ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC VÀ KỸ THUẬT THÔNG TIN

NGUYỄN THỊ NGUYỆT
NGUYỄN THỊ PHƯƠNG THẢO

KHÓA LUẬN TỐT NGHIỆP

HỆ THỐNG GIÁM SÁT TRÁI CÂY VÀ RAU CỦ THỜI GIAN THỰC

Real-Time Fruit And Vegetable Monitoring System

CỬ NHÂN NGÀNH KHOA HỌC DỮ LIỆU

TP. HỒ CHÍ MINH, 2024

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KHOA HỌC VÀ KỸ THUẬT THÔNG TIN

NGUYỄN THỊ NGUYỆT – 20521689

NGUYỄN THỊ PHƯƠNG THẢO - 20521936

KHÓA LUẬN TỐT NGHIỆP

HỆ THỐNG GIÁM SÁT TRÁI CÂY VÀ RAU CỦ THỜI GIAN THỰC

Real-Time Fruit And Vegetable Monitoring System

CỬ NHÂN NGÀNH KHOA HỌC DỮ LIỆU

GIẢNG VIÊN HƯỚNG DẪN

TS. ĐỖ TRỌNG HỢP

TS. TRẦN VĂN THÀNH

TP. HỒ CHÍ MINH, 2024

THÔNG TIN HỘI ĐỒNG CHẤM KHÓA LUẬN TỐT NGHIỆP

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số …… ngày …….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

LỜI CẢM ƠN

Chúng em muốn bày tỏ lòng biết ơn sâu sắc tới Ban Giám hiệu cùng các thầy cô của Khoa Khoa học và Kỹ thuật Thông tin, Trường Đại học Công nghệ Thông tin (UIT) - Đại học Quốc gia TP. Hồ Chí Minh, vì đã tạo ra môi trường học tập thuận lợi và hỗ trợ chúng em trong suốt thời gian học và nghiên cứu.

Đặc biệt, chúng em xin gửi lời cảm ơn chân thành đến TS. Đỗ Trọng Hợp và TS. Trần Văn Thành. Sự hướng dẫn tận tình và động viên liên tục từ các thầy đã giúp chúng em vượt qua nhiều khó khăn trong quá trình thực hiện đề tài "***Hệ thống giám sát trái cây và rau củ thời gian thực***".

Chúng em cũng muốn bày tỏ lòng biết ơn đến các thầy cô và bạn bè trong nhóm ngành Khoa học dữ liệu, những người luôn đồng hành, chia sẻ và đóng góp những ý kiến quý giá, giúp chúng em cải thiện và hoàn thiện khóa luận này.

Cuối cùng, chúng em xin cảm ơn gia đình và bạn bè đã luôn ủng hộ, tiếp thêm sức mạnh và nghị lực cho chúng em trong suốt quá trình học tập và nghiên cứu.

TP. Hồ Chí Minh, ngày 01 tháng 07 năm 2024

Nguyễn Thị Nguyệt

Nguyễn Thị Phương Thảo

MỤC LỤC

Chương 1. MỞ ĐẦU 6

1.1. Lý do chọn đề tài 6

1.2. Mục tiêu nghiên cứu 6

1.3. Đối tượng nghiên cứu 7

1.4. Phạm vi nghiên cứu 7

Chương 2. TỔNG QUAN 9

2.1. Các nghiên cứu trước đây 9

2.1.1. Nghiên cứu trong nước 9

2.1.2. Nghiên cứu quốc tế 10

2.2. Những vấn đề còn tồn tại 11

2.3. Các giải pháp nghiên cứu cần thiết 12

Chương 3. PHƯƠNG PHÁP NGHIÊN CỨU VÀ PHÁT TRIỂN HỆ THỐNG 13

3.1. Khung làm việc 13

3.2. Xây dựng bộ dữ liệu 14

3.2.1. Nguồn dữ liệu và mô tả 14

3.2.2. Gán nhãn dữ liệu 15

3.2.3. Tiền xử lý dữ liệu 16

3.2.3.1. Tăng cường dữ liệu mosaic 18

3.2.3.2. Tăng cường dữ liệu đa loại (MTDA) 19

3.3. Mô hình huấn luyện 22

3.3.1. Phát hiện đối tượng (Object detection) 22

3.3.2. Phát hiện đối tượng hai giai đoạn (Two-stages) 24

3.3.3. Phát hiện đối tượng một giai đoạn (One-stage) 26

3.3.3.1. Mô hình mạng YOLOv5 27
3.3.3.2. Mô hình mạng YOLOv9 30
3.3.3.3. Mô hình mạng YOLOv10 32
3.4. Xử lý dữ liệu thời gian thực 33
3.4.1. Apache Kafka 33
3.4.2. Apache Spark 36
3.4.3. Theo dõi đối tượng DeepSORT 38
3.4.4. Đếm và giám sát đối tượng 40
Chương 4. NGHIÊN CỨU THỰC NGHIỆM 42
4.1. Cài đặt thực nghiệm 42
4.1.1. Phần ngoại tuyến 42
4.1.2. Phần trực tuyến 44
4.2. Phương pháp và độ đo đánh giá 46
Chương 5. KẾT QUẢ VÀ THẢO LUẬN 48
5.1. Kết quả thực nghiệm 48
5.2. Thảo luận 51
5.2.1. Thảo luận về dữ liệu 51
5.2.2. Thảo luận về kết quả các mô hình trên từng tập dữ liệu 51
5.2.3. Thảo luận về phần thời gian thực 55
Chương 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 56
6.1. Kết luận 56
6.2. Hướng phát triển 56

DANH MỤC HÌNH

Hình 3. 1. Sơ đồ Hệ thống giám sát trái cây và rau củ thời gian thực 13

Hình 3. 2. Một số hình ảnh mẫu của tập dữ liệu FVID. (a) Trái cây hoặc rau củ tươi; (b) Trái cây hoặc rau củ thối/hỏng; (c) Trái cây hoặc rau củ trạng thái tươi và hỏng; (d) Trái cây và rau củ hỗn hợp. 15

Hình 3. 3. Tăng cường dữ liệu với phương pháp xoay. 17

Hình 3. 4. Hình ảnh mô tả quá trình tăng cường dữ liệu Mosaic. (a) Bốn hình ảnh gốc ngẫu nhiên cùng với bounding box tương ứng; (b) Kết hợp bốn hình ảnh gốc cùng với bounding box tương ứng; (c) Cắt ngẫu nhiên và điều chỉnh bounding box 18

Hình 3. 5. Sơ đồ phương pháp Tăng cường dữ liệu đa loại 20

Hình 3. 6. Một số thuật toán phát hiện đối tượng (2014 - 2024) 24

Hình 3. 7. Phát hiện đối tượng với Faster R-CNN.....25

Hình 3. 8. Sơ đồ ResNet-101 25

Hình 3. 9. Sơ đồ phát triển các phiên bản YOLO 27

Hình 3. 10. Kiến trúc của Yolov5 28

Hình 3. 11. Sơ đồ kiến trúc của Apache Kafka.....35

Hình 3. 12. Sơ đồ cấu trúc thuật toán DeepSORT 39

Hình 3. 13. Minh họa việc xác định định danh riêng cho mỗi đối tượng 40

Hình 5. 1. Phân bố mẫu trước và sau khi sử dụng phép biến đối xoay cho mỗi lớp 48

Hình 5. 2. Minh họa quy trình đếm các đối tượng trên từng khung hình. 50

DANH MỤC BẢNG

Bảng 3. 1. Bảng chi tiết các thuộc tính của bộ dữ liệu FVID 14

Bảng 3. 2. Thuật toán tăng cường dữ liệu Mosaic 19

Bảng 3. 3. Chiến lược tăng cường dữ liệu đa loại (MTDA) 20

Bảng 4. 1. Dữ liệu thực nghiệm 43

Bảng 4. 2. Thông số cấu hình và môi trường cài đặt cho phần ngoại tuyến.....44

Bảng 4. 3. Thông số cấu hình và môi trường cài đặt cho phần trực tuyến 45

Bảng 4. 4. Thông tin triển khai và sử dụng hệ thống trực tuyến.....45

Bảng 5. 1. Hiệu suất mô hình Faster-RCNN-Resnet-101 48

Bảng 5. 2. Thông số của các mô hình YOLO 49

Bảng 5. 3. Hiệu suất các mô hình YOLO trên tập dữ liệu gốc 49

Bảng 5. 4. Hiệu suất các mô hình YOLO trên tập dữ liệu xoay 49

Bảng 5. 5. Hiệu suất các mô hình YOLO trên tập dữ liệu Xoay + Mosaic 50

Bảng 5. 6. Hiệu suất mô hình YOLOv9 trên tập dữ liệu Xoay + MTDA + Mosaic.50

DANH MỤC TỪ VIẾT TẮT

1. CNN: Convolutional Neural Network (mạng nơ-ron tích chập)
2. DAug: Data Augmentation (Phương pháp tăng cường dữ liệu)
3. MTDA: Multi-Type Data Augmentation (Phương pháp tăng cường dữ liệu đa loại)
4. CAT: Context-Aware Transformation (Phương pháp biến đổi theo ngữ cảnh)
5. APB: Assembled Piecewise Boundaries (Phương pháp ghép các mảnh ảnh)
6. OD: Object Detection (Phát hiện đối tượng)
7. Bbox: Bounding box
8. RPN: Region Proposal Network
9. R-CNN: Region-based Convolutional Neural Network
10. YOLO: You Only Look Once
11. NMS: Non-Maximum Suppression
12. CSP: Cross Stage Partial
13. PGI: Programmable Gradient Information
14. E-ELAN: Extended Efficient Layer Aggregation Networks
15. DeepSORT: Simple Online and Realtime Tracking with a Deep Association Metric.
16. OpenCV: Open Source Computer Vision Library

TÓM TẮT KHÓA LUẬN

Nghiên cứu này nhằm phát triển và đánh giá hệ thống giám sát trái cây và rau củ thời gian thực bằng cách sử dụng các mô hình nhận diện đối tượng tiên tiến. Các mô hình được áp dụng bao gồm Faster-RCNN-Resnet-101 và các phiên bản khác nhau của YOLO (YOLOv5, YOLOv9, YOLOv10). Bộ dữ liệu phong phú đã được xây dựng bằng cách thu thập hình ảnh của các loại trái cây và rau củ như táo, chuối, cà rốt và cà tím, sau đó áp dụng các kỹ thuật tăng cường dữ liệu như xoay hình, mosaic và MTDA để đảm bảo tính đa dạng và phản ánh nhiều điều kiện thực tế.

Hệ thống được triển khai với Apache Spark và Kafka để xử lý dữ liệu liên tục và đảm bảo phản ứng thời gian thực. Kết quả cho thấy YOLOv9 đạt các chỉ số cao nhất với precision là 0.884, recall là 0.84, mAP 0.5 đạt 0.912, và mAP 0.5:0.95 đạt 0.739 trên tập kiểm thử khi áp dụng các kỹ thuật tăng cường dữ liệu. YOLOv10 có thời gian suy luận nhanh nhất là 4.1 ms trên tập xác thực, chứng tỏ khả năng phản ứng nhanh trong các ứng dụng thời gian thực. Trong khi đó, Faster-RCNN-Resnet-101 đạt AP50 cao nhất trên tập kiểm thử với giá trị 85.49%, nhưng gặp khó khăn trong việc phát hiện các đối tượng nhỏ.

Hệ thống giám sát không chỉ nhận diện mà còn theo dõi và đếm chính xác từng đối tượng nhờ kết hợp mô hình YOLO tốt nhất với DeepSORT, đảm bảo quản lý số lượng trái cây và rau củ theo thời gian thực. Nghiên cứu này đã cung cấp những kết quả ấn tượng về hiệu suất và khả năng ứng dụng, mở ra hướng phát triển mới cho các hệ thống giám sát thời gian thực trong tương lai.

Cấu trúc của Khoá luận tốt nghiệp

Nhóm xin trình bày nội dung của Khóa luận tốt nghiệp theo cấu trúc như sau:

- Chương 1: Mở đầu

- Chương 2: Tổng quan

- Chương 3: Phương pháp nghiên cứu và phát triển hệ thống

- Chương 4: Cài đặt thực nghiệm

- Chương 5: Kết quả thực nghiệm

- Chương 6: Đánh giá và bàn luận kết quả

Chương 1. MỞ ĐẦU

1.1. Lý do chọn đề tài

Trong những năm gần đây, ngành nông nghiệp và công nghệ thực phẩm đã có những bước tiến đáng kể nhờ vào việc áp dụng các công nghệ tiên tiến như trí tuệ nhân tạo và phân tích dữ liệu lớn. Một trong những thách thức lớn nhất mà ngành nông nghiệp đang đối mặt là việc kiểm soát và quản lý chất lượng của trái cây và rau củ từ giai đoạn thu hoạch cho đến khi sản phẩm đến tay người tiêu dùng. Việc đảm bảo chất lượng thực phẩm nông sản không những ảnh hưởng trực tiếp đến lợi ích của người sử dụng mà còn quyết định sự thành công và phát triển lâu dài của các doanh nghiệp trong lĩnh vực này.

Hiện nay, các phương pháp kiểm tra chất lượng trái cây và rau củ chủ yếu dựa vào kiểm tra thủ công và cảm quan, điều này không những mất nhiều thời gian và chi phí mà còn có độ chính xác và hiệu quả không cao. Trong điều kiện nhu cầu tiêu dùng ngày càng cao và yêu cầu đối với chất lượng sản phẩm ngày một khắt khe, việc ứng dụng các hệ thống theo dõi và phân tích dữ liệu thời gian thực trở thành một yêu cầu cấp thiết. Các hệ thống này không những giúp tối ưu hoá quá trình sản xuất mà còn đảm bảo chất lượng sản phẩm đầu ra, giúp nâng cao giá trị sản phẩm và bảo vệ sức khoẻ người tiêu dùng.

Để giải quyết các vấn đề trên, nghiên cứu sẽ tập trung vào việc xây dựng "***Hệ thống giám sát trái cây và rau củ thời gian thực***". Sử dụng các công nghệ tiên tiến như Apache Spark và Kafka, hệ thống này sẽ giúp thu thập và phân tích dữ liệu video thời gian thực từ các camera giám sát, cho phép nhận dạng, phân tích dữ liệu và đếm số lượng trái cây và rau củ một cách tự động và hiệu quả.

1.2. Mục tiêu nghiên cứu

Mục đích nghiên cứu của đề tài này là xây dựng "***Hệ thống giám sát trái cây và rau củ thời gian thực***", nhằm đơn giản hoá quy trình kiểm soát chất lượng trong nông

nghiệp. Hệ thống sẽ sử dụng các công nghệ Big Data như Apache Spark và Apache Kafka nhằm xử lý dữ liệu thời gian thực từ các camera giám sát, giúp xử lý một lượng lớn dữ liệu video được lưu trữ trong thời gian dài. Hệ thống không những cung cấp khả năng phân tích cao mà còn lưu giữ thông tin một cách hiệu quả, nhờ đó tối ưu hoá quá trình kiểm soát chất lượng trong sản xuất.

1.3. Đối tượng nghiên cứu

- Xây dựng một bộ dữ liệu toàn diện chứa hình ảnh của hai loại trái cây (táo và chuối) và hai loại rau củ (cà rốt và cà tím).
- Thực hiện các kỹ thuật DAug mosaic, MTDA và phương pháp biến đổi xoay ba góc độ khác nhau để tăng số lượng và sự đa dạng của bộ dữ liệu.
- Huấn luyện mô hình Faster RCNN ResNet-101 cùng với ba phiên bản của mô hình YOLO là YOLOv5-S, YOLOv9-C và YOLOv10-L cho tác vụ phát hiện, phân loại trái cây và rau củ.
- Áp dụng công nghệ Big Data, sử dụng Apache Spark và Kafka để xử lý và phân tích dữ liệu thời gian thực từ các hệ thống phát hiện. Kết hợp với DeepSort để theo dõi và đếm chính xác các đối tượng.

1.4. Phạm vi nghiên cứu

Nghiên cứu này được triển khai với hai thành phần chính là phần ngoại tuyến và phần trực tuyến.

Phần ngoại tuyến:

- Thu thập, gán nhãn và DAug nhằm tạo ra bộ dữ liệu phong phú và đa dạng.
- Dùng bộ dữ liệu đã xây dựng để huấn luyện và đào tạo từng mô hình học sâu (Faster RCNN, YOLO), phân tích hiệu suất của từng mô hình nhằm lựa chọn mô hình tối ưu.

Phần trực tuyến:

- Xây dựng hệ thống dùng Apache Spark và Kafka để xử lý luồng dữ liệu thời gian thực.

- Tích hợp DeepSort để theo dõi và đếm chính xác các đối tượng qua các khung hình.

Phạm vi ứng dụng trong chuỗi cung ứng:

- *Ngoài vườn*: Ứng dụng của hệ thống có thể theo dõi và đếm số lượng cây trồng, xác định tình trạng sức khỏe của cây và phát hiện kịp thời các dấu hiệu bệnh tật.
- *Vận chuyển*: Hệ thống giúp giám sát quá trình vận chuyển hàng hóa, đếm số lượng và theo dõi tình trạng hàng hóa trong quá trình di chuyển, đảm bảo tính an toàn và chất lượng.
- *Kho lưu trữ*: Ứng dụng có thể theo dõi và quản lý số lượng hàng hóa trong kho, đảm bảo sự chính xác trong kiểm kê và tối ưu hóa không gian lưu trữ.

Bằng việc giải quyết các đối tượng nghiên cứu trong phạm vi đã định, nghiên cứu nhằm xây dựng một "***Hệ thống giám sát trái cây và rau củ thời gian thực***" mạnh mẽ và hiệu quả, đóng góp vào sự phát triển của ngành nông nghiệp và tự động hóa. Các chương còn lại của bài luận này được tổ chức như sau: Chương 2 sẽ tổng quan tài liệu, phân tích và đánh giá các nghiên cứu liên quan của các tác giả trong và ngoài nước, đồng thời xác định các vấn đề cần giải quyết trong đề tài này. Quy trình thu thập dữ liệu cùng với cơ sở lý thuyết và giả thuyết khoa học về các phương pháp DAug, OD và đếm rau quả được sử dụng trong nghiên cứu, được mô tả trong chương 3. Chương 4 mô tả chi tiết về quá trình nghiên cứu thực nghiệm, bao gồm cài đặt thực nghiệm và các phương pháp đánh giá. Tiếp đến, chương 5 sẽ thảo luận các kết quả thu được từ các thử nghiệm, đánh giá hiệu quả của hệ thống. Cuối cùng, chương 6 sẽ tổng kết nghiên cứu, đưa ra các kết luận và đề xuất hướng phát triển cho các nghiên cứu tương lai.

Chương 2. TỔNG QUAN

Chương này sẽ tập trung phân tích và đánh giá các nghiên cứu liên quan đến đề tài "***Hệ thống giám sát trái cây và rau củ thời gian thực***". Trong đó, sẽ nêu lên những vấn đề còn tồn tại và xác định những khía cạnh cần được tập trung nghiên cứu và giải quyết.

2.1. Các nghiên cứu trước đây

Phần này giới thiệu về các nghiên cứu liên quan đến đề tài ở trong và ngoài nước.

2.1.1. Nghiên cứu trong nước

Năm 2019, Nguyễn Văn Phúc đã cho thấy học sâu có thể cải thiện đáng kể độ chính xác và hiệu quả của hệ thống giám sát nông sản, sử dụng mô hình CNN để phân loại ảnh trái cây, giúp quy trình nhận dạng và phân loại trở nên tự động hơn[1]. Trong cùng năm, tác giả này cộng tác với Vũ Thanh Hiền, đã thiết kế mạng Deep Convolutional Neural Network để phân loại trái cây dựa trên ảnh màu, áp dụng các kỹ thuật DAug và phương pháp "dropout" nhằm giảm thiểu hiện tượng overfitting[1] trong quá trình huấn luyện[2].

Năm 2021, Trịnh Trung Hải và cộng sự đã phát triển một hệ thống để nhận dạng và phân loại trái cây chín bằng cách kết hợp thuật toán học sâu với các thuộc tính như kích thước, màu sắc, hình dạng và cấu trúc của trái cây[3]. Trong cùng năm, nhóm nghiên cứu này cũng đã áp dụng một phiên bản cải tiến của mô hình Fast R-CNN để nhận dạng và phát hiện trái dứa, giúp giảm bớt sức lao động cho nông dân và nâng cao hiệu quả quản lý chất lượng trái cây[4].

Gần đây nhất, vào năm 2024, Nguyễn Văn Mạnh và cộng sự đã nghiên cứu và ứng dụng thuật toán YOLOv7 vào phân loại cà chua, áp dụng phương pháp huấn luyện đa kích thước (multi-scale training) nhằm đạt được hiệu quả cao trong quá trình nhận dạng và phân loại[5].

[1] Overfitting là hiện tượng mô hình hoạt động tốt trên dữ liệu huấn luyện nhưng kém trên dữ liệu kiểm thử.

2.1.2. Nghiên cứu quốc tế

Năm 2021, Yanfei Li và cộng sự đã ứng dụng CNN để nhận diện và phân loại chất lượng táo từ các hình ảnh thực tế bị nhiễu. Đây là một bước tiến quan trọng, giúp tăng độ chính xác của hệ thống phân loại trong điều kiện thực tế[6]. Cũng trong năm này, Kirill Bogomasov và nhóm nghiên cứu đã phát triển mô hình kết hợp giữa EfficientNet và Decision Tree để phân loại và đếm trái cây và rau củ, giúp tiết kiệm thời gian và giảm sai sót trong quy trình kiểm đếm[7].

Năm 2022, Mukhriddin Mukhiddinov và cộng sự đã nâng cấp mô hình YOLOv4 để phân loại trái cây và rau củ thành hai loại: tươi và hỏng. Bằng cách sử dụng hàm kích hoạt Mish và các mạng tích hợp, tác giả đã cải thiện đáng kể độ chính xác và tốc độ của mô hình. Bộ dữ liệu gồm 12,000 hình ảnh từ 10 loại trái cây và rau củ dưới nhiều điều kiện khác nhau đã được sử dụng để huấn luyện và kiểm tra mô hình. Tác giả cũng áp dụng các kỹ thuật tăng cường dữ liệu như xoay hình ảnh và ghép ảnh Mosaic[8]. Trong cùng năm, Jagannadha Swamy Tata và đồng nghiệp đã phát triển một hệ thống trí tuệ nhân tạo để phân loại và đánh giá chất lượng trái cây và rau củ theo thời gian thực. Hệ thống sử dụng các kỹ thuật xử lý hình ảnh để trích xuất đặc trưng quan trọng và CNN để nhận diện và phân loại. Hệ thống được triển khai trên nền tảng Android, giúp người dùng đánh giá chất lượng trái cây và rau củ dễ dàng trên điện thoại di động[9].

Năm 2023, Jing Hu và cộng sự đã cải tiến mô hình YOLOv7 bằng cách kết hợp cơ chế attention[2] của Vision Transformer, giúp tăng cường khả năng phát hiện và đếm trái cây trong vườn táo. Ngoài ra, tác giả còn tích hợp phương pháp theo dõi đối tượng đa mục tiêu để nâng cao độ chính xác trong việc đếm trái cây qua các khung hình video. Các kỹ thuật tăng cường dữ liệu như MixUP và Mosaic cũng được sử dụng để giảm hiện tượng overfitting[10].

[2] Attention là cơ chế giúp mô hình học máy tập trung vào các phần quan trọng của đầu vào khi đưa ra dự đoán, đặc biệt hữu ích trong xử lý ngôn ngữ tự nhiên và thị giác máy tính.

Gần đây, vào năm 2024, Jiachuang Zhang và nhóm nghiên cứu đã cải tiến mô hình YOLOv5 để phát hiện chính xác các mục tiêu nhỏ trong môi trường vườn tự nhiên[11].

2.2. Những vấn đề còn tồn tại

Mặc dù có nhiều nghiên cứu đã được thực hiện nhằm cải thiện chất lượng và hiệu quả của các hệ thống giám sát trái cây và rau củ, vẫn còn tồn tại một số vấn đề chưa được giải quyết hoàn toàn:

- Một trong những thách thức chính là sự thiếu hụt về dữ liệu phong phú và đa dạng. Các bộ dữ liệu được sử dụng trong các nghiên cứu trước đây thường không đủ về số lượng và chủng loại, dẫn đến khó khăn trong việc phân biệt và nhận dạng chính xác các loại trái cây và rau củ trong nhiều điều kiện khác nhau. Điều này ảnh hưởng đến hiệu suất của hệ thống khi áp dụng vào thực tế.
- Thêm vào đó, việc sử dụng các kỹ thuật tăng cường dữ liệu gặp phải vấn đề lớn về sự thích ứng quá mức. Nhiều mô hình bị quá phụ thuộc vào dữ liệu huấn luyện, làm giảm khả năng dự đoán chính xác trên dữ liệu mới. Khả năng xử lý dữ liệu liên tục và kịp thời cũng là một điểm yếu cần cải thiện. Nhiều hệ thống hiện tại chưa thể xử lý dữ liệu một cách liên tục và nhanh chóng, gây ra sự chậm trễ và không đáp ứng được yêu cầu của người sử dụng.
- Ngoài ra, độ chính xác trong việc nhận dạng và phân loại nông sản trong điều kiện thực tế vẫn chưa đạt được mức độ mong muốn. Điều này đòi hỏi các mô hình cần phải có khả năng theo dõi và quản lý số lượng nông sản một cách hiệu quả hơn để đáp ứng nhu cầu thực tế.
- Ngoài ra, khả năng xử lý dữ liệu thời gian thực cũng là một điểm cần cải thiện. Nhiều hệ thống hiện tại chưa tối ưu hóa được việc xử lý liên tục, gây ra độ trễ trong phản hồi và không đáp ứng kịp thời các yêu cầu của ứng dụng thực tế.

2.3. Các giải pháp nghiên cứu cần thiết

Để khắc phục các hạn chế hiện tại, nghiên cứu tập trung vào các hướng chính sau:

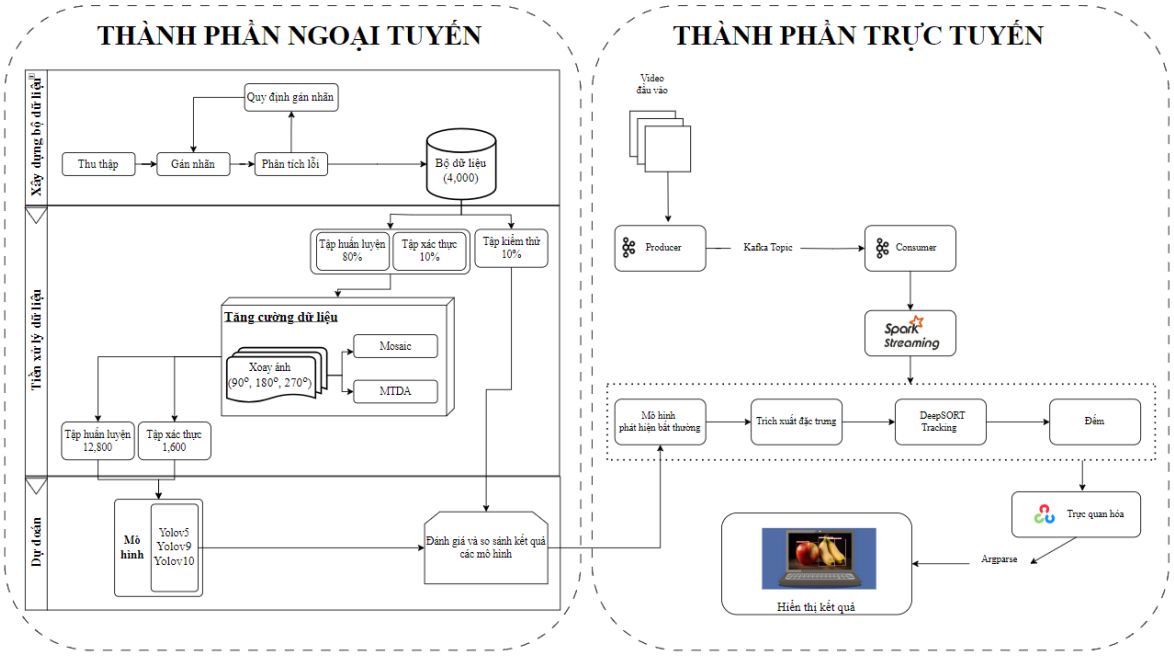
- *Xây dựng bộ dữ liệu phong phú* (mục 3.2): Thu thập hình ảnh của táo, chuối, cà rốt và cà tím, và áp dụng các kỹ thuật như xoay hình, ghép ảnh mosaic và chiến lược tăng cường đa loại (MTDA) để đảm bảo tính đa dạng và phản ánh nhiều điều kiện thực tế.
- *Triển khai và đánh giá mô hình* (mục 3.3): Sử dụng các mô hình nhận diện hiện đại như Faster RCNN và các phiên bản YOLO (YOLOv5, YOLOv9, YOLOv10) để phát hiện và phân loại trái cây và rau củ.
- *Xử lý dữ liệu thời gian thực* (mục 3.4): Áp dụng Apache Spark và Kafka để xử lý và phân tích dữ liệu liên tục, giúp hệ thống phản ứng nhanh chóng với những thay đổi trong môi trường.
- *Theo dõi và đếm đối tượng*: Kết hợp mô hình YOLO tốt nhất với DeepSORT để theo dõi và đếm chính xác từng đối tượng, đảm bảo hệ thống không chỉ nhận diện mà còn quản lý số lượng trái cây và rau củ theo thời gian thực.

Chương 3. PHƯƠNG PHÁP NGHIÊN CỨU VÀ PHÁT TRIỂN HỆ THỐNG

Chương 3 sẽ trình bày cơ sở lí thuyết, lí luận, giả thiết khoa học và phương pháp nghiên cứu đã được sử dụng.

3.1. Khung làm việc

Nghiên cứu đề xuất một khung làm việc cho "***Hệ thống giám sát trái cây và rau củ thời gian thực***". Hệ thống thiết kế bao gồm hai phần chính: phần ngoại tuyến và phần trực tuyến, được trình bày cụ thể trong hình 3.1.



Hình 3. 1. Sơ đồ Hệ thống giám sát trái cây và rau củ thời gian thực

Thành phần ngoại tuyến: Phần ngoại tuyến tập trung vào xây dựng và phát triển bộ dữ liệu cũng như các mô hình học máy. Quy trình này bao gồm: Thu thập và gán nhãn dữ liệu, DAug, và huấn luyện các mô hình CNN, YOLO để nhận diện và phân loại trái cây và rau củ.

Thành phần trực tuyến: Phần trực tuyến đảm bảo xử lý dữ liệu thời gian thực, với các công cụ như Apache Spark và Apache Kafka giúp lưu trữ và xử lý dữ liệu một cách nhanh chóng và hiệu quả. Bên cạnh đó, mô hình được đánh giá cao trong phần ngoại tuyến sẽ được tích hợp với DeepSORT cho phép theo dõi và quan sát từng đối

tượng trong các khung hình. Kết quả sau khi xử lý sẽ được trực quan để dễ dàng theo dõi.

3.2. Xây dựng bộ dữ liệu

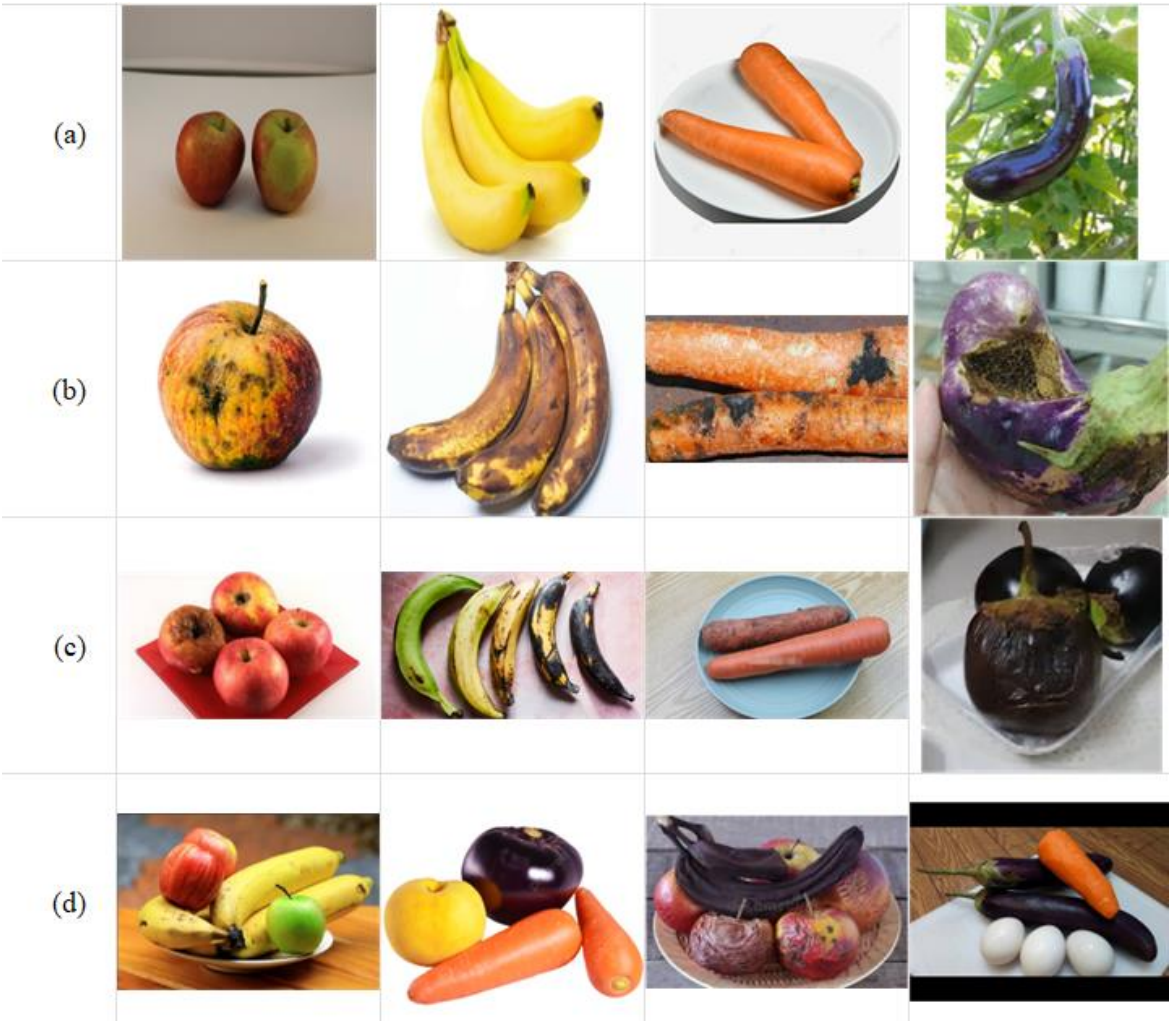
3.2.1. Nguồn dữ liệu và mô tả

Bộ dữ liệu Fruit And Vegetable Images Dataset (FVID) được xây dựng dựa trên bộ dữ liệu Fruits and Vegetables dataset[3] có sẵn và công khai trên kho lưu trữ Kaggle. Bên cạnh đó, bổ sung thêm vào một lượng dữ liệu được tìm kiếm thủ công từ nhiều nguồn khác nhau trên Internet như Google, Shutterstock, Dreamtime, Alamy, Youtube, và camera điện thoại di động. Bộ dữ liệu được thu thập bao gồm tổng cộng 4.000 điểm dữ liệu, chia thành hai loại trái cây (táo và chuối) và hai loại rau củ (cà rốt và cà tím). Mỗi loại trái cây và rau củ đều được phân loại thành hai nhóm: tươi và thối/hỏng, tạo thành tổng cộng 8 nhãn, được miêu tả chi tiết trong bảng 3.1. Trong hình 3.2 minh họa một số hình ảnh mẫu từ tập dữ liệu này, đối với mỗi hình ảnh, có thể được chụp ngoài trời hoặc trong nhà để thu được thông tin dưới các điều kiện ánh sáng khác nhau. Ngoài ra, các rau củ quả có thể bị che khuất một phần hoặc bị che phủ bởi các vật thể khác, tạo ra các trường hợp khó khăn trong việc nhận diện.

Loại	Nhãn/ Lớp (Class)
Trái cây	Táo tươi
	Táo hỏng
	Chuối tươi
	Chuối hỏng
Rau củ	Cà rốt tươi
	Cà rốt hỏng
	Cà tím tươi
	Cà tím hỏng

Bảng 3. 1. Bảng chi tiết các thuộc tính của bộ dữ liệu FVID

[3] https://www.kaggle.com/datasets/muhriddinmuxiddinov/fruits-and-vegetables-dataset



Hình 3. 2. Một số hình ảnh mẫu của tập dữ liệu FVID. (a) Trái cây hoặc rau củ tươi; (b) Trái cây hoặc rau củ thối/hỏng; (c) Trái cây hoặc rau củ trạng thái tươi và hỏng; (d) Trái cây và rau củ hỗn hợp.

3.2.2. Gán nhãn dữ liệu

Quá trình gán nhãn được tiến hành dựa trên quy tắc gán nhãn dưới sự nhất trí của cả hai thành viên, cộng với một vài lần phân tích khác, cùng với một số lần phân tích lỗi. Quy định gán nhãn được mô tả như sau:

Đảm bảo yêu cầu về các thuộc tính có trong ảnh:

(a) Ảnh chứa một trong tám loại được đề cập trong bảng 3.1, tối thiểu là một loại, tối đa là tám loại. Chấp nhận ảnh của các loại bị cắt mất một phần, bị

mờ trong khoảng chấp nhận được (< 0.4) hoặc bị cắt ra thành từng phần nhưng vẫn nhận dạng được.

(b) Nhận diện loại: Nhận diện trực quan, dựa vào hình dạng, màu sắc và kích thước để nhận diện từng loại trái cây và rau củ. Chấp nhận hình ảnh những loại rau củ bị biến dạng và biến đổi về màu sắc, tuy nhiên loại rau củ đó phải nhận diện được dựa trên những thuộc tính khác. Các loại trái cây và rau củ khác không thuộc bốn loại được nêu trong bảng 3.1 sẽ không được ghi nhãn.

(c) Phân loại chất lượng:

- Chất lượng tốt (tươi): Trái cây, rau củ có màu sắc đồng đều, sáng bóng, không có dấu hiệu của sự hư hỏng, không mốc, bề mặt mịn màng, không có vết thâm, vết nứt , không dập nát, và giữ nguyên hình dáng tự nhiên.
- Chất lượng xấu (hỏng): Có dấu hiệu thối rữa, bầm dập, mốc, hoặc biến dạng nghiêm trọng.

(d) Đếm số lượng: Thực hiện tính số lượng dựa trên số lượng các bounding box tương ứng của từng loại được gán ở phần phân loại.

(e) Trong quá trình gán nhãn người gán nhãn loại bỏ cảm xúc cá nhân.

Bộ dữ liệu hình ảnh này sau khi gán nhãn, được phân thành hai loại chính, với 80% mẫu bao gồm các đối tượng đơn lẻ và phần còn lại bao gồm nhiều đối tượng từ 2 đến 8, như trong hình 3.2. Toàn bộ tập dữ liệu được chú thích thủ công bằng công cụ MakeSense.AI[4].

3.2.3. Tiền xử lý dữ liệu

Tiến hành chia bộ dữ liệu 4.000 ảnh thành 3 tập là tập huấn luyện, tập xác thực và tập kiểm thử với tỉ lệ lần lượt là 8:1:1.

DAug được đề cập như một phần quan trọng của regularization[5] qua dữ liệu [12], bằng cách áp dụng các phương pháp biến đổi vào tập huấn luyện để tạo ra một tập

[4] https://www.makesense.ai/

[5] Regularization là kỹ thuật ngăn chặn overfitting bằng cách thêm ràng buộc vào mô hình học máy.

dữ liệu mới hoặc mở rộng tập dữ liệu sẵn có, khiến cho mô hình không bị lệ thuộc vào tập dữ liệu huấn luyện ban đầu, qua đó hạn chế tình trạng overfitting.

Các phương pháp DAug có thể được phân vào nhiều loại khác nhau, mỗi một phương pháp đều làm biến đổi dữ liệu, tạo ra sự đa dạng và mở rộng cho tập dữ liệu, giúp cải thiện độ chính xác của mô hình. Các phép biến đổi hình học như xoay (rotation), dịch chuyển (translation), chia tỷ lệ (scaling ratio) và lật ảnh (flipping) giúp thay đổi hướng và cấu trúc của hình ảnh. Điều chỉnh màu sắc (color space) và độ tương phản (contrast) giúp thay đổi hình thức của hình ảnh, bao gồm thay đổi độ sáng, độ tương phản và cân bằng màu sắc. Việc chèn nhiễu (noise injection), chẳng hạn như thêm nhiễu Gaussian hoặc salt-and-pepper, đưa ra các biến thể ngẫu nhiên. Các kỹ thuật cắt, ghép và trộn như Mixup [13] và CutMix [14] sửa đổi hình ảnh hoặc các thành phần của chúng để tạo ra các mẫu mới. Hơn nữa, kỹ thuật mosaic và dynamic mosaic [15] tạo ra hình ảnh tổng hợp từ nhiều ảnh gốc, giúp đa dạng hóa dữ liệu một cách toàn diện. Các kỹ thuật trên đã được chứng minh là hữu ích cho các bộ dữ liệu cụ thể, cũng như tác vụ phân loại (classification), lẫn phát hiện đối tượng (object detection).

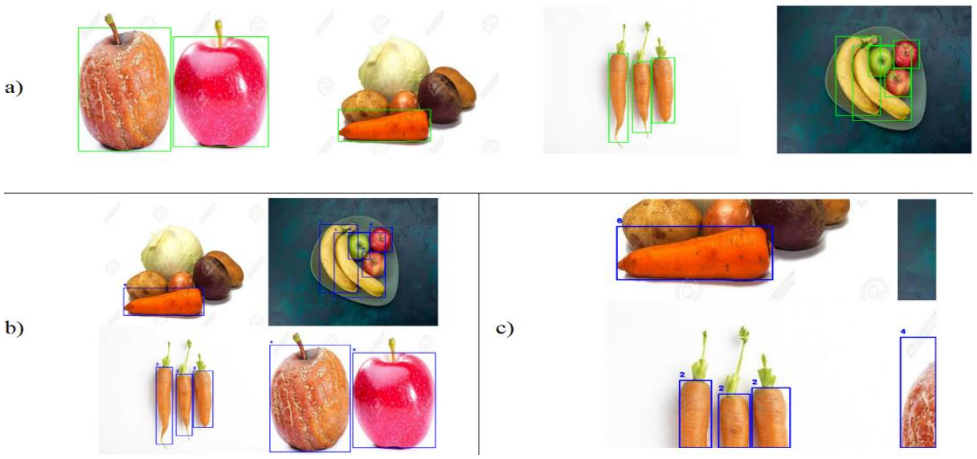
Hình 3. 3. Tăng cường dữ liệu với phương pháp xoay.

Dựa trên cơ sở như vậy, trong bài luận này, nhằm nâng cao độ phong phú và chất lượng của dữ liệu, nghiên cứu sẽ sử dụng và kết hợp các kỹ thuật DAug. Cụ thể, các bức ảnh trong tập huấn luyện và tập xác thực sẽ được xoay theo các góc 90°, 180°, và 270°, chi tiết hình 3.3 nhằm tạo ra nhiều biến thể khác nhau của cùng một ảnh. Bên cạnh đó, các kỹ thuật như mosaic và MTDA, được sử dụng để kết hợp, tạo ra các mẫu mới và phức tạp hơn, hai kỹ thuật được mô tả chi tiết cách thức thực hiện

tương ứng trong mục 3.2.3.1 và 3.2.3.1. Nhờ vào các bước DAug trên, tập huấn luyện cuối cùng bao gồm 12.800 ảnh và tập xác thực bao gồm 1.600 ảnh.

3.2.3.1. Tăng cường dữ liệu mosaic

DAug với mosaic giới thiệu lần đầu trong YOLOv4, bởi tác giả của YOLOv5. Đây là một kỹ thuật DAug cực kỳ mạnh mẽ trong OD, nhờ vào tính hiệu quả và độ ổn định cao mà nó mang lại. Mosaic là một cải tiến của phương pháp DAug CutMix. Ý tưởng đằng sau là bốn hình ảnh được ghép lại với nhau để tạo thành một ảnh duy nhất, chi tiết thuật toán được mô tả trong bảng 3.2.



Hình 3. 4. Hình ảnh mô tả quá trình tăng cường dữ liệu Mosaic. (a) Bốn hình ảnh gốc ngẫu nhiên cùng với bounding box tương ứng; (b) Kết hợp bốn hình ảnh gốc cùng với bounding box tương ứng; (c) Cắt ngẫu nhiên và điều chỉnh bounding box

Theo thuật toán được trình bày trong bảng 3.2, sự biến đổi trong hình ảnh được thể hiện như trong hình 3.4. Quá trình DAug Mosaic bắt đầu bằng việc điều chỉnh kích thước của từng ảnh sao cho khớp với kích thước đầu ra mong muốn. Sau đó, các hình ảnh đã được thay đổi kích thước sẽ được ghép lại với nhau, chia thành bốn góc phần tư, mỗi góc phần tư sẽ được lấp đầy bằng một miếng vá từ một trong bốn hình ảnh gốc, tạo thành một bức ảnh mosaic kết hợp các yếu tố từ cả bốn bức ảnh. Tiếp theo, một phần cắt ngẫu nhiên của hình ảnh kết hợp sẽ được lấy ra để tạo ra hình ảnh mosaic cuối cùng. Lưu ý rằng, các hình ảnh mosaic sẽ không bị trùng lặp. Kỹ

thuật này giúp tạo ra đầu vào đa dạng và phức tạp cho mô hình, đưa ra các biến thể về không gian và ngữ cảnh, buộc mô hình phải tìm hiểu các tính năng mạnh mẽ hơn. Bốn hình ảnh gốc là những hình ảnh được chọn ngẫu nhiên từ tập dữ liệu huấn luyện và các chú thích (nhãn) của chúng được điều chỉnh tương ứng để đảm bảo thông tin hộp giới hạn (bounding box) chính xác trong hình ảnh mosaic.

Đầu vào: Images = {I_1, I_2, I_3, I_4}; Labels = { L_1, L_2, L_3, L_4}, Kích thước ảnh: S

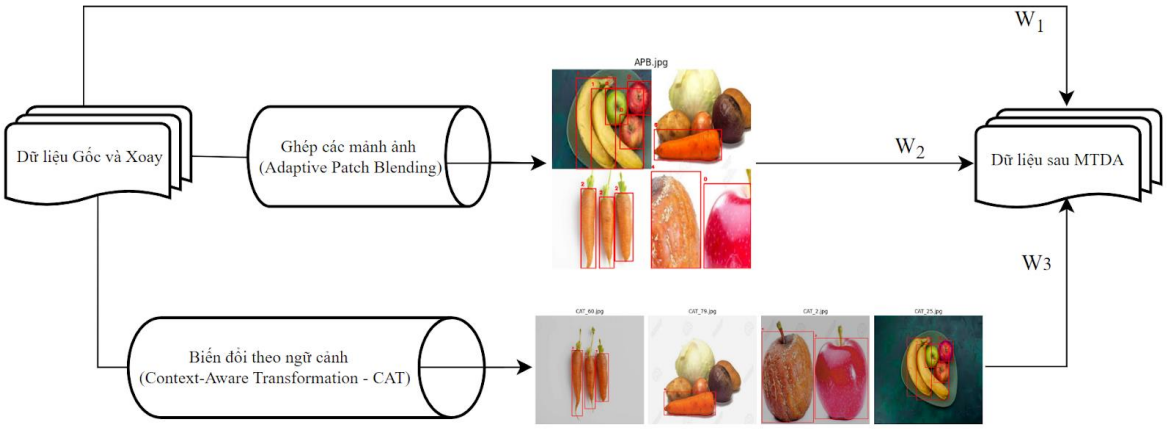
Đầu ra: I_{mosaic} ; L_{mosaic}

```
Iresize = []
For i in Images do
    Iresized.append(Resize(i, S))
End for
Icombined = TaoAnhXam(2S)
Lcombined = []
vitri = [(0, 0), (S, 0), (0, S), (S, S)]
For i in range(0, 4) do
    ThayAnh(Icombined, Iresized[i], vitri[i])
    DieuChinhBBox(Lcombined, Labels[i], vitri[i])
End for
xcenter, ycenter = (Random(S/2, 2S - S/2), Random(S/2, 2S - S/2))
Imosaic = CatAnh(Icombined, (xcenter, ycenter), S)
Lmosaic = []
For bbox in Lcombined do
    If bbox trong vùng cắt then
        DieuChinhBBox (bbox)
        Lmosaic.append(bbox)
    End if
End for
return Imosaic, Lmosaic
```

Bảng 3. 2. Thuật toán tăng cường dữ liệu Mosaic

3.2.3.2. Tăng cường dữ liệu đa loại (MTDA)

MTDA được thiết kế để tăng cường sự đa dạng của dữ liệu huấn luyện bằng việc kết hợp nhiều loại và phương pháp tăng cường khác nhau[15]. Trong nghiên cứu này, MTDA được minh họa trong hình 3.5, và chi tiết thuật toán được mô tả trong bảng 3.3. Ý tưởng chính là phân chia tập dữ liệu huấn luyện làm ba phần, mỗi phần sử dụng một phương pháp tăng cường dữ liệu riêng lẻ.



Hình 3. 5. Sơ đồ phương pháp Tăng cường dữ liệu đa loại

Đầu vào: Ảnh gốc I_{orig}, Nhãn gốc L_{orig}, Trọng số W_1, W_2, W_3

Đầu ra: I_{mtda} ; L_{mtda}

Function MTDA(I_{orig}, L_{orig}, W_1, W_2, W_3)
    P_1, P_2, P_3 = ChiaDuLieu(I_{orig}, L_{orig}, W_1, W_2, W_3)
    If len(P_2) >=4 **then**
        APB = AdaptivePatchBlending(P_2)
    End if
    If len(P_3) >=0 **then**
        CAT = ContextAwareTransformation(P_3)
    End if
    I_{mtda}, L_{mtda} = Merge(P_1, APB, CAT)
End function
Return I_{mtda} ; L_{mtda}

Bảng 3. 3. Chiến lược tăng cường dữ liệu đa loại (MTDA)

Các phương pháp DAug được áp dụng trong MTDA bao gồm:

- *Ảnh gốc:* Các ảnh trong tập dữ liệu mà không thực hiện bất kỳ biến đổi nào.
- *Áp dụng kỹ thuật Ghép các mảnh ảnh (Adaptive Patch Blending - APB):* Kỹ thuật này dựa trên ý tưởng của mosaic nhưng phần cắt ngẫu nhiên xảy ra trên từng ảnh trước khi ghép. Phương pháp này giúp giữ nguyên thông tin quan trọng của ảnh, khắc phục điểm yếu của mosaic về khả năng làm mất thông tin khi thực hiện cắt ngẫu nhiên. Như đã được đề cập trong phần cuối của mục 3.2.2, ảnh chứa nhiều đối tượng chỉ chiếm 20% trên cả tập dữ liệu, đây cũng là cơ sở khi áp dụng kỹ thuật này. Cụ thể, có bốn bức ảnh gốc, với mỗi hình ảnh được chọn, trích xuất một mảnh ngẫu nhiên chứa ít nhất một đối

tượng quan tâm như trong bảng 3.1 và điều chỉnh các nhãn cho phù hợp với tọa độ mới trong mảnh trích xuất. Sau đó, tạo một ảnh ghép trống với kích thước mong muốn SxS. Tiếp đó, chia ảnh ghép thành bốn phần (mỗi phần S/2xS/2), đặt các mảnh vào bốn phần này, điều chỉnh các nhãn để phù hợp với vị trí và kích thước mới trong ảnh ghép. Ngoài ra, nghiên cứu còn đảm bảo rằng các hình ảnh sau khi ghép sẽ không bị trùng lặp, bằng cách duyệt qua từng mảnh, sau đó chọn ngẫu nhiên ba mảnh còn lại trong tập mảnh và ghép chúng lại. Bằng cách trên, tuy có khả năng sẽ bị trùng các mảnh, nhưng vị trí khi ghép vào lại có sự khác nhau, vì thế, sẽ đảm bảo được tập ảnh đầu ra của phương pháp này sẽ không bị trùng.

- *Áp dụng kỹ thuật Biến đổi theo ngữ cảnh (Context-Aware Transformation - CAT):* Phương pháp tăng cường này dựa trên yêu cầu của bài toán về nhận dạng rau củ quả tươi/hỏng, các kỹ thuật biến đổi dữ liệu truyền thống bao gồm điều chỉnh độ tương phản và độ bão hoà thường được sử dụng một cách độc lập mà không quan tâm đến ngữ cảnh của toàn bộ hình ảnh. Điều này có thể dẫn đến sự biến dạng không mong muốn và làm mất đi các đặc trưng quan trọng của hình ảnh. Tập trung vào điều chỉnh các yếu tố của hình ảnh dựa trên ngữ cảnh tổng thể của nó, như độ sáng trung bình, độ tương phản, và các thành phần màu sắc. Cụ thể, bước đầu tính toán độ sáng trung bình của từng hình ảnh để xác định các thuộc tính ngữ cảnh, sau đó, dựa trên giá trị này, nếu độ sáng trung bình thấp hơn ngưỡng nhất định, tăng độ sáng của hình ảnh; ngược lại, giảm độ sáng nếu độ sáng trung bình cao hơn ngưỡng. Cuối cùng, điều chỉnh độ tương phản và các thành phần màu sắc HSV[6].

Việc phân tách tập dữ liệu huấn luyện và sử dụng các phương pháp tăng cường khác nhau giúp mô hình tiếp xúc với nhiều dạng khác nhau của dữ liệu gốc, từ đó học được các đặc trưng quan trọng một cách toàn diện. Phương pháp ghép các mảnh ảnh (APB) giúp giữ nguyên thông tin quan trọng của ảnh, trong khi phương pháp biến đổi theo ngữ cảnh (CAT) đảm bảo các biến đổi không làm mất đi thông tin ngữ cảnh quan trọng của ảnh. Sử dụng chiến lược này trong bài toán nhận diện rau củ quả tươi/hỏng bước đầu đặt ra giả thuyết mô hình có khả năng phân loại tốt hơn giữa các loại rau củ quả, qua đó tăng độ chính xác của việc nhận diện.

[6] Hue - sắc độ, Saturation - độ bão hòa, Value - giá trị độ sáng của màu sắc

3.3. Mô hình huấn luyện

Phần này sẽ trình bày các nội dung liên quan đến tác vụ OD, tiếp theo là khảo sát và phân tích chi tiết hai phương pháp OD chính là phương pháp hai giai đoạn (Two-stages) và phương pháp một giai đoạn (One-stage). Mục tiêu là cung cấp một cái nhìn tổng quan về cách thức vận hành và tác dụng của mỗi phương pháp. Đây cũng sẽ là nền tảng cho bài luận này, nhằm áp dụng các mô hình từ cả hai phương pháp để đánh giá các kết quả trên tập dữ liệu đã được xây dựng và nghiên cứu trong mục 3.2.

3.3.1. Phát hiện đối tượng (Object detection)

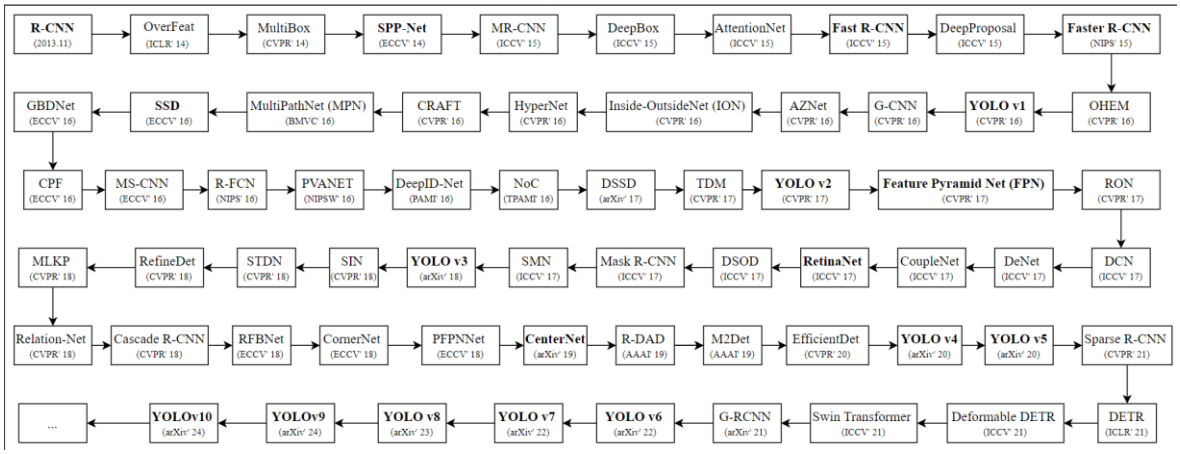
OD là một lĩnh vực quan trọng trong thị giác máy tính và trí tuệ nhân tạo[16]. Kỹ thuật này cho phép các hệ thống xác định và định vị các đối tượng trong hình ảnh hoặc video. Đây không phải là một bài toán mới, và trong những năm gần đây, lĩnh vực này đã đạt được nhiều thành tựu đáng kể, bao gồm cả các ứng dụng thực tiễn và sự phát triển của các mô hình thuật toán.

Với phần ứng dụng, mở ra nhiều ứng dụng trong các lĩnh vực như xe tự hành, giám sát an ninh, y học, và phân tích bán lẻ[7]. Chẳng hạn, các hệ thống giám sát đã trở nên thông minh hơn, có khả năng xác định các mối đe dọa tiềm ẩn hoặc sự bất thường trong thời gian thực. Trong lĩnh vực y tế, tính năng phát hiện vật thể đang hỗ trợ các quy trình chẩn đoán, hỗ trợ bác sĩ X quang phát hiện chính xác các khối u hoặc dị thường bằng cách quét hình ảnh y tế[17].

Trong phần thuật toán, các phương pháp OD đã có sự tiến bộ đáng kể qua nhiều giai đoạn phát triển, bắt đầu từ các kỹ thuật xử lý ảnh truyền thống đến các phương pháp học sâu (deep learning) hiện đại. Với sự phát triển của công nghệ deep learning, các phương pháp OD ngày càng trở nên hiệu quả hơn. Cụ thể, trước khi deep learning trở nên phổ biến từ khoảng năm 2013, hầu như mọi hoạt động OD đều được thực hiện bằng các kỹ thuật OD cổ điển, chủ yếu dựa trên các kỹ thuật xử lý ảnh hoặc kỹ thuật học máy cổ điển. Các kỹ thuật xử lý ảnh như phát hiện biên (edge detection), giúp xác định ranh giới đối tượng bằng cách tìm các điểm có sự thay đổi bất chợt về

màu sắc [18]; Histogram of Oriented Gradients (HOG) biểu diễn hình dạng của đối tượng bằng cách sử dụng thông tin về sự phân bố của các cường độ gradient (intensity gradient) hoặc các hướng biên (edge directions) [19]; và Scale Invariant Feature Transform (SIFT), sử dụng Gaussian filters và Difference of Gaussian (DoG) để tìm các điểm quan trọng (cực trị) trong ảnh (các điểm đặc trưng) ở các tỷ lệ khác nhau, tinh chỉnh vị trí và xác định hướng cho mỗi điểm đặc trưng dựa trên gradient của cường độ pixel xung quanh, sau đó so sánh các vector đặc trưng giữa ảnh tham chiếu và ảnh cần nhận diện để nhận dạng vật thể [20]. Kỹ thuật học máy cổ điển như Viola-Jones, sẽ tính toán đặc trưng Haar-like dựa trên ảnh tích phân, sau đó sử dụng Cascade classifier để kiểm tra các vùng ảnh và phát hiện khuôn mặt, loại bỏ nhanh các vùng không tiềm năng [21]; hoặc Deformable Parts Models (DPM) sử dụng phương pháp cửa sổ trượt (sliding window), trong đó các bộ phân loại được áp dụng một cách đều đặn trên toàn bộ bức ảnh, theo kiểu brute force [22]. Ngày nay, các kỹ thuật dựa trên deep learning (hình 3.6) vượt trội hơn rất nhiều so với những kỹ thuật trước đó. Đối với phương pháp này, các mô hình OD thường sẽ được xếp vào hai loại là phát hiện đối tượng hai giai đoạn (two-stage OD) và phát hiện đối tượng một giai đoạn (one-stage OD). Cùng điểm sơ qua phần 3.3.2 và 3.3.3, trình bày về mô hình được sử dụng trong bài luận này đối với mỗi loại. Two-stage trước tiên sử dụng một mạng backbone để lấy các đặc trưng từ hình ảnh. Những đặc trưng này sau đó được đưa vào một module nhằm tạo ra các vùng đề xuất. Cuối cùng, các vùng đề xuất này được mạng head xử lý để phân loại đối tượng và dự đoán các bounding boxes. Một số mô hình phổ biến thuộc loại này bao gồm họ nhà R-CNN. Khác với two-stage, one-stage sử dụng một mạng backbone để trích xuất các đặc trưng và một mạng head để tiến hành phân loại và hồi quy bounding boxes thông qua một bước duy nhất, một số mô hình phổ biến như họ YOLO, SSD, RetinaNet. Lựa một trong hai loại mô hình trên để sử dụng sẽ có sự đánh đổi khác nhau, trong khi one-stage sẽ ưu tiên hơn về tốc độ xử lý thì two-stage sẽ cho ra kết quả chính xác hơn. Ngoài hai loại mô hình phổ biến trên thì còn có loại khác là end-

to-end OD được phát triển sau này (ví dụ như DETR, DeFCN, DINO), nhưng sẽ không được để cập trong bài luận này.



Hình 3. 6. Một số thuật toán phát hiện đối tượng (2014 - 2024)

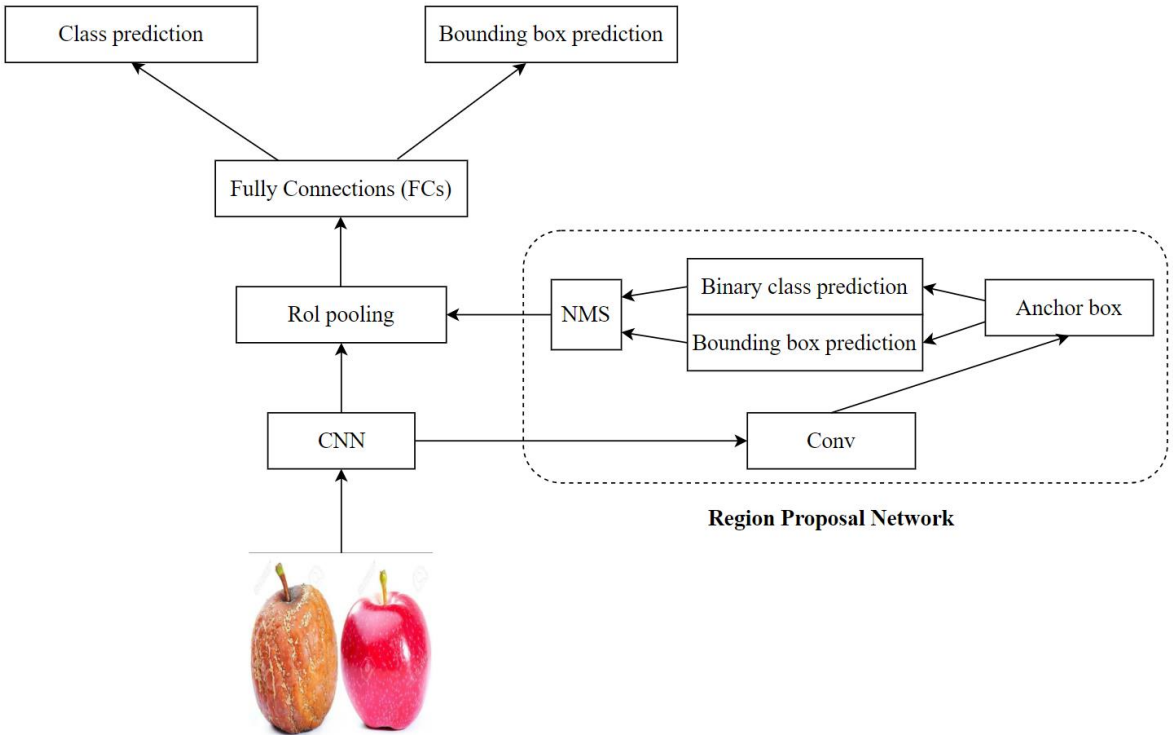
3.3.2. Phát hiện đối tượng hai giai đoạn (Two-stages)

Chi tiết, bao gồm hai giai đoạn:

- *Giai đoạn đầu*: Tạo đề xuất vùng (Region proposal), là giai đoạn để trích xuất các vùng trên ảnh có khả năng chứa đối tượng, bằng cách sử dụng một hàm hoặc một mạng nào đó (tùy vào mô hình được lựa chọn) để đưa ra một số lượng giới hạn các bounding boxes chứa các đối tượng tiềm năng. Các bounding boxes này được gọi là "anchors"[7]. Thuật toán đề xuất vùng này đã giải quyết được vấn đề về thời gian của phương pháp truyền thống là sử dụng cửa sổ trượt để trượt qua toàn bộ các vùng trong ảnh (brute force), tạo ra một số lượng windows cực kỳ lớn khiến thời gian huấn luyện và inference bị chậm.
- *Giai đoạn hai*: Phân loại và xác định vị trí đối tượng (Object classification and localization), giai đoạn này xác định liệu các anchors được tạo ra ở giai đoạn đầu có chứa đối tượng hay không, và nếu có, nó sẽ phân loại loại đối tượng đó. Cụ thể là xác định class và offset[8] của bounding box.

Trong bài luận này, tập trung vào lý thuyết về mô hình Faster R-CNN[23].

[7] Anchors là các bounding boxes cố định với các kích thước và tỷ lệ khác nhau.
[8] Offset là giá trị điều chỉnh để dự đoán vị trí và kích thước chính xác của đối tượng so với các anchors ban đầu.

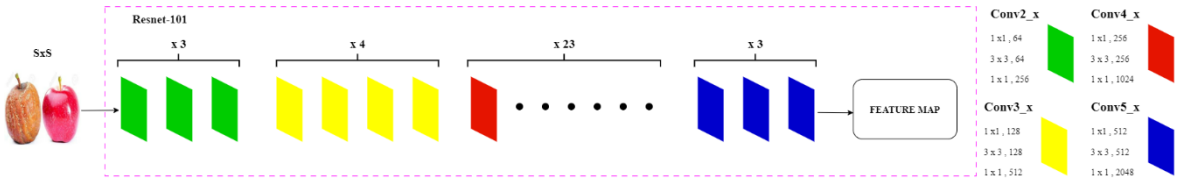


Hình 3. 7. Phát hiện đối tượng với Faster R-CNN

Khác với R-CNN sử dụng thuật toán Selective Search[27] để tạo ra các region proposals, dẫn đến tốc độ xử lý chậm, và Fast R-CNN cải tiến bằng cách sử dụng RoI Pooling nhưng vẫn phụ thuộc vào Selective Search, Faster R-CNN đã loại bỏ sự phụ thuộc này bằng cách sử dụng Region Proposal Network (RPN)[25], giúp tạo ra các region proposals nhanh chóng và hiệu quả hơn.

Hình 3.7 mô tả quy trình hoạt động của Faster R-CNN bao gồm các bước chính sau:

- *CNN:* Toàn bộ ảnh đầu vào được truyền qua một mạng CNN đã được huấn luyện trước trên tập dữ liệu lớn để tạo ra một feature map duy nhất. Trong nghiên cứu này, sử dụng ResNet101 (hình 3.8).



Hình 3. 8. Sơ đồ ResNet-101

- *RPN*: Một mạng nơ-ron nhỏ được áp dụng trực tiếp trên feature map để tạo ra các region proposals. RPN hoạt động bằng cách trượt một cửa sổ trên feature map[9] và sử dụng các anchor boxes tại mỗi vị trí cửa sổ. Mạng RPN sau đó dự đoán liệu mỗi anchor box có chứa đối tượng hay không (classification) và điều chỉnh bounding box để phù hợp hơn với đối tượng thực sự (regression). Để lọc ra các region proposals tốt nhất từ những kết quả dự đoán của RPN, Faster R-CNN sử dụng kỹ thuật NMS, loại bỏ các region proposals trùng lặp và chỉ giữ lại những proposals có điểm số cao nhất, đảm bảo rằng mỗi đối tượng chỉ được đề xuất một lần.
- *RoI Pooling*: Các region proposals được tạo ra từ RPN sau khi qua NMS sẽ được ánh xạ lên feature map và được chuẩn hóa về cùng một kích thước cố định bằng RoI Pooling. Tại đây, lớp Region of Interest Pooling (RoI Pooling) sẽ chia các region proposals thành các ô nhỏ hơn, và thực hiện phương pháp pooling (thường là max pooling) trên mỗi ô để chuyển đổi kích thước của các region proposals về một kích thước cố định.
- *Fully Connected Layers*: Các vùng có kích thước đồng nhất sau RoI Pooling được đưa qua các lớp fully connected để tạo ra các vector đặc trưng cố định. Các vector đặc trưng này được sử dụng để phân loại (classification) và dự đoán bounding box (regression). Một lớp softmax được sử dụng để phân loại các đối tượng trong region proposals và một lớp hồi quy (regression) để điều chỉnh các bounding boxes.

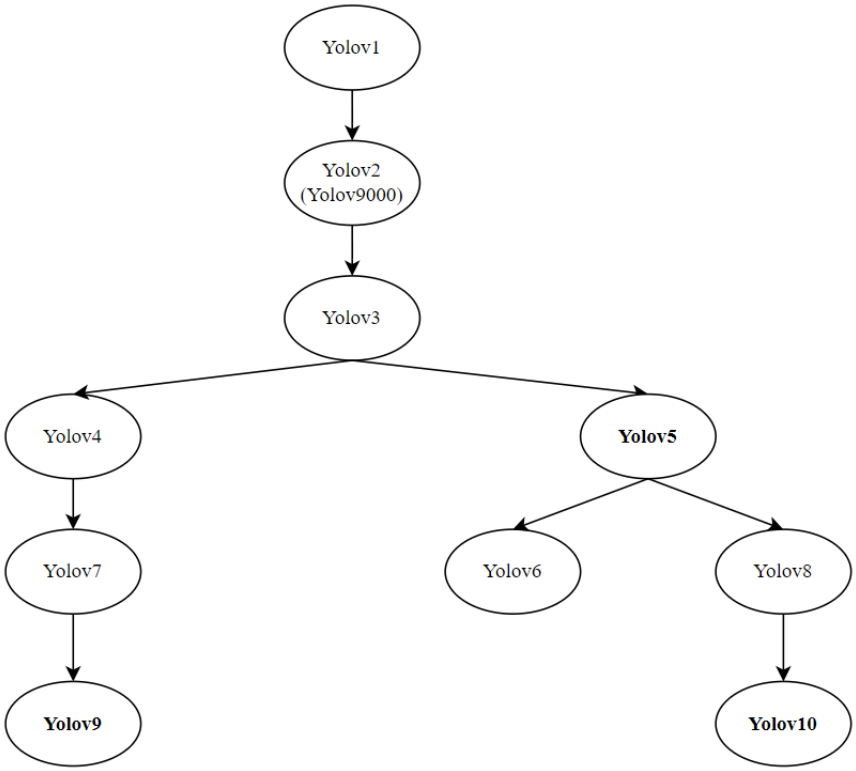
3.3.3. Phát hiện đối tượng một giai đoạn (One-stage)

Cách tiếp cận one-stage tập trung vào việc tối ưu hóa tốc độ xử lý, cho phép thực hiện OD trong thời gian thực một cách hiệu quả. Phương pháp one-stage OD, đúng như tên gọi, thực hiện dự đoán bounding box và phân loại đối tượng trong một bước duy nhất, bỏ qua bước tạo vùng đề xuất trung gian. Cụ thể, YOLO chia ảnh đầu vào thành một lưới SxS, trong đó mỗi ô lưới chịu trách nhiệm dự đoán B bounding boxes và C nhãn lớp cho các đối tượng có trung tâm nằm trong ô đó. Mô hình này

[9] Feature map là biểu diễn không gian của các đặc trưng được phát hiện bởi các lớp trong mạng nơ-ron.

sử dụng mạng nơ-ron sâu để trích xuất đặc trưng từ ảnh và trực tiếp dự đoán bounding boxes và nhãn lớp từ ảnh đầu vào. Việc thực hiện tất cả các dự đoán này trong một lần giúp cải thiện đáng kể tốc độ xử lý so với các phương pháp hai giai đoạn.

Một thuật toán tiêu biểu cho phương pháp one-stage là họ mô hình YOLO. Bài luận này sẽ tập trung vào lý thuyết của các phiên bản YOLOv5[26], YOLOv9[27], và YOLOv10[28], cùng với việc phân tích các ưu điểm của chúng.



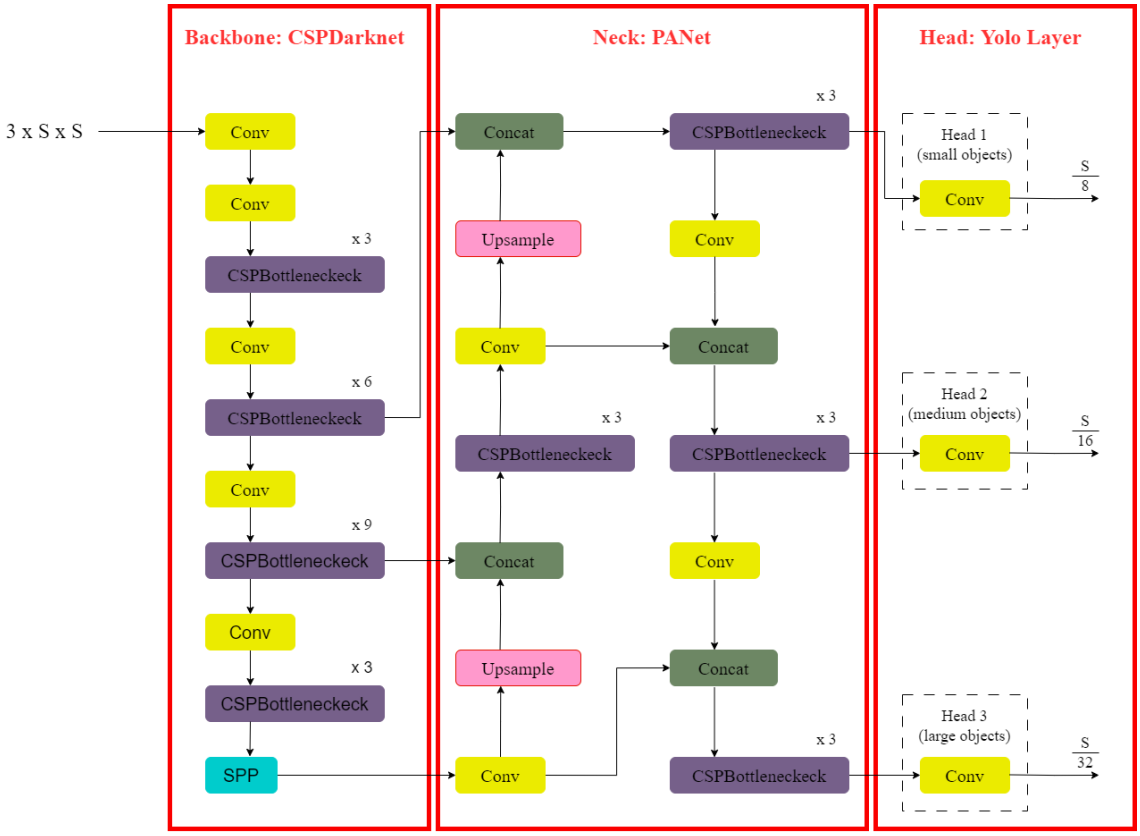
Hình 3. 9. Sơ đồ phát triển các phiên bản YOLO

3.3.3.1. Mô hình mạng YOLOv5

Tổng quan: Được phát triển bởi công ty Ultralytics và được cộng đồng sử dụng rộng rãi từ năm 2020.

Kiến trúc mạng (hình 3.10): Dựa trên CNN với ba phần chính: Backbone, Neck, và Head.

- *Backbone*: CSP-Darknet53, đây là phần chính của mô hình, dùng để trích xuất các đặc trưng từ ảnh đầu vào.
- *Neck*: PANet**,** được sử dụng để tổng hợp các đặc trưng từ nhiều mức độ khác nhau của Backbone**,** giúp cải thiện hiệu suất tổng hợp đặc trưng và tăng khả năng phát hiện đối tượng ở nhiều kích thước khác nhau.
- *Head*: Sử dụng các lớp tích chập để dự đoán bounding box, confidence score, và xác suất lớp.



Hình 3. 10. Kiến trúc của Yolov5

Quá trình hoạt động: YOLOv5 chia hình ảnh đầu vào thành một lưới và dự đoán các bounding box và xác suất lớp cho từng ô trong lưới. Mỗi ô lưới sẽ dự đoán một số lượng cố định các bounding box. Các bounding box này được đánh giá bởi các confidence scores để xác định xem chúng có chứa đối tượng hay không.

Hàm mất mát: Sử dụng một hàm mất mát tổng hợp từ ba thành phần chính:

$$\text{loss} = l_{\{box\}} + l_{\{cls\}} + l_{\{obj\}}$$

- Bounding Box Regression Loss: Đo lường sự khác biệt giữa vị trí và kích thước của bbox dự đoán và bbox thực tế.

$$l_{\{box\}} = \lambda_{ccoord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbf{1}_{ij}^{obj} \left[\left(x - \hat{x_i}\right)^2 + \left[(y - \hat{y_i})^2\right]\right.$$

- Class Prediction Loss: Đo lường sự khác biệt giữa các lớp dự đoán và ground truth

$$l_{\{cls\}} = \sum_{i=0}^{S^2} 1_i^{obj} \sum_{c \in classes} (p_i(c) - \hat{p_i}(c))^2$$

- Objectness Loss: Đo lường sự khác biệt giữa confidence score dự đoán và ground truth.

$$l_{\{obj\}} = \lambda_{obj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} (C_i - \hat{C_i})^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} (C_i - \hat{C_i})^2$$

SSS: Kích thước của lưới.

BBB: Số lượng hộp giới hạn được dự đoán cho mỗi ô lưới.

CCC: Xác suất có đối tượng trong hộp giới hạn.

1_{ij}^{obj}: Chỉ báo có đối tượng trong ô lưới thứ i và hộp giới hạn thứ j.

$\lambda_{coord}, \lambda_{obj}, \lambda_{noobj}$: Trọng số cho các thành phần loss.

Đặc điểm:

- Hiệu suất cao: YOLOv5 được tối ưu hóa để hoạt động hiệu quả trên cả CPU và GPU, giúp dễ dàng triển khai trên nhiều loại phần cứng khác nhau.
- Dễ sử dụng: Cung cấp các tập lệnh huấn luyện và dự đoán đơn giản, dễ hiểu, kèm theo tài liệu hướng dẫn chi tiết.

- Chất lượng dự đoán cao: Cung cấp các dự đoán chính xác với độ trễ thấp, phù hợp cho các ứng dụng thời gian thực.
- Khả năng tùy chỉnh: Hỗ trợ nhiều tùy chọn để điều chỉnh các tham số huấn luyện và dự đoán, giúp người dùng dễ dàng tinh chỉnh mô hình theo nhu cầu cụ thể.

Những cải tiến của YOLOv5 so với các phiên bản trước:

- Triển khai trong PyTorch: YOLOv5 được triển khai trong PyTorch, tận dụng hệ sinh thái linh hoạt và dễ triển khai của PyTorch.
- Tối ưu hóa tốc độ và độ chính xác: Tối ưu hóa tốc độ và độ chính xác, lý tưởng cho các ứng dụng thời gian thực.
- Hỗ trợ nhiều cài đặt mới: Hỗ trợ AutoAnchor, Mosaic augmentation, CIoU loss, và sử dụng tệp YAML thay cho tệp CFG để cấu hình mô hình.
- Hỗ trợ nhiều phiên bản: Cung cấp các phiên bản YOLOv5s, YOLOv5m, YOLOv5l, và YOLOv5x, tối ưu hóa cho nhiều ứng dụng khác nhau.

3.3.3.2. Mô hình mạng YOLOv9

Tổng quan: Là phiên bản tiên tiến gần như là mới nhất trong dòng mô hình YOLO hiện tại, tập trung vào việc cải thiện độ chính xác và tốc độ phát hiện đối tượng, đồng thời duy trì tính hiệu quả và dễ sử dụng.

Điểm mới: Sử dụng hai kỹ thuật mới là PGI và GELAN.

- ***PGI***: Một kỹ thuật mới giúp bảo toàn dữ liệu thiết yếu qua các lớp mạng sâu. Khi dữ liệu đầu vào đi qua nhiều lớp của mạng, mất mát thông tin là điều không thể tránh khỏi, điều này ảnh hưởng đến quá trình cập nhật trọng số và làm giảm hiệu suất của mô hình. PGI giải quyết vấn đề này bằng cách cấu trúc thành ba thành phần chính, mỗi thành phần có một nhiệm vụ riêng biệt nhưng hoạt động song song và hỗ trợ lẫn nhau. Các thành phần chính bao gồm:

- *Main Branch*: Nhánh chính xử lý dữ liệu đầu vào thông qua các bước pooling và unpooling để điều chỉnh không gian đặc trưng và đưa ra dự đoán.
- *Auxiliary Reversible Branch*: Nhánh phụ này được thiết kế hoạt động song song với nhánh chính, giúp bảo toàn thông tin và cấu trúc dữ liệu để giảm thiểu mất mát trong quá trình xử lý, từ đó cải thiện độ chính xác của mô hình.
- *Multi-level Auxiliary Information*: Cung cấp các điều kiện hỗ trợ thông tin đa cấp cho nhánh chính, giúp điều chỉnh và tinh chỉnh học tập dựa trên nhiều mức độ thông tin khác nhau.

- ***GELAN***: Một bước đổi mới trong kiến trúc mạng nơ-ron, mở rộng khả năng của ELAN bằng cách tích hợp với cách tiếp cận của CSPNet. Điểm đặc biệt của GELAN là tính linh hoạt cao, cho phép sử dụng bất kỳ khối tính toán nào, không chỉ giới hạn ở lớp tích chập. Kiến trúc này hỗ trợ nhiều loại khối tính toán khác nhau, tăng cường khả năng tùy biến và hiệu quả của mạng thông qua cấu trúc chia nhỏ và tái kết hợp các đặc trưng. Nhờ đó, GELAN không chỉ giảm bớt tải tính toán mà còn cải thiện đáng kể độ chính xác, đáp ứng nhu cầu hiệu suất cao trong các ứng dụng thực tế.

Điểm nổi bật:

- *Hiệu suất cao*: Tốc độ và độ chính xác tối ưu, phù hợp ứng dụng thời gian thực.
- *Cải tiến về thông tin gradient*: PGI giảm mất mát thông tin, tối ưu hóa quá trình học.
- *Cấu trúc mạng linh hoạt*: GELAN tăng cường phát hiện đối tượng, cải thiện hiệu suất tính toán.
- *Độ chính xác cao*: Thuật toán và cấu trúc cải tiến nâng cao độ chính xác.
- *Cần thêm thử nghiệm*: Dù có nhiều tiềm năng, YOLOv9 vẫn cần thêm các thử nghiệm và đánh giá để khẳng định vị trí của nó trong lĩnh vực phát hiện đối tượng.

3.3.3.3. Mô hình mạng YOLOv10

Tổng quan: YOLOv10 là phiên bản mới nhất trong dòng mô hình YOLO, được nhóm nghiên cứu tại Đại học Thanh Hoa phát triển. Đặt nghi vấn về sự tối ưu trong việc phụ thuộc vào kỹ thuật hậu xử lý NMS và cách thiết kế mô hình của các phiên bản YOLO trước đó.

Điểm mới:

- *Huấn luyện không cần NMS (NMS-free Training):* Cài đặt một chiến lược huấn luyện mới là sự kết hợp của one-to-one assignment và one-to-many assignment, mang tên "Dual label assignments", kết hợp được ưu điểm và khắc phục nhược điểm của cả hai phương pháp, làm tăng cường độ chính xác trong khi giảm độ trễ suy luận.
  - Trong giai đoạn huấn luyện, sử dụng cả hai phương pháp để mô hình học tốt nhất.
  - Trong giai đoạn suy luận, chỉ dùng one-to-one để tránh xử lý NMS.
  - Đánh giá dự đoán: Sử dụng "Consistent Matching Metric" để đảm bảo dự đoán phù hợp với đối tượng thực tế, duy trì sự nhất quán và chất lượng cao trong suốt quá trình huấn luyện.
- Sử dụng chiến lược "Holistic efficiency-accuracy driven model design" với các thành phần như sau:
  - *Lightweight classification Head:* Sử dụng hai lớp convolution loại depth-wise và point-wise để giảm số lượng tham số và tăng hiệu quả xử lý, giảm bớt gánh nặng cho nhánh phân loại mà vẫn đảm bảo độ chính xác.
  - *Spatial-Channel decoupled Downsampling*: Áp dụng kết hợp giữa convolution loại point-wise và depth-wise để thay thế phương pháp downsampling truyền thống, giảm chi phí tính toán và bảo toàn thông tin quan trọng.
  - *Rank-guided block design*: Sử dụng Compact Inverted Block (CIB) thay thế cho các block cơ bản trong các stage cuối cùng để giảm thiểu sự dư thừa tham số và tối ưu hóa chi phí tính toán.
  - *Large-kernel convolution và Partial Self-Attention (PSA):* Mở rộng receptive field và cải thiện khả năng phát hiện bằng cách tăng kích thước

kernel trong convolution và áp dụng PSA để tận dụng sức mạnh của self-attention, giúp cải thiện độ chính xác với chi phí tính toán thấp.

Những cải tiến trên giúp mô hình trở thành công cụ hữu ích và hiệu quả trong nhiều ứng dụng thực tiễn. Cũng như với YOLOv9, YOLOv10 vẫn còn khá mới. Mặc dù kết quả ban đầu đầy hứa hẹn nhưng vẫn cần thử nghiệm và đánh giá thêm.

3.4. Xử lý dữ liệu thời gian thực

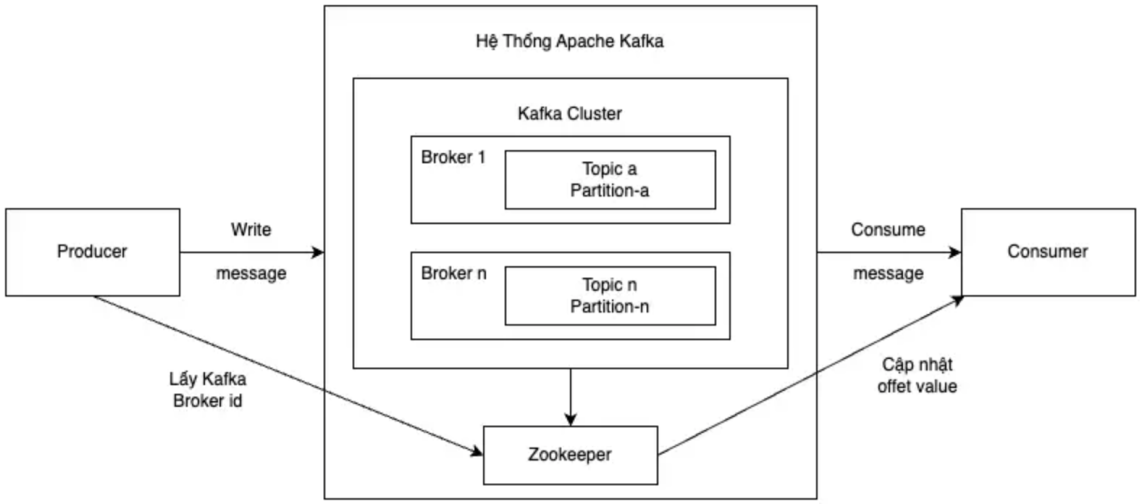
3.4.1. Apache Kafka

Trong bối cảnh công nghệ hiện đại, việc xử lý và phân tích thời gian thực trở nên quan trọng ở các ngành công nghiệp, đặc biệt là nông nghiệp và công nghệ thực phẩm. Big Data đã trở thành cụm từ không quá xa lạ và nó tạo ra những sáng kiến hữu ích cho việc giám sát và quản lý chất lượng sản phẩm nông nghiệp, từ quá trình thu hoạch đến khi sản phẩm đến tay người tiêu dùng. Các thiết bị IoT như cảm biến, máy ảnh ngày càng được triển khai rộng rãi, giúp thu thập một lượng lớn dữ liệu về tình trạng nông sản, và chất lượng sản phẩm. Tuy nhiên, điều này cũng đặt ra những thách thức về việc xử lý một lượng lớn dữ liệu phát sinh liên tục từ các thiết bị giám sát, cảm biến.

Với nền nông nghiệp hiện đại ngày nay, thông tin liên quan đến thời gian thực có thể cung cấp được những tri thức quý giá về các yếu tố ảnh hưởng đến cây trồng và chất lượng sản phẩm. Ví dụ, dữ liệu về nhiệt độ, độ ẩm, ánh sáng, và sự hiện diện của sâu bệnh có thể được thu thập liên tục để đưa ra các giải pháp kịp thời giúp tối ưu hóa quy trình sản xuất và giảm thiểu rủi ro. Tuy nhiên phải đảm bảo được rằng các hệ thống phải có khả năng phân tích dữ liệu và xử lí ngay khi nó được thu thập, bảo đảm rằng các thông tin quan trọng không bị bỏ lỡ và có thể được sử dụng ngay lập tức. Một trong những thách thức lớn nhất là khả năng xử lý và quản lý khối lượng lớn dữ liệu này một cách hiệu quả và kịp thời. Các hệ thống truyền thống thường gặp khó khăn khi phải xử lý dữ liệu liên tục từ các nguồn đa dạng, đồng thời đảm bảo độ chính xác và độ tin cậy của thông tin. Việc tích hợp Apache Kafka vào

mô hình giám sát theo thời gian thực là một giải pháp tối ưu và hiệu quả để giải quyết những thách thức này.

Apache Kafka là hệ thống dữ liệu phân tán, tối ưu cho việc nạp và xử lý dữ liệu theo thời gian thực. Dữ liệu dòng (data streaming) là dữ liệu được tạo ra một cách liên tục từ hàng ngàn nguồn dữ liệu không giống nhau, chẳng hạn như cảm biến IoT, mobile, ứng dụng web. Những nguồn dữ liệu này thường gửi các bản ghi dữ liệu đồng thời, tạo ra dòng dữ liệu liên tục và phong phú. Nền tảng xử lý dữ liệu dòng phải có khả năng xử lý lượng dữ liệu lớn này một cách tuần tự và gia tăng, đảm bảo dữ liệu được xử lý ngay khi nó được thu thập. Kafka bao gồm thành phần chính như broker, Zookeeper, topics, partitions, producers, và consumers. Các broker lưu trữ và quản lý dữ liệu, trong khi Zookeeper duy trì cấu hình và thực hiện chức năng đồng bộ hóa. Dữ liệu được tổ chức thành các topics, chia nhỏ thành các partitions để phân phối tải công việc. Producers gửi dữ liệu vào các topics và consumers đọc dữ liệu từ đó, thường thông qua các consumer groups để xử lý dữ liệu đồng thời. Ngoài ra Kafka còn giúp sao chép dữ liệu để bảo đảm được tính sẵn sàng và độ tin cậy cao và thuận tiện trong việc tích hợp với các công nghệ Big Data khác[29]. Với khả năng mở rộng, hiệu suất cao và độ tin cậy, Kafka là giải pháp lý tưởng cho việc xử lý dữ liệu thời gian thực trong các hệ thống giám sát hiện đại và được minh họa như ở Hình 3.11.



Hình 3. 11. Sơ đồ kiến trúc của Apache Kafka

Kafka cung cấp ba chức năng chính giúp nhà phát triển có thể đảm bảo việc quản lý và xử lý dữ liệu dòng hiệu quả:

- *Đăng ký và theo dõi các luồng bản ghi*: Kafka hỗ trợ đăng ký và theo dõi các luồng dữ liệu đa dạng. Người dùng có thể tạo ra các chủ đề (topics) để quản lý và tổ chức dữ liệu, đồng thời sử dụng các producers để gửi dữ liệu vào các chủ đề này và các consumers để lấy dữ liệu ra từ đó. Việc này giúp đảm bảo dữ liệu được luân chuyển liên tục và có thể truy cập bởi các ứng dụng và dịch vụ khác nhau.
- *Lưu trữ các luồng bản ghi*: Kafka tạo ra giúp lưu trữ một cách hiệu quả dữ liệu theo thứ tự mà chúng được tạo ra. Điều này có nghĩa là các bản ghi dữ liệu được ghi vào Kafka theo trình tự thời gian, hỗ trợ người dùng có thể truy xuất và xử lý dữ liệu theo đúng thứ tự mà nó được tạo ra. Kafka sử dụng một mô hình nhật ký (log) để lưu trữ dữ liệu, giúp tối ưu hóa hiệu suất ghi và đọc dữ liệu, đồng thời cung cấp khả năng phục hồi dữ liệu cao.
- *Xử lý các luồng bản ghi trong thời gian thực*: Các công cụ được tích hợp giúp xử lý dòng như Apache Storm, Apache Flink và Apache Spark Streaming. Người dùng có thể thiết lập các luồng xử lý dữ liệu để xử lý, biến đổi, và phân tích dữ liệu ngay khi nó được nạp vào Kafka. Điều này giúp các ứng dụng có thể phản hồi nhanh chóng với các sự kiện và thay đổi trong dữ liệu, từ đó cải thiện hiệu suất và độ chính xác của hệ thống.

Kafka hỗ trợ giúp triển khai các đường dẫn dữ liệu dòng thời gian thực và các ứng dụng thích nghi với các luồng dữ liệu. Đường dẫn dữ liệu này có thể chuyển dữ liệu từ các nguồn phát sinh đến các hệ thống đích như cơ sở dữ liệu, kho dữ liệu, hoặc các công cụ phân tích. Kafka kết hợp ba khía cạnh quan trọng là messaging, storage, stream processing để cho phép lưu trữ và phân tích.

Nhìn chung, Apache Kafka là một công cụ mạnh và rất linh hoạt, nó giúp các tổ chức quản lý và khai thác dữ liệu dòng thời gian thực một cách tối ưu hơn. Với Kafka, chúng ta có thể xây dựng các hệ thống dữ liệu hiện đại, đáp ứng nhanh chóng và chính xác với các thay đổi và sự kiện trong môi trường hoạt động. [29]

Hệ thống giám sát trái cây và rau củ được triển khai với việc tích hợp Apache Kafka đóng vai trò then chốt. Apache Kafka cho phép thu thập dữ liệu video từ các camera giám sát, cảm biến tại các địa điểm đa dạng, đảm bảo được độ trễ được giảm thiểu tối đa và giúp gửi thông tin ổn định và nhanh chóng hơn tránh mất mát dữ liệu. Với khả năng xử lý hàng triệu tin nhắn mỗi giây, Kafka hỗ trợ xử lý đồng thời các khung hình video từ nhiều camera, phân tán tải công việc và đảm bảo được hiệu suất thực thi ngay khi khối lượng của dữ liệu có tăng lên đột ngột. Thêm vào đó, nghiên cứu mở ra tiềm năng tích hợp Kafka với Apache Spark để thực hiện các phân tích phức tạp và tối ưu hóa chuỗi cung ứng. Sự kết hợp này không chỉ đảm bảo độ tin cậy cao và khả năng phục hồi mạnh mẽ mà còn bảo vệ dữ liệu và đảm bảo rằng được tính sẵn sàng của thông tin ngay khi cần cũng đẩy mạnh hiệu quả sản xuất và quản lý nông sản.

3.4.2. Apache Spark

Apache Spark là một hệ thống xử lý dữ liệu phân tán mã nguồn mở, được thiết kế để xử lý các khối lượng công việc dữ liệu lớn. Spark tận dụng khả năng lưu trữ tạm thời trong bộ nhớ và thực hiện truy vấn tối ưu hóa, giúp phân tích dữ liệu nhanh chóng với bất kỳ kích thước nào. Hệ thống này cung cấp các API cho nhiều ngôn ngữ lập trình như Java, Scala, Python và R, đồng thời hỗ trợ tái sử dụng mã cho nhiều loại công việc khác nhau, bao gồm xử lý dữ liệu theo lô, truy vấn tương tác, phân tích thời gian thực, học máy và xử lý đồ thị.Với khả năng xử lý dữ liệu theo mô hình phân tán, Spark giúp hệ thống dễ dàng mở rộng quy mô xử lý khi khối lượng dữ liệu tăng lên. Điều này đặc biệt quan trọng đối với các trang trại lớn hoặc các cơ sở sản xuất có nhiều điểm giám sát. Khi tích hợp với Kafka, Spark có thể xử lý dữ liệu từ đa dạng các nguồn một cách đồng thời và hiệu quả. Spark còn cung

cấp nhiều thư viện phân tích dữ liệu mạnh mẽ như MLlib cho machine learning, GraphX cho phân tích đồ thị, và Spark SQL cho xử lý dữ liệu quan hệ, cho phép hệ thống giám sát thực hiện các phân tích phức tạp từ dự đoán sản lượng đến phát hiện các bất thường trong chất lượng sản phẩm.

Trong nghiên cứu "***Hệ thống giám sát trái cây và rau củ thời gian thực***" các thành phần Apache Spark được sử dụng bao gồm: Spark Core và Spark Streaming.

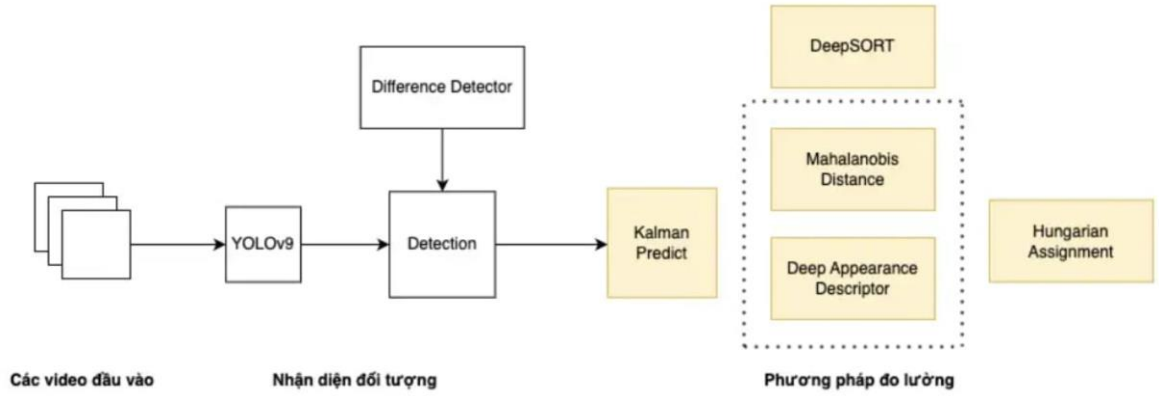
- *Spark Core* là thành phần cơ bản và trung tâm của Apache Spark. Spark Core cung cấp các chức năng cơ bản cần thiết để thực hiện các tác vụ xử lý dữ liệu phân tán, bao gồm quản lý bộ nhớ và lưu trữ, lập lịch công việc, khả năng chịu lỗi, và sử dụng RDD (Resilient Distributed Datasets). Quản lý bộ nhớ và lưu trữ giúp Spark Core quản lý bộ nhớ và sử dụng bộ nhớ hiệu quả để lưu trữ dữ liệu tạm thời và tăng tốc độ xử lý. Lập lịch công việc cho phép Spark Core lập lịch và phân phối các tác vụ xử lý dữ liệu đến các nút trong cụm Spark, đảm bảo các tác vụ được thực hiện đồng thời và tối ưu hóa hiệu suất. Khả năng chịu lỗi của Spark Core cung cấp các cơ chế phục hồi từ lỗi, đảm bảo rằng dữ liệu và các tác vụ xử lý không bị mất mát khi có sự cố xảy đến. RDD là cấu trúc dữ liệu chính được Spark Core sử dụng để lưu trữ và xử lý dữ liệu phân tán, cung cấp các transformations và actions giúp dữ liệu được hiệu quả và linh hoạt hơn trong quá trình xử lí [30].
- Một khái niệm cần lưu tâm hơn đó là *Spark Streaming*, nó cũng hộ trợ xử lý dữ liệu theo luồng thời gian thực[31]. Khi kết hợp với Kafka, Spark Streaming có thể nhận và xử lý dữ liệu video liên tục từ các camera giám sát, giúp hệ thống giám sát cập nhật thông tin về số lượng và chất lượng sản phẩm một cách liên tục và chính xác. Ngoài ra, Spark còn có thể kết hợp với nhiều công nghệ Big Data khác như Hadoop, Cassandra, và HBase, giúp hệ thống giám sát phát huy được sức mạnh của các công cụ phân tích hiện đại để nhanh chóng, kịp thời đưa ra giải pháp dựa trên cốt lõi của dữ liệu.

Các công việc trong Spark có thể được kiểm tra và khởi động lại nếu gặp sự cố, giúp dữ liệu được an toàn và giảm thiểu được gián đoạn trong quá trình phân tích cũng như đưa ra quyết định. Trong bối cảnh xử lý dữ liệu video từ các video thời gian thực, Apache Spark có thể kết hợp với Kafka để liên tục nhận và xử lý các khung hình video. Dữ liệu từ các khung hình này sẽ được xử lý bằng các mô hình học sâu như YOLO để nhận dạng các đối tượng. Sau đó, DeepSORT được đưa vào để theo dõi các đối tượng này qua các khung hình. Các thông số về số lượng và chất lượng sản phẩm sẽ được cập nhật theo thời gian thực, giúp nông dân và các nhà quản lý có những giải pháp kịp thời và có tính tin cậy cao nhằm tối ưu hóa quy trình sản xuất.

3.4.3. Theo dõi đối tượng DeepSORT

DeepSORT, viết tắt của Deep Simple Online and Realtime Tracking, là thuật toán theo dõi đối tượng thời gian thực sử dụng học sâu để cải thiện độ chính xác và độ tin cậy của việc giám sát đối tượng trong video. Việc tích hợp DeepSORT vào dự án nhằm hỗ trợ thêm nhiều lợi ích thực tiễn, đặc biệt là trong việc theo dõi và nhận dạng đối tượng một cách liên tục và cần độ tin cậy cao[32].

Thuật toán này hoạt động bằng cách sử dụng mạng nơ-ron thần kinh để trích xuất các đặc trưng từ các đối tượng phát hiện trong video. Bao gồm hình dạng, màu sắc đặc trưng, và những thông tin trực quan khác. Sau đó, thuật toán sử dụng các đặc trưng này để gán định danh cho các đối tượng qua các khung hình khác nhau, giúp hệ thống không chỉ nhận diện được các đối tượng (trái cây, rau củ) mà còn theo dõi được chúng theo thời gian.



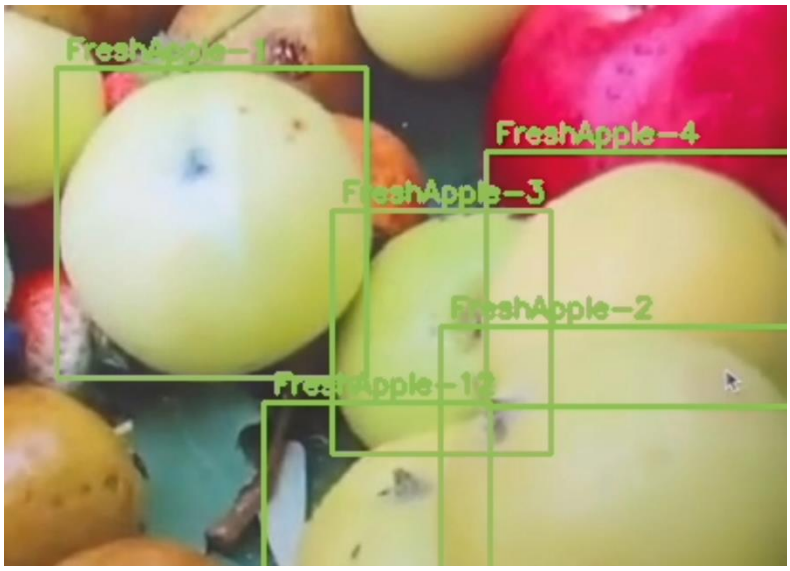
Hình 3. 12. Sơ đồ cấu trúc thuật toán DeepSORT

Việc kết hợp YOLO cho nhận diện đối tượng với DeepSORT tạo nên một công cụ mạnh mẽ có khả năng vừa phát hiện vừa theo dõi đối tượng một cách liên tục. YOLO bảo đảm được việc tìm ra đối tượng nhanh chóng trong mỗi khung hình, trong khi DeepSORT đảm bảo theo dõi liên tục các đối tượng này qua các khung hình liên tiếp.

Một trong những điểm nổi bật của DeepSORT là khả năng theo dõi chính xác các đối tượng ngay cả khi chúng di chuyển hoặc thay đổi hình dạng. DeepSORT sử dụng Kalman Filter để phỏng đoán vị trí tiếp theo của các đối tượng dựa trên các quan sát trước đó. Kalman Filter dùng để dự đoán trạng thái của một hệ thống động qua thời gian, và đặc biệt giúp ích trong việc theo dõi các đối tượng di động[33]. Sau khi Kalman Filter dự đoán vị trí, Hungarian Algorithm sẽ giải quyết bài toán gán định danh cho các đối tượng trong từng khung hình được minh họa cụ thể như ở Hình 3.12. Hungarian Algorithm giúp việc gán định danh một cách tối ưu hóa cho các đối tượng bằng cách tìm ra cách gán kết hợp tốt nhất giữa các đối tượng được phát hiện và các đối tượng đang được theo dõi, giảm thiểu nhầm lẫn và mất theo dõi (tracking loss) được mô tả ở Hình 3.13.

Một lợi ích khác của DeepSORT là giảm thiểu các vấn đề như mất theo dõi và nhầm lẫn định danh (ID switching). Các đặc trưng trích xuất từ mạng nơ-ron giúp DeepSORT có thể phân biệt các đối tượng dựa trên hình dạng và màu sắc, ngay cả khi có nhiều đối tượng tương tự nhau trong cùng một khung hình. Điều này giúp mô hình có thể theo dõi chính xác các đối tượng qua nhiều khung hình, đảm bảo rằng

thông tin về chất lượng cũng như số lượng của sản phẩm được theo dõi chính xác hơn.



Hình 3. 13. Minh họa việc xác định định danh riêng cho mỗi đối tượng

DeepSORT cũng có khả năng mở rộng tốt, xử lý nhiều đối tượng cùng lúc trong môi trường phức tạp mà không làm giảm hiệu suất. Điều này có ý nghĩa lớn với các hệ thống giám sát lớn, nơi có nhiều điểm giám sát và khối lượng dữ liệu lớn cần được xử lý đồng thời. Với DeepSORT, hệ thống giám sát trái cây và rau củ có thể hoạt động hiệu quả trong trường hợp khối lượng dữ liệu tăng đột biến.

Tóm lại, việc sử dụng DeepSORT vào mô hình giám sát trái cây và rau củ góp phần cải thiện độ chính xác và tin cậy của việc theo dõi đối tượng, đồng thời cung cấp các giá trị quan trọng về số lượng và chất lượng sản phẩm theo thời gian thực. DeepSORT không chỉ giúp cải thiện quy trình sản xuất và quản lý trong nông nghiệp mà còn đảm bảo rằng các quyết định được đưa ra dựa trên dữ liệu chính xác và kịp thời. Điều này giúp quy trình sản xuất được cải thiện, giảm thiểu tổn thất góp phần nâng cao hiệu quả và năng suất trong ngành nông nghiệp.

3.4.4. Đếm và giám sát đối tượng

Quá trình tự động đếm số lượng từng loại quả trong mỗi khung hình video từ các camera giám sát bắt đầu bằng việc sử dụng công cụ phát hiện đối tượng YOLO, đã

được huấn luyện trước để nhận diện và phân loại các loại quả phổ biến dựa trên các mẫu khác nhau của từng loại.

Sau khi một khung hình được nhận diện và phân loại, hệ thống sử dụng trình theo vết đối tượng DeepSORT được trình bày trong phần 3.4.3, để theo dõi từng đối tượng quả từ khung hình này sang khung hình khác. DeepSORT sử dụng các đặc trưng như hình dạng và vị trí để duy trì và cập nhật định danh của từng đối tượng trong suốt quá trình di chuyển[34].

Thông qua việc liên kết giữa YOLO và DeepSORT, hệ thống có thể đảm bảo rằng mỗi quả được phát hiện sẽ được theo dõi và đếm một cách chính xác, kể cả chúng có di chuyển hay xuất hiện ở các góc độ khác nhau trên camera[35]. Giải pháp này không chỉ giúp cải thiện năng suất và hiệu quả trong quản lý nông nghiệp mà còn hỗ trợ các quyết định chiến lược dựa trên dữ liệu chính xác và kịp thời về sản lượng và chất lượng của các loại quả trồng trọt. Hơn nữa, hệ thống còn cho phép xử lý đồng thời nhiều loại quả và dữ liệu lớn từ các điểm giám sát khác nhau, đảm bảo tính liên tục và khả năng áp dụng trong các môi trường nông nghiệp phức tạp, hiện đại.

Chương 4. NGHIÊN CỨU THỰC NGHIỆM

4.1. Cài đặt thực nghiệm

4.1.1. Phần ngoại tuyến

Nghiên cứu tập trung triển khai và đánh giá các mô hình phát hiện đối tượng trong cả hai phương pháp là one-stage OD và two-stages OD, được mô tả chi tiết trong mục 3.3, nhằm có cái nhìn toàn diện về bài toán "*phát hiện trái cây và rau củ*" đặt ra trong nghiên cứu này. Các cài đặt và môi trường thực nghiệm được mô tả chi tiết bên dưới.

Dữ liệu (bảng 4.1):

- Gán nhãn: Dữ liệu được gán nhãn thủ công bằng công cụ MakeSense.AI.
- Phân chia dữ liệu: Chia theo tỷ lệ 8:1:1 cho các tập huấn luyện (train), xác thực (validation) và kiểm thử (test).
- Áp dụng các phương pháp tăng cường dữ liệu bao gồm: Phép biến đổi xoay, xoay kết hợp với mosaic và MTDA (được mô tả chi tiết trong mục 3.2.3), các phương pháp như mosaic và MTDA không làm tăng kích thước dữ liệu, nhưng tạo ra các biến thể khác nhau từ dữ liệu hiện có.
- Tổ chức và sắp xếp dữ liệu: Định dạng COCO cho mô hình Faster RCNN, và sử dụng định dạng tiêu chuẩn của YOLO cho các mô hình YOLO.

Tăng cường dữ liệu:

- Sử dụng albumentations[10] để triển khai phương pháp xoay theo ba góc độ: 90, 180, 270.
- Phương pháp mosaic được áp dụng trực tiếp trong quá trình huấn luyện, sử dụng các giá trị trong file hyperameter mặc định của YOLO.
- Phương pháp MTDA với các thông số đầu vào là W1 = 0.3, W2 = 0.3, W3 = 0.4, và kết hợp cuối với mosaic trong mô hình YOLO.

[10] https://albumentations.ai/docs/

- Các phương pháp tăng cường dữ liệu chỉ được áp dụng cho các mô hình YOLO, được thực hiện trên dữ liệu huấn luyện và xác thực.

Bộ dữ liệu Fruit And Vegetable Images Dataset (FVID)				
Loại	Số lượng			Tổng
	Huấn luyện	Xác thực	Kiểm thử	
Dữ liệu ban đầu	3.200	400	400	4.000
Sau khi áp dụng tăng cường dữ liệu	12.800	1.600	400	14.800

Bảng 4. 1. Dữ liệu thực nghiệm

Cấu hình mô hình:

- Mô hình Faster R-CNN: Faster R-CNN sử dụng ResNet-101 làm mạng trích xuất đặc trưng, áp dụng kỹ thuật học chuyển giao (Transfer learning), với bộ trọng số đã được huấn luyện trước trên bộ dữ liệu lớn COCO Dataset.
- Các mô hình YOLO: Sử dụng cấu hình phiên bản YOLOv5-S, YOLOv9-C, YOLOv10-L, các mô hình này sẽ bắt đầu học từ đầu, không sử dụng trọng số được huấn luyện từ trước.

Thông số huấn luyện: Kích thước ảnh đầu vào cho tất cả các mô hình là 640x640 pixels. Cài đặt chạy mô hình Faster RCNN với kích thước batch 2, iterations 20000; và kích thước batch 16, epoch 100 cho các mô hình YOLO.

Thiết lập đánh giá: Sử dụng tập xác thực đánh giá trong quá trình huấn luyện, và tập kiểm thử là dữ liệu ban đầu chưa qua các phép biến đổi/tăng cường, được sử dụng để đánh giá cuối cho các mô hình.

Mô trường thực nghiệm: Nghiên cứu được thực nghiệm trực tuyến với cấu hình môi trường được mô tả trong bảng 4.2, hỗ trợ hiệu quả cho quá trình huấn luyện và kiểm thử với dữ liệu đầu vào ảnh kích thước 640x640.

	Faster RCNN - Resnet101	YOLO (YOLOv5-S, YOLOv9-C, YOLOv10-L)
Môi trường	Google Colab	
Phần cứng	GPU NVIDIA Tesla T4	GPU NVIDIA A100-SXM4-40GB
Phần mềm	Python 3.10.12, Torch 1.5.0, Detectron2, và các thư viện liên quan	Python 3.10.12, Torch 2.3.0+cu121, và các thư viện liên quan

Bảng 4. 2. Thông số cấu hình và môi trường cài đặt cho phần ngoại tuyến

4.1.2. Phần trực tuyến

Trong khuôn khổ khóa luận, ứng dụng nghiên cứu vào thời gian thực được mô tả là một hệ thống bao gồm nhiều bước tích hợp, từ thu thập dữ liệu đến trình bày kết quả cuối cùng. Quá trình bắt đầu với việc thu thập video từ các camera giám sát, sau đó gửi dữ liệu qua Kafka, và cuối cùng là phân tích bằng Spark Streaming. Sử dụng mô hình YOLO được phát triển trong phần ngoại tuyến, để trích xuất đặc trưng và phát hiện tình trạng của trái cây, rau củ. Bên cạnh đó, tích hợp thuật toán DeepSORT để theo dõi và đếm các đối tượng. Kết quả được trực quan hóa và hiển thị qua một giao diện dễ sử dụng. Nghiên cứu được thực hiện trên hai video, sử dụng Apache Kafka và Apache Spark trên macOS với CPU Apple M1. Bảng 4.3 chi tiết cài đặt và môi trường thực nghiệm, trong khi bảng 4.4 mô tả các bước khởi chạy và vận hành hệ thống.

Thành phần	Chi tiết
Hệ điều hành	macOS 14.0 (BuildVersion: 23A344)
Phần cứng	CPU: Apple M1, RAM: 16 GB
Phần mềm & Frameworks	Apache Kafka: phiên bản kafka_2.13-3.5.0 Apache Spark: phiên bản 3.4.0 Môi trường: Python 3.10.9
Cấu hình Kafka Listeners	listeners=PLAINTEXT://localhost:9092 advertised.listeners=PLAINTEXT://localhost:9092

Bảng 4. 3. Thông số cấu hình và môi trường cài đặt cho phần trực tuyến

Thành phần	Chi tiết
Thiết lập môi trường Kafka	- Khởi động Zookeeper: *bin/zookeeper-server-start.sh config/zookeeper.properties* - Khởi động Kafka Server: *bin/kafka-server-start.sh config/server.properties* - Tạo kafka topic phù hợp.
Khởi chạy và điều khiển Producer&Consumer	Tiến hành chạy file producer và consumer song song trên 2 terminal khác nhau và áp dụng "*argparse*" hỗ trợ thay đổi các tham số.

Bảng 4. 4. Thông tin triển khai và sử dụng hệ thống trực tuyến

4.2. Phương pháp và độ đo đánh giá

Precision (Độ chính xác):

$$\text{Precision} = \frac{\text{Số lượng dự đoán đúng (TP)}}{\text{Số lượng dự đoán đúng (TP)} + \text{Số lượng dự đoán sai (FP)}}$$

Recall (Độ phủ):

$$\text{Recall} = \frac{\text{Số lượng dự đoán đúng (TP)}}{\text{Số lượng dự đoán đúng (TP)} + \text{Số lượng bỏ sót (FN)}}$$

Average Precision (AP) là trung bình các giá trị Precision (độ chính xác) tại các điểm khác nhau của đường cong Precision-Recall. Đường cong này được tạo ra bằng cách tính độ chính xác và nhớ cho một loạt các ngưỡng xác suất phân loại khác nhau. Cụ thể, công thức tính AP là:

$$AP = \Sigma_n (R_n - R_{n-1}) P_n$$

Trong đó:

+ P_n là giá trị Precision tại ngưỡng n.

+ R_n là giá trị Recall tại ngưỡng n.

+ n là các ngưỡng khác nhau.

- *IoU (Intersection over Union):* Để xác định một phát hiện là đúng hay sai, AP sử dụng IoU, chỉ số đo mức độ chồng chéo giữa hộp giới hạn dự đoán và hộp giới hạn thật. Một phát hiện được coi là đúng nếu IoU của nó với một hộp giới hạn thật lớn hơn ngưỡng nhất định (thường từ 0.5 đến 0.95).
- *AP Small, AP Medium, AP Large:* Một tiêu chuẩn trong đánh giá mô hình phát hiện đối tượng, phân loại các đối tượng dựa trên kích thước của chúng. Điều này giúp nhận định mô hình hoạt động tốt như thế nào với các đối

tượng ở các kích thước khác nhau, một yếu tố quan trọng trong các ứng dụng thực tế nơi kích thước đối tượng có thể rất đa dạng.

- Small: Diện tích nhỏ hơn 32x32 pixels.
- Medium: Diện tích từ 32x32 đến 96x96 pixels.
- Large: Diện tích lớn hơn 96x96 pixels.

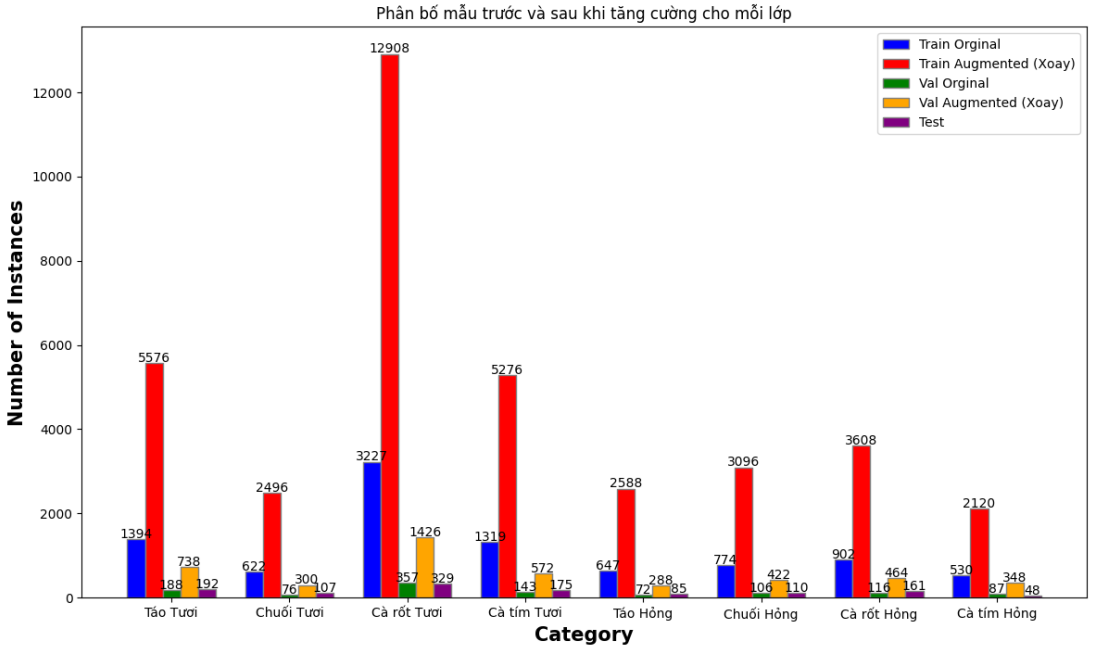
mAP: Tính toán giá trị trung bình của các chỉ số Average Precision (AP) tại nhiều ngưỡng khác nhau. AP là trung bình của precision tại mỗi mức recall từ 0 đến 1.

- mAP 0.5: Là giá trị mAP tại ngưỡng IoU (Intersection over Union) là 0.5. Nghĩa là, một dự đoán được xem là đúng nếu nó chồng khớp với đối tượng thật ít nhất 50%.
- mAP 0.5:0.95: Là giá trị mAP trung bình tại các ngưỡng IoU từ 0.5 đến 0.95.

Average Recall (AR) là độ đo tổng số đối tượng mà mô hình có thể phát hiện được, không kể số lượng phát hiện mà mô hình tạo ra. AR được tính bằng cách đo lường tỷ lệ các đối tượng thực sự được phát hiện so với tổng số đối tượng thực tế, dựa trên các ngưỡng khác nhau của số lượng đối tượng tối đa được phép phát hiện (Max Dets).

Chương 5. KẾT QUẢ VÀ THẢO LUẬN

5.1. Kết quả thực nghiệm



Hình 5. 1. Phân bố mẫu trước và sau khi sử dụng phép biến đổi xoay cho mỗi lớp

Hiệu suất đánh giá chung			Hiệu suất theo loại nhãn		
Độ đo	Dữ liệu		Loại nhãn	AP	
	Xác thực	Kiểm thử		Xác thực	Kiểm thử
AP (IoU=0.50:0.95)	61.33	62.81	Táo Tươi	**82.64**	**84.19**
AP50 (IoU=0.50)	**82.45**	**85.49**	Chuối Tươi	76.81	70.35
AP75 (IoU=0.75)	68.94	70.92	Cà rốt Tươi	50.78	58.51
AP Small	20.33	50.48	Cà tím Tươi	50.86	55.76
AP Medium	40.46	42.69	Táo Hỏng	**78.66**	**79.83**
AP Large	66.71	66.61	Chuối Hỏng	63.07	59.34
AR (Max Dets=100)	71.1	74.0	Cà rốt Hỏng	55.64	51.5
Thời gian huấn luyện: 3.1024 (giờ)			Cà tím Hỏng	32.17	42.97

Bảng 5. 1. Hiệu suất mô hình Faster-RCNN-Resnet-101

Mô hình	Params	GFLOPs	Layers	Kích thước (MB)
YOLOv5-S	7.03M	15.8	157	14.5
YOLOv9-C	50.97M	237.7	724	102.8
YOLOv10-L	25.73M	126.4	461	52.2

Bảng 5. 2. Thông số của các mô hình YOLO

Mô hình	Time (giờ)	Dữ liệu	Inference (ms)	Precision	Recall	mAP 0.5	mAP 0.5:0.95
YOLO v5	1.839	Xác thực	9.4	0.74	0.526	0.605	0.383
		Kiểm thử	12.7	0.647	0.52	0.561	0.347
YOLO v9	2.161	Xác thực	60.1	**0.837**	**0.659**	**0.767**	**0.606**
		Kiểm thử	61.0	**0.83**	**0.637**	**0.751**	**0.59**
YOLO v10	1.31	Xác thực	**4.5**	0.68	0.486	0.558	0.42
		Kiểm thử		0.64	0.446	0.522	0.383

Bảng 5. 3. Hiệu suất các mô hình YOLO trên tập dữ liệu gốc

Mô hình	Time (giờ)	Dữ liệu	Inference (ms)	Precision	Recall	mAP 0.5	mAP 0.5:0.95
YOLO v5	7.81	Xác thực	5.3	0.772	0.707	0.749	0.523
		Kiểm thử	13.1	0.754	0.635	0.675	0.465
YOLO v9	8.476	Xác thực	67.7	**0.846**	**0.752**	**0.827**	**0.667**
		Kiểm thử	72.5	**0.84**	**0.749**	**0.841**	**0.669**
YOLO v10	4.885	Xác thực	**4.1**	0.831	0.672	0.757	0.593
		Kiểm thử		0.812	0.645	0.746	0.571

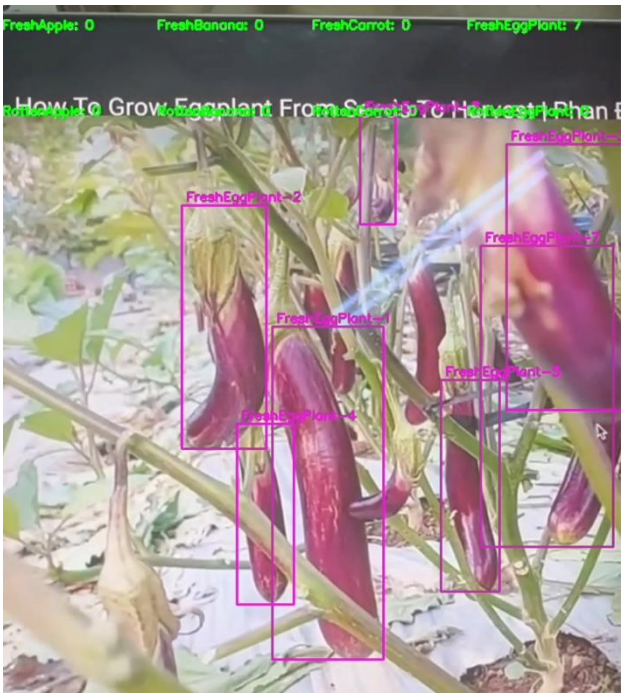
Bảng 5. 4. Hiệu suất các mô hình YOLO trên tập dữ liệu xoay

Mô hình	Time (giờ)	Dữ liệu	Inference (ms)	Precision	Recall	mAP 0.5	mAP 0.5:0.95
YOLO v5	7.9	Xác thực	5.3	0.815	0.789	0.824	0.617
		Kiểm thử	10.8	0.818	0.798	0.853	0.625
YOLO v9	8.494	Xác thực	69.0	0.871	**0.821**	**0.873**	**0.713**
		Kiểm thử	71.5	**0.884**	**0.86**	**0.912**	**0.739**
YOLO v10	4.651	Xác thực	**4.1**	**0.873**	0.81	0.862	0.697
		Kiểm thử		0.855	0.838	0.9	0.721

Bảng 5. 5. Hiệu suất các mô hình YOLO trên tập dữ liệu Xoay + Mosaic

Mô hình	Time (giờ)	Dữ liệu	Inference (ms)	Precision	Recall	mAP 0.5	mAP 0.5:0.95
YOLO v9	8.77	Xác thực	65.9	0.834	0.769	0.839	0.668
		Kiểm thử	64.8	0.854	0.84	0.894	0.712

Bảng 5. 6. Hiệu suất mô hình YOLOv9 trên tập dữ liệu Xoay + MTDA + Mosaic



Hình 5. 2. Minh họa quy trình đếm các đối tượng trên từng khung hình.

Các thông tin về vị trí và số lượng của từng loại quả được tổng hợp và hiển thị trực tiếp trên video, cung cấp cho người dùng một cái nhìn toàn diện và chi tiết về hiệu suất sản xuất và chất lượng sản phẩm trong thời gian thực được minh họa cụ thể trong Hình 5.2, chi tiết trong video[11] được thực hiện với mô hình Yolov9.

5.2. Thảo luận

5.2.1. Thảo luận về dữ liệu

Phương pháp xoay, như được thể hiện qua biểu đồ hình 5.1, đã thành công trong việc mở rộng tập dữ liệu, nhưng cũng làm nổi bật một số vấn đề cần được thảo luận:

- *Tăng số lượng mẫu nhưng chưa đạt cân bằng:* Dù phương pháp xoay đã hiệu quả trong việc tăng số lượng mẫu cho tất cả các lớp, nó vẫn chưa giải quyết được vấn đề cân bằng giữa các lớp. Lớp "Cà rốt Tươi" nhận được sự tăng trưởng mạnh mẽ hơn so với các lớp khác, dẫn đến sự chênh lệch lớn hơn về số lượng mẫu giữa các lớp.
- *Mở ra nhu cầu cho các kỹ thuật phức tạp hơn*: Khám phá việc kết hợp các phương pháp mosaic và MTDA cùng với xoay để xem liệu chúng có giải quyết được vấn đề mất cân bằng hay không.
- *Phản ánh tính khách quan trong đánh giá mô hình*: Bằng việc giữ nguyên tập kiểm tra, nghiên cứu đảm bảo rằng việc đánh giá mô hình dựa trên dữ liệu không qua xử lý tăng cường, phản ánh khả năng hoạt động thực tế của mô hình. Điều này là cần thiết để đánh giá khách quan hiệu quả của mô hình khi áp dụng vào thực tiễn.

5.2.2. Thảo luận về kết quả các mô hình trên từng tập dữ liệu

Kết quả mô hình Faster-RCNN-Resnet-101 trên dữ liệu gốc (bảng 5.1):

Hiệu suất tổng thể:

- Mô hình cho thấy hiệu suất tốt trên cả tập xác định và tập kiểm thử với AP50 đạt trên 85% trên tập kiểm thử, điều này chỉ ra rằng mô hình có khả năng phát hiện đối tượng tốt khi giả định rằng IoU threshold là 0.50.

[11] https://drive.google.com/file/d/1RuX5rMGkYGRCTvodeGYCLVC429TwlmjP/view?usp=sharing

- Tăng nhẹ trong AP từ tập xác thực sang tập kiểm thử (từ 61.328% lên 62.806%) cho thấy mô hình có sự ổn định và khả năng tổng quát hóa tốt.
- Chỉ số AP Small có sự chênh lệch lớn giữa tập dữ liệu xác thực và kiểm thử, với kết quả kiểm thử cao hơn đáng kể (20.33 so với 50.48), cho thấy mô hình có thể còn gặp khó khăn trong việc phát hiện các đối tượng nhỏ. Điều này cần được cải thiện trong các nghiên cứu tiếp theo.

Đánh giá theo loại nhãn:

- Hiệu suất cao: Táo Tươi và Táo Hỏng cho thấy hiệu suất cao nhất trên cả hai tập, cho thấy mô hình hiệu quả trong việc phát hiện và phân loại các loại táo
- Hiệu suất trung bình và thấp: Các loại nhãn khác như Cà rốt Tươi và Cà rốt Hỏng có kết quả thấp hơn, đặc biệt là trên tập dữ liệu kiểm thử, cho thấy mô hình có thể cần cải thiện khả năng nhận dạng các loại nhãn này.

Phân tích ảnh hưởng của tập dữ liệu lên hiệu năng của mô hình:

- Số lượng mẫu lớn không phải lúc nào cũng đảm bảo hiệu suất nhận dạng cao. Đặc điểm hình ảnh và sự đa dạng của các mẫu cũng đóng vai trò quan trọng.
- Các loại nhãn có đặc điểm hình ảnh dễ phân biệt và ít biến đổi thường đạt hiệu suất cao hơn, như trường hợp của Táo Tươi và Táo Hỏng.
- Các loại nhãn có sự đa dạng lớn về hình dạng, màu sắc và khả năng che khuất lẫn nhau trong hình ảnh có thể làm giảm hiệu suất nhận dạng, như trường hợp của Cà rốt Tươi và Cà rốt Hỏng.

Cấu trúc và kích thước của các mô hình YOLO (bảng 5.2):

- YOLOv5: Với YOLOv5-S là mô hình nhẹ và nhỏ nhất với (7.03M params, 15.8 GFLOPs, 157 layers, 14.5 MB), rất phù hợp cho các ứng dụng thời gian thực trên các thiết bị có tài nguyên hạn chế. Tuy nhiên, do cấu trúc đơn giản, nó có thể không đạt được độ chính xác cao trong các tác vụ phức tạp.
- YOLOv9: Với YOLOv9-C là mô hình phức tạp nhất với (50.97M params, 237.7 GFLOPs, 724 layers, 102.8 MB), khả năng xử lý chi tiết tốt hơn. Tuy nhiên, nó đòi hỏi tài nguyên tính toán lớn và thời gian suy luận dài hơn, phù hợp cho các ứng dụng yêu cầu độ chính xác cao.

- YOLOv10: Với YOLOv10-L là mô hình cân bằng giữa số lượng tham số, kích thước và độ phức tạp với (25.73M params, 126.4 GFLOPs, 461 layers, 52.2 MB), cung cấp sự linh hoạt cho nhiều ứng dụng khác nhau mà không đòi hỏi quá nhiều tài nguyên.

Hiệu suất của các mô hình YOLO trên tập dữ liệu gốc (bảng 5.3):

- YOLOv5: Mô hình này có thời gian suy luận gần như là nhanh nhất với 9.4 ms trên tập xác thực và 12.7 ms trên tập kiểm thử, phù hợp cho các ứng dụng yêu cầu tốc độ cao. Tuy nhiên, precision và recall thấp hơn các mô hình khác (precision: 0.74, recall: 0.526, mAP 0.5: 0.605, mAP 0.5:0.95: 0.383 trên tập xác thực).
- YOLOv9: Có precision và recall cao nhất với (precision: 0.837, recall: 0.659, mAP 0.5: 0.767, mAP 0.5:0.95: 0.606 trên tập xác thực), phù hợp cho các ứng dụng yêu cầu độ chính xác cao. Tuy nhiên, thời gian suy luận và huấn luyện lâu hơn, đòi hỏi tài nguyên tính toán mạnh mẽ..
- YOLOv10: Cung cấp sự cân bằng giữa tốc độ suy luận, độ chính xác và tài nguyên tính toán (4.5 ms trên tập xác thực, precision: 0.68, recall: 0.486, mAP 0.5: 0.558, mAP 0.5:0.95: 0.42 trên tập xác thực). Nó phù hợp cho nhiều ứng dụng khác nhau mà không yêu cầu tài nguyên quá lớn hay độ chính xác cực cao.

Hiệu suất của các mô hình YOLO trên tập dữ liệu xoay (bảng 5.4):

- YOLOv5: Có thời gian suy luận khá nhanh, đặc biệt là trên tập xác thực (5.3 ms). Precision và recall ở mức trung bình (precision: 0.772, recall: 0.707, mAP 0.5: 0.749, mAP 0.5:0.95: 0.523 trên tập xác thực), phù hợp cho các ứng dụng không yêu cầu quá cao về độ chính xác nhưng vẫn cần tốc độ xử lý nhanh.
- YOLOv9: Có precision và recall cao nhất với (precision: 0.846, recall: 0.752, mAP 0.5: 0.827, mAP 0.5:0.95: 0.667 trên tập xác thực), phù hợp cho các ứng dụng yêu cầu độ chính xác cao nhất. Tuy nhiên, thời gian suy luận dài hơn so với các mô hình khác (67.7 ms trên tập xác thực), điều này có thể là hạn chế đối với các ứng dụng yêu cầu thời gian thực.

- YOLOv10: Có thời gian suy luận nhanh nhất đặc biệt trên tập xác thực (4.1 ms), và có precision và recall khá tốt (precision: 0.831, recall: 0.672, mAP 0.5: 0.757, mAP 0.5:0.95: 0.593 trên tập xác thực). Đây là mô hình cân bằng tốt giữa tốc độ và độ chính xác, phù hợp cho nhiều ứng dụng khác nhau.

Hiệu suất của các mô hình YOLO trên tập dữ liệu xoay kết hợp với phương pháp tăng cường mosaic (bảng 5.5):

- YOLOv5: Có thời gian suy luận khá nhanh, đặc biệt là trên tập xác thực (5.3 ms). Precision và recall ở mức trung bình khá (precision: 0.815, recall: 0.789, mAP 0.5: 0.824, mAP 0.5:0.95: 0.617 trên tập xác thực), phù hợp cho các ứng dụng yêu cầu tốc độ xử lý nhanh nhưng vẫn đảm bảo độ chính xác nhất định.
- YOLOv9: Có precision và recall cao nhất, đặc biệt là trên tập kiểm thử với (precision: 0.884, recall: 0.86, mAP 0.5: 0.912, mAP 0.5:0.95: 0.739). Điều này phù hợp cho các ứng dụng yêu cầu độ chính xác cao nhất, mặc dù thời gian suy luận dài hơn so với các mô hình khác (69.0 ms trên tập xác thực) có thể là một hạn chế đối với các ứng dụng thời gian thực.
- YOLOv10: Có thời gian suy luận nhanh nhất (4.1 ms) và thời gian huấn luyện ngắn nhất (4.651 giờ). Precision trên tập xác thực cao nhất trong các mô hình với (0.873). Mặc dù precision và recall trên tập kiểm thử không phải là cao nhất trên tất cả các chỉ số, nhưng vẫn ở mức tốt và phù hợp cho nhiều ứng dụng yêu cầu tốc độ và hiệu suất ổn định (mAP 0.5: 0.862, mAP 0.5:0.95: 0.697 trên tập xác thực).

Hiệu suất của mô hình YOLOv9 trên tập dữ liệu xoay kết hợp với phương pháp tăng cường MTDA và mosaic (bảng 5.6): Từ các phân tích bên trên, nhận thấy rằng Yolov9 có độ chính xác hầu như là cao nhất trong cả ba mô hình YOLO. Để tiết kiệm chi phí tính toán, bài luận chỉ áp dụng phương pháp xoay kết hợp với MTDA và mosaic đối với mô hình Yolov9. Kết quả được tổng quan trong bảng 5.6 với kết quả thời gian suy luận cho tập xác thực là 65.9 ms và kiểm thử là 64.8 ms. Precision đạt 0.834 (xác thực) và 0.854 (kiểm thử), recall là 0.769 (xác thực) và 0.84 (kiểm thử). mAP tại ngưỡng 0.5 là 0.839 (xác thực) và 0.894 (kiểm thử), tại

ngưỡng 0.5-0.95 là 0.668 (xác thực) và 0.712 (kiểm thử). Nhìn chung, mô hình thể hiện tốt hơn trên tập dữ liệu kiểm thử, cho thấy khả năng tổng quát hóa tốt hơn khi áp dụng các tăng cường dữ liệu.

5.2.3. Thảo luận về phần thời gian thực

Khối lượng dữ liệu lớn từ nhiều camera gây ra độ trễ trong xử lý và phân tích, ảnh hưởng đến khả năng phản ứng kịp thời của hệ thống. Trong hệ thống hiện tại, mô hình YOLOv9 được sử dụng cho nhiệm vụ nhận diện và phân loại đối tượng. Tuy nhiên, thời gian suy luận của YOLOv9, như được ghi nhận, khá dài (65.9 ms trên tập xác thực và 64.8 ms trên tập kiểm thử). Độ trễ này có thể là một thách thức lớn khi yêu cầu giám sát thời gian thực, đặc biệt khi hệ thống phải xử lý nhiều khung hình từ nhiều camera đồng thời.

Một thách thức khác là độ chính xác của mô hình nhận diện và phân loại sẽ bị giảm khi phải làm việc trong những điều kiện ánh sáng và góc quay khác nhau, đặc biệt là khi có nhiều vật thể chồng chéo hoặc khi phải phân biệt những loại hoa quả và rau củ có hình dáng và màu sắc tương đồng. Điều này đòi hỏi hệ thống phải có khả năng điều chỉnh và tinh chỉnh mô hình để duy trì độ chính xác cao trong các điều kiện thực tế phức tạp.

Bên cạnh đó, chất lượng hình ảnh từ các camera cũng có thể không đồng nhất, làm giảm hiệu quả nhận diện. Các camera có thể có độ phân giải khác nhau, mức độ nhiễu khác nhau, và chất lượng ánh sáng không nhất quán, điều này dẫn đến việc cần có giải pháp để đồng nhất chất lượng hình ảnh.

Chương 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1. Kết luận

Nghiên cứu này đã đạt được các mục tiêu chính đã đề ra ban đầu, bao gồm xây dựng bộ dữ liệu phong phú, triển khai và đánh giá các mô hình nhận diện, xử lý dữ liệu thời gian thực, và theo dõi và đếm đối tượng. Dưới đây là những điểm nổi bật:

- Nghiên cứu này đã áp dụng và so sánh các mô hình nhận diện đối tượng khác nhau, bao gồm Faster-RCNN-Resnet-101 và các biến thể của YOLO, trên tập dữ liệu bao gồm cả dữ liệu gốc và dữ liệu đã qua tăng cường. Các kết quả cho thấy:
  - Mô hình Faster-RCNN-Resnet-101 có hiệu suất tốt, đặc biệt hiệu quả với các đối tượng lớn nhưng gặp khó khăn với các đối tượng nhỏ.
  - YOLOv9 có độ chính xác và recall cao nhất, đặc biệt khi áp dụng các kỹ thuật tăng cường dữ liệu như MTDA và mosaic, nhưng có thời gian suy luận dài.
  - YOLOv5 và YOLOv10 cung cấp sự cân bằng giữa tốc độ suy luận và độ chính xác, phù hợp cho các ứng dụng yêu cầu tốc độ cao và độ chính xác tương đối.
- Apache Spark và Kafka đã được áp dụng để xử lý và phân tích dữ liệu liên tục, giúp hệ thống phản ứng nhanh chóng với những thay đổi trong môi trường. Mô hình YOLO tốt nhất đã được kết hợp với DeepSORT để theo dõi và đếm chính xác từng đối tượng. Hệ thống không chỉ nhận diện mà còn quản lý số lượng trái cây và rau củ theo thời gian thực, đảm bảo tính chính xác và hiệu quả trong việc giám sát.

6.2. Hướng phát triển

Dựa trên những kết quả đã đạt được, nghiên cứu đề xuất một số hướng phát triển nhằm tiếp tục cải thiện hệ thống giám sát trái cây và rau củ thời gian thực:

- *Cải thiện cân bằng dữ liệu*: Tiếp tục nghiên cứu và áp dụng các kỹ thuật tăng cường dữ liệu khác để đạt được sự cân bằng tốt hơn giữa các lớp nhãn, nhằm cải thiện hiệu suất tổng thể của mô hình.

- *Tối ưu hóa thời gian suy luận*: Nghiên cứu và áp dụng các phương pháp tối ưu hóa mô hình như pruning, quantization, hoặc sử dụng phần cứng chuyên dụng như GPU và TPU để giảm thời gian suy luận của các mô hình YOLO, đặc biệt là YOLOv9.
- *Cải thiện độ chính xác trong điều kiện thực tế:* Tiếp tục sử dụng các kỹ thuật tăng cường dữ liệu và điều chỉnh, cải tiến mô hình để nâng cao độ chính xác trong các điều kiện ánh sáng và góc quay khác nhau, cũng như trong các tình huống có nhiều vật thể chồng chéo.
- *Xây dựng cơ sở hạ tầng mở rộng*: Phát triển kế hoạch và cấu trúc mở rộng hợp lý để xử lý dữ liệu từ nhiều camera khác nhau, đảm bảo hệ thống có thể mở rộng một cách hiệu quả mà không làm giảm hiệu suất.

TÀI LIỆU THAM KHẢO

TÀI LIỆU THAM KHẢO TRONG NƯỚC (VIỆT NAM)

[1] Phúc N.V., T. viện khóa luận luận luận văn thạc sĩ, báo cáo thực tập, tiểu, "Luận văn Công nghệ thông tin Ứng dụng học sâu trong phân loại trái cây", tháng 10 2019.

[2] Thành N. Đ., "Luận văn Nhận dạng và phân loại hoa quả trong ảnh màu - Luận văn, đồ án, đề tài tốt nghiệp", 2016.

[3] Trịnh T. H., Bùi X. T., Nguyễn L. T. K., và Nguyễn H. H. C., "Xây dựng hệ thống nhận dạng phân loại trái cây chín tiếp cận mạng Nơron tích chập (CNN)", Nhà xuất bản Đà Nẵng, Working Paper, 2021. Available at: https://elib.vku.udn.vn/handle/123456789/1882

[4] Hải T. T., Cường N. H. H., và Duy N. K., "Mô hình Fast R-CNN cải tiến cho giải pháp nhận dạng, phát hiện trái dứa thời kỳ chín", *Tạp Chí Khoa Học Và Công Nghệ - Đại Học Đà Nẵng*, tr 94–98, tháng 7 2022.

[5] Mạnh N. V., "Hướng dẫn phân loại cà chua bằng YOLOv7", THỊ GIÁC MÁY TÍNH. Available at: https://thigiacmaytinh.com/huong-dan-phan-loai-ca-chua-bang-yolov7/

[17] Thái N. H. *và c.s.*, "NGHIÊN CỨU MÔ HÌNH HỌC SÂU FASTER R-CNN ĐỂ PHÁT HIỆN VÀ PHÂN LOẠI CÁC TỔN THƯƠNG KHU TRÚ THƯỜNG GẶP Ở GAN TRÊN ẢNH CHỤP CẮT LỚP VI TÍNH", *Tạp Chí Dược Học Cần Thơ*, số p.h 57, Art. số p.h 57, tháng 2 2023, doi: 10.58490/ctump.2023i57.598.

TÀI LIỆU THAM KHẢO QUỐC TẾ

[6] Y. Li, X. Feng, Y. Liu, và X. Han, "Apple quality identification and classification by image processing based on convolutional neural networks", *Sci. Rep.*, vol 11, tháng 8 2021, doi: 10.1038/s41598-021-96103-2.

[7] K. Bogomasov và S. Conrad, "Efficient Fruit and Vegetable Classification and Counting for Retail Applications Using Deep Learning", trong *Proceedings of the 5th International Conference on Advances in Artificial Intelligence*, trong ICAAI '21. New York, NY, USA: Association for Computing Machinery, 2022, tr 65–71. doi: 10.1145/3505711.3505720.

[8] M. Mukhiddinov, A. Muminov, và J. Cho, "Improved Classification Approach for Fruits and Vegetables Freshness Based on Deep Learning", *Sensors*, vol 22, số p.h 21, 2022, doi: 10.3390/s22218192.

[9] J. S. Tata, N. K. V. Kalidindi, H. Katherapaka, S. K. Julakal, và M. Banothu, "Real-Time Quality Assurance of Fruits and Vegetables with Artificial Intelligence", *J. Phys. Conf. Ser.*, vol 2325, số p.h 1, tr 012055, tháng 8 2022, doi: 10.1088/1742-6596/2325/1/012055.

[10] J. Hu, C. Fan, Z. Wang, J. Ruan, và S. Wu, "Fruit Detection and Counting in Apple Orchards Based on Improved Yolov7 and Multi-Object Tracking Methods", *Sensors*, vol 23, số p.h 13, 2023, doi: 10.3390/s23135903.

[11] J. Zhang, M. Tian, Z. Yang, J. Li, và L. Zhao, "An improved target detection method based on YOLOv5 in natural orchard environments", *Comput. Electron. Agric.*, vol 219, tr 108780, 2024, doi: https://doi.org/10.1016/j.compag.2024.108780.

[12] J. Kukacka, V. Golkov, và D. Cremers, "Regularization for Deep Learning: A Taxonomy", tháng 10 2017.

[13] H. Zhang, M. Cisse, Y. N. Dauphin, và D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization". arXiv, 27 Tháng Tư 2018. doi: 10.48550/arXiv.1710.09412.

[14] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, và Y. J. Yoo, “CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features”, *2019 IEEECVF Int. Conf. Comput. Vis. ICCV*, tr 6022–6031, 2019.

[15] Y. Li, R. Cheng, C. Zhang, M. Chen, H. Liang, và Z. Wang, “Dynamic Mosaic algorithm for data augmentation.”, *Math. Biosci. Eng. MBE*, vol 20 4, tr 7193–7216, 2023.

[16] Y. Amit, P. Felzenszwalb, và R. Girshick, “Object Detection”, 2020, tr 1–9. doi: 10.1007/978-3-030-03243-2_660-1.

[18] J. Canny, “A Computational Approach to Edge Detection”, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol PAMI-8, số p.h 6, tr 679–698, 1986, doi: 10.1109/TPAMI.1986.4767851.

[19] N. Dalal và B. Triggs, “Histograms of oriented gradients for human detection”, trong *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, 2005, tr 886–893 vol 1. doi: 10.1109/CVPR.2005.177.

[20] T. Lindeberg, “Scale Invariant Feature Transform”, trong *Scholarpedia*, vol 7, 2012. doi: 10.4249/scholarpedia.10491.

[21] P. Viola và M. Jones, “Rapid object detection using a boosted cascade of simple features”, trong *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 2001, tr I–I. doi: 10.1109/CVPR.2001.990517.

[22] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, và D. Ramanan, “Object Detection with Discriminatively Trained Part-Based Models”, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol 32, số p.h 9, tr 1627–1645, 2010, doi: 10.1109/TPAMI.2009.167.

[23] S. Ren, K. He, R. Girshick, và J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, trong *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, và R. Garnett, B.t.v, Curran Associates, Inc., 2015. [Online]. Available at:

https://proceedings.neurips.cc/paper_files/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf

[24] J. Uijlings, K. Sande, T. Gevers, và A. W. M. Smeulders, "Selective Search for Object Recognition", *Int. J. Comput. Vis.*, vol 104, tr 154–171, tháng 9 2013, doi: 10.1007/s11263-013-0620-5.

[25] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, và S. Belongie, "Feature Pyramid Networks for Object Detection", trong *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, tr 936–944. doi: 10.1109/CVPR.2017.106.

[26] Ultralytics, "YOLOv5: A state-of-the-art real-time object detection system". 2021. [Online]. Available at: https://docs.ultralytics.com

[27] C.-Y. Wang, I.-H. Yeh, và H. Liao, "YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information", *ArXiv*, vol abs/2402.13616, 2024,. Available at: https://api.semanticscholar.org/CorpusID:267770251

[28] A. Wang *và c.s.*, "YOLOv10: Real-Time End-to-End Object Detection". 2024. [Online]. Available at: https://arxiv.org/abs/2405.14458

[29] T. Raptis và A. Passarella, "A Survey on Networked Data Streaming With Apache Kafka", *IEEE Access*, vol PP, tr 1–1, tháng 1 2023, doi: 10.1109/ACCESS.2023.3303810.

[30] X. Meng *và c.s.*, "MLlib: Machine Learning in Apache Spark". arXiv, 26 Tháng Năm 2015. doi: 10.48550/arXiv.1505.06807.

[31] M. A. Nurkemal, E. Mulyana, A. B. Suksmono, A. Taniwidjaja, và W. Shalannanda, "Data Streaming and Visualization Results of Object Detection Systems (Case Study of Human Object)", trong *2019 IEEE 5th International Conference on Wireless and Telematics (ICWT)*, 2019, tr 1–3. doi: 10.1109/ICWT47785.2019.8978261.

[32] Z. Kuang, S. Tan, và B. Jin, "Appleyolo: Apple Yield Estimation Method Using Improved Yolov8 Based on Deep Oc-Sort". Rochester, NY, 11 Tháng Sáu 2024. doi: 10.2139/ssrn.4861527.

[33] R. Pereira, G. Carvalho, L. Garrote, và U. J. Nunes, “Sort and Deep-SORT Based Multi-Object Tracking for Mobile Robotics: Evaluation with New Data Association Metrics”, *Appl. Sci.*, vol 12, số p.h 3, 2022, doi: 10.3390/app12031319.

[34] J. Barbedo, “A Review on Methods for Automatic Counting of Objects in Digital Images”, *Lat. Am. Trans. IEEE Rev. IEEE Am. Lat.*, vol 10, tr 2112–2124, tháng 9 2012, doi: 10.1109/TLA.2012.6362356.

[35] C. Pornpanomchai, F. Stheitsthienchai, và S. Rattanachuen, “Object Detection and Counting System”, trong *2008 Congress on Image and Signal Processing*, 2008, tr 61–65. doi: 10.1109/CISP.2008.108.