quản lý người dùng

Các chức năng của quản lý người dùng bao gồm các chức năng như: quản lý tài khoản người dùng (dành cho quản trị viên) và đăng nhập, đăng ký.

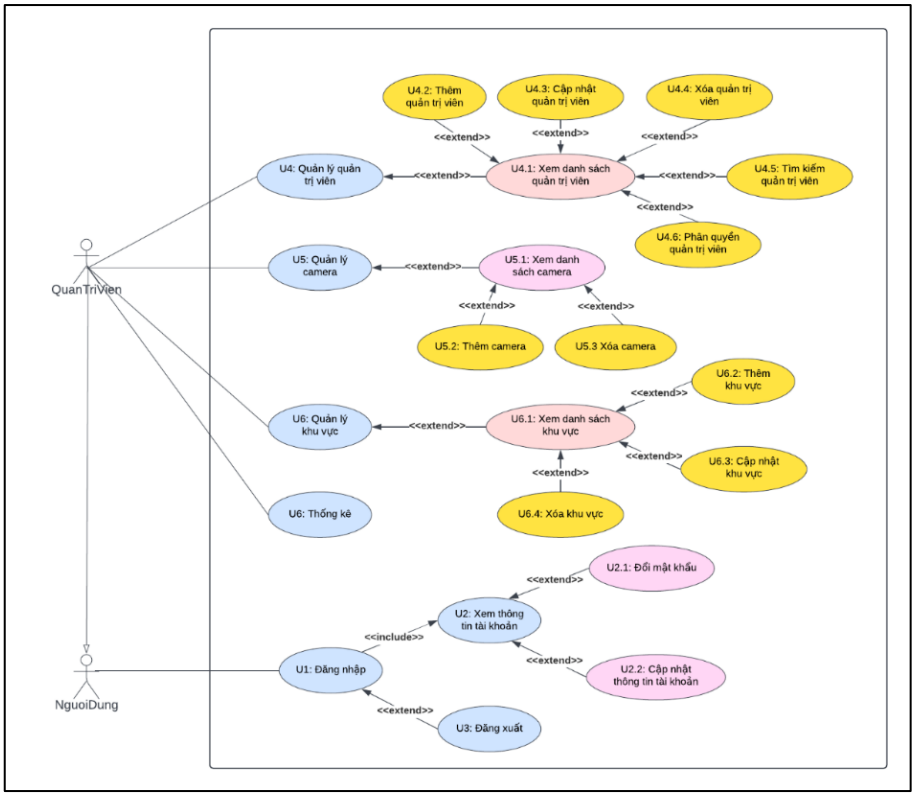


Hình 4.7: Use case sử dụng nhận diện biển số xe

Để nhận diện biển số xe thì có thể được triển khai thông qua nhận diện hình ảnh thông thường hoặc hình ảnh đã được chuẩn hóa dưới dạng base64.



Hình 4.8: Use Case Sử dụng realtime nhận diện biển số xe



Hình 4.9: UseCase cho ứng dụng web quản lý

Đối với ứng dụng web quản lý thông tin phương tiện tại bãi đậu xe có các chức năng sau: Đăng nhập, quản lý camera, quản lý khu vực, thống kê, quản lý quản trị viên (hay nói cách khác là quản lý thông tin các tài khoản đã đăng ký và các tài khoản được cấp quyền quản trị viên).

Hình 4.10: UseCase cho ứng dụng di động

Các chức năng phát triển dành cho ứng dụng di động bao gồm: Đăng nhập, đăng ký, quản lý danh sách xe ra/vào (dành cho quản trị viên). Ở chức năng đăng nhập, người dùng sau khi đã đăng nhập có thể thực hiện các chức năng khác như xem và cập nhật thông tin cá nhân, đổi mật khẩu, đặt lịch hẹn.

4. Chương trình

4.1. API

Trong báo cáo này, sẽ sử dụng công nghệ FastAPI để phát triển bên phía server, vì FastAPI là một web framework hiện đại, có hiệu suất cao để xây dựng API với ngôn ngữ Python3.7+, FastAPI được xây dựng dựa trên Uvicorn máy chủ web ASGI (Giao diện cổng máy chủ bất đồng bộ).

Sử dụng cơ sở dữ liệu là MongoDB để lưu trữ các quá trình ANPR nhận diện biển số xe, và dùng để truy vấn các thông tin liên quan tới chủ xe, thời gian ra vào của xe, quản lý người dùng, cho phép xe ra vào theo từng khu vực.

Lý do sử dụng MongoDB là vì nó truy vấn rất nhanh và là NoSQL nên dễ dàng mở rộng so với SQL. Vì thư viện sử dụng của MongoDB là bất đồng bộ nên phù hợp với FastAPI dẫn đến việc xây dựng ứng dụng có hiệu suất cao.

Đối với bên phía người dùng, có thể phát triển theo 2 hướng: ứng dụng di động cho người dùng đăng ký và theo dõi thời gian ra vào của xe và trang web quản trị cho admin quản lý chương trình.

4.2. Ứng dụng di động

Công nghệ sử dụng để phát triển ứng dụng điện thoại là Flutter. Flutter là một công nghệ phát triển ứng dụng mobile đa nền tảng mạnh mẽ và tiện dụng, được tập đoàn công nghệ Google phát triển và ra mắt vào năm 2017. Flutter cho phép lập trình viên phát triển ứng dụng cho cả iOS và Android từ một mã nguồn duy nhất, giúp tiết kiệm thời gian, công sức và chi phí trong việc phát triển ứng dụng.

Một số lợi ích của việc sử dụng Flutter để phát triển ứng dụng mobile như đa nền tảng (Android, iOS), tốc độ phát triển, thiết kế đẹp mắt và tương tác tốt, hiệu suất cao, cộng đồng lớn và hỗ trợ tốt, tích hợp tốt với các dịch vụ khác (Firebase, FCM, AdMod), hỗ trợ cho các ứng dụng phức tạp (animation, hoạt hình, tích hợp các API, …), độc lập với hệ thống. Với những lợi ích trên, Flutter là một lựa chọn tốt cho việc phát triển ứng dụng mobile đa nền tảng, giúp tiết kiệm thời gian và chi phí, đồng thời tạo ra các ứng dụng đẹp mắt, tương tác tốt với người dùng và có hiệu suất cao.

Ứng dụng di động cho hệ thống quản lý bãi giữ xe có nhiều tính năng và chức năng khác nhau để người dùng có thể dễ dàng đăng ký và theo dõi các phương tiện cá nhân. Dưới đây là một số tính năng và chức năng được tích hợp vào ứng dụng di động này:

1. Đăng ký và đăng nhập: Cho phép người dùng đăng ký tài khoản mới và đăng nhập vào ứng dụng để sử dụng các tính năng và chức năng của hệ thống.

2. Quản lý thông tin xe và khách hàng: Cho phép quản lý thông tin khách hàng,bao gồm biển số xe, loại xe, họ tên khách hàng, số điện thoại, cho phép chỉnh sửa thông tin cá nhân.

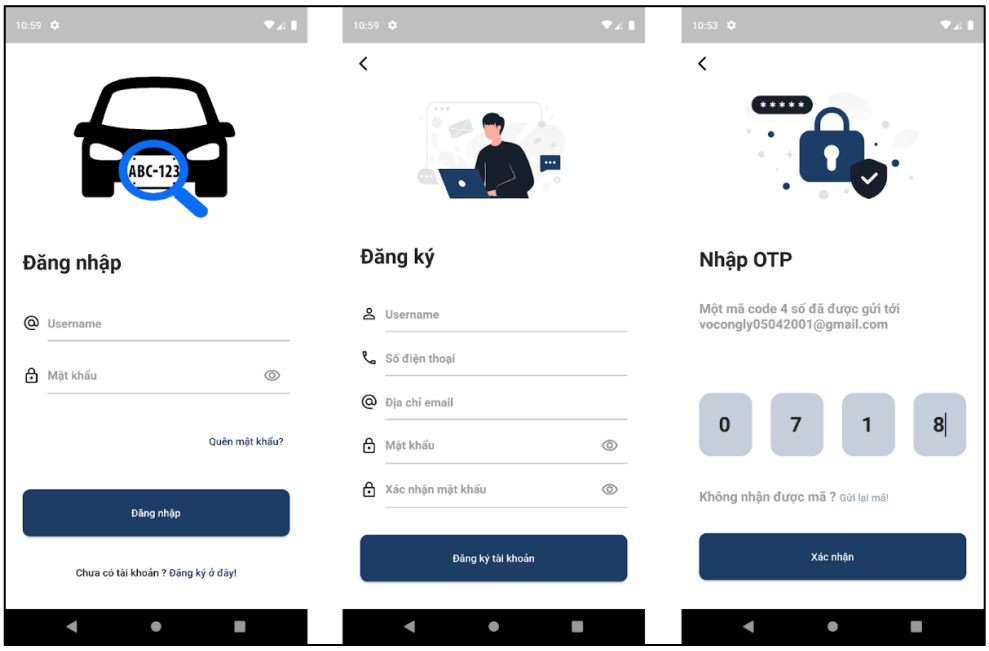
3. Thông báo và cập nhật: Thông báo về lịch sử ra vào các phương tiện cá nhân của khách, thời gian của các phương tiện và các chương trình khuyến mãi khi đăng ký xe theo tháng, theo năm.

4. Liên lạc và hỗ trợ: Cung cấp các thông tin liên lạc để khách hàng có thể liên hệ với nhân viên quản lý bãi giữ xe để giải đáp các thắc mắc hoặc khi cần hỗ trợ.

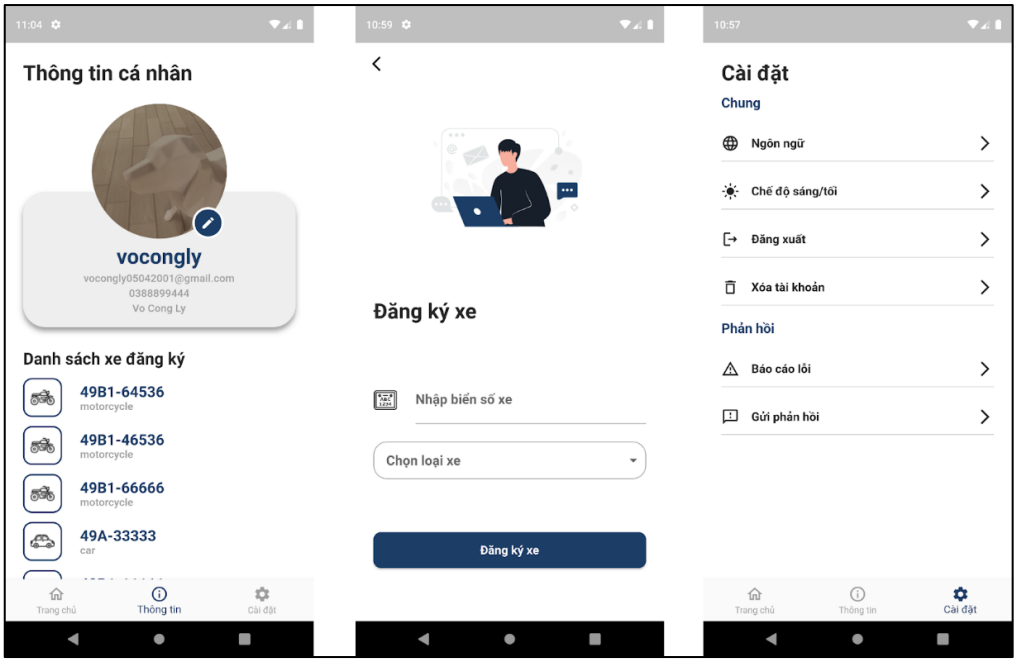
5. Hỗ trợ nhiều ngôn ngữ: Cung cấp nhiều ngôn ngữ khác nhau để hỗ trợ khách hàng đến từ các quốc gia và vùng lãnh thổ khác nhau.

6. Đánh giá và phản hồi: Cho phép khách hàng đánh giá và phản hồi về chất lượng dịch vụ, giúp quản lý bãi giữ xe cải thiện và nâng cao chất lượng dịch vụ.

Tất cả những tính năng và chức năng trên tích hợp vào một ứng dụng di động cho hệ thống quản lý bãi giữ xe, giúp tăng tính tiện ích và hiệu quả cho khách hàng.



Hình 4.11: Giao diện đăng nhập, đăng ký và OTP



Hình 4.12: Giao diện profile, đăng ký xe và settings

4.3. Ứng dụng web

Ứng dụng web là trang web quản lý bãi xe thông minh sử dụng công nghệ Reactjs. Reactjs là một thư viện JavaScript mã nguồn mở được Facebook xây dựng và phát triển, được sử dụng để tạo ra các ứng dụng web với hiệu quả cao, tốc độ tải nhanh giảm thiểu số lượng dòng lệnh lập trình. Mỗi website sử dụng ReactJS phải chạy nhanh, mượt và có khả năng mở rộng cao, thao tác thực hiện đơn giản.

Trong ứng dụng trên nền web này, sử dụng các thư viện có sẵn giúp tối ưu hóa quá trình phát triển cũng như thuận tiện mở rộng các chức năng về sau. Mỗi chức năng được thực hiện nhờ vào việc sử dụng các thư viện kết hợp với quá trình chỉnh sửa và bổ sung sẽ đáp ứng được các yêu cầu của người sử dụng cũng như là quá trình trải nghiệm của người dùng. Cụ thể như để có thể tương tác với các biến và dữ liệu của trang web sử dụng thư viện Hook. Bên cạnh đó, để tạo được giao diện thân thiện với người dùng mà không tốn quá nhiều thời gian để xử lý là một giải pháp hướng đến trong quá trình phát triển ứng dụng. Vì vậy, trong ứng dụng này sử dụng thư viện MUI để tạo ra các giao diện tương tác với người dùng.

Dưới đây là một số tính năng và chức năng được tích hợp vào ứng dụng web này:

1. Đăng nhập: Cho phép người dùng đăng nhập vào tài khoản đã được cấp từ trước để sử dụng các tính năng và chức năng của hệ thống.

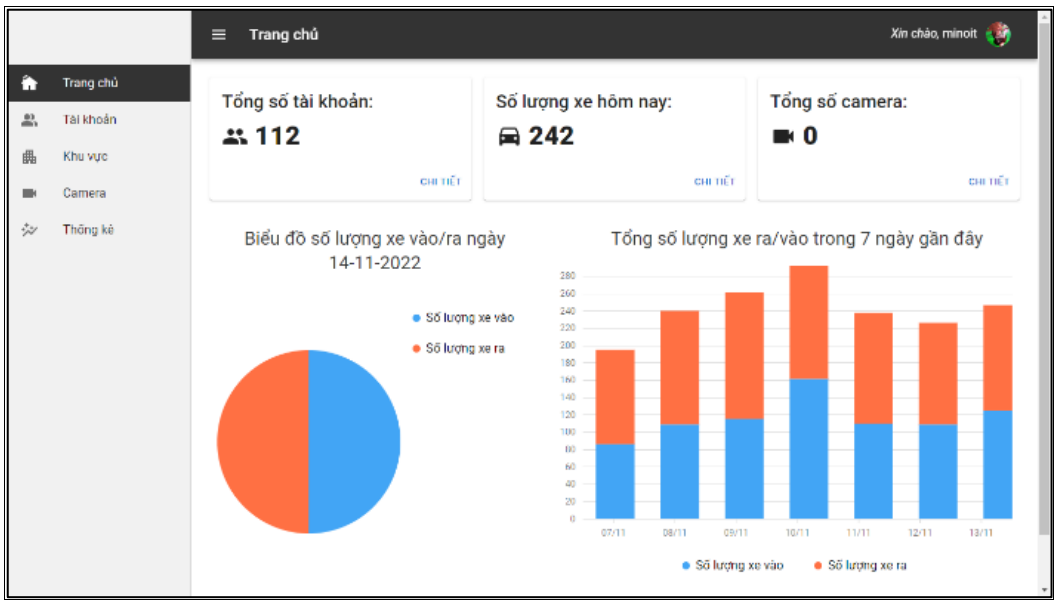
2. Quản lý thông tin người dùng: Cho phép quản lý thông tin cá nhân, bao gồm họ tên, số điện thoại, ảnh đại diện và mật khẩu.

3. Thống kê số lượng xe vào / ra: Tại trang chủ, người dùng có thể thống kê số lượng vào / ra trong ngày và xem các biểu đồ trực quan về số lượng xe vào / ra trong ngày và trong 7 ngày gần nhất để so sánh, cập nhật thông tin mới nhất cho người dùng. Ngoài ra, người dùng còn có thể tự mình thống kê số lượng xe vào / ra tại các thời gian cụ thể như số lượng xe trong khoảng ngày, khoảng thời gian, khu vực, biển số,... và xem chi tiết thông tin của xe đó.

4. Xem danh sách các tài khoản đã đăng ký: Người dùng có thể xem danh sách các tài khoản đã đăng ký biển số xe và thực hiện phân quyền cho các tài khoản và tạo các tài khoản mới.

5. Xem danh sách khu vực có trong bãi đậu xe: Người dùng có thể quản lý các khu vực trong bãi đậu xe để kịp thời khắc phục hay xử lý các vấn đề có thể xảy ra tại các khu vực này.

Tất cả những tính năng và chức năng trên tích hợp vào một ứng dụng web cho hệ thống quản lý bãi đậu xe, giúp tăng tính tiện ích và hiệu quả cho người dùng.



Hình 4.13: Giao diện trang web quản lý

KẾT LUẬN VÀ KIẾN NGHỊ

1. Kết luận

Mô hình quản lý bãi xe thông minh là một giải pháp hiệu quả để quản lý và vận hành bãi giữ xe một cách thông minh và tiết kiệm chi phí. Bằng cách sử dụng các công nghệ và thiết bị thông minh, mô hình này giúp tăng tính tiện ích và hiệu quả trong việc quản lý bãi giữ xe, đồng thời giảm thiểu sự cố và tăng tính an toàn cho người sử dụng.

Trong báo cáo này, chúng tôi đã tìm hiểu, nghiên cứu lý thuyết về mô hình xử lý hình ảnh, nhận diện vùng biển số và biển số, xử lý hình ảnh để nhận diện khuôn mặt của người dùng. Bài báo này tập trung vào công nghệ và ứng dụng công nghệ vào thực tiễn để quản lý phương tiện. Nội dung chính của bài báo cáo được chia thành 5 phần riêng biệt: Phần thứ nhất giới thiệu mô hình YOLO, mô hình MTCNN, FaceNet và kiến trúc Microservice. Phần thứ hai là các giai đoạn nhận diện biển số. Phần thứ ba nói rõ về các giai đoạn nhận diện khuôn mặt. Phần thứ 5 của báo cáo là xây dựng các ứng dụng thực tiễn áp dụng các mô hình ở trên.

Rõ ràng ANPR là một hệ thống khó có chính xác tuyệt đối vì có nhiều giai đoạn nhận diện và mỗi giai đoạn phụ thuộc vào giai đoạn trước. Một số yếu tố như ánh sáng khác nhau điều kiện, bóng xe và kích thước không đồng nhất của biển số, phông chữ và màu nền khác nhau ảnh hưởng đến hiệu suất của ANPR. Một số hệ thống hoạt động trong những điều kiện khác nhau và có thể không tạo ra số độ chính xác cao trong điều kiện bất lợi [7].

Vì vậy, để triển khai mô hình quản lý bãi xe thông minh, cần đầu tư các thiết bị và công nghệ phù hợp, đồng thời có kế hoạch và quy trình vận hành và bảo trì để đảm bảo hiệu quả và độ tin cậy của hệ thống. Ngoài ra, việc đào tạo và hướng dẫn người sử dụng và quản lý cũng rất quan trọng để tăng tính sử dụng và hiệu quả của mô hình quản lý bãi xe thông minh.

2. Hướng phát triển

Với sự phát triển của giao thông hiện nay, mô hình quản lý bãi đậu xe thông minh đang được áp dụng rộng rãi và có rất nhiều tiềm năng phát triển trong tương lai. Một số hướng phát triển của mô hình này:

Phát triển các giải pháp thanh toán tiện lợi: Các hình thức thanh toán thông thường như tiền mặt hoặc thẻ tín dụng đã trở nên lỗi thời, các ví điện tử ngày càng phổ biến và tiện lợi. Áp dụng các hình thức thanh toán mới để tăng tính tiện lợi cho khách hàng.

Phát triển các tính năng thông minh: Mô hình bãi đậu xe thông minh có thể phát triển thêm các tính năng thông minh như là kiểm tra các vị trí trống và thông báo cho người dùng biết về tình trạng bãi đậu xe, hệ thống định vị vị trí xe của người dùng khi người dùng đưa xe vào bãi và các tính năng khác để tăng tính tiện lợi và hiệu quả của hệ thống.

Tăng cường tính bảo mật và an ninh: Mô hình bãi đậu xe thông minh cần tăng cường tính bảo mật và an ninh các phương tiện, đảm bảo an toàn cho xe và người sử dụng ứng dụng. Điều này có thể đạt được bằng cách phát triển hệ thống giám sát, hệ thống báo động, hệ thống nhận diện khuôn mặt và các công nghệ khác để ngăn chặn hành vi gian lận, trộm cắp và các hành vi phạm pháp.

Tích hợp với hệ thống giao thông thông minh: Mô hình quản lý bãi đậu xe thông minh có thể tích hợp với hệ thống giao thông thông minh để tối ưu hóa quản lý bãi đậu xe và giảm tắc đường. Bằng cách tích hợp các dữ liệu về tình trạng giao thông và tình trạng đỗ xe, mô hình quản lý bãi đậu xe thông minh có thể dự đoán được tình trạng đỗ xe và giúp người dùng tìm kiếm bãi đỗ xe nhanh chóng và tiết kiệm thời gian.

Đưa vào ứng dụng trên diện rộng: Mô hình quản lý bãi đậu xe thông minh cần được đưa vào ứng dụng trên diện rộng, không chỉ tại các thành phố lớn mà còn tại các thành phố nhỏ và vùng nông thôn. Điều này sẽ giúp giảm tắc đường, giảm ô nhiễm môi trường và tăng tính tiện lợi cho người dân.

Tóm lại, mô hình quản lý bãi đậu xe thông minh có nhiều tiềm năng phát triển trong tương lai. Các hướng phát triển tiềm năng bao gồm tăng cường tính tiện lợi và đa dạng hóa hệ thống, phát triển các tính năng thông minh, tăng cường tính bảo mật và an ninh, tích hợp với hệ thống giao thông thông minh, và đưa vào ứng dụng trên diện rộng. Để đạt được các mục tiêu phát triển này, cần có sự đầu tư nghiên cứu các mô hình và phát triển công nghệ ứng dụng, cũng như đối tượng sử dụng đang sử dụng bãi đậu xe thông minh. Ngoài ra, cần phải đảm bảo tính bền vững của hệ thống và đáp ứng mục tiêu của cộng đồng.

TÀI LIỆU THAM KHẢO

[1] Rayson Laroca, Evair Severo, Luiz A. Zanlorensi, Luiz S. Oliveira, etc, "A Robust Real-Time Automatic License Plate Recognition Based on the YOLO Detector," p. 10, 28 4 2018.

[2] Yonten Jamtsho, Panomkhawn Riyamongkol, Rattapoom Waranusast, etc, "Real-time license plate detection for non-helmeted motorcyclist using YOLO," p. 6, 22 8 2020.

[3] Thanh-Sach Le, Van-Nhan Tran, Kazuhiko Hamamoto, "A robust and flexible license plate detection method," p. 4, 15 10 2014.

[4] Florian Schroff, Dmitry Kalenichenko, James Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," p. 10, 17 6 2015.

[5] Yu. A. Malkov, D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs," p. 13.

[6] VinParking, "MÁY GIỮ XE THÔNG MINH 2 LÀN," VinParking, 2022. [Online]. Available: https://vinparking.com/san-pham/may-giu-xe-thong-minh-2-lan-1. [Accessed 15 5 2023].

[7] D. S. A. P. Chirag Patel, "Automatic Number Plate Recognition System (ANPR): A Survey ," p. 14, 5 2013.

[8] Jaradat, Mohammed-Issa Riad, "A New Approach for License Plate Detection and Localization: Between Reality and Applicability," p. 14, 5 6 2017.

[9] Wenjing Jia, Huaifeng Zhang and Xiangjian He, "Region-Based License Plate Detection," *Faculty of Information Technology, University of Technology, Sydney,* p. 13, 3 3 2006.

[10] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," p. 10, 9 5 2016.

[11] A. Barguzar, "Python Flask versus FastAPI: Which Should You Choose?," 21 4 2022. [Online]. Available: https://www.netguru.com/blog/python-flask-versus-fastapi.

[12] A. Bassett, "Introducing FARM Stack - FastAPI, React, and MongoDB," 6 2 2022. [Online]. Available:

https://www.mongodb.com/developer/languages/python/farm-stack-fastapi-react-mongodb/.

[13] Chánh, "Tổng quan về flutter, cài đặt flutter và chạy chương trình đầu tiên," 29 10 2021. [Online]. Available: https://phannhatchanh.com/blog/cai-dat-flutter-va-chay-chuong-trinh-dau-tien.

[14] N. V. Dung, "Tổng quan về Flutter," 22 5 2019. [Online]. Available: https://viblo.asia/p/tong-quan-ve-flutter-Eb85oyAkZ2G.

[15] Q. Hòa, "Giới thiệu về kiến trúc Microservices," 1 2 2019. [Online]. Available: https://viblo.asia/p/gioi-thieu-ve-kien-truc-microservices-4P8566O35Y3.

[16] ITNavi, "Microservices là gì? Từ A - Z về Microservices và Microservices Architecture - ITNavi," 7 1 2021. [Online]. Available: https://itnavi.com.vn/blog/microservices-la-gi.

[17] LCDUNG, "Phát triển phần mềm theo kiến trúc microservice," 28 12 2018. [Online]. Available: https://lcdung.top/phat-trien-phan-mem-theo-kien-truc-microservice/.