TRƯỜNG ĐẠI HỌC NGOẠI NGỮ-TIN HỌC TP. HỒ CHÍ MINH  
KHOA: KHOA HỌC DỮ LIỆU

**KHÓA LUẬN TỐT NGHIỆP**

**NGHIÊN CỨU VÀ CẢI TIẾN   
MÔ HÌNH NHẬN DIỆN   
HÌNH ẢNH YOLOV8**

GIẢNG VIÊN HƯỚNG DẪN: Phan Thị Ngọc Hân

SINH VIÊN THỰC HIỆN: Nguyễn Cao Hoàng - 21DH112490

**TP. HỒ CHÍ MINH – 8-2024**

# Lời cảm ơn

Trước tiên, nhóm chúng em gửi lời cảm ơn chân thành và sâu sắc đến cô Phan Thị Ngoc Hân, là giảng viên đã tận tình hướng dẫn, chỉ bảo, giúp đỡ em trong suốt thời gian em nghiên cứu khóa luận. Và cũng là người đưa ra những ý tưởng, kiểm tra sự phù hợp của khóa luận.

Em cũng xin gửi lời cảm ơn đến toàn thể giảng viên trường Đại học Ngoại ngữTin học thành phố Hồ Chí Minh, trong thời gian qua đã giảng dạy, và tạo điều kiện cho chúng em trong quá trình học tập và nghiên cứu tại trường. Những kiến thức mà chúng em nhận được sẽ là hành trang giúp chúng em vững bước trong tương lai.

Cuối cùng, em xin cảm ơn gia đình, bạn bè, người thân đã luôn ở bên để động viên và là nguồn cổ vũ lớn lao, là động lực giúp em hoàn thành khóa luận này. Mặc dù đã cố gắng hoàn thành khóa luận trong phạm vi và khả năng có thể. Tuy nhiên sẽ không tránh khỏi những thiếu sót. Em rất mong nhận được sự cảm thông và tận tình chỉ bảo của quý thầy cô và toàn thể các bạn.

Em xin chân thành cảm ơn.

# Lời cam đoan

Em xin cam đoan đề tài này là một công trình nghiên cứu độc lập với toàn bộ nội dung và kết quả là sản phẩm mà em đã nỗ lực nghiên cứu trong quá trình học tập tại trường cũng như dưới sự dẫn dắt của giảng viên hướng dẫn.

Trong quá trình viết bài có sự tham khảo một số tài liệu có nguồn gốc rõ ràng và đã được trích dẫn đầy đủ.

Em xin cam đoan nếu có vấn đề gì em xin chịu hoàn toàn trách nhiệm.

MỤC LỤC

[Lời cảm ơn i](#_Toc173733549)

[Lời cam đoan ii](#_Toc173733550)

Mục lục……………………………………………………………………………………………………….iii

[Danh mục các từ viết tắt vii](#_Toc173733551)

[Danh mục các hình vẽ v](#_Toc173733552)iii

[Chương 1: Mở đầu 9](#_Toc173733553)

[1.1 Lý do chọn đề tài 9](#_Toc173733554)

[1.2 Nội dung nghiên cứu 9](#_Toc173733555)

[Chương 2: Tổng quan 10](#_Toc173733556)

[2.1 Lịch sử YOLO 10](#_Toc173733557)

[2.1.1 YOLO (You Only Look Once) - 2015 10](#_Toc173733558)

[2.1.2 YOLOv2 (YOLO9000) - 2016 10](#_Toc173733559)

[2.1.3 YOLOv3 - 2018 10](#_Toc173733560)

[2.1.4 YOLOv4 - 2020 11](#_Toc173733561)

[2.1.5 Scaled-YOLOv4 - 2020 11](#_Toc173733562)

[2.1.6 YOLOv5 - 2020 11](#_Toc173733563)

[2.1.7 PP-YOLO - 2020 12](#_Toc173733564)

[2.1.8 PP-YOLOv2 - 2021 12](#_Toc173733565)

[2.1.9 YOLOR - 2021 12](#_Toc173733566)

[2.1.10 YOLOX - 2021 12](#_Toc173733567)

[2.1.11 YOLOv6 - 2022 13](#_Toc173733568)

[2.1.12 YOLOv7 - 2022 13](#_Toc173733569)

[2.1.13 PP-YOLOE - 2022 13](#_Toc173733570)

[2.1.14 YOLOv8 - 2023 14](#_Toc173733571)

[2.2 Kiến trúc YOLOv8 14](#_Toc173733572)

[2.2.1 Tổng quan 14](#_Toc173733573)

[2.2.2 Cải tiến quan trọng 18](#_Toc173733574)

[2.2.3 Ưu, nhược điểm 19](#_Toc173733575)

[Chương 3: Phương pháp đề xuất 20](#_Toc173733576)

[3.1 Điều chỉnh siêu tham số 20](#_Toc173733577)

[3.1.1 Learning Rate (Tốc độ học) 20](#_Toc173733578)

[3.1.2 Batch Size 21](#_Toc173733579)

[3.1.3 Epochs 21](#_Toc173733580)

[3.1.4 Optimizer (Bộ tối ưu hóa): 22](#_Toc173733581)

[3.1.5 Momentum 22](#_Toc173733582)

[3.1.6 Dropout Rate 22](#_Toc173733583)

[3.1.7 Hiệu quả của sự thay đổi các siêu tham số 23](#_Toc173733584)

[3.2 Tăng cường dữ liệu 23](#_Toc173733585)

[3.2.1 HSV (Hue, Saturation, Value) Augmentation (hsv\_h, hsv\_s, hsv\_v) 23](#_Toc173733586)

[3.2.2 Degrees (Rotation) 24](#_Toc173733587)

[3.2.3 Translate 25](#_Toc173733588)

[3.2.4 Scale 25](#_Toc173733589)

[3.2.5 Mosaic 26](#_Toc173733590)

[3.2.6 Mixup 26](#_Toc173733591)

[3.2.7 Flipud (Flip Up-Down) 27](#_Toc173733592)

[3.2.8 Fliplr (Flip Left-Right) 27](#_Toc173733593)

[3.2.9 Shear 28](#_Toc173733594)

[3.2.10 Perspective 28](#_Toc173733595)

[3.2.11 Ảnh hưởng của tăng cường dữ liệu 29](#_Toc173733596)

[3.3 Thay đổi kiến trúc YOLOv8 29](#_Toc173733597)

[3.4 TensorRT 33](#_Toc173733598)

[3.4.1 Khái niệm 33](#_Toc173733599)

[3.4.2 Ưu, nhược điểm 34](#_Toc173733600)

[Chương 4: Thực nghiệm 36](#_Toc173733601)

[4.1 Thiết lập thực nghiệm 36](#_Toc173733602)

[4.1.1 Chuẩn bị Dữ Liệu: 36](#_Toc173733603)

[4.1.2 Huấn luyện mô hình nguyên bản của YOLOv8 36](#_Toc173733604)

[4.1.3 Điều chỉnh siêu tham số 38](#_Toc173733605)

[4.1.4 Tăng cường dữ liệu 41](#_Toc173733606)

[4.1.5 Thay đổi kiến trúc YOLOv8 43](#_Toc173733607)

[4.1.6 TensorRT 50](#_Toc173733608)

[4.2 Kết quả 52](#_Toc173733609)

[Chương 5: Kết luận 54](#_Toc173733610)

[TÀI LIỆU THAM KHẢO 55](#_Toc173733611)

# Danh mục các từ viết tắt

|  |  |  |
| --- | --- | --- |
| **STT** | **Từ viết tắt** | **Mô tả** |
| 1 | API | Application Programming Interface – phương thức trung gian kết nối |

# Danh mục các hình vẽ

Hình 1: Ảnh hưởng của learning rate [ii](#_Toc173738452)0

[Hình 2: Tìm kiếm learning rate thích hợp 21](#_Toc173738453)

[Hình 3: SGD với momentum 22](#_Toc173738454)

[Hình 4:Box plot và hình ảnh minh họa của của các giá trị hue, sarturation và value 24](#_Toc173738455)

Hình 5: Kĩ thuật quay ảnh  [25](#_Toc173738456)

[Hình 6: Kĩ thuật translate 25](#_Toc173738457)

[Hình 7: Kĩ thuật scale 26](#_Toc173738458)

[Hình 8: Kĩ thuật mosiac 26](#_Toc173738459)

[Hình 9: Kĩ thuật mixup 27](#_Toc173738460)

[Hình 10: Kĩ thuật flipup 27](#_Toc173738461)

[Hình 11: Kĩ thuật fliplr 28](#_Toc173738462)

[Hình 12: Kĩ thuật shear 28](#_Toc173738463)

[Hình 13: Kĩ thuật perspective 29](#_Toc173738464)

# 

# Chương 1: Mở đầu

## 1.1 Lý do chọn đề tài

- Lý do chọn đề tài này là do sự phát triển vượt bậc của các mô hình YOLO (You Only Look Once) trong lĩnh vực nhận diện và phát hiện vật thể. YOLOv8 là phiên bản với nhiều cải tiến đáng kể so với các phiên bản cũ hơn có tốc độ xữ lý nhanh hơn vả độ chính xác cao, việc nghiên cứu và cải thiện mô hình này có thể đóng góp lớn cho các ứng dụng thực tế.  
- Sự phổ biến của YOLO trong các lĩnh vực như an ninh, xe tự hành, và công nghiệp đã chứng minh tiềm năng của mô hình này. Việc cải thiện hiệu suất của YOLOv8 sẽ giúp nâng cao độ chính xác và tốc độ xử lý, từ đó ứng dụng rộng rãi hơn.

## 1.2 Nội dung nghiên cứu

- Nghiên cứu và hiểu rõ về mô hình YOLOv8.  
- Thực hiện các thí nghiệm và cải thiện mô hình này.  
- Xây dựng một demo minh họa về hiệu quả của các cải tiến đã thực hiện.

# 

# Chương 2: Tổng quan

## 2.1 Lịch sử YOLO

- YOLO (You Only Look Once) là một dòng mô hình mạng nơ-ron nhân tạo nổi tiếng trong lĩnh vực phát hiện vật thể, được phát triển bởi Joseph Redmon và các đồng sự. Dưới đây là lịch sử phát triển của các phiên bản YOLO từ đầu tới phiên bn YOLOv8:

### 2.1.1 YOLO (You Only Look Once) - 2015

**- Giới thiệu**: Joseph Redmon và cộng sự phát triển.

**- Đặc điểm nổi bật**:

+ Phương pháp thống nhất: sử dụng một mạng CNN duy nhất để dự đoán bounding boxes và class probabilities.

+ Xử lý tốc độ cao: 45 FPS.

### 2.1.2 YOLOv2 (YOLO9000) - 2016

**- Giới thiệu**: Cải thiện độ chính xác và tốc độ.

**- Đặc điểm nổi bật**:

+ Chuẩn hóa theo batch.

+ Anchor boxes.

+ Bộ phân loại có độ phân giải cao.

+ YOLO9000: Kết hợp phát hiện đối tượng trên cả COCO và ImageNet.

### 2.1.3 YOLOv3 - 2018

**- Giới thiệu**: Tăng cường khả năng phát hiện đối tượng nhỏ.

**- Đặc điểm nổi bật**:

+ Dự đoán đa kích thước.

+ Darknet-53.

+ Dự đoán bounding box bằng logistic regression.

### 2.1.4 YOLOv4 - 2020

**- Giới thiệu**: Cải tiến đáng kể về tốc độ và độ chính xác.

**- Đặc điểm nổi bật**:

+ CSPDarknet53.

+ Bag of freebies và bag of specials.

+ Sử dụng nhiều kỹ thuật tối ưu hóa và tăng cường dữ liệu.

### 2.1.5 Scaled-YOLOv4 - 2020

**- Giới thiệu**: Phiên bản mở rộng của YOLOv4.

**- Đặc điểm nổi bật**:

+ Kiến trúc đa mô hình (P5, P6, P7).

+ Hợp nhất đường dẫn đặc trưng (PAN).

+ Tăng cường Mosaic, đào tạo tự đối kháng.

### 2.1.6 YOLOv5 - 2020

**- Giới thiệu**: Tối ưu hóa cho sử dụng trong thực tế.

**- Đặc điểm nổi bật**:

+ Triển khai trong PyTorch.

+ Cải thiện tăng cường dữ liệu.

+ Nhiều kích thước mô hình (s, m, l, x).

### 2.1.7 PP-YOLO - 2020

**- Giới thiệu**: Được phát triển bởi Baidu.

**- Đặc điểm nổi bật**:

+ ResNet50-vd.

+ Tối ưu hóa phần cứng.

+ Tăng cường dữ liệu MixUp, CutMix.

### 2.1.8 PP-YOLOv2 - 2021

**- Giới thiệu**: Phiên bản cải tiến của PP-YOLO.

**- Đặc điểm nổi bật**:

+ ResNet50-vd và CSPDarknet.

+ IoU Loss, Grid Sensitive, Matrix NMS.

+ Hiệu suất vượt trội.

### 2.1.9 YOLOR - 2021

**- Giới thiệu**: Kết hợp phát hiện đối tượng và phân loại.

**- Đặc điểm nổi bật**:

+ Học đại diện hợp nhất.

+ Phân loại và phát hiện đối tượng.

+ Hiệu suất cao.

### 2.1.10 YOLOX - 2021

**- Giới thiệu**: Mô hình YOLO tiên tiến.

**- Đặc điểm nổi bật**:

+ Không có neo.

+ Gán nhãn động.

+ Đa quy mô (s, m, l, x).

### 2.1.11 YOLOv6 - 2022

**- Giới thiệu**: Tập trung vào ứng dụng công nghiệp.

**- Đặc điểm nổi bật**:

+ Tăng cường dữ liệu nâng cao.

+ Cải thiện kiến ​​trúc mạng.

+ Tập trung vào hiệu quả triển khai.

### 2.1.12 YOLOv7 - 2022

**- Giới thiệu**: Được cho là nhanh nhất và chính xác nhất trong dòng YOLO.

**- Đặc điểm nổi bật**:

+ E-ELAN.

+ Hàm mất mát nâng cao.

+ Độ chính xác cao và hiệu suất thời gian thực.

### 2.1.13 PP-YOLOE - 2022

**- Giới thiệu**: Phiên bản mới của PP-YOLOv2.

**- Đặc điểm nổi bật**:

+ Kiến trúc mở rộng.

+ Mạng tổng hợp đường dẫn.

+ Hợp nhất tính năng không gian thích ứng.

+ Hiệu suất cao.

### 2.1.14 YOLOv8 - 2023

**- Giới thiệu**: Phiên bản nổi bật với khả năng kết hợp giữa tốc độ và độ chính xác.

**- Đặc điểm nổi bật**:

+ Cải tiến trong thiết kế mạng.

+ Tăng cường dữ liệu hiện đại.

+ Tập trung vào triển khai thực tế.

## 2.2 Kiến trúc YOLOv8

### 2.2.1 Tổng quan

- YOLOv8 là một phiên bản cải tiến đáng kể của dòng mô hình YOLO, với nhiều thay đổi và bổ sung quan trọng trong kiến trúc để nâng cao hiệu suất và độ chính xác. Kiến trúc YOLOv8 có thể được chia thành các thành phần chính sau:  
+ Backbone: Là phần đầu vào của mạng, YOLOv8 sử dụng một phiên bản cải tiến của Darknet hoặc một mạng backbone khác như CSPNet để trích xuất các đặc trưng từ ảnh đầu vào.  
+ Neck: Gồm các lớp convolution và các cơ chế attention để tiếp tục xử lý và kết hợp các đặc trưng được trích xuất từ Backbone.  
+ Head: Phần cuối cùng của mạng, chứa các lớp phát hiện để dự đoán các hộp giới hạn và xác suất lớp cho mỗi vật thể trong ảnh.  
Các công thức toán học quan trọng:  
+ Hàm mất mát: YOLOv8 sử dụng một hàm mất mát tổng hợp bao gồm các thành phần cho lỗi vị trí (location loss), lỗi kích thước (size loss), lỗi phân loại (classification loss), và lỗi IOU (Intersection over Union loss).  
+ Hàm kích hoạt: Sử dụng các hàm kích hoạt như Leaky ReLU, Mish, và các biến thể khác để cải thiện khả năng học của mạng.  
+ Cơ chế attention: Sử dụng các cơ chế như Squeeze-and-Excitation (SE) và Convolutional Block Attention Module (CBAM) để tăng cường khả năng chú ý vào các vùng quan trọng trong ảnh.

A diagram of a computer program

Description automatically generated with medium confidence- Kiến trúc của mô hình YOLOv8 với 2 khối SPFF và C2f được truyền cảm hứng từ kiến trúc khối DarknetBottleneck:

Backbone

Neck

Head

+ DarknetBottleneck:

A diagram of a bottleneck

Description automatically generated

+ Conv:

A diagram of a structure

Description automatically generated

+ C2f:

A diagram of a diagram

Description automatically generated

+ SPFF:

A diagram of a computer program

Description automatically generated with medium confidence

+ Detect:

A diagram of a computer network

Description automatically generated

### 2.2.2 Cải tiến quan trọng

- YOLOv8 mang lại nhiều cải tiến quan trọng so với các phiên bản trước, bao gồm:  
+ Kiến trúc CSPNet: Sử dụng CSPNet để cải thiện hiệu suất và giảm chi phí tính toán bằng cách chia nhỏ các feature Các chỉ số đo độ chính xác và kết hợp chúng lại sau khi thực hiện các phép biến đổi.  
+ Các lớp convolution sâu hơn: Sử dụng các lớp convolution sâu hơn và phức tạp hơn để cải thiện khả năng trích xuất đặc trưng.  
+ Attention mechanism: Áp dụng các cơ chế attention để tập trung vào các vùng quan trọng của ảnh, giúp cải thiện độ chính xác trong việc phát hiện các vật thể nhỏ và khó nhận diện.  
+ Hyperparameter optimization: Tối ưu hóa các siêu tham số thông qua các kỹ thuật như grid search và Bayesian optimization để tìm ra các cấu hình tốt nhất cho mạng.

### 2.2.3 Ưu, nhược điểm

Ưu điểm:  
- Tốc độ cao: YOLOv8 vẫn giữ được tốc độ xử lý nhanh, giúp nó phù hợp với các ứng dụng thời gian thực.  
- Độ chính xác cao: Nhờ vào các cải tiến về kiến trúc và thuật toán, YOLOv8 đạt được độ chính xác cao hơn trong việc phát hiện vật thể.  
- Khả năng nhận diện vật thể nhỏ: Với các cơ chế attention và các lớp convolution sâu, YOLOv8 cải thiện đáng kể khả năng nhận diện các vật thể nhỏ và khó phát hiện.  
Nhược điểm:  
- Chi phí tính toán cao: Các cải tiến về kiến trúc và sử dụng các lớp convolution sâu hơn dẫn đến chi phí tính toán cao hơn, yêu cầu phần cứng mạnh mẽ hơn.  
- Độ phức tạp: Kiến trúc phức tạp hơn đòi hỏi nhiều thời gian và công sức trong việc huấn luyện và tối ưu hóa mô hình.  
- Cần nhiều dữ liệu: Để đạt được hiệu suất tốt nhất, YOLOv8 yêu cầu một lượng lớn dữ liệu huấn luyện chất lượng cao.

# 

# Chương 3: Phương pháp đề xuất

## 3.1 Điều chỉnh siêu tham số

- Siêu tham số (hyperparameter) là những tham số được thiết lập trước khi quá trình huấn luyện bắt đầu và không được cập nhật trong quá trình huấn luyện của mô hình. Chúng có tác động lớn đến hiệu suất của mô hình và quá trình học của mạng nơ-ron nhân tạo. Dưới đây là một số siêu tham số phổ biến trong mô hình học sâu:

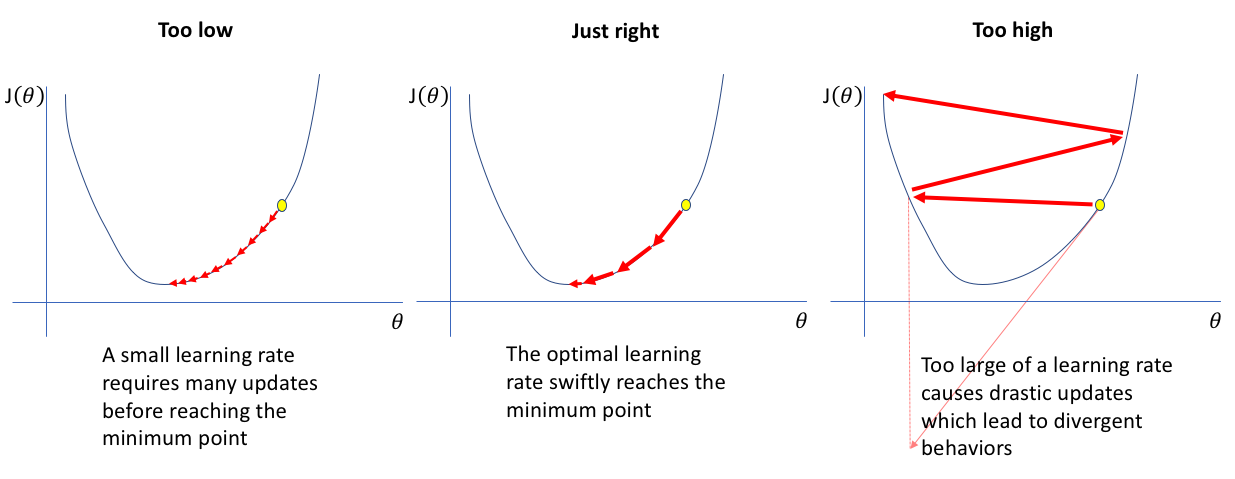
### 3.1.1 Learning Rate (Tốc độ học)

**- Định nghĩa**: Là một hệ số xác định mức độ điều chỉnh trọng số của mạng nơ-ron đối với mỗi bước cập nhật dựa trên độ dốc của hàm mất mát.

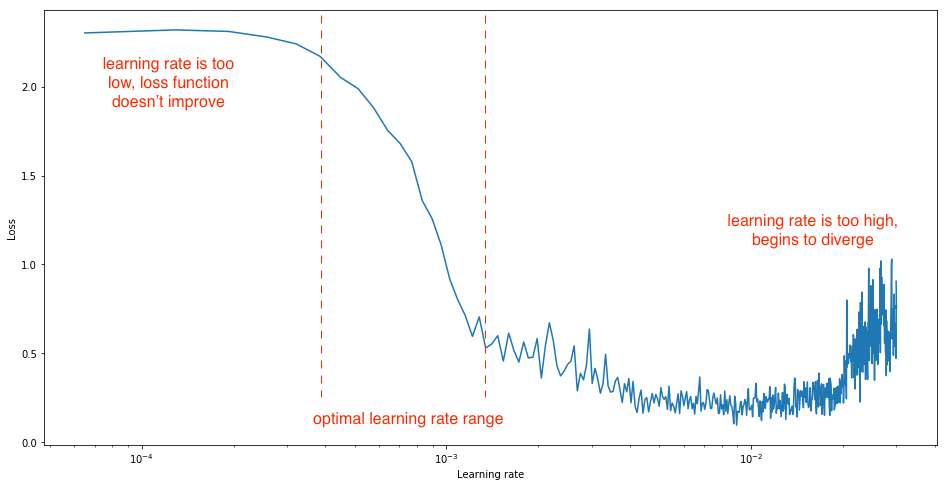
**- Ý nghĩa thay đổi**:

**+ Learning rate quá cao**: Mô hình có thể dao động xung quanh giá trị tối ưu và không hội tụ.

**+ Learning rate quá thấp**: Quá trình huấn luyện diễn ra chậm và mô hình có thể bị kẹt tại các cực trị cục bộ.



*Hình 1:* [1] *Ảnh hưởng của learning rate.*



*Hình 2:* [2] *Tìm kiếm learning rate thích hợp.*

### 3.1.2 Batch Size

**- Định nghĩa**: Số lượng mẫu được sử dụng để tính toán mỗi bước cập nhật của trọng số.

**- Ý nghĩa thay đổi**:

**+ Batch size lớn**: Giúp tính toán gradient chính xác hơn nhưng yêu cầu bộ nhớ lớn và thời gian huấn luyện lâu hơn.

**+ Batch size nhỏ**: Giảm nhu cầu bộ nhớ và tăng tốc độ huấn luyện, nhưng gradient tính toán ít chính xác hơn.

### 3.1.3 Epochs

**- Định nghĩa**: Số lần toàn bộ tập dữ liệu huấn luyện được đưa qua mô hình.

**- Ý nghĩa thay đổi**:

**+ Số lượng epochs quá ít**: Mô hình chưa học đủ, dẫn đến underfitting.

**+ Số lượng epochs quá nhiều**: Mô hình có thể học quá mức, dẫn đến overfitting.

### 3.1.4 Optimizer (Bộ tối ưu hóa):

**- Định nghĩa**: Thuật toán được sử dụng để điều chỉnh trọng số của mô hình nhằm tối thiểu hóa hàm mất mát.

**- Ý nghĩa thay đổi**: Các bộ tối ưu hóa khác nhau (như SGD, Adam, RMSprop) có các chiến lược cập nhật trọng số khác nhau, ảnh hưởng đến tốc độ và khả năng hội tụ của mô hình.

### 3.1.5 Momentum

**- Định nghĩa**: Tham số này giúp tăng tốc độ hội tụ của mô hình bằng cách thêm một phần của gradient trước đó vào gradient hiện tại.

**- Ý nghĩa thay đổi**:

**+ Momentum quá cao**: Có thể dẫn đến dao động quanh giá trị tối ưu.

**+ Momentum quá thấp**: Có thể làm chậm quá trình hội tụ.  
A diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of a diagram of

Description automatically generated

*Hình 3:* [3] *SGD với momentum.*

### 3.1.6 Dropout Rate

**- Định nghĩa**: Tỷ lệ bỏ qua ngẫu nhiên các đơn vị (neurons) trong quá trình huấn luyện để ngăn ngừa overfitting.

**- Ý nghĩa thay đổi**:

**+ Dropout rate quá cao**: Mô hình khó học được các đặc trưng quan trọng.

**+ Dropout rate quá thấp**: Không đủ hiệu quả trong việc ngăn chặn overfitting.

### 3.1.7 Hiệu quả của sự thay đổi các siêu tham số

**- Hiệu suất mô hình**: Điều chỉnh siêu tham số có thể cải thiện hoặc làm giảm hiệu suất của mô hình trên tập kiểm tra.

**- Tốc độ hội tụ**: Các siêu tham số như learning rate và batch size ảnh hưởng trực tiếp đến tốc độ hội tụ của mô hình.

**- Overfitting và Underfitting**: Việc thiết lập các siêu tham số hợp lý giúp ngăn ngừa overfitting và underfitting, từ đó cải thiện khả năng tổng quát hóa của mô hình.

**- Sử dụng tài nguyên**: Siêu tham số như batch size và number of epochs ảnh hưởng đến tài nguyên tính toán và thời gian huấn luyện.

- Việc chọn lựa và điều chỉnh siêu tham số là một quá trình quan trọng và đòi hỏi sự thử nghiệm, thường thông qua các phương pháp như grid search, random search, hoặc các thuật toán tối ưu hóa bayesian để tìm ra cấu hình tốt nhất cho mô hình.

## 3.2 Tăng cường dữ liệu

- Tăng cường dữ liệu (data augmentation) là một kỹ thuật được sử dụng để tạo ra các biến thể của dữ liệu huấn luyện hiện có nhằm cải thiện khả năng tổng quát hóa của mô hình học sâu. Mục tiêu của việc tăng cường dữ liệu là giúp mô hình học được nhiều đặc trưng hơn từ dữ liệu bằng cách làm phong phú thêm tập huấn luyện mà không cần thu thập thêm dữ liệu mới. Sau đây là 1 số tham số và ý nghĩa của chúng:

### 3.2.1 HSV (Hue, Saturation, Value) Augmentation (hsv\_h, hsv\_s, hsv\_v)

**- hsv\_h (Hue Shift)**: Điều chỉnh sắc độ của ảnh. Tham số này thay đổi màu sắc của ảnh mà không thay đổi độ sáng hoặc độ bão hòa.

**- hsv\_s (Saturation Shift)**: Điều chỉnh độ bão hòa của ảnh. Tham số này thay đổi độ rực rỡ của màu sắc trong ảnh.

**- hsv\_v (Value Shift)**: Điều chỉnh độ sáng của ảnh. Tham số này thay đổi độ sáng tổng thể của ảnh.

**- Ý nghĩa**: Những thay đổi này giúp mô hình học được cách nhận diện các đối tượng dưới các điều kiện ánh sáng và màu sắc khác nhau.

A diagram of different types of saturation

Description automatically generated

*Hình 4:* [4] *Box plot và hình ảnh minh họa của của các giá trị hue, sarturation và value.*

### 3.2.2 Degrees (Rotation)

**- Định nghĩa**: Xoay ảnh theo một góc ngẫu nhiên trong khoảng giá trị cho trước.

**- Ý nghĩa**: Giúp mô hình học được cách nhận diện đối tượng từ nhiều góc nhìn khác nhau.

A collage of a cat

Description automatically generated

*Hình 5: Kĩ thuật quay ảnh.*

### 3.2.3 Translate

**- Định nghĩa**: Dịch chuyển ảnh theo chiều ngang hoặc chiều dọc với một khoảng ngẫu nhiên.

**- Ý nghĩa**: Giúp mô hình nhận diện đối tượng ngay cả khi chúng không nằm ở vị trí trung tâm của ảnh.

A cat lying on a surface

Description automatically generated

*Hình 6: Kĩ thuật translate.*

### 3.2.4 Scale

**- Định nghĩa**: Thay đổi kích thước của ảnh bằng cách phóng to hoặc thu nhỏ theo một tỷ lệ ngẫu nhiên.

**- Ý nghĩa**: Giúp mô hình học cách nhận diện đối tượng ở các kích thước khác nhau.

A cat lying down on a surface

Description automatically generated

*Hình 7: Kĩ thuật scale.*

### 3.2.5 Mosaic

**- Định nghĩa**: Kết hợp nhiều ảnh lại với nhau để tạo thành một ảnh duy nhất.

**- Ý nghĩa**: Tăng tính đa dạng của tập dữ liệu bằng cách tạo ra các bối cảnh phức tạp hơn, giúp mô hình nhận diện đối tượng trong nhiều tình huống khác nhau.



*Hình 8: Kĩ thuật mosaic.*

### 3.2.6 Mixup

**- Định nghĩa**: Trộn lẫn hai ảnh với nhau bằng cách lấy trung bình có trọng số của các giá trị pixel.

**- Ý nghĩa**: Giúp mô hình tổng quát hóa tốt hơn bằng cách tạo ra các mẫu mới từ các mẫu hiện có.

A cat sitting on a white blanket

Description automatically generated

*Hình 9: Kĩ thuật MixUp.*

### 3.2.7 Flipud (Flip Up-Down)

**- Định nghĩa**: Lật ngược ảnh theo chiều dọc.

**- Ý nghĩa**: Tăng cường tính đa dạng của tập dữ liệu bằng cách thêm các biến thể lật ngược của ảnh.

A collage of a black cat lying on a carpet

Description automatically generated

*Hình 10: Kĩ thuật flipud.*

### 3.2.8 Fliplr (Flip Left-Right)

**- Định nghĩa**: Lật ngược ảnh theo chiều ngang.

**- Ý nghĩa**: Tăng cường tính đa dạng của tập dữ liệu bằng cách thêm các biến thể lật ngang của ảnh.

A collage of a black cat

Description automatically generated

*Hình 11: Kĩ thuật fliplr.*

### 3.2.9 Shear

**- Định nghĩa**: Biến dạng ảnh theo một hướng cố định, tạo ra hiệu ứng kéo dài hoặc nén.

**- Ý nghĩa**: Giúp mô hình học được cách nhận diện đối tượng ngay cả khi chúng bị biến dạng.

A collage of a cat

Description automatically generated

*Hình 12: Kĩ thuật shear.*

### 3.2.10 Perspective

**- Định nghĩa**: Thay đổi phối cảnh của ảnh, tạo ra hiệu ứng như nhìn từ một góc khác.

**- Ý nghĩa**: Giúp mô hình nhận diện đối tượng dưới các góc nhìn phối cảnh khác nhau.

A collage of a cat lying down

Description automatically generated

*Hình 13: Kĩ thuật perspective.*

### 3.2.11 Ảnh hưởng của tăng cường dữ liệu

**- Cải thiện khả năng tổng quát hóa**: Tăng cường dữ liệu giúp mô hình học được nhiều đặc trưng hơn từ dữ liệu, từ đó cải thiện khả năng tổng quát hóa của mô hình đối với dữ liệu chưa từng thấy trước đó.

**- Giảm overfitting**: Bằng cách tạo ra nhiều biến thể của dữ liệu huấn luyện, tăng cường dữ liệu giúp ngăn ngừa mô hình học quá mức vào các đặc trưng cụ thể của tập huấn luyện.

**- Tăng kích thước tập dữ liệu**: Tăng cường dữ liệu giúp tăng kích thước tập dữ liệu huấn luyện mà không cần thu thập thêm dữ liệu mới, từ đó cải thiện hiệu suất huấn luyện mô hình.

**- Đa dạng hóa dữ liệu**: Các kỹ thuật tăng cường dữ liệu giúp mô hình học được cách nhận diện đối tượng trong nhiều tình huống và điều kiện khác nhau, từ đó cải thiện hiệu suất tổng thể của mô hình.

- Việc sử dụng các kỹ thuật tăng cường dữ liệu một cách hợp lý có thể mang lại hiệu suất tốt hơn cho mô hình học sâu, đặc biệt là khi dữ liệu huấn luyện bị hạn chế.

## 3.3 Thay đổi kiến trúc YOLOv8

- Kiến trúc YOLOv8 sử dụng 3 đầu detect cho từng tỉ lệ kích cỡ vật thể cần nhận diện, cùng với đó là những khối tương ứng hỗ trợ. Có thể chia ra làm 3 kiến trúc nhỏ hơn là small, medium, large:

+ cho kích thước nhỏ (small).

+ cho kích thước trung bình (medium).

+ cho kích thước lớn (large).

- Với những công việc có sự chính xác và lặp lại cao như sản xuất và chế tạo trong công nghiệp sẽ tạo ra hình ảnh có góc chụp không đổi và tỉ lệ kích cỡ tương đối giống nhau.

- Từ hiện thực nêu trên, em đề xuất rằng với từng công việc khác nhau tạo ra những bức ảnh có tỉ lệ tương đồng, chúng ta sẽ loại bỏ những đầu detect những khối hỗ trợ tương ứng không cần thiết với tỉ lệ vật thể cần nhận diện để giảm số lượng tham số cần huấn luyện của mô hình từ đó tăng tốc độ detect của model:

+ Small:

A computer screen shot of a diagram

Description automatically generated

+ Medium:

A diagram of a structure

Description automatically generated

+ Large:

A screenshot of a computer

Description automatically generated

- Tùy thuộc vào tỉ lệ vật cần detect so với khung hình, chúng ta có thể sử dụng riêng hoặc kết hợp các kiến trúc trên để huấn luyên mô hình.

- Tuy nhiên kĩ thuật này cũng tồn tại 1 vài nhược điểm:

+ Sự đánh đổi độ chính xác với tốc độ phụ thuộc rất nhiều vào sự nhất quán của tỉ lệ vật cần detect so với khung hình trong tập dữ liệu, bỏ đi các kiến trúc dùng để nhận diện các vật thể tỉ lệ khác đồng nghĩa với việc khó hoặc gần như không thể nhận diện các vật thể tỉ lệ đó.

+ Chưa có khoảng cụ

## 3.4 TensorRT

- Có rất nhiều thư viện bổ trợ cho mô hình YOLOv8 như TensorRT và OpenVINO, sử dụng các thuật toán tối ưu hóa như hiệu chỉnh các lớp và hiệu chỉnh độ chính xác như FP16, INT18 để cải thiện hiệu suất của các mô hình học sâu. Tuy nhiên trong báo cáo này chúng ta chỉ đề cập tượng trưng 1 thư viện và đó chính là TensorRT.

### 3.4.1 Khái niệm

- TensorRT là một thư viện và bộ công cụ được phát triển bởi NVIDIA. Nó được thiết kế để tối ưu hóa và triển khai các mô hình mạng nơ-ron trên GPU của NVIDIA, tăng tốc độ và hiệu quả của các hoạt động suy luận:

**- Các Tính Năng Chính của TensorRT:**

**+ Tối ưu mô hình**: TensorRT thực hiện nhiều tối ưu hóa trên các mô hình mạng nơ-ron đã huấn luyện, như hợp nhất các lớp, hiệu chỉnh độ chính xác (FP32, FP16, INT8), và tự động điều chỉnh nhân, để tối đa hóa hiệu suất trên phần cứng của NVIDIA.

**+ Suy luận hiệu suất cao**: Bằng cách tối ưu hóa mô hình, TensorRT tăng tốc suy luận, giảm cả độ trễ và tài nguyên tính toán cần thiết. Điều này rất quan trọng cho các ứng dụng thời gian thực như lái xe tự động, robot và phân tích video.

**+ Hiệu chỉnh độ chính xác**: TensorRT cho phép các mô hình chạy ở các chế độ độ chính xác giảm (FP16 và INT8), giúp cải thiện hiệu suất đáng kể và giảm sử dụng bộ nhớ mà không làm giảm độ chính xác quá nhiều.

**+ Quản lý bộ nhớ động Tensor**: TensorRT quản lý bộ nhớ tensor một cách động, giúp sử dụng bộ nhớ GPU hiệu quả, cho phép triển khai các mô hình phức tạp hơn hoặc chạy nhiều mô hình đồng thời.

**+ Tích hợp với hệ sinh thái NVIDIA**: TensorRT được thiết kế để hoạt động liền mạch với các khung và thư viện học sâu của NVIDIA, bao gồm CUDA, cuDNN và DeepStream SDK, tạo điều kiện thuận lợi cho quy trình triển khai từ đầu đến cuối.

**+ Hỗ trợ các khung chính**: TensorRT hỗ trợ các mô hình từ các khung học sâu phổ biến như TensorFlow, PyTorch, ONNX và Caffe. Điều này giúp dễ dàng triển khai các mô hình được huấn luyện trên các khung này lên GPU của NVIDIA.

### 3.4.2 Ưu, nhược điểm

- **Ưu điểm:**

**+ Tối ưu hiệu suất**: TensorRT cung cấp các cải tiến về tốc độ đáng kể cho suy luận thông qua tối ưu hóa các lớp mạng, hợp nhất lớp, và lựa chọn độ chính xác tốt nhất (FP32, FP16, INT8) cho hiệu suất.

**+ Giảm độ trễ**: TensorRT giảm độ trễ của các tác vụ suy luận, rất quan trọng cho các ứng dụng thời gian thực như nhận diện đối tượng trong xe tự lái hoặc giám sát video.

**+ Sử dụng tài nguyên hiệu quả**: TensorRT tối ưu hóa việc sử dụng bộ nhớ GPU và tài nguyên tính toán, cho phép triển khai nhiều mô hình hơn hoặc xử lý nhiều dữ liệu hơn trong cùng một hạn chế phần cứng.

**+ Hiệu chỉnh độ chính xác**: TensorRT hỗ trợ độ chính xác INT8, giúp giảm đáng kể bộ nhớ và tăng thông lượng mà không làm giảm độ chính xác nhiều, đặc biệt hữu ích cho triển khai trên các thiết bị biên.

**+ Tích hợp hệ thống**: TensorRT tích hợp tốt với phần cứng và phần mềm của NVIDIA, bao gồm CUDA, cuDNN và DeepStream SDK, tạo điều kiện thuận lợi cho các quy trình triển khai từ đầu đến cuối.

**+ Bộ nhớ Tensor động**: TensorRT phân bổ và giải phóng bộ nhớ tensor động, giúp quản lý bộ nhớ GPU hiệu quả hơn và cho phép xử lý các kích thước đầu vào khác nhau.

**- Nhược Điểm:**

**+ Độ phức tạp**: Quá trình chuyển đổi mô hình sang TensorRT có thể phức tạp và đòi hỏi hiểu biết tốt về cả kiến trúc mô hình và TensorRT. Việc khắc phục sự cố phát sinh trong quá trình chuyển đổi có thể khó khăn.

**+ Vấn đề tương thích**: TensorRT có thể gặp vấn đề tương thích với một số lớp hoặc thao tác trong mô hình gốc, đòi hỏi các triển khai tùy chỉnh hoặc điều chỉnh mô hình.

**+ Hỗ trợ khung hạn chế**: Mặc dù TensorRT hỗ trợ nhiều khung phổ biến như TensorFlow và PyTorch, nhưng hỗ trợ cho các khung mới hoặc ít phổ biến hơn có thể bị chậm hoặc cần nỗ lực bổ sung để tích hợp.

**+ Đánh đổi độ chính xác**: Mặc dù độ chính xác thấp hơn (như INT8) mang lại lợi ích về hiệu suất, nhưng có thể dẫn đến giảm độ chính xác của mô hình. Cần hiệu chỉnh và thử nghiệm kỹ lưỡng để đảm bảo hiệu suất chấp nhận được.

**+ Yêu cầu tài nguyên**: Tối ưu hóa và suy luận sử dụng TensorRT có thể yêu cầu nhiều tài nguyên, đòi hỏi GPU mạnh mẽ, điều này có thể không khả thi cho tất cả các môi trường triển khai.

**+ Tối ưu hóa tĩnh**: Các tối ưu hóa của TensorRT được thực hiện tại thời điểm tạo engine. Điều này có nghĩa là các thay đổi động của mô hình trong thời gian chạy (ví dụ: cập nhật mô hình) yêu cầu chuyển đổi và tối ưu hóa lại.

**+ Khóa Vendor**: TensorRT là sản phẩm của NVIDIA, điều này ràng buộc việc triển khai vào phần cứng của NVIDIA. Đây có thể là một hạn chế nếu cần triển khai trên phần cứng của các nhà cung cấp khác.

# 

# Chương 4: Thực nghiệm

## 4.1 Thiết lập thực nghiệm

- Thực hiện các thực nghiệm trên các tập dữ liệu là Fruit Freshness Detection Dataset với 12410 ảnh thuộc 4 lớp gồm có 'Fresh Apple', 'Fresh Banana', 'Rotten Apple', 'Rotten Banana' được thu thập bởi Brac University [5] và fm với 801 ảnh thuộc 3 lớp gồm có 'Mask', 'No Mask', 'Bad Mask'. Các bước thực hiện chủ yếu thông qua API của Ultralystics [6] và triển khai trên Google Colab bao gồm:

4.1.1 Chuẩn bị Dữ Liệu:

- Đổi tên tập dữ liệu Fruit Freshness Detection Dataset thành fruit\_fr để tiện việc sử dụng sau này và sau đó chia cả 2 tập dữ liệu thành các tập huấn luyện, xác thực và kiểm tra.

- Các dữ liệu được chuẩn bị với các đặc trưng sau:

+ Huấn luyện: 70% mẫu

+ Xác thực: 20% mẫu

+ Kiểm tra: 10% mẫu

### 4.1.2 Huấn luyện mô hình nguyên bản của YOLOv8

#### 4.1.2.1 fm

- Sử dụng tập dữ liệu fm để huấn luyện mô hình nguyên bản của YOLOv8, đặt tên là fm\_original:



- Kết quả huấn luyện:

A screenshot of a computer program

Description automatically generated

- Các chỉ số đo độ chính xác:

A screenshot of a computer

Description automatically generated

- Tốc độ:

A screenshot of a computer

Description automatically generated

#### 4.1.2.2 fruit\_fr

- Sử dụng tập dữ liệu fruit\_fr để huấn luyện mô hình nguyên bản của YOLOv8, đặt tên là fruit\_original:

- Kết quả huấn luyện:

A screenshot of a computer

Description automatically generated

- Các chỉ số đo độ chính xác:

A screenshot of a computer

Description automatically generated

- Tốc độ:

A screenshot of a computer

Description automatically generated

### 4.1.3 Điều chỉnh siêu tham số

#### 4.1.3.1 fm

- Điều chỉnh các siêu tham số của YOLOv8 để huấn luyện mô hình fm\_v1:





- Kết quả huấn luyện:

A white background with numbers and text

Description automatically generated with medium confidence

- Các chỉ số đo độ chính xác:

A screenshot of a computer

Description automatically generated

- Tốc độ:

A screenshot of a computer

Description automatically generated

#### 4.1.3.2 fruit\_fr

- Điều chỉnh các siêu tham số của YOLOv8 để huấn luyện mô hình fruit\_v1:





- Kết quả huấn luyện:

A screenshot of a computer

Description automatically generated

- Các chỉ số đo độ chính xác:

A screen shot of a computer

Description automatically generated

- Tốc độ:

A screenshot of a computer

Description automatically generated

### 4.1.4 Tăng cường dữ liệu

#### 4.1.4.1 fm

- Áp dụng các kĩ thuật tăng cường dữ liệu để huấn luyện mô hình fm\_v2:





- Kết quả huấn luyện:

A screenshot of a computer

Description automatically generated

- Các chỉ số đo độ chính xác:

A screenshot of a computer

Description automatically generated

- Tốc độ:

A screenshot of a computer

Description automatically generated

#### 4.1.4.2 fruit\_fr

- Áp dụng các kĩ thuật tăng cường dữ liệu để huấn luyện mô hình fruit\_v3:





- Kết quả huấn luyện:

A screenshot of a computer screen

Description automatically generated

- Các chỉ số đo độ chính xác:

A screenshot of a computer

Description automatically generated

- Tốc độ:

A screenshot of a computer

Description automatically generated

### 4.1.5 Thay đổi kiến trúc YOLOv8

#### 4.1.5.1 fm

- Plot biểu đồ histogram về tỉ lệ hình ảnh:

A graph of a line graph

Description automatically generated

A graph of a person with a blue line

Description automatically generated

A graph with blue lines

Description automatically generated

- Tạo file YOLOv8l-s\_m.yaml:

A screenshot of a computer

Description automatically generated

- Huấn luyện mô hình fm\_v3:



- Kết quả huấn luyện:

A screenshot of a computer code

Description automatically generated

- Các chỉ số đo độ chính xác:

A screenshot of a computer

Description automatically generated

- Tốc độ:

A screenshot of a computer

Description automatically generated

#### 4.1.5.2 fruit\_fr

- Plot biểu đồ histogram về tỉ lệ hình ảnh:

A graph of a line graph

Description automatically generated

A graph of a person in a blue shirt

Description automatically generated with medium confidence

A graph of a line graph

Description automatically generated

- Tạo file YOLOv8l-m.yaml:

A screenshot of a computer program

Description automatically generated

- Huấn luyện mô hình fruit\_v2:



- Kết quả huấn luyện:

A close-up of a screen

Description automatically generated

- Các chỉ số đo độ chính xác:

A screenshot of a computer

Description automatically generated

- Tốc độ:

A screenshot of a computer

Description automatically generated

### 4.1.6 TensorRT

#### 4.1.6.1 fm

- Chuyển đổi fm\_original sang TensorRT:

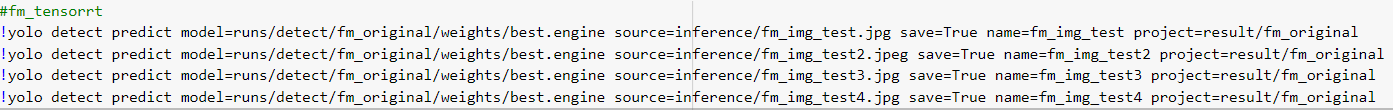


- Các chỉ số đo độ chính xác:

A screenshot of a computer

Description automatically generated

- Tốc độ:



A screenshot of a computer

Description automatically generated

#### 4.1.6.2 fruit\_fr

- Chuyển đổi fruit\_original sang TensorRT:

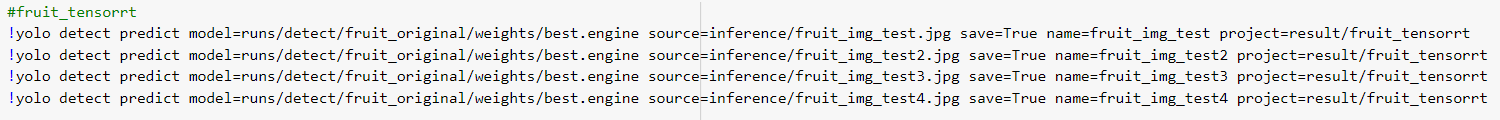


- Các chỉ số đo độ chính xác:

A screenshot of a computer

Description automatically generated

- Tốc độ:



A screenshot of a computer

Description automatically generated

## 4.2 Kết quả

- Kết quả các phiên bản của mô hình đào tạo từ tập dữ liệu fm:

+ Các chỉ số đo độ chính xác:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Precision | Recall | mAP50 | mAP50-95 |
| fm\_original | 0.919 | 0.825 | 0.911 | 0.633 |
| fm\_v1 | 0.881 (-0.038) | 0.883 (+0.058) | 0.915 (+0.004) | 0.643 (+0.01) |
| fm\_v2 | 0.942 (+0.023) | 0.827 (+0.002) | 0.928 (+0.017) | 0.607 (-0.026) |
| fm\_v3 | 0.895 (-0.024) | 0.829 (+0.004) | 0.893 (-0.018) | 0.593 (-0.04) |
| fm\_tensorrt | 0.913 (-0.006) | 0.824 (-0.001) | 0.909 (-0.002) | 0.633 |

+ Tốc độ (trung bình từ 4 lần detect ảnh):

|  |  |
| --- | --- |
| Model | Inference |
| fm\_original | 87.4 |
| fm\_v1 | 94.2 (+6.8) |
| fm\_v2 | 129.8 (+42.4) |
| fm\_v3 | 67.25 (-20.05) |
| fm\_tensorrt | 17.4 (-70) |

- Kết quả các phiên bản của mô hình đào tạo từ tập dữ liệu fruit\_fr:

+ Các chỉ số đo độ chính xác:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Precision | Recall | mAP50 | mAP50-95 |
| fruit\_original | 0.928 | 0.842 | 0.891 | 0.633 |
| fruit\_v1 | 0.926 (-0.002) | 0.832 (-0.01) | 0.893 (+0.002) | 0.639 (+0.006) |
| fruit\_v2 | 0.921 (-0.007) | 0.827 (-0.015) | 0.89 (-0.001) | 0.489 (-0.144) |
| fruit\_v3 | 0.898 (-0.03) | 0.829 (-0.013) | 0.876 (-0.015) | 0.567 (-0.066) |
| fruit\_tensorrt | 0.911 (-0.017) | 0.829 (-0.013) | 0.872 (-0.019) | 0.639 (+0.066) |

+ Tốc độ (trung bình từ 4 lần detect ảnh):

|  |  |
| --- | --- |
| Model | Inference |
| fruit\_original | 99.98 |
| fruit\_v1 | 144.58 (+44.6) |
| fruit\_v2 | 62.225 (-37.755) |
| fruit\_v3 | 114.65 (+14.67) |
| fruit\_tensorrt | 17.84 (-82.14) |

# 

# Chương 5: Kết luận

- Từ kết quả thực nghiệm chúng ta có thể kết luận rằng:

+ Điều chỉnh siêu tham số làm tốt về giảm tài nguyên với việc dừng sớm tại fm\_v1 và tăng độ chính xác với fruit\_fr.

+ Tăng cường dữ liệu tuy là những phương pháp có thể tăng cường hiệu suất nhưng có nhược điểm là tốn thời gian để có thể tìm ra các giá trị phù hợp, minh chứng rằng tăng cường dữ liệu tăng độ chính xác tại fm\_v2 nhưng lại giảm tại fruit\_v3 do dữ liệu phần lớn là sử dụng các kĩ thuật tăng cường dữ liệu.

+ Thay đổi kiến trúc YOLOv8 tuy phụ thuộc vào sự nhất quán của dữ liệu nhưng tốc độ và việc giảm tài nguyên để huấn luyện do giảm số lượng param rất để đánh đổi với độ chính xác.

+ Chuyển đổi mô hình sang TensorRT mang lại tốc độ vượt trội so với thông thường nhưng phải đánh đổi với độ chính xác tuy nhiên giảm rất ít, có thể cân nhắc sử dụng.

# 

# TÀI LIỆU THAM KHẢO

[1] Jeremy Jordan, Setting the learning rate of your neural network. (1/3/2018). Ngày truy cập : 5/8/2024.[Hình ảnh trực tuyến] https://www.jeremyjordan.me/content/images/2018/02/Screen-Shot-2018-02-24-at-11.47.09-AM.png

[2] Jeremy Jordan, Learning rate finder - Setting the learning rate of your neural network. (1/3/2018). Ngày truy cập: 5/8/2024.[Hình ảnh trực tuyến] https://www.jeremyjordan.me/content/images/2018/02/lr\_finder.png

[3] Cathrine Jeeva, SGD with momentum - Optimizers in Deep Learning. (11/1/2024). Ngày truy cập: 5/8/2024. [Hình ảnh trực tuyến] https://www.scaler.com/topics/images/sgd-with-momentum.webp

[4] Zhangxi Ye, Spectral augmentation, a HSV boxplot b spectral transformation - Extraction of Olive Crown Based on UAV Visible Images and the U2-Net Deep Learning Model. (21/3/2022). Ngày truy cập: 5/8/2024.[Hình ảnh trực tuyến] https://www.researchgate.net/publication/359401395/figure/fig5/AS:11431281212511218@1702672691169/Spectral-augmentation-a-HSV-boxplot-b-spectral-transformation.tif

[5] Brac University, Fruit Freshness Detection Dataset. (9/2022). Ngày truy cập: 5/8/2024. [Tập dữ liệu] https://universe.roboflow.com/brac-university-v9w2y/fruit-freshness-detection-08shj

[6] Glenn Jocher - CEO của Ultralystics, Ultralystics, (2014). Ngày truy cập: 5/8/2024. [Thư viện] https://github.com/ultralytics/ultralytics