TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM
KHOA CÔNG NGHỆ THÔNG TIN

HCMUTE

BÁO CÁO CUỐI KỲ

MÔN: XỬ LÝ ẢNH SỐ

Đề tài: ỨNG DỤNG XỬ LÝ ẢNH SỐ VÀ HỌC SÂU VÀO NHẬN DIỆN NGUYÊN LIỆU VÀ ĐỀ XUẤT MÓN ĂN

MÃ MÔN HỌC: DIPR430685_23_2_05CLC

HỌC KỲ II – NĂM HỌC 2023-2024

Giảng viên hướng dẫn : Hoàng Văn Dũng

Thực hiện: Nhóm 13

Bùi Quốc Khang 21110202

Lê Minh Quang 21110279

Đặng Kim Thành 21110298

Trương Thị Thùy Dung 21110820

Thành phố Hồ Chí Minh, tháng 5 năm 2024

BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC SƯ PHẠM KỸ THUẬT TPHCM
KHOA CÔNG NGHỆ THÔNG TIN
MÔN XỬ LÝ ẢNH SỐ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA
VIỆT NAM
Độc lập – Tự do – Hạnh phúc

HCMUTE

BÁO CÁO CUỐI KỲ

MÔN: XỬ LÝ ẢNH SỐ

Đề tài: ỨNG DỤNG XỬ LÝ ẢNH SỐ VÀ HỌC SÂU VÀO NHẬN DIỆN NGUYÊN LIỆU VÀ ĐỀ XUẤT MÓN ĂN

MÃ MÔN HỌC: DIPR430685_23_2_05CLC
HỌC KỲ II – NĂM HỌC 2023-2024
Giảng viên hướng dẫn : Hoàng Văn Dũng
Thực hiện: Nhóm 13

Bùi Quốc Khang 21110202
Lê Minh Quang 21110279
Đặng Kim Thành 21110298
Trương Thị Thùy Dung 21110820

Thành phố Hồ Chí Minh, tháng 5 năm 2024

DANH SÁCH THÀNH VIÊN THAM GIA ĐỒ ÁN

Đề tài: Ứng dụng xử lý ảnh số và học sâu vào nhận diện nguyên liệu và đề xuất món ăn

---

STT	HỌ VÀ TÊN THÀNH VIÊN	MÃ SỐ SINH VIÊN	TỶ LỆ THAM GIA
1	Bùi Quốc Khang	21110202	100%
2	Lê Minh Quang	21110279	100%
3	Đặng Kim Thành	21110298	100%
4	Trương Thị Thùy Dung	21110820	100%

Ghi chú:

Tỷ lệ %: Mức độ phần trăm hoàn thành của từng sinh viên tham gia.

Trưởng nhóm*:* Bùi Quốc Khang

Nhận xét của giáo viên

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp. Hồ Chí Minh - Tháng 5 năm 2024

I. LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn sâu sắc đến thầy Hoàng Văn Dũng, thầy đã tận tình giảng dạy trong suốt quá trình tìm hiểu và học tập môn đồ án công nghệ thông tin. Trong từng buổi học, thầy đã nhiệt huyết trong từng lời giải đáp, những kiến thức, kỹ năng, tài liệu quý giá mà thầy đã truyền đạt học không chỉ là nền tảng cho quá trình thực hiện bài tập cuối kì mà còn là kiến thức nền tảng cho nhiều phần sau này.

Rất nhiều công sức và nỗ lực đã được bỏ ra, nhưng do chưa có kinh nghiệm trong việc xây dựng đề tài và những hạn chế về kiến thức, bài báo cáo này không tránh khỏi sai sót. Chúng em luôn sẵn sàng tiếp thu những ý kiến từ thầy để cải thiện bài báo cáo cũng như đề tài lần này.

Một lần nữa, em xin bày tỏ lòng biết ơn đến thầy. Xin kính chúc thầy luôn dồi dào sức khỏe, đạt được nhiều thành công trong công việc.

II. MỤC LỤC

III. MỤC LỤC HÌNH ẢNH

CHƯƠNG 1: MỞ ĐẦU

1.1. Giới thiệu về ứng dụng mobile

Trong thời đại công nghệ phát triển như hiện nay, việc áp dụng học sâu và xử lý ảnh số vào lĩnh vực nấu ăn đang trở thành một xu hướng phổ biến. Dự án ***"Ứng dụng xử lý ảnh số và học sâu vào việc xây dựng ứng dụng mobile cho phép nhận diện nguyên liệu và đề xuất món ăn - FoodLens"*** là một ví dụ điển hình cho việc sử dụng công nghệ này để giúp người dùng nhận diện nguyên liệu từ hình ảnh và đề xuất các món ăn tương ứng.

1.2. Mục đích, yêu cầu cần thực hiện

a. Mục đích

Hỗ trợ lập kế hoạch bữa ăn: Mục đích chính của ứng dụng là cung cấp một công cụ hữu ích để người dùng có thể dễ dàng lên kế hoạch cho bữa ăn hàng ngày của mình.

Tiết kiệm thời gian: Bằng cách nhận diện nguyên liệu từ hình ảnh và đề xuất các món ăn phù hợp, ứng dụng giúp người dùng tiết kiệm thời gian trong quá trình tìm kiếm và chọn lựa công thức nấu ăn.

Thúc đẩy sự sáng tạo trong nấu ăn: Bằng cách đề xuất các món ăn mới dựa trên các nguyên liệu có sẵn, ứng dụng kích thích sự sáng tạo và động viên người dùng thử nghiệm và khám phá ẩm thực mới.

b. Yêu cầu đề ra

Nhận diện nguyên liệu chính xác: Ứng dụng cần có khả năng nhận diện và phân tích các nguyên liệu từ hình ảnh một cách chính xác và đáng tin cậy. Điều này đòi hỏi sự phát triển và tinh chỉnh của các thuật toán xử lý ảnh số.

Đề xuất món ăn phù hợp: Sau khi nhận diện nguyên liệu, ứng dụng cần đề xuất một loạt các món ăn phù hợp với những nguyên liệu đã được nhận diện. Các đề xuất này cần được cá nhân hóa dựa trên khẩu vị và sở thích của người dùng.

Giao diện thân thiện và dễ sử dụng: Giao diện của ứng dụng cần được thiết kế một cách thân thiện và dễ sử dụng, để người dùng có thể tận dụng các tính năng một cách dễ dàng và nhanh chóng.

Bảo mật thông tin người dùng: Đảm bảo rằng dữ liệu cá nhân và thông tin về thói quen ăn uống của người dùng được bảo vệ và không bị lộ ra ngoài mạng.

1.3. Các giai đoạn thực hiện đồ án

1.3.1. Giai đoạn 1: Xây dựng mô hình để nhận diện nguyên liệu và đề xuất

❖ **Thu Thập Dữ Liệu**

Thu thập một tập dữ liệu đa dạng chứa các hình ảnh về nguyên liệu và các món ăn khác nhau.

Dữ liệu cần phải được gắn nhãn chính xác để phục vụ quá trình huấn luyện mô hình.

❖ **Tiền xử lý dữ liệu:**

Tiền xử lý dữ liệu hình ảnh để chuẩn hóa kích thước, độ sáng, và độ tương phản.

Áp dụng các kỹ thuật tiền xử lý khác nhau như cắt ghép, xoay, và phản chiếu để tăng cường độ đa dạng của dữ liệu.

❖ **Xây dựng mô hình nhận diện nguyên liệu:**

Sử dụng các kiến trúc mạng nơ-ron tích chập (CNN) để xây dựng mô hình nhận diện nguyên liệu.

Huấn luyện mô hình trên tập dữ liệu đã được gắn nhãn, sử dụng các kỹ thuật như transfer learning để tăng tốc quá trình huấn luyện.

❖ **Kiểm thử và đánh giá mô hình**

Đánh giá hiệu suất của mô hình trên tập dữ liệu kiểm tra không được sử dụng trong quá trình huấn luyện.

Đo lường các chỉ số như độ chính xác, độ phủ, và độ nhớ để đánh giá khả năng nhận diện của mô hình.

❖ **Xây dựng hệ thống đề xuất món ăn:**

Dựa trên kết quả của mô hình nhận diện nguyên liệu, xây dựng một hệ thống để đề xuất các món ăn phù hợp.

Sử dụng các phương pháp như học sâu và học máy để tạo ra các đề xuất món ăn đa dạng và hấp dẫn.

❖ **Tối ưu hóa và cải tiến:**

Tối ưu hóa mô hình nhận diện và đề xuất để cải thiện độ chính xác và hiệu suất của hệ thống.

Sử dụng các kỹ thuật như tinh chỉnh siêu tham số và kỹ thuật tăng cường để cải thiện khả năng nhận diện và đề xuất.

❖ **Kiểm thử và đánh giá toàn diện:**

Tiến hành kiểm thử toàn diện trên hệ thống đã xây dựng để đảm bảo tính ổn định và hiệu suất.

Đánh giá lại mô hình nhận diện và đề xuất dựa trên các thử nghiệm thực tế và phản hồi từ người dùng.

1.3.2. Giai đoạn 2: Xây dựng ứng dụng mobile

❖ **Thiết kế giao diện người dùng (ui/ux):**

Phát triển wireframes và thiết kế giao diện người dùng dựa trên yêu cầu của người dùng và mục tiêu của ứng dụng.

Tối ưu hóa trải nghiệm người dùng để đảm bảo tính thân thiện và dễ sử dụng.

❖ **Phát triển frontend:**

Sử dụng các công nghệ như Swift cho iOS và Kotlin/Java cho Android để phát triển phần giao diện người dùng của ứng dụng.

Kết hợp giao diện người dùng với các tính năng nhận diện nguyên liệu và đề xuất món ăn.

❖ **Xây dựng backend:**

Phát triển backend của ứng dụng để quản lý dữ liệu nguyên liệu, món ăn và thông tin người dùng.

Xây dựng API để kết nối frontend với backend.

1.3.3. Giai đoạn 3: Tích hợp ứng dụng mobile – FoodLens và mô hình đã xây dựng

❖ **Tích hợp mô hình nhận diện nguyên liệu**

Sử dụng API hoặc thư viện phù hợp để tích hợp mô hình nhận diện nguyên liệu vào phần backend của ứng dụng FoodLens.

Xác định giao thức hoặc cách thức giao tiếp giữa frontend và backend để truyền dữ liệu hình ảnh từ ứng dụng di động đến mô hình nhận diện.

❖ **Kết nối và gửi dữ liệu**

Xây dựng giao diện người dùng trên ứng dụng di động để cho phép người dùng chụp hình ảnh của nguyên liệu.

Khi người dùng chụp ảnh, gửi hình ảnh đến backend của ứng dụng FoodLens để phân tích và nhận diện nguyên liệu bằng mô hình đã tích hợp.

❖ **Xử lý kết quả**

Backend xử lý kết quả từ mô hình nhận diện và trả về thông tin về các nguyên liệu được nhận diện cho frontend của ứng dụng.

Dựa trên thông tin này, ứng dụng FoodLens có thể hiển thị danh sách các nguyên liệu đã nhận diện cho người dùng.

❖ **Đề xuất món ăn**

Sử dụng thông tin về các nguyên liệu đã nhận diện để đề xuất danh sách các món ăn phù hợp.

Kết quả của mô hình nhận diện nguyên liệu có thể được đưa vào mô hình đề xuất món ăn để tạo ra các gợi ý món ăn tương ứng.

❖ **Hiển thị kết quả**

Hiển thị danh sách nguyên liệu đã nhận diện và các món ăn đề xuất lên giao diện người dùng của ứng dụng FoodLens.

Cho phép người dùng xem chi tiết về mỗi nguyên liệu và món ăn đề xuất và lựa chọn món ăn phù hợp với sở thích và nguyên liệu có sẵn

1.3.4. Giai đoạn 4: Tiến hành thực nghiệm

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT ĐỂ THỰC HIỆN PROJECT

2.1. Công cụ và môi trường để lập trình

- **IDE:** Microsoft Visual Studio
- **Platform:** Figma
- **Ngôn ngữ:** Dart, Kotlin
- **Các thư viện:** TensorFlow Lite
- **Framework:** Flutter

2.2. Giới thiệu tổng quát về TensorFlow Lite

2.2.1. TensorFlow Lite

TensorFlow Lite là một thư viện nhỏ, tối ưu hóa cho thiết bị di động, giúp thực thi các mô hình học máy nhanh chóng và hiệu quả. Nó cho phép ứng dụng của chúng tôi nhận diện và theo dõi các đối tượng trong thời gian thực mà không làm giảm hiệu suất thiết bị.

2.2.2. Ưu điểm và nhược điểm của công nghệ

❖ ***Ưu điểm***

Hiệu Suất Cao: TensorFlow Lite được tối ưu hóa cho thiết bị di động, giúp giảm thời gian và tài nguyên cần thiết cho việc nhận diện đối tượng.

Hoạt Động Trên Thiết Bị (On-device): Việc xử lý tất cả các tác vụ trực tiếp trên thiết bị giúp bảo mật dữ liệu người dùng và hoạt động mà không cần kết nối internet.

Hỗ Trợ Đa Dạng Đối Tượng: Có khả năng nhận diện và phân loại một loạt rộng lớn các nguyên liệu nấu ăn và trái cây, nông sản.

❖ ***Nhược điểm***

Giới Hạn về Độ Chính Xác: Dù đã được tối ưu, độ chính xác của việc nhận diện đối tượng vẫn có thể bị ảnh hưởng bởi điều kiện ánh sáng, góc chụp và chất lượng camera.

Cần Huấn Luyện Mô Hình Cụ Thể: Để đạt được hiệu suất tốt nhất, mô hình cần được huấn luyện với một tập dữ liệu cụ thể và đa dạng của các nguyên liệu nấu ăn và trái cây, nông sản.

2.3. Giới thiệu tổng quát về Flutter

2.3.1. Flutter

Flutter là một framework phát triển ứng dụng di động đa nền tảng được phát triển bởi Google, cho phép bạn xây dựng ứng dụng di động đồng nhất trên cả Android và iOS từ một mã nguồn duy nhất.

Khi kết hợp với TensorFlow Lite, Flutter cung cấp một giải pháp toàn diện để phát triển ứng dụng phát hiện đối tượng trên thiết bị di động.

Flutter có tốc độ phát triển nhanh, giao diện người dùng linh hoạt và dễ dàng tương tác. Đồng thời, TensorFlow Lite hỗ trợ tích hợp mô hình machine learning trực tiếp vào ứng dụng của bạn.

Kết hợp cả hai giúp đơn giản hóa quá trình phát triển và triển khai ứng dụng phát hiện đối tượng trên cả hai nền tảng Android và iOS.

Việc sử dụng Flutter và TensorFlow Lite giúp tạo ra một ứng dụng phát hiện đối tượng mạnh mẽ, hiệu quả và có hiệu suất cao, đồng thời đảm bảo tính nhất quán trên cả hai nền tảng di động. Điều này giúp tăng cường trải nghiệm người dùng và giảm thiểu thời gian và công sức cần thiết cho việc phát triển ứng dụng.

2.3.2. Ưu điểm và nhược điểm của Flutter

- ***Ưu điểm***

Đa nền tảng (Cross-platform): phát triển ứng dụng cho cả IOS và Android từ một nguồn mã duy nhất.

Giao diện đẹp (impressive UI) : Flutter cung cấp rất nhiều các widget, với nhiều sự lựa chọn. Giao diện khá đẹp và sắc nét.

Hiệu suất cao: Flutter sử dụng ngôn ngữ Dart giúp hiệu quả về mặt hiệu suất.

- ***Nhược điểm***

Kích thước ứng dụng lớn: Ứng dụng được phát triển bằng Flutter có thể có kích thước lớn hơn so với các ứng dụng được phát triển bằng các công nghệ khác.

Hạn chế của các plugin: Mặc dù Flutter có một số lượng lớn các plugin, nhưng không phải tất cả các tính năng đều được hỗ trợ. Điều này có thể tạo ra một số hạn chế trong việc phát triển ứng dụng.

2.4. Giới thiệu tổng quát về ngôn ngữ Dart

2.4.1. Dart

Dart là một ngôn ngữ lập trình được phát triển bởi Google, được sử dụng chủ yếu cho việc phát triển ứng dụng web, di động và máy chủ. Dart được thiết kế để cung cấp một nền tảng linh hoạt, hiệu quả và dễ dàng để phát triển ứng dụng, đặc biệt là trong việc xây dựng các ứng dụng sử dụng Google's Flutter framework.

2.4.2. Ưu điểm và nhược điểm của Dart

❖ ***Ưu điểm***

Hiệu suất cao: Dart cung cấp hiệu suất cao, đặc biệt là khi được sử dụng với framework Flutter, giúp ứng dụng chạy mượt mà và đáp ứng nhanh chóng.

Dễ học và sử dụng: Dart có cú pháp đơn giản và dễ hiểu, giúp người lập trình dễ dàng học và sử dụng ngôn ngữ này.

Flutter framework: Dart được sử dụng chủ yếu cho việc phát triển ứng dụng di động và web bằng cách sử dụng framework Flutter, một trong những framework phát triển ứng dụng di động hàng đầu hiện nay.

Kiểu tĩnh và kiểu động: Dart hỗ trợ cả kiểu tĩnh và kiểu động, giúp người lập trình linh hoạt trong việc xây dựng ứng dụng.

❖ ***Nhược điểm***

Hạn chế trong việc tích hợp: Mặc dù Dart có thể được sử dụng để phát triển ứng dụng web, nhưng việc tích hợp với các dự án web hiện có không phải lúc nào cũng dễ dàng.

2.5. Giới thiệu tổng quát về ngôn ngữ Kotlin

2.5.1. Kotlin

Kotlin là một ngôn ngữ lập trình đa nền tảng được phát triển bởi JetBrains. Nó được thiết kế để chạy trên JVM (Java Virtual Machine), nhưng cũng có thể được biên dịch thành mã máy của nền tảng khác. Kotlin được xem là một ngôn ngữ thay thế hoàn hảo cho Java, mang lại nhiều tính năng tiện ích và hiệu suất cao hơn.

2.5.2. Ưu điểm và nhược điểm của Kotlin

❖ ***Ưu điểm***

Tương thích ngược với Java: Kotlin tương thích hoàn toàn với mã nguồn Java, cho phép dễ dàng tích hợp mã Kotlin vào dự án Java hiện có mà không cần phải chuyển đổi mã.

An toàn về loại (Type-safe): Kotlin là một ngôn ngữ an toàn về loại, giúp phát hiện lỗi trước khi chạy ứng dụng, từ đó giảm thiểu lỗi và tăng tính ổn định của ứng dụng.

Cú pháp ngắn gọn và dễ đọc: Kotlin có cú pháp ngắn gọn, dễ đọc và dễ hiểu hơn so với Java, giúp tăng hiệu suất lập trình và giảm thời gian phát triển.

Hỗ trợ mạnh mẽ cho lập trình hàm: Kotlin hỗ trợ lập trình hàm (functional programming) một cách mạnh mẽ, bao gồm lambda, higher-order

❖ ***Nhược điểm***

Tài nguyên hạn chế: So với Java, Kotlin có ít tài liệu và nguồn lực học tập hơn, đặc biệt là đối với các người mới bắt đầu học lập trình.

CHƯƠNG 3: PHÂN TÍCH, THIẾT KẾ GIẢI PHÁP

3.1. Giới thiệu tổng quát về Object Detection (phát hiện đối tượng)

3.1.1. Object Detection (phát hiện đối tượng)

Phát hiện đối tượng là một kỹ thuật thị giác máy tính hoạt động để xác định và định vị các đối tượng trong hình ảnh hoặc video. Cụ thể, tính năng phát hiện đối tượng sẽ vẽ các hộp giới hạn xung quanh các đối tượng được phát hiện này, cho phép chúng tôi xác định vị trí của các đối tượng nói trên (hoặc cách chúng di chuyển qua) một cảnh nhất định.

Đó là một trong những thành tựu lớn nhất của deep learning và xử lý hình ảnh. Một trong những cách tiếp cận phổ biến để tạo bản địa hóa cho các đối tượng là sử dụng các hộp giới hạn. Có thể huấn luyện mô hình phát hiện đối tượng để xác định và phát hiện nhiều đối tượng cụ thể, do đó mô hình này rất linh hoạt.

3.1.2. Các chế độ và kiểu phát hiện đối tượng

Tính năng phát hiện đối tượng có thể được chia thành: các phương pháp tiếp cận dựa trên máy học và các phương pháp tiếp cận dựa trên học tập sâu.

Trong các phương pháp tiếp cận dựa trên học máy truyền thống hơn, các kỹ thuật thị giác máy tính được sử dụng để xem xét các đặc điểm khác nhau của hình ảnh, chẳng hạn như biểu đồ màu hoặc các cạnh, để xác định các nhóm pixel có thể thuộc về một đối tượng. Các tính năng này sau đó được đưa vào mô hình hồi quy để dự đoán vị trí của đối tượng cùng với nhãn của nó.

Mặt khác, các phương pháp tiếp cận dựa trên học sâu sử dụng mạng thần kinh tích chập (CNN) để thực hiện phát hiện đối tượng không giám sát, từ đầu đến cuối, trong đó các tính năng không cần phải được xác định và trích xuất riêng biệt.

3.1.3. Tầm quan trọng của phát hiện đối tượng

Phát hiện đối tượng được liên kết chặt chẽ với các kỹ thuật thị giác máy tính tương tự khác như nhận dạng hình ảnh và phân đoạn hình ảnh, trong đó nó giúp chúng ta hiểu và phân tích các cảnh trong hình ảnh hoặc video.

Nhưng có những khác biệt quan trọng. Nhận dạng hình ảnh chỉ xuất ra nhãn lớp cho một đối tượng được xác định và phân đoạn hình ảnh tạo ra sự hiểu biết ở cấp độ pixel về các thành phần của cảnh. Điều khác biệt giữa tính năng phát hiện đối tượng với

các tác vụ khác này là khả năng độc đáo của nó trong việc định vị các đối tượng trong hình ảnh hoặc video. Điều này cho phép đếm và theo dõi các đối tượng đó.

Với những điểm khác biệt chính này và khả năng độc đáo của tính năng phát hiện đối tượng, có thể thấy cách nó có thể được áp dụng theo một số cách:

- Đếm đám đông
- Xe tự lái
- Video theo dõi
- Phát hiện khuôn mặt
- Phát hiện bất thường

3.2. Data augmentation (trong dataset)

Trong thời đại dữ liệu đóng vai trò trung tâm trong việc phát triển các mô hình học sâu, làm giàu dữ liệu (data augmentation) đã trở thành một công cụ không thể thiếu nhằm cải thiện khả năng tổng quát hóa của các mô hình này. Làm giàu dữ liệu là kỹ thuật mở rộng bộ dữ liệu huấn luyện thông qua việc tạo ra các biến thể của dữ liệu hiện có mà không cần thu thập thêm thông tin. Các phương pháp này có thể bao gồm nhưng không giới hạn ở việc biến đổi hình ảnh, tổng hợp văn bản, hoặc sử dụng các kỹ thuật như làm mờ, thêm nhiễu, và biến đổi ngữ điệu.

Mục đích chính của làm giàu dữ liệu là giúp mô hình phát triển khả năng phân biệt và nhận diện dữ liệu trong các tình huống thực tế mà nó có thể chưa từng gặp phải trong quá trình huấn luyện. Điều này giúp giảm thiểu nguy cơ overfitting, nơi mà mô hình quá khớp với bộ dữ liệu huấn luyện nhưng lại không thể thực hiện tốt trên dữ liệu mới. Trong ngành công nghiệp xử lý ngôn ngữ tự nhiên, làm giàu dữ liệu còn có thể bao gồm việc thay đổi cú pháp hoặc từ vựng để tạo ra các biến thể của văn bản, qua đó nâng cao khả năng hiểu và xử lý ngôn ngữ của mô hình.

Các kỹ thuật data augmentation áp dụng trong xử lý tập dữ liệu để huấn luyện mô hình trong bài báo cáo này:

Auto-adjust contrast:

Auto-adjust contrast là một phương pháp làm giàu dữ liệu trong đó độ tương phản của ảnh được tự động điều chỉnh để tăng cường chất lượng visual của dữ liệu. Phương pháp này thường được sử dụng để cải thiện độ nét và khả năng phân biệt các

đặc điểm trên ảnh, giúp mô hình học máy dễ dàng học hỏi và phân loại thông tin hình ảnh một cách chính xác hơn.

Kỹ thuật này thường bao gồm các bước sau:

- **Phân tích histogram của ảnh**: Xác định phân bố cường độ pixel hiện tại trong ảnh.
- **Áp dụng phép biến đổi**: Dựa trên histogram, áp dụng một phép biến đổi nhằm mục đích mở rộng phạm vi của histogram, từ đó làm tăng độ tương phản. Phương pháp phổ biến là sử dụng "histogram stretching" hoặc "histogram equalization".

Noise:

Tăng cường tính ổn định khi mô hình được huấn luyện để chịu đựng nhiễu, nó trở nên ổn định hơn đối với các sai lệch và sự không hoàn hảo trong dữ liệu đầu vào. Nhiễu có thể giúp mô hình không tập trung quá mức vào các chi tiết nhỏ, không đại diện cho toàn bộ bộ dữ liệu, từ đó hạn chế hiện tượng overfitting.

- **Nhiễu Gauss (Gaussian Noise)**: Loại nhiễu phổ biến nhất, có đặc điểm là các giá trị nhiễu được thêm vào một cách ngẫu nhiên theo phân phối Gauss. Trong hình ảnh, điều này có thể được hiểu như là một sự phân bố ngẫu nhiên của các điểm sáng và tối.
- **Nhiễu muối tiêu (Salt-and-Pepper Noise)**: Loại nhiễu này tạo ra các điểm trắng và đen ngẫu nhiên trong hình ảnh, giống như hạt muối và hạt tiêu rắc lên trên bề mặt.
- **Nhiễu định lượng (Quantization Noise)**: Nhiễu này xảy ra do sự giới hạn của quá trình lượng tử hóa, khi dữ liệu được biểu diễn trong một số lượng hữu hạn các mức giá trị.

Rotation:

Xoay trong làm giàu dữ liệu là quá trình biến đổi hình ảnh bằng cách quay hình ảnh xung quanh trung tâm của nó ở các góc độ khác nhau. Kỹ thuật này không thay đổi nội dung của hình ảnh mà chỉ thay đổi vị trí và góc nhìn của các đối tượng trong hình ảnh. Mô hình được huấn luyện với hình ảnh đã được xoay sẽ học cách nhận diện các đối tượng ở nhiều hướng khác nhau, giúp cải thiện độ chính xác khi áp dụng mô hình vào thực tế, nơi các đối tượng có thể xuất hiện ở bất kỳ góc độ nào.

- **Xoay tuyến tính**: Hình ảnh được xoay một góc cụ thể xung quanh trục trung tâm. Góc xoay có thể được chọn một cách ngẫu nhiên hoặc theo một phân phối cụ thể trong một khoảng cho phép (ví dụ, từ -45 đến +45 độ).
- **Xử lý sau khi xoay**: Do quá trình xoay có thể dẫn đến việc một số phần của hình ảnh bị chệch ra ngoài biên của khung hình, có thể cần phải áp dụng các phương pháp như cắt xén hoặc điền các pixel mới vào các khu vực trống.

Shear:

Shearing (cắt lệch) là một phép biến đổi affine trong xử lý hình ảnh, nơi mà các điểm trong hình ảnh được dịch chuyển theo một góc nhất định dọc theo một trong hai trục của ảnh, không làm thay đổi kích thước tổng thể của hình ảnh nhưng thay đổi hình dạng của các đối tượng trong hình ảnh. Shearing giúp mô hình học được cách nhận diện đối tượng không chỉ trong tư thế tiêu chuẩn mà còn ở những tư thế bị lệch hoặc méo mó, điều này phản ánh đúng hơn đa dạng của dữ liệu trong thực tế.

Kỹ thuật này thường bao gồm các bước sau:

- **Xác định trục và góc shear:** Trong phép cắt lệch, bạn cần xác định trục (ngang hoặc dọc) và góc shear. Góc này thường được đo bằng độ hoặc radian và quyết định mức độ lệch của hình ảnh.
- **Biến đổi affine:** Sử dụng ma trận biến đổi affine để thực hiện shear. Ví dụ, một shear ngang có thể được biểu diễn như một phép nhân ma trận với ma trận đầu vào.

Mosaic

Mosaic trong xử lý ảnh là một kỹ thuật nơi mà nhiều hình ảnh nhỏ được ghép lại thành một hình ảnh lớn, mỗi hình ảnh nhỏ chiếm một phần của hình ảnh lớn. Kỹ thuật này cho phép mô hình nhìn thấy và học hỏi từ nhiều đối tượng cùng một lúc, mỗi đối tượng đến từ một phần khác nhau của dữ liệu huấn luyện. Mô hình được huấn luyện với dữ liệu dạng Mosaic sẽ học được cách nhận diện nhiều đối tượng cùng lúc, cải thiện khả năng xử lý tình huống khi nhiều đối tượng xuất hiện cùng một khung hình.

Kỹ thuật này thường bao gồm các bước sau:

- **Ghép hình ảnh**: Các hình ảnh này được ghép lại với nhau theo một cách thức cụ thể, thường là bốn hình ảnh được ghép thành một theo hình chữ thập hoặc theo hình vuông.

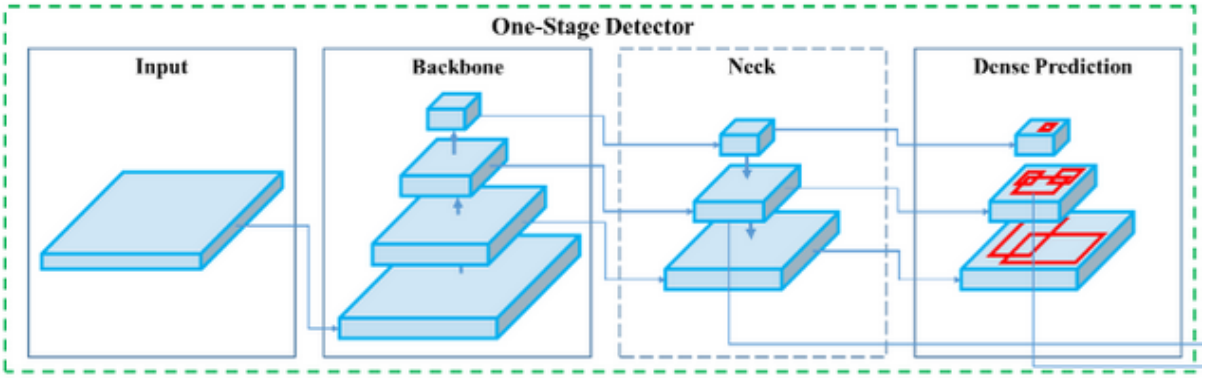
- **Xử lý kích thước**: Điều chỉnh kích thước của các hình ảnh con sao cho phù hợp với kích thước mong muốn của hình ảnh lớn.

3.3. YOLOv5

3.3.1. Kiến trúc cấp cao cho single-stage object detectors:

Có hai loại mô hình object detection: two-stage object detectors và single-stage object detectors.

Kiến trúc của Single-stage object detectors (như YOLO) bao gồm ba thành phần: Backbone, Neck và Head để đưa ra các dự đoán dày đặc như trong hình dưới đây.



Hình 1. Kiến trúc của Single-stage object detector

Backbone của mô hình: là một mạng được đào tạo trước được sử dụng để trích xuất biểu diễn tính năng phong phú cho hình ảnh. Điều này giúp giảm độ phân giải không gian của hình ảnh và tăng độ phân giải tính năng (kênh) của nó.

Neck của mô hình: được sử dụng để trích xuất các kim tự tháp đặc trưng. Điều này giúp mô hình có thể khái quát hóa tốt các đối tượng ở các kích cỡ và tỷ lệ khác nhau.

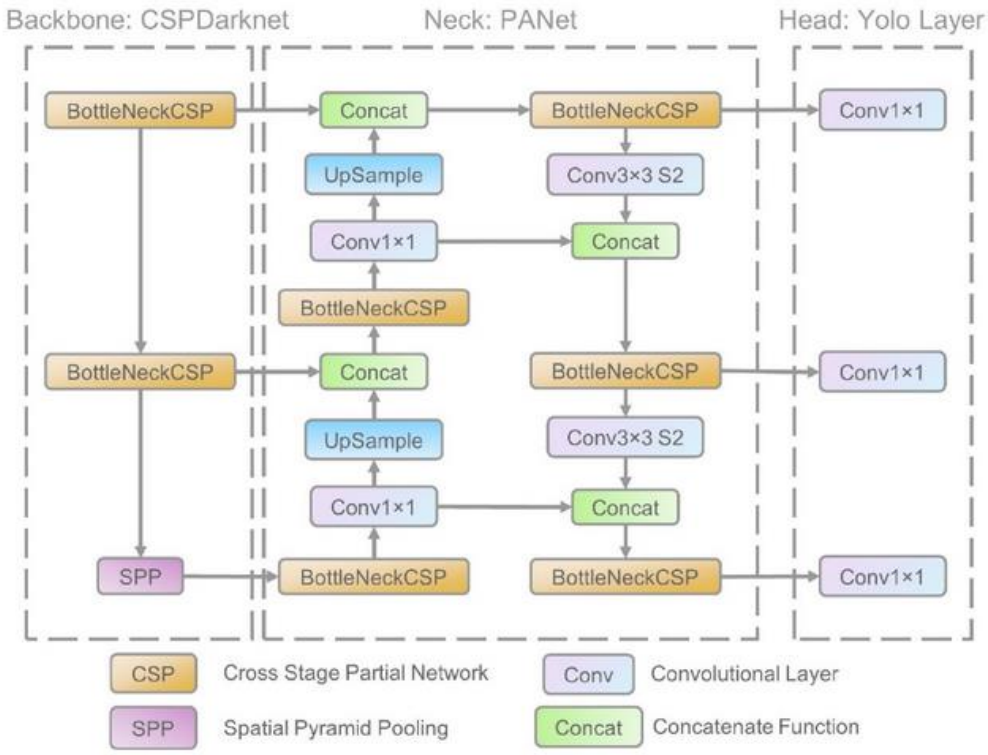
Head của mô hình: được sử dụng để thực hiện các hoạt động ở giai đoạn cuối. Nó áp dụng các hộp neo trên bản đồ tính năng và hiển thị đầu ra cuối cùng: các lớp, điểm đối tượng và hộp giới hạn.

3.3.2. Kiến trúc của YOLOv5

Có 5 phiên bản YOLOv5: đó là YOLOv5x, YOLOv5l, YOLOv5m, YOLOv5s và YOLOv5n.

Ngoài ra còn có 5 phiên bản lớn hơn tương ứng là YOLOv5x6, YOLOv5l6, YOLOv5m6, YOLOv5s6 và YOLOv5n6.

Tất cả các mô hình YOLOv5 đều bao gồm 3 thành phần giống nhau: CSP-Darknet53 như backbone, SPP và PANet ở phần neck của model và phần head được sử dụng trong YOLOv4.



Hình 2. Kiến trúc của YOLOv5

3.2.2.1. YOLOv5: backbone

YOLOv5 sử dụng CSP-Darknet53 làm backbone. CSP-Darknet53 là một phiên bản cải tiến của mạng Darknet53, một mạng nơ-ron tích chập sâu được sử dụng làm lõi (backbone) cho YOLOv3. CSP-Darknet53 được tạo ra bằng cách áp dụng chiến lược mạng Cross Stage Partial (CSP), là một phần quan trọng trong việc tối ưu hóa mạng nơ-ron cho việc nhận diện đối tượng.

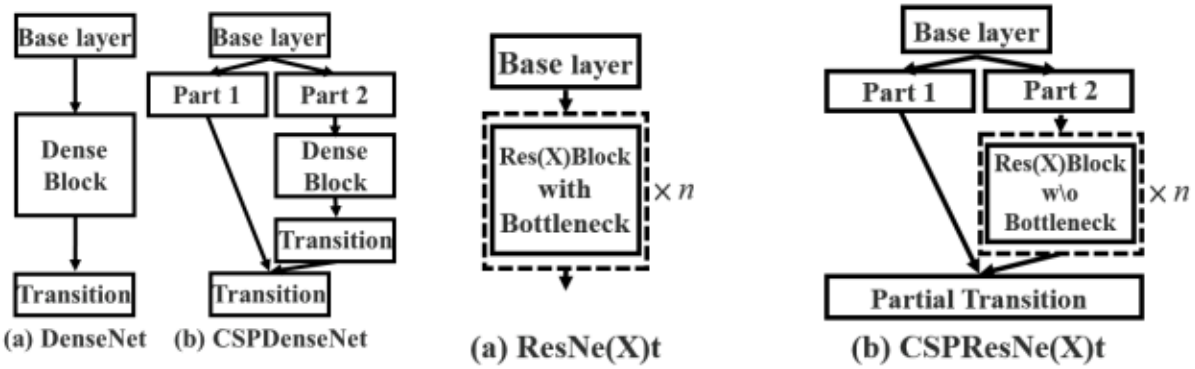
❖ **Cross Stage Partial (CSP)**

YOLO (You Only Look Once) là một mạng nơ-ron sâu được sử dụng cho việc nhận diện và định vị đối tượng trong ảnh. Nó sử dụng các khối residual và dense để giúp thông tin truyền từ các lớp sâu nhất, giúp mạng học được hiệu quả hơn và vượt qua vấn đề biến mất gradient.

Khối residual và dense được sử dụng trong YOLO giúp mạng học được sâu hơn bằng cách xây dựng các "đường ngắn" hoặc "đường dày" cho thông tin từ lớp

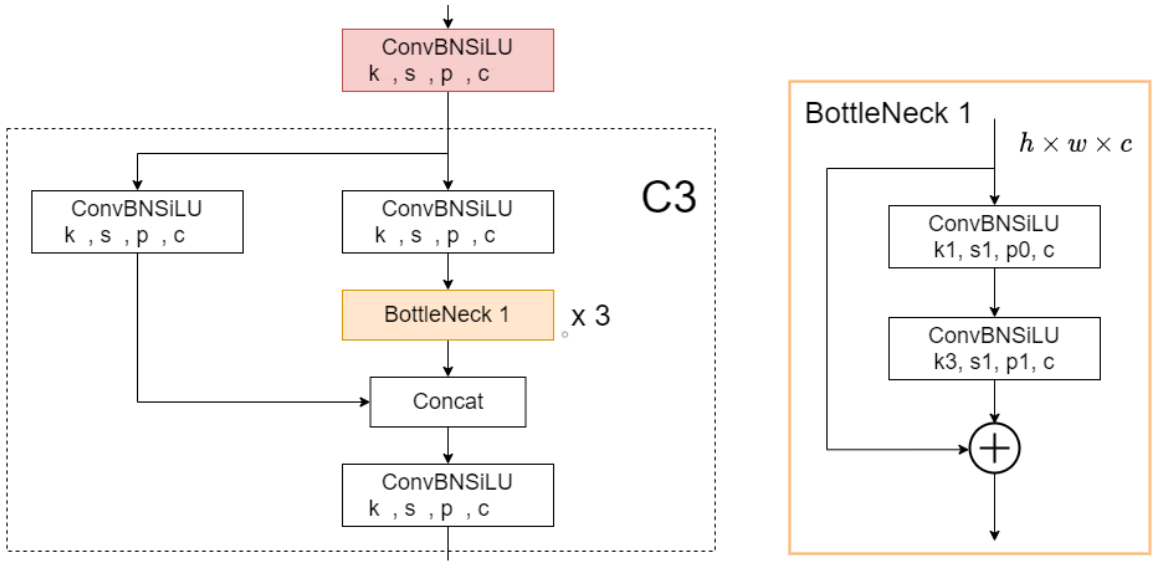
này đến lớp khác, giúp tránh được vấn đề gradient biến mất. Tuy nhiên, một vấn đề phát sinh khi sử dụng các khối này là có thể tạo ra các gradient lặp lại hoặc dư thừa, dẫn đến tăng thời gian huấn luyện và tăng cường bộ nhớ yêu cầu.

CSPNet (Cross Stage Partial Network) giúp giải quyết vấn đề này bằng cách cắt ngang luồng gradient tại một số giai đoạn của mạng. Bằng cách này, thông tin không cần thiết được loại bỏ và đồng thời giảm bớt sự lặp lại của gradient, từ đó cải thiện hiệu suất huấn luyện của mạng.



Hình 3. Áp dụng CSPNet cho ResNet và DenseNet

YOLOv5 sử dụng chiến lược CSPNet để phân chia bản đồ đặc trưng của lớp cơ sở thành hai phần và sau đó hợp nhất chúng thông qua hệ thống phân cấp nhiều giai đoạn như trong hình bên dưới



Hình 4. Kiến trúc mô-đun BottleNeckCSP.

❖ **CSPNet hoạt động như sau:**

Chia bản đồ đặc trưng: Ban đầu, bản đồ đặc trưng của lớp cơ sở được chia thành hai phần bằng cách cắt ngang theo kích thước của kênh đặc trưng. Mỗi phần sẽ chứa một phần của thông tin đặc trưng.

Tổ chức cấp bậc chéo: Sau đó, hai phần của bản đồ đặc trưng được đưa qua một cấp bậc chéo, nơi thông tin từ cả hai phần được kết hợp và tinh chỉnh. Cấp bậc chéo này

giúp hợp nhất thông tin từ các phần khác nhau của bản đồ đặc trưng, từ đó tạo ra một biểu diễn tổng thể của dữ liệu đầu vào.

Mục tiêu: Mục tiêu của việc này là tạo ra một biểu diễn đặc trưng tổng thể mạnh mẽ hơn bằng cách kết hợp thông tin từ nhiều phần của bản đồ đặc trưng và cải thiện khả năng của mô hình trong việc nhận diện và định vị đối tượng trong ảnh.

→ Áp dụng chiến lược này mang lại lợi ích lớn cho YOLOv5, vì nó giúp giảm số lượng tham số và giúp giảm lượng tính toán quan trọng (ít FLOPS hơn) dẫn đến tăng tốc độ suy luận, tham số quan trọng trong các mô hình phát hiện đối tượng thời gian thực.

3.2.1.1. Neck của YOLOv5

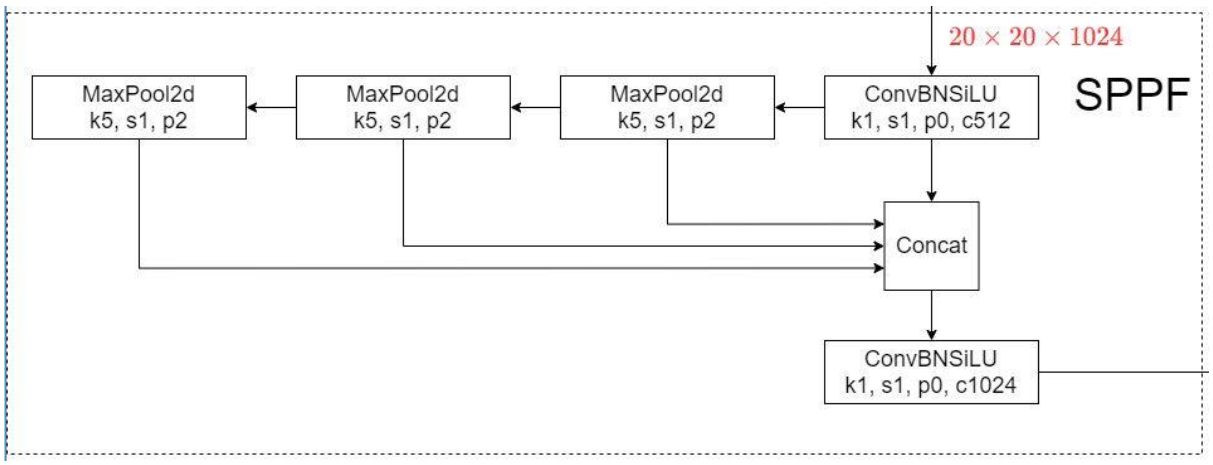
Trong YOLOv5, có hai cấu trúc được sử dụng để kết nối phần "backbone" và phần "head", đó là SPPF (Spatial Pyramid Pooling Fusion) và New CSP-PAN (Cross Stage Partial Path Aggregation Network).

❖ **SPPF (Spatial Pyramid Pooling Fusion):**

SPPF là một cấu trúc được sử dụng để tạo ra một biểu diễn đặc trưng đa tỷ lệ từ các đặc trưng được trích xuất từ phần "backbone" của mô hình.

Cấu trúc này tích hợp kỹ thuật Spatial Pyramid Pooling (SPP) để chế tạo đặc trưng từ các ảnh có kích thước khác nhau mà không cần phải thay đổi kích thước của ảnh đầu vào.

Sau đó, các đặc trưng từ SPP được kết hợp thông qua một quá trình gọi là "fusion", nơi thông tin từ các tỷ lệ và kích thước khác nhau được kết hợp lại để tạo ra một biểu diễn đặc trưng toàn diện và đa tỷ lệ.



Hình 5. Cấu trúc của khối SPPF.

❖ **New CSP-PAN (Cross Stage Partial Path Aggregation Network):**

New CSP-PAN là một cấu trúc kết hợp giữa CSPNet (Cross Stage Partial Network) và PAN (Path Aggregation Network), được sử dụng để tăng cường khả năng học và hiệu suất của mô hình.

CSPNet giúp giảm thiểu sự lặp lại của gradient và cải thiện khả năng học của mô hình bằng cách cắt ngang luồng gradient tại các giai đoạn của mạng. PAN giúp kết hợp

thông tin từ các lớp cùng độ sâu để tăng cường độ chính xác của việc nhận diện đối tượng.

Bằng cách kết hợp CSPNet và PAN, New CSP-PAN trong YOLOv5 tạo ra một quá trình kết hợp thông tin hiệu quả từ các lớp cùng độ sâu và từ các tỷ lệ và kích thước khác nhau của đặc trưng, giúp cải thiện hiệu suất của mô hình trong việc nhận diện và định vị đối tượng trong ảnh.

3.2.1.2. Head của YOLOv5

Trong YOLOv5, phần "head" của mô hình sử dụng cùng một kiến trúc với YOLOv3 và YOLOv4. Nó bao gồm ba lớp convolutional để dự đoán vị trí của các hộp giới hạn (x, y, chiều cao, chiều rộng), điểm số và các lớp đối tượng.

YOLOv5 có sự thay đổi trong cách tính toán tọa độ của các hộp giới hạn. Cách tính toán mới này có thể cải thiện hiệu suất của mô hình hoặc đảm bảo rằng các dự đoán tọa độ được chính xác hơn.

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

(b)

Hình 6. Công thức tính toán trong YOLOv5

3.3.3. Chiến lược training

YOLOv5 áp dụng một số chiến lược đào tạo phức tạp để nâng cao hiệu suất của mô hình. Chúng bao gồm:

- Training đa tỷ lệ: Hình ảnh đầu vào được thay đổi tỷ lệ ngẫu nhiên trong phạm vi từ 0,5 đến 1,5 lần kích thước ban đầu của chúng trong quá trình đào tạo.
- AutoAnchor: Chiến lược này tối ưu hóa các hộp neo trước đó để phù hợp với đặc điểm thống kê của các hộp chân lý cơ bản trong dữ liệu tùy chỉnh của bạn.
- Warmup và Cosine LR Scheduler: Một phương pháp điều chỉnh tốc độ học tập để nâng cao hiệu suất mô hình.

- Đường trung bình động hàm mũ (EMA): Một chiến lược sử dụng giá trị trung bình của các tham số qua các bước trước đây để ổn định quá trình đào tạo và giảm lỗi khái quát hóa.
- Huấn luyện độ chính xác hỗn hợp: Một phương pháp thực hiện các thao tác ở định dạng có độ chính xác một nửa, giảm mức sử dụng bộ nhớ và nâng cao tốc độ tính toán.
- Tiến hóa siêu tham số: Một chiến lược tự động điều chỉnh siêu tham số để đạt được hiệu suất tối ưu.

3.3.4. Tính toán mất mát:

Trong YOLOv5, ba thành phần mất mát được sử dụng để tính toán mức độ sai số trong quá trình huấn luyện:

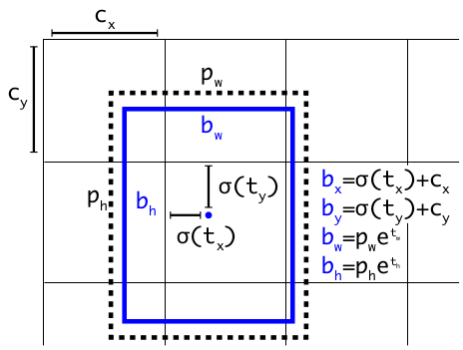
- **Classes Loss (BCE Loss):** Đo lường sai số trong việc phân loại các đối tượng.
- **Objectness Loss (BCE Loss):** Đo lường sai số trong việc xác định sự hiện diện của đối tượng trong ô lưới.
- **Location Loss (CIoU Loss):** Đo lường sai số trong việc xác định vị trí chính xác của đối tượng trong ô lưới.

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

Hình 7. Công thức tính toán mất mát

3.3.5. Loại bỏ độ nhạy của lưới

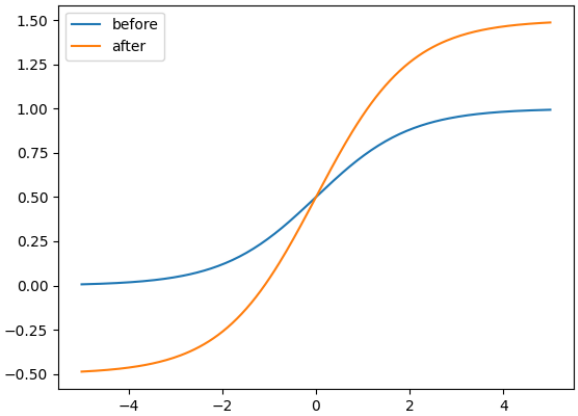
Trong YOLOv2 và YOLOv3, tọa độ của hộp giới hạn được dự đoán trực tiếp từ các kích hoạt của lớp cuối cùng của mô hình



Hình 8. Tính toán tọa độ của các hộp giới hạn (YOLOv2, YOLOv3)

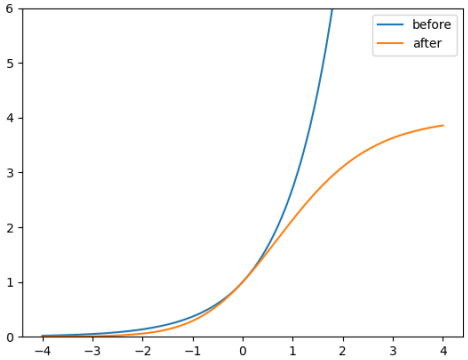
Tuy nhiên, trong YOLOv5, công thức để dự đoán các tọa độ hộp giới hạn đã được cập nhật để giảm độ nhạy cảm của lưới và ngăn mô hình dự đoán các kích thước hộp không giới hạn.

So sánh sự thay đổi của trung tâm trước và sau khi được tỉ lệ. Phạm vi trung tâm được điều chỉnh từ (0, 1) thành (-0.5, 1.5). Do đó, việc điều chỉnh có thể dễ dàng đạt được giá trị 0 hoặc 1.



Hình 9. Sự thay đổi của trung tâm trước và sau khi được tỉ lệ

So sánh tỷ lệ co giãn chiều cao và chiều rộng (tương đối với anchor) trước và sau khi điều chỉnh. Công thức ban đầu của YOLO/darknet box có một lỗi nghiêm trọng. Chiều rộng và chiều cao không giới hạn hoàn toàn vì chúng đơn giản là out=exp(in), điều này nguy hiểm vì nó có thể dẫn đến độ dốc chạy trốn, không ổn định, mất mát NaN và cuối cùng là mất hoàn toàn quá trình huấn luyện.



Hình 10. Tỷ lệ co giãn chiều cao và chiều rộng (tương đối với anchor) trước và sau khi điều chỉnh

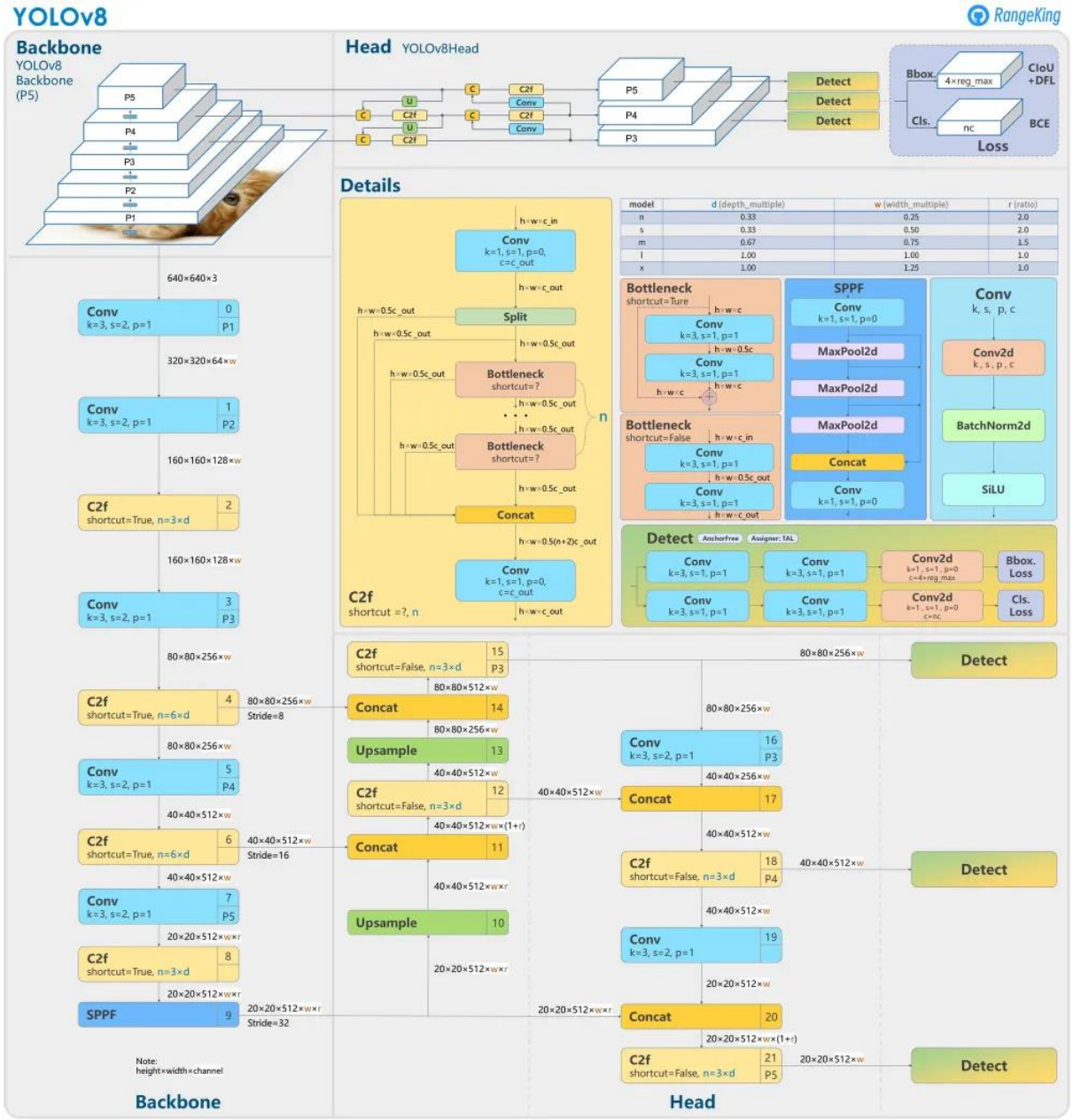
3.4. YOLOv8

3.4.1. Giới thiệu chung

YOLOv8 là mô hình YOLO tiên tiến mới nhất có thể được sử dụng để phát hiện đối tượng, phân loại hình ảnh và phân đoạn cá thể. YOLOv8 được phát triển bởi Ultralytics,

người cũng đã tạo ra mô hình YOLOv5 có sức ảnh hưởng và mang tính định hình trong ngành. YOLOv8 bao gồm nhiều thay đổi và cải tiến về trải nghiệm kiến trúc và nhà phát triển so với YOLOv5.

3.4.2. Kiến trúc của YOLOv8



Hình 11. Kiến trúc của YOLOv8

Phiên bản sửa đổi của kiến trúc CSPDarknet53 tạo thành phần backbone của YOLOv8. Kiến trúc này bao gồm 53 lớp tích chập và sử dụng các kết nối một phần xuyên giai đoạn để cải thiện luồng thông tin giữa các lớp khác nhau.

Phần head của YOLOv8 bao gồm nhiều lớp chập theo sau là một loạt các lớp được kết nối đầy đủ. Các lớp này chịu trách nhiệm dự đoán các hộp giới hạn, điểm đối tượng và xác suất lớp cho các đối tượng được phát hiện trong ảnh.

Một trong những tính năng chính của YOLOv8 là sử dụng cơ chế tự chú ý trong phần đầu của mạng. Cơ chế này cho phép mô hình tập trung vào các phần khác nhau của hình ảnh và điều chỉnh tầm quan trọng của các tính năng khác nhau dựa trên mức độ liên quan của chúng với nhiệm vụ.

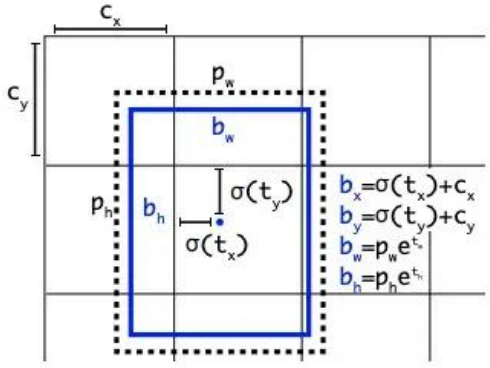
Một tính năng quan trọng khác của YOLOv8 là khả năng thực hiện phát hiện đối tượng ở nhiều tỷ lệ. Mô hình sử dụng mạng kim tự tháp đặc trưng để phát hiện các vật thể có kích thước và tỷ lệ khác nhau trong một hình ảnh.

Mạng kim tự tháp đặc trưng này bao gồm nhiều lớp phát hiện các vật thể ở các tỷ lệ khác nhau, cho phép mô hình phát hiện các vật thể lớn và nhỏ trong một hình ảnh.

3.4.2.1. Anchor Free Detection

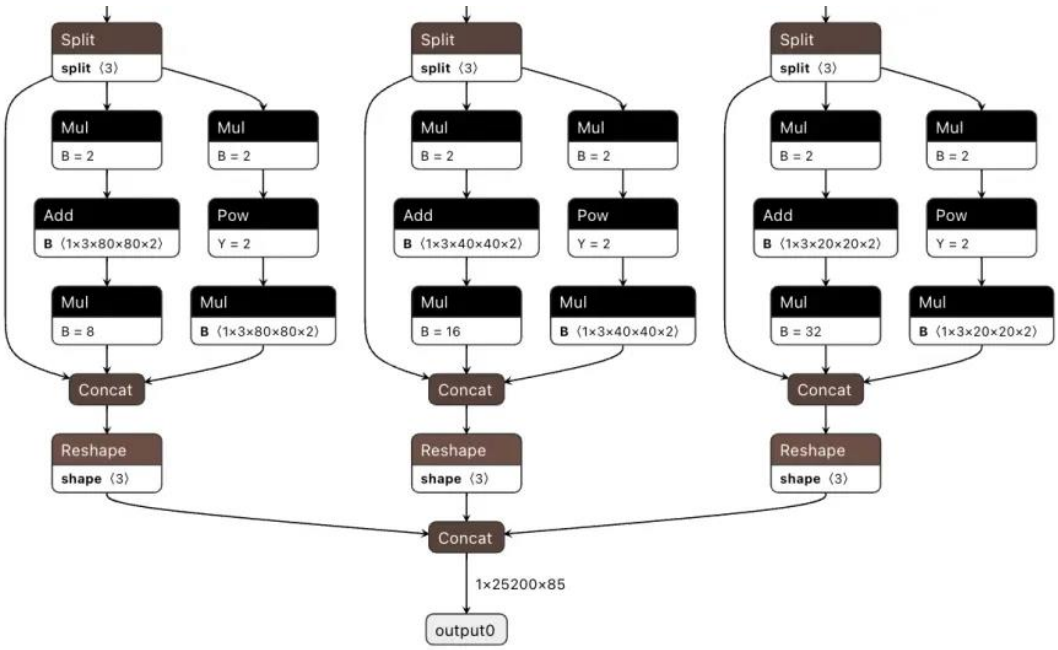
YOLOv8 không sử dụng anchor như các phiên bản trước. Thay vào đó, nó dự đoán trực tiếp trung tâm của đối tượng thay vì offset từ một anchor box đã biết. Điều này có nghĩa là mỗi ô lưới trên hình ảnh không cần phải được gắn với một tập hợp cố định các anchor box như trước. Thay vào đó, mô hình dự đoán trung tâm của đối tượng và kích thước của hộp giới hạn dựa trên các đặc trưng của ô lưới đó.

Việc loại bỏ anchor giúp đơn giản hóa quá trình dự đoán và giảm thiểu số lượng siêu tham số cần tinh chỉnh trong mô hình. Nó cũng có thể làm cho mô hình linh hoạt hơn trong việc xử lý các đối tượng có kích thước và tỷ lệ khác nhau trong ảnh. Tuy nhiên, để đạt được hiệu suất tốt, YOLOv8 cần có cách tiếp cận dự đoán trung tâm và kích thước hộp giới hạn một cách chính xác và ổn định.



Hình 12. Trực quan hóa anchor box trong YOLO

Trong các mô hình YOLO trước đây, việc sử dụng anchor boxes là một phần khá phức tạp, vì chúng có thể đại diện cho phân phối của các hộp giới hạn trong bộ dữ liệu thử nghiệm chuẩn nhưng không phản ánh phân phối của bộ dữ liệu tùy chỉnh.

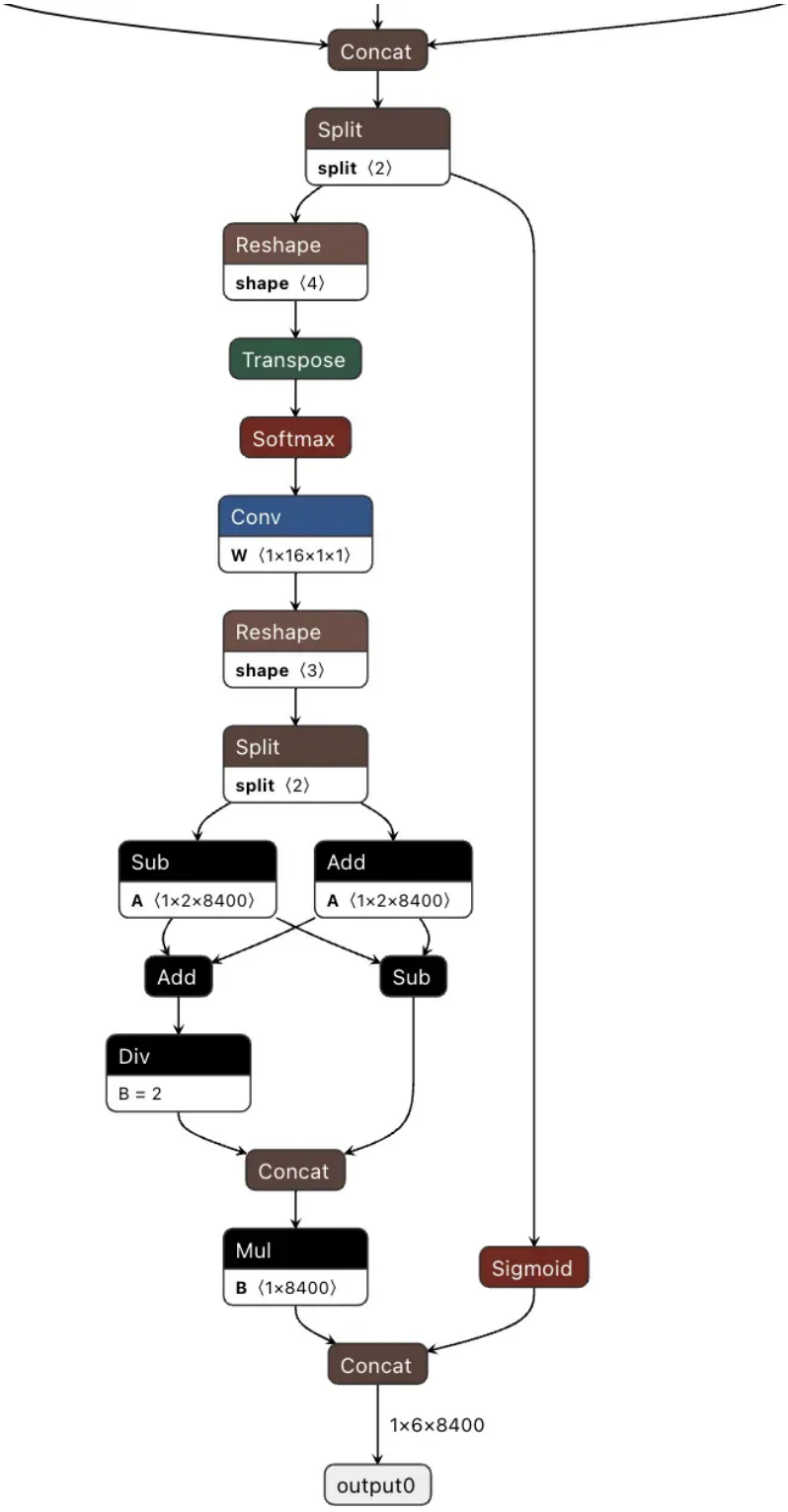


Hình 13. Phần đầu phát hiện (detection head) của YOLOv5

Phát hiện không cần anchor giảm số lượng dự đoán hộp, điều này làm tăng tốc độ của bước "Non-Maximum Suppression" (NMS), một bước xử lý phức tạp sau khi dự đoán mà lọc các dự đoán ứng viên sau khi dự đoán.

Bước "Non-Maximum Suppression" (NMS) là quá trình quan trọng trong việc loại bỏ các dự đoán trùng lặp và không cần thiết sau khi mô hình đã dự đoán các hộp giới hạn cho các đối tượng trong ảnh. Khi có nhiều hộp giới hạn trùng lặp cho cùng một đối tượng, NMS sẽ giữ lại hộp có độ tin cậy cao nhất và loại bỏ các hộp khác.

Khi sử dụng phát hiện không cần anchor, số lượng dự đoán hộp giảm đi đáng kể so với việc sử dụng anchor. Điều này giúp làm giảm thời gian tính toán của bước NMS, vì có ít hơn các dự đoán cần được xử lý sau khi dự đoán. Do đó, việc sử dụng phát hiện không cần anchor có thể làm tăng tốc độ toàn bộ quá trình phát hiện đối tượng.



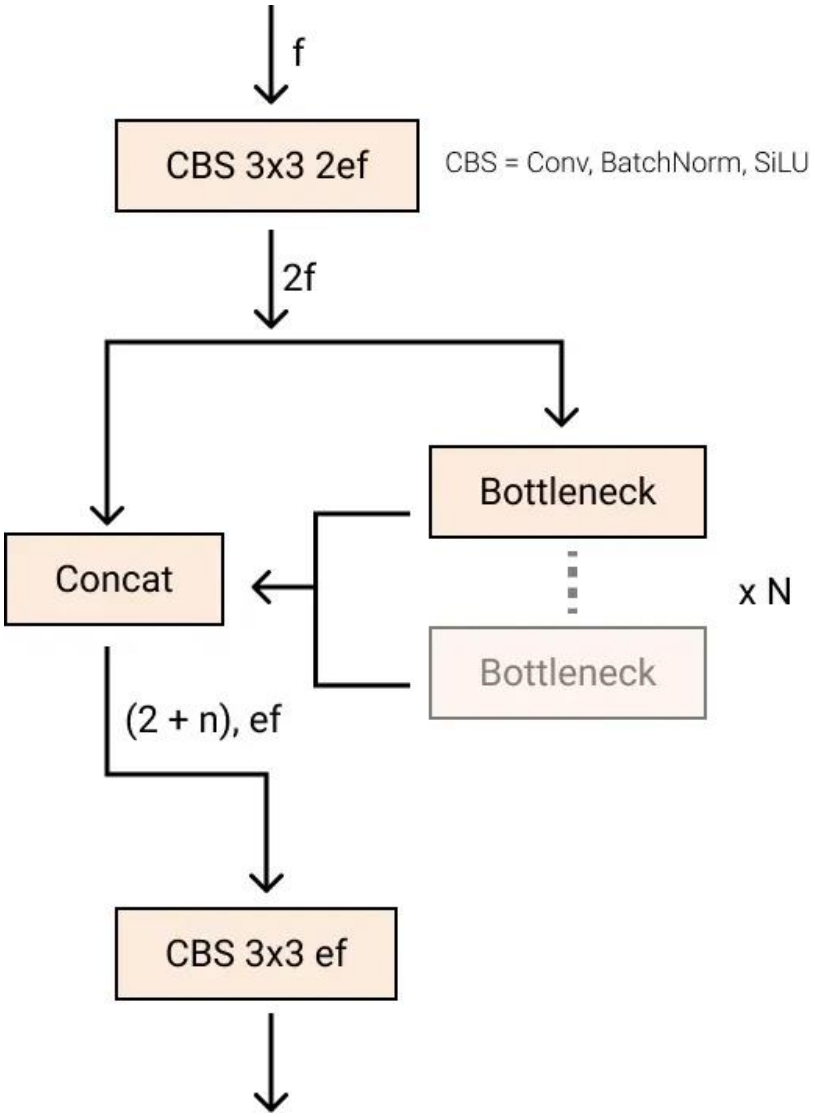
Hình 14. Phần đầu phát hiện (detection head) của YOLOv8

3.4.2.2. New convolutions

Trong phần stem của YOLOv8, lớp convolution 6x6 ban đầu được thay thế bằng một lớp convolution 3x3. Kiến trúc chính của module cũng đã được thay đổi và C2f đã

thay thế C3. Module này được tóm tắt trong hình dưới đây, trong đó "f" là số đặc trưng, "e" là tỷ lệ mở rộng và CBS là một khối gồm một lớp Convolution, một lớp Batch Normalization và một lớp SiLU (Scaled Exponential Linear Unit).

Trong C2f, tất cả các đầu ra từ Bottleneck (tên phức tạp cho hai lớp convolution 3x3 với kết nối dư) được nối lại. Trong khi đó, trong C3 chỉ có đầu ra của Bottleneck cuối cùng được sử dụng.



Hình 15. C2f module trong YOLOv8

YOLOv8 có Bottleneck giống như trong YOLOv5 nhưng kích thước kernel của conv đầu tiên đã được thay đổi từ 1x1 thành 3x3. Từ thông tin này, chúng ta có thể thấy rằng YOLOv8 đang bắt đầu quay trở lại khối ResNet được định nghĩa từ năm 2015.

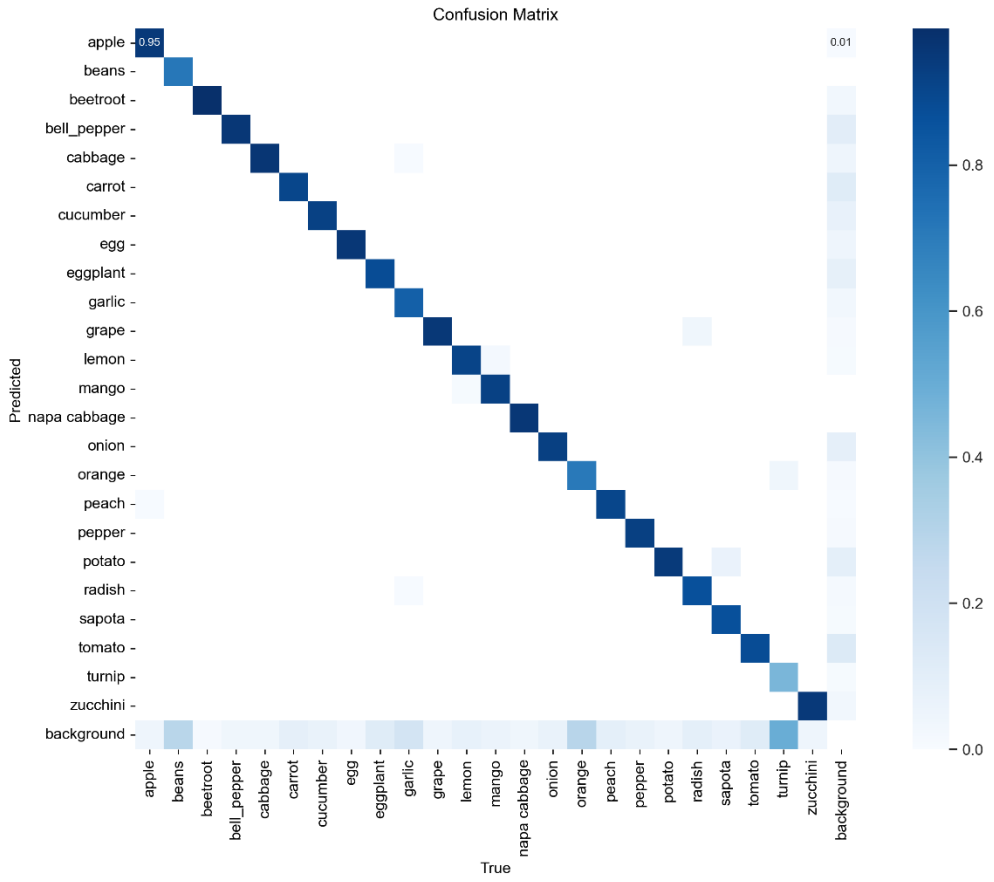
Tại phần neck, các đặc trưng được nối trực tiếp mà không ép buộc các chiều kênh giống nhau. Điều này giảm số lượng tham số và kích thước tổng thể của các tensor.

3.5. So sánh giữa YOLOv5s và YOLOv8n

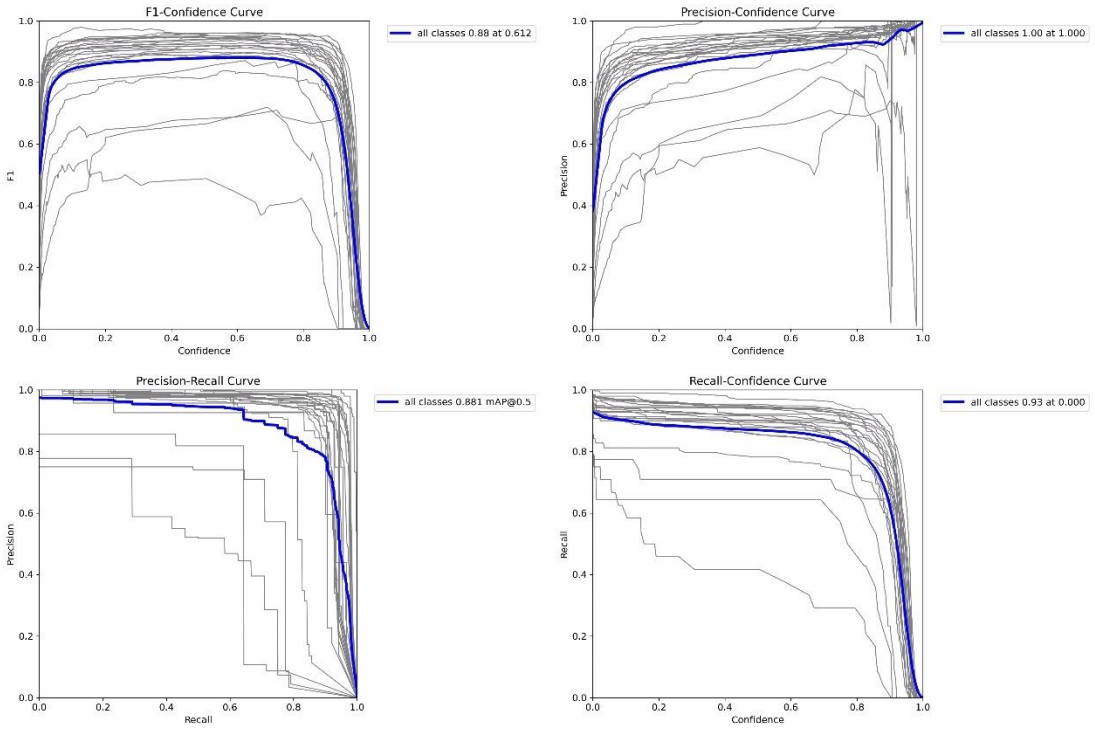
Tiêu chí		**YOLOv8n**	**YOLOv5s**
Kiến trúc	Stem	Lớp convolution kernel (3x3)	Lớp convolution kernel (6x6)
	Backbone	Lớp bottleneck convolution kernel (3x3)	Lớp bottleneck convolution kernel (1x1)
		Khối xây dựng C2f	Khối xây dựng C3
	Neck	Kết hợp các đặc trưng mà không thay đổi kích thước	Kết hợp các đặc trưng với việc thay đổi kích thước
Kích thước		Nhỏ hơn	Lớn hơn
Độ chính xác		Cao hơn	Thấp hơn
Tốc độ		Chậm hơn trong một số trường hợp	Nhanh hơn trong nhiều trường hợp

3.5.1. Kết quả thực tế của độ chính xác giữa YOLOv5 và YOLOv8

3.5.1.1. Kết quả về độ chính xác của YOLOv5

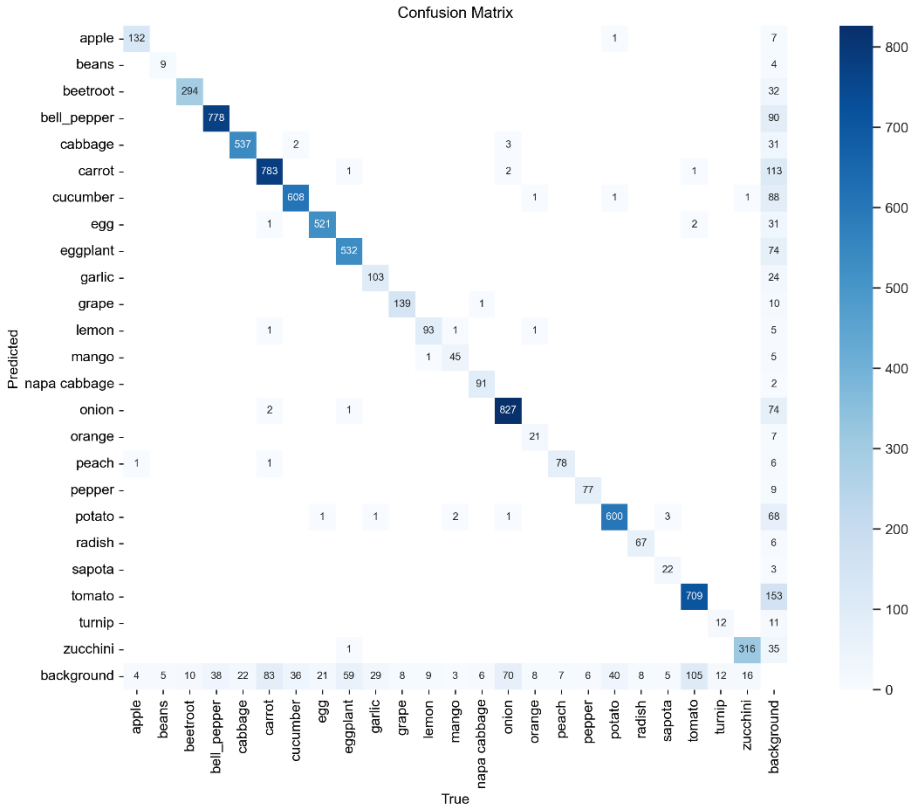


Hình 16. Biểu đồ confusion matrix đánh giá độ chính xác mô hình YOLOv5

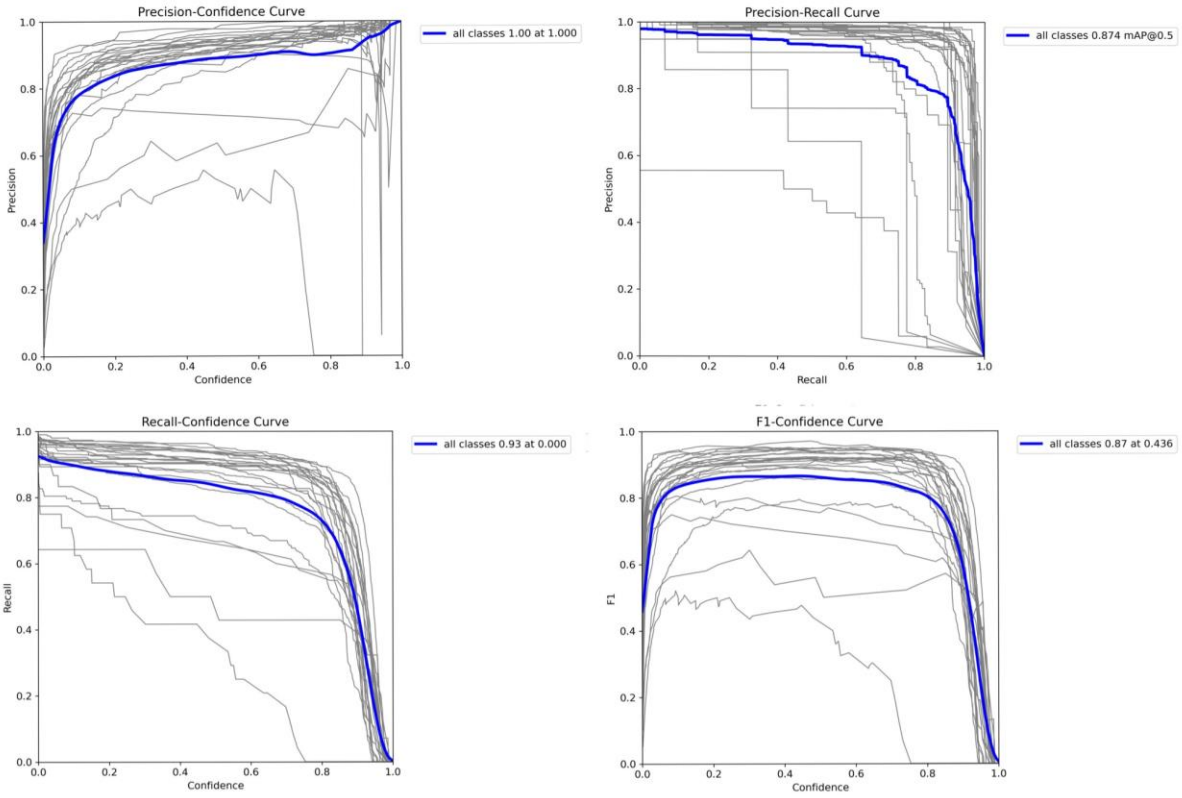


Hình 17. Các biểu đồ đánh giá độ chính xác của YOLOv5

3.5.1.2. Kết quả về độ chính xác của YOLOv8



Hình 18. Biểu đồ confusion matrix đánh giá độ chính xác mô hình YOLOv8



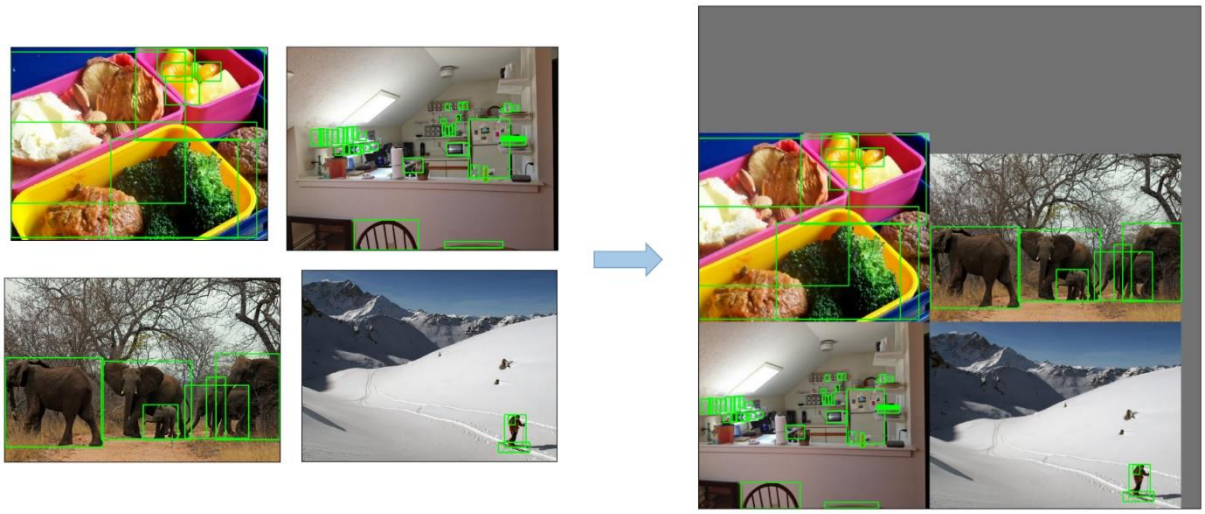
Hình 19. Các biểu đồ đánh giá độ chính xác của YOLOv8

3.5. Kỹ thuật tăng cường dữ liệu (trong mô hình)

YOLOv5 sử dụng nhiều kỹ thuật tăng cường dữ liệu khác nhau để cải thiện khả năng khái quát hóa và giảm thiểu tình trạng quá khớp của mô hình. Những kỹ thuật này bao gồm:

❖ **Tăng cường khảm (Mosaic Augmentation):**

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



Hình 20. Tăng cường khảm (Mosaic Augmentation)

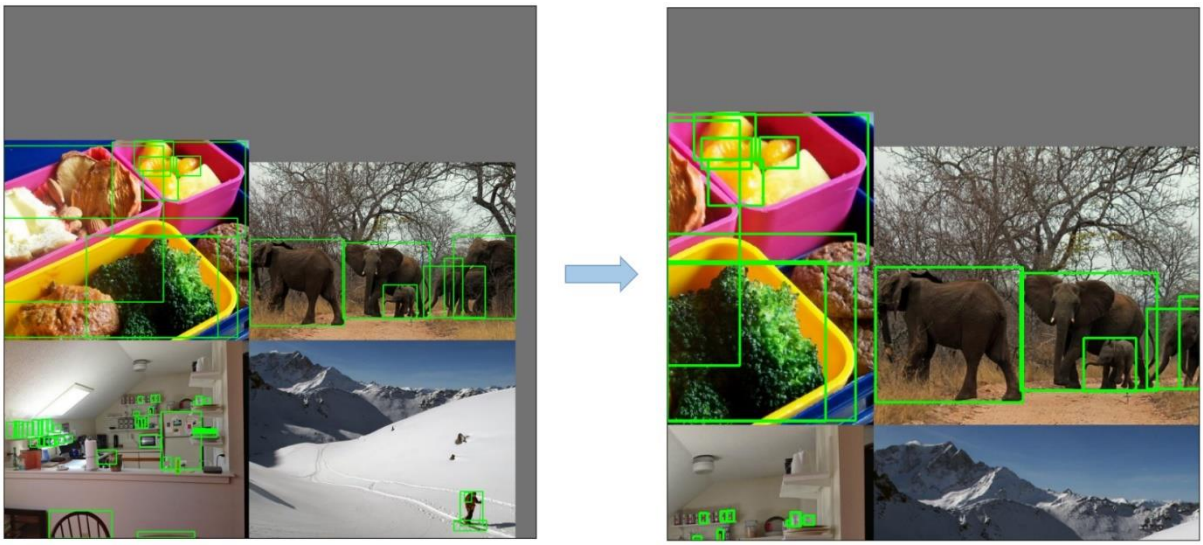
❖ **Tăng cường sao chép-dán (Copy-Paste Augmentation):**

Một phương pháp tăng cường dữ liệu cải tiến sao chép các mảng ngẫu nhiên từ một hình ảnh và dán chúng vào một hình ảnh được chọn ngẫu nhiên khác, tạo ra mẫu đào tạo mới một cách hiệu quả.

Hình 21. Tăng cường sao chép-dán (Copy-Paste Augmentation)

❖ **Biến đổi Affine ngẫu nhiên (Random Affine Transformations)**

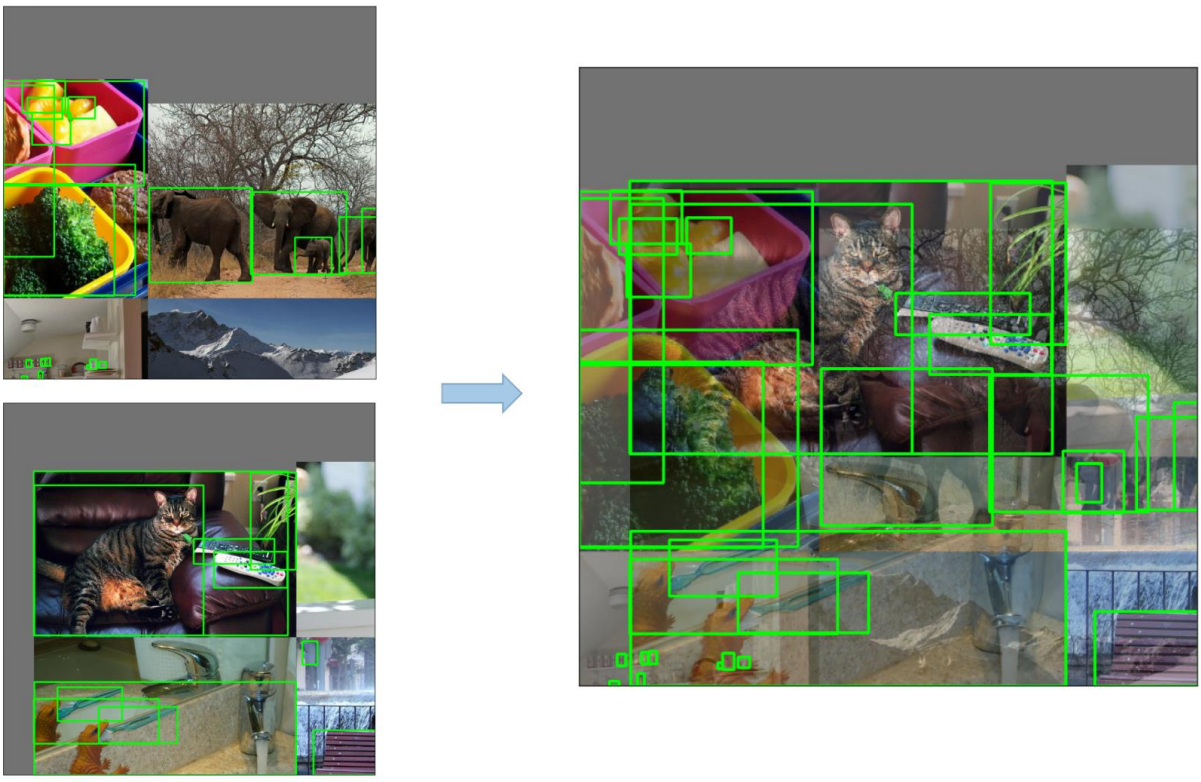
Điều này bao gồm xoay ngẫu nhiên, chia tỷ lệ, dịch và cắt hình ảnh.



Hình 22. Biến đổi Affine ngẫu nhiên (Random Affine Transformations)

❖ **Tăng cường MixUp (MixUp Augmentation):**

Phương pháp tạo hình ảnh tổng hợp bằng cách kết hợp tuyến tính hai hình ảnh và nhãn liên quan của chúng.



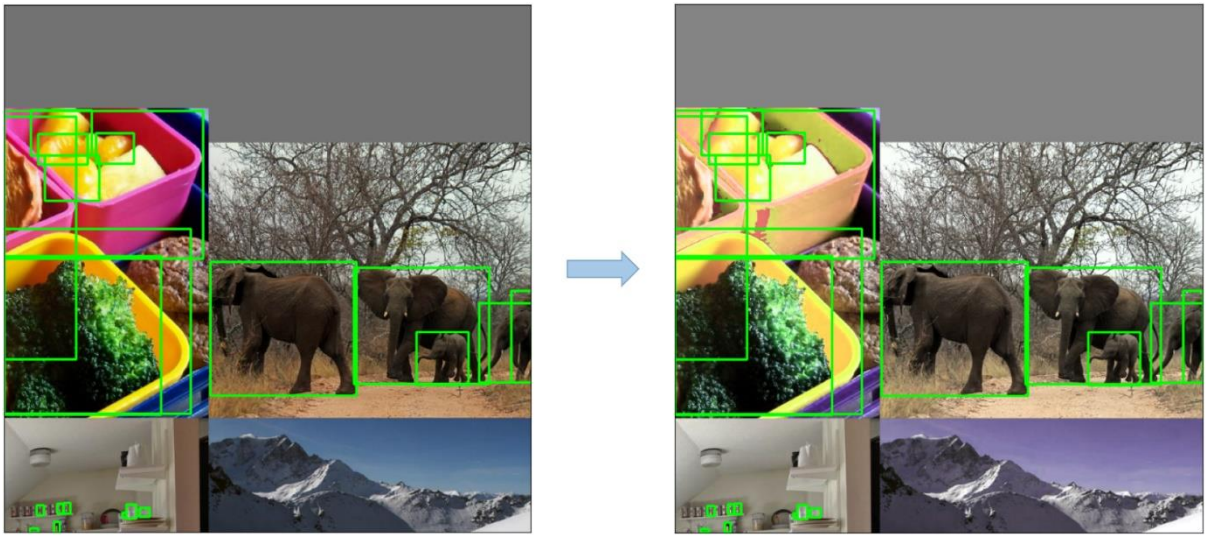
Hình 23. Tăng cường MixUp (MixUp Augmentation)

❖ **Albumentations:**

Một thư viện mạnh mẽ để tăng cường hình ảnh hỗ trợ nhiều kỹ thuật tăng cường khác nhau.

❖ **Tăng cường HSV (HSV Augmentation):**

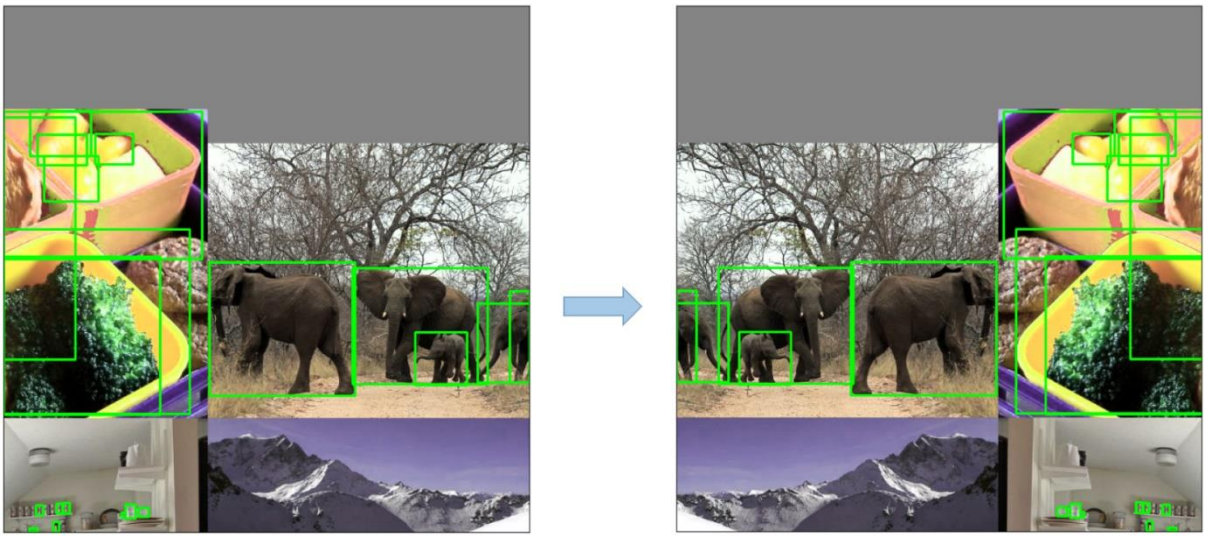
Thay đổi ngẫu nhiên về Màu sắc, Độ bão hòa và Giá trị của hình ảnh.



Hình 24. Tăng cường HSV (HSV Augmentation)

❖ **Lật ngang ngẫu nhiên (Random Horizontal Flip):**

Một phương pháp tăng cường lật ngẫu nhiên hình ảnh theo chiều ngang.



Hình 25. Lật ngang ngẫu nhiên (Random Horizontal Flip)

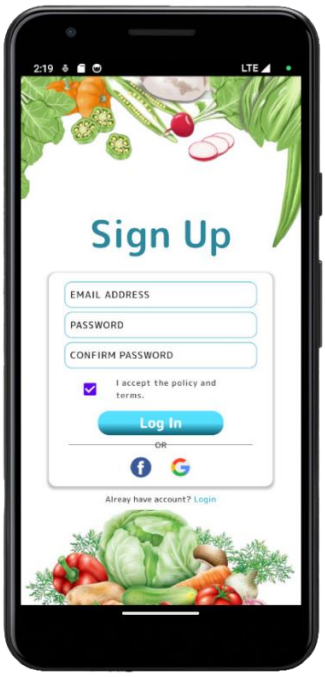
CHƯƠNG 4: HÌNH ẢNH THỰC TẾ CỦA ỨNG DỤNG

4.1. Hình ảnh giao diện ứng dụng

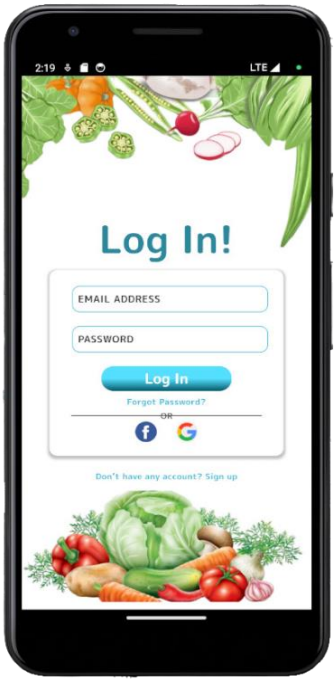
Hình 26. Ứng dụng ngoài màn hình nền

Hình 27. Màn hình tải vào ứng dụng

4.2. Hình ảnh giao diện chức năng đăng ký, đăng nhập và thay đổi mật khẩu

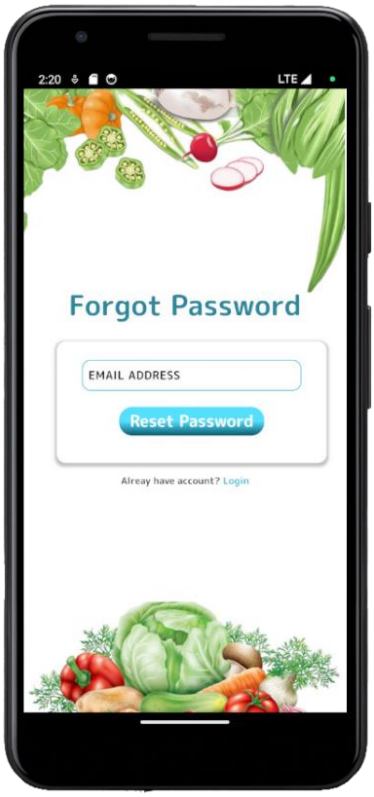


Hình 28. Màn hình đăng ký tài khoản



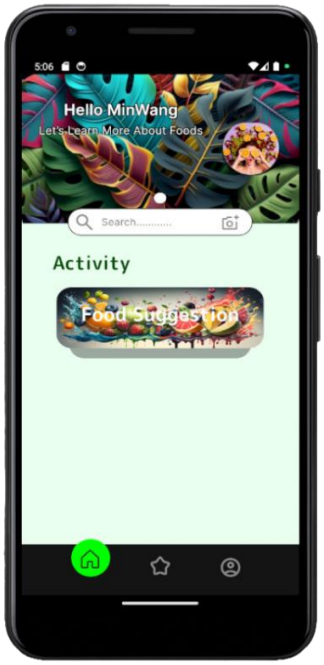
Hình 29. Màn hình đăng nhập tài khoản

Hình 30. Màn hình thay đổi mật khẩu



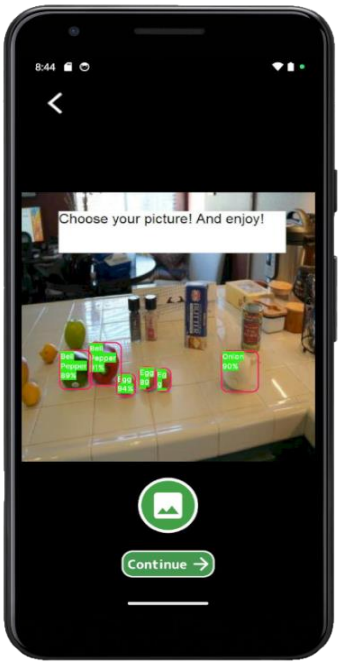
Hình 31. Màn hình lấy lại mật khẩu

4.3. Hình ảnh giao diện chức năng gợi ý món ăn thông qua mô hình nhận diện rau củ quả

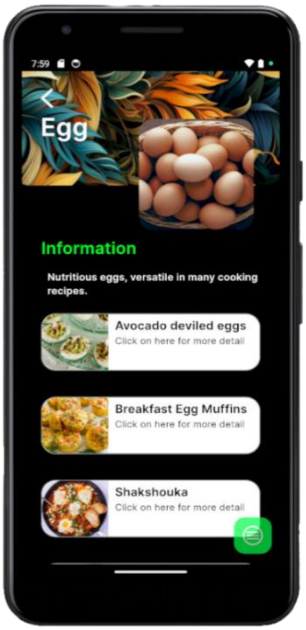


Hình 32. Màn hình trang chủ

Hình 33. Màn hình chức năng

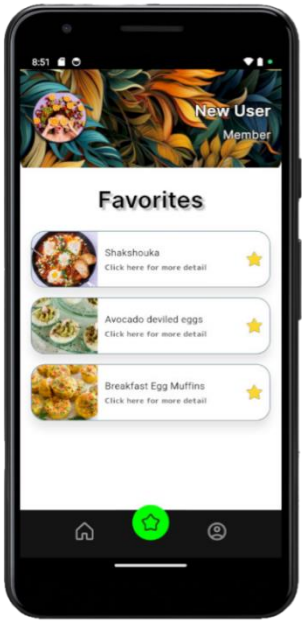


Hình 34. Màn hình chọn hình ảnh



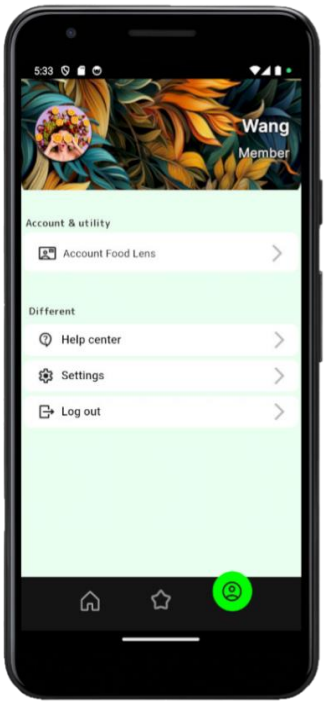
Hình 35. Màn hình đề xuất món ăn

Hình 36. Màn hình chi tiết món ăn



Hình 37. Màn hình lưu món ăn ưa thích

4.4. Hình ảnh giao diện quản lý tài khoản cá nhân

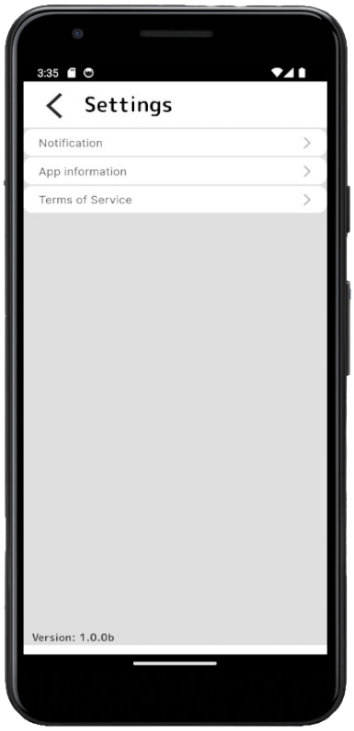


Hình 38. Màn hình tài khoản cá nhân

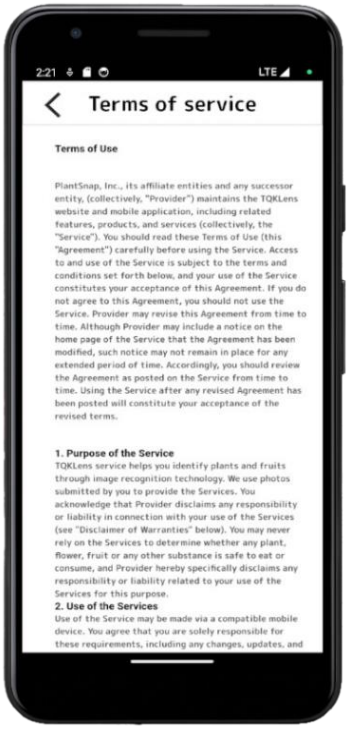
Hình 39. Màn hình account setup

Hình 40. Màn hình đổi tên User

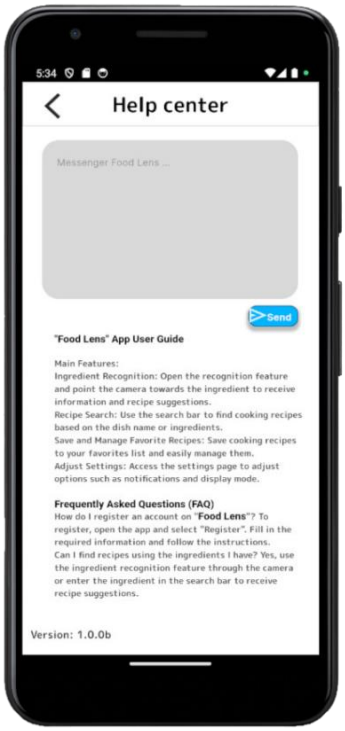
4.5. Hình ảnh giao diện chung về ứng dụng



Hình 41. Màn hình Setting



Hình 42. Màn hình điều khoản và dịch vụ



Hình 43. Màn hình help Center

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Tổng kết

FoodLens trên di động là một ứng dụng được phát triển để nhận diện và gợi ý các món ăn dựa trên hình ảnh của rau củ quả. Ứng dụng này sử dụng các mô hình phát hiện đối tượng YOLOv5 và YOLOv8 để nhận diện và phân loại các loại rau củ quả từ hình ảnh chụp bởi người dùng.

Quá trình huấn luyện này giúp mô hình hiểu được các đặc điểm và đặc trưng của từng loại rau củ quả, từ đó có thể nhận diện chính xác trong thời gian thực.

Sau khi nhận diện được các loại rau củ quả, ứng dụng sẽ gợi ý các món ăn dựa trên loại rau củ quả mà người dùng đã chụp.

FoodLens trên di động mang lại lợi ích lớn cho người dùng bằng cách giúp họ dễ dàng và nhanh chóng tìm ra các món ăn phù hợp với nguyên liệu có sẵn trong tủ lạnh hoặc quanh khu vực của họ. Đồng thời, ứng dụng cũng thúc đẩy lối sống lành mạnh và chăm sóc sức khỏe thông qua việc tạo ra các gợi ý món ăn cân đối và dinh dưỡng từ các loại rau củ quả.

5.2. Hướng phát triển

Phát triển một đề tài về ứng dụng nhận diện rau củ quả để gợi ý món ăn bao gồm nhiều khía cạnh, từ quá trình train mô hình đến việc xây dựng ứng dụng cung cấp nền tảng sử dụng cho người dùng. Dưới đây là một số hướng phát triển cụ thể:

❖ Cải thiện mô hình:

Tiếp tục nghiên cứu và cải thiện mô hình nhận diện đối tượng bằng cách sử dụng các phương pháp như tăng cường dữ liệu, đa tác vụ học, và tối ưu hóa hyperparameters để tăng độ chính xác và độ tin cậy của mô hình.

Khám phá các kiến trúc mạng mới hoặc cải tiến như YOLOv5x và YOLOv8s để cải thiện khả năng nhận diện và phân loại các loại rau củ quả.

❖ Huấn luyện mô hình mới:

Huấn luyện các mô hình khác nhau cho các loại rau củ quả cụ thể và các món ăn phổ biến dựa trên dữ liệu huấn luyện phong phú và đa dạng.

Tạo ra các mô hình phân loại thực phẩm dựa trên hình ảnh để đề xuất các món ăn dựa trên yêu cầu dinh dưỡng, thói quen ăn uống, và sở thích cá nhân của người dùng.

❖ Cải thiện giao diện người dùng:

Tối ưu hóa giao diện người dùng để cung cấp trải nghiệm mượt mà và dễ sử dụng, bao gồm cả việc tăng tính tương tác và tùy chỉnh cho người dùng.

Bổ sung các tính năng mới như gợi ý món ăn dựa trên thời tiết, mùa vụ, và lịch sử tìm kiếm của người dùng để cung cấp trải nghiệm cá nhân hóa.

❖ Bổ sung đa dạng dữ liệu:

Mở rộng tập dữ liệu huấn luyện bằng cách thu thập dữ liệu từ nhiều nguồn khác nhau, bao gồm cả ảnh chụp thực tế và ảnh được tạo ra tự động từ môi trường mô phỏng.

Tăng cường dữ liệu bằng cách thêm các biến thể về ánh sáng, góc chụp, và nền để cải thiện khả năng tổng quát hóa của mô hình.

- ❖ Nâng cao tính ứng dụng thực tế:

Tích hợp tính năng nhận diện thực phẩm trong thời gian thực để người dùng có thể chụp ảnh và nhận được gợi ý món ăn ngay lập tức.

Phát triển tính năng tùy chỉnh chế độ ăn dựa trên thông tin dinh dưỡng cá nhân và mục tiêu sức khỏe của người dùng.

- ❖ Mở rộng phạm vi ứng dụng:

Nghiên cứu và phát triển tính năng mới như gợi ý công thức món ăn dựa trên các nguyên liệu có sẵn hoặc dựa trên thị hiếu và yêu cầu của người dùng.

Tích hợp tính năng tìm kiếm sản phẩm thực phẩm hoặc cửa hàng thực phẩm gần nhất dựa trên các sản phẩm được nhận diện từ hình ảnh.

TÀI LIỆU THAM KHẢO

[1]. Glenn Jocher, Sergiu Waxmann. Jan 14, 2024. Ultralytics YOLOv5 Architecture. From: https://docs.ultralytics.com/yolov5/tutorials/architecture_description/

[2]. WZMIAOMIAO, Glenn Jocher. Mar 16, 2022. YOLOv5 (6.0/6.1) brief summary #6998. From: https://github.com/ultralytics/yolov5/issues/6998#2

[3]. OpenGenus Tech Review Team. 2024. YOLO v5 model architecture [Explained]. From: https://iq.opengenus.org/yolov5/#:~:text=Single-stage%20object%20detectors%20(like,shown%20in%20the%20figure%20bellow.&text=The%20backbone%20is%20a%20pre,rich%20feature%20representation%20for%20images.

[4]. Jacob Solawetz. Jan 11, 2023. What is YOLOv8? The Ultimate Guide. [2024]. From: https://blog.roboflow.com/whats-new-in-yolov8/

[5]. VK_Venkatkumar. Nov 22, 2023. YoloV8 Architecture & Cow Counter With Region Based Dragging Using YoloV8. From: https://medium.com/@VK_Venkatkumar/yolov8-architecture-cow-counter-with-region-based-dragging-using-yolov8-e75b3ac71ed8

[6]. Augmented A.I. Feb 20,2023. YOLOv8 vs. YOLOv5: Choosing the Best Object Detection Model. From: https://www.augmentedstartups.com/blog/yolov8-vs-yolov5-choosing-the-best-object-detection-model

[7]. EG_Johnson. Dec 20, 2023. Yolov8 Architecture vs Yolov5. From: https://medium.com/@EG_Johnson/yolov8-architecture-vs-yolov5-49d23b462ea6