

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC ĐẠI NAM



BÀI TẬP LỚN

TÊN HỌC PHẦN: NÔNG NGHIỆP THÔNG MINH

& THÀNH PHỐ THÔNG MINH

ĐỀ TÀI: NHẬN DIỆN HOA QUẢ

Giáo viên hướng dẫn: Ts.Trần Đăng Công,Thầy Nguyễn Văn Nhân

Sinh viên thực hiện:

Stt	Mã sv	Họ và tên	Lớp
1	1571020088	Phùng Đăng Hậu	CNTT 15-04

Hà Nội, năm 2025

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC ĐẠI NAM**



BÀI TẬP LỚN

**TÊN HỌC PHẦN: NÔNG NGHIỆP THÔNG MINH
& THÀNH PHỐ THÔNG MINH
ĐỀ TÀI: NHẬN DIỆN HOA QUẢ**

STT	Mã Sinh Viên	Họ và Tên	Ngày Sinh	Điểm	
				Bảng Số	Bảng Chữ
1	1571020088	Phùng Đăng Hậu	07/10/2003		

CÁN BỘ CHẤM THI 1

CÁN BỘ CHẤM THI 2

Hà Nội, năm 202

LỜI NÓI ĐẦU

Trong những năm gần đây, trí tuệ nhân tạo (AI) và thị giác máy tính (Computer Vision) đã trở thành những lĩnh vực cốt lõi thúc đẩy sự đổi mới trong nhiều ngành công nghiệp, đặc biệt là trong nông nghiệp thông minh, tự động hóa và bán lẻ. Nhận diện đối tượng là một trong những bài toán then chốt của thị giác máy, cho phép máy tính có khả năng "nhìn thấy" và "hiểu được" hình ảnh trong thế giới thực.

Trong bối cảnh đó, việc xây dựng một hệ thống nhận diện hoa quả tự động không chỉ mang lại giá trị ứng dụng cao trong kiểm kê, phân loại sản phẩm nông nghiệp mà còn đóng vai trò quan trọng trong việc giảm thiểu chi phí lao động, tăng độ chính xác và nâng cao hiệu quả sản xuất.

Với mục tiêu áp dụng các phương pháp học sâu tiên tiến, đặc biệt là mô hình YOLOv8 – một kiến trúc hiện đại và tối ưu trong bài toán nhận diện thời gian thực – Em thực hiện đã tiến hành nghiên cứu, thiết kế và xây dựng một hệ thống có khả năng nhận diện chính xác các loại hoa quả phổ biến như táo, chuối, cam, xoài.

Nội dung báo cáo bao gồm toàn bộ quá trình thực hiện đề tài từ khảo sát công nghệ, xây dựng bộ dữ liệu, huấn luyện mô hình cho đến đánh giá hiệu quả và đề xuất hướng phát triển trong tương lai. Em hy vọng rằng kết quả nghiên cứu này không chỉ là một minh chứng cho khả năng ứng dụng AI trong thực tiễn mà còn là bước đệm để phát triển các giải pháp tự động hóa hiệu quả trong lĩnh vực nông nghiệp và tiêu dùng thông minh.

LỜI CẢM ƠN

Trước tiên, em xin gửi lời cảm ơn chân thành và sâu sắc nhất đến thầy Trần Đăng Công và Thầy Nguyễn Văn Nhân, người đã tận tình hướng dẫn, chỉ bảo chúng em trong suốt quá trình thực hiện báo cáo thực tập này. Thầy không chỉ giúp chúng em hiểu rõ hơn về chuyên môn, cách tiếp cận và giải quyết vấn đề mà còn truyền động lực để chúng em không ngừng nỗ lực, hoàn thiện bản thân.

Nhóm sinh viên thực hiện đề tài “Nhận diện hoa quả bằng mô hình YOLOv8” xin gửi lời cảm ơn chân thành và sâu sắc đến Thầy Trần Đăng Công và Thầy Nguyễn Văn Nhân – người đã trực tiếp hướng dẫn, tận tình chỉ bảo và định hướng cho nhóm trong suốt quá trình thực hiện đề tài.

Nhờ sự hướng dẫn tận tâm, kiến thức chuyên môn sâu rộng và tinh thần hỗ trợ nhiệt tình của thầy, nhóm đã có cơ hội tiếp cận với những kiến thức hiện đại về trí tuệ nhân tạo, học sâu và thị giác máy tính, từ đó ứng dụng thành công vào việc xây dựng một hệ thống nhận diện hoa quả hiệu quả.

Một lần nữa, nhóm xin trân trọng cảm ơn thầy và mong rằng sẽ tiếp tục nhận được sự đồng hành, định hướng của thầy trong những chặng đường học tập và nghiên cứu tiếp theo.

MỤC LỤC

CHƯƠNG 1: GIỚI THIỆU	7
1.1. Giới thiệu đề tài.....	7
1.2. Mục tiêu nghiên cứu	8
1.3. Lý do chọn đề tài.....	8
1.4. Lý do lựa chọn Yolo V8.....	8
1.5. phạm vi nghiên cứu	8
1.6. Đối tượng và ý nghĩa nghiên cứu	9
1.6.1. <i>Đối tượng:</i>	9
1.6.2. <i>Ý nghĩa thực tiễn:</i>	9
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT.....	10
2.1. Xử lý ảnh và thị giác máy tính	10
2.2. Nhận diện vật thể.....	11
2.3. Công nghệ sử dụng	11
CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG	14
3.1. Kiến trúc hệ thống Hệ thống nhận diện hoa quả sử dụng mô hình YOLOv8 được thiết kế với các thành phần chính như sau:	14
3.2. Quy trình thực hiện	14
CHƯƠNG 4: THỰC NGHIỆM VÀ KẾT QUẢ	15
4.1. Dữ liệu huấn luyện.....	15
4.2. Thông số huấn luyện	26

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	28
5.1. Kết luận	28
5.2. Hướng phát triển	28
KẾT LUẬN	29
DANH MỤC TÀI LIỆU THAM KHẢO	30

MỤC LỤC HÌNH ẢNH

CHƯƠNG 1: GIỚI THIỆU

1.1. Giới thiệu đề tài

Trong thời đại công nghệ số hiện nay, thị giác máy tính (Computer Vision) đã trở thành một trong những lĩnh vực then chốt của trí tuệ nhân tạo (AI), góp phần thay đổi cách mà máy móc nhận thức và tương tác với thế giới thực. Trong đó, xử lý ảnh đóng vai trò quan trọng trong việc trích xuất và phân tích thông tin hình ảnh nhằm phục vụ cho các ứng dụng như giám sát an ninh, y tế, xe tự lái, nhận diện khuôn mặt, và đặc biệt là nhận diện vật thể.

Nhận diện vật thể là một bài toán nền tảng trong thị giác máy tính, cho phép hệ thống xác định chính xác sự xuất hiện, vị trí và nhãn của các đối tượng trong một hình ảnh hoặc đoạn video. Nhờ đó, máy có thể hiểu được môi trường xung quanh và đưa ra những phản ứng phù hợp. Trong nông nghiệp, công nghệ nhận diện vật thể mang lại giải pháp tự động hóa hiệu quả như phân loại trái cây, giám sát mùa vụ, và phát hiện sâu bệnh. Việc ứng dụng mô hình nhận diện vào nhận dạng hoa quả không chỉ giúp tối ưu hoá quy trình sản xuất mà còn mang lại nhiều giá trị thực tiễn trong phân phối và bán hàng tự động.

Trong đề tài này, nhóm nghiên cứu lựa chọn sử dụng mô hình YOLO (You Only Look Once) – một trong những mô hình phát hiện vật thể tiên tiến và phổ biến nhất hiện nay – để triển khai bài toán nhận diện hoa quả. Với tốc độ nhanh và độ chính xác cao, YOLO đặc biệt phù hợp với các ứng dụng cần xử lý thời gian thực. Qua đề tài, nhóm hướng tới việc xây dựng một hệ thống đơn giản nhưng hiệu quả để phát hiện và phân loại các loại trái cây như táo, chuối, cam, xoài,... thông qua hình ảnh.

1.2. Mục tiêu nghiên cứu

- Tìm hiểu và ứng dụng mô hình YOLO trong bài toán nhận diện vật thể.
- Xây dựng bộ dữ liệu hoa quả có gán nhãn đầy đủ.
- Triển khai mô hình nhận diện hình ảnh các loại hoa quả.
- Đánh giá hiệu quả của mô hình trên tập dữ liệu thử nghiệm.

1.3. Lý do chọn đề tài

Trong thời đại trí tuệ nhân tạo phát triển mạnh mẽ, nhận diện vật thể bằng xử lý ảnh đang được ứng dụng rộng rãi trong nhiều lĩnh vực, đặc biệt là nông nghiệp thông minh. Đề tài “Ứng dụng YOLO trong nhận diện hoa quả” được chọn vì có tính thực tiễn cao, giúp tự động hóa việc phân loại trái cây, tiết kiệm thời gian, nhân lực, đồng thời giúp nhóm sinh viên hiểu rõ hơn về mô hình YOLO và quy trình xây dựng hệ thống nhận diện hình ảnh.

1.4. Lý do lựa chọn Yolo V8

YOLOv8 là phiên bản mới nhất trong dòng mô hình YOLO, nổi bật với độ chính xác cao, tốc độ xử lý nhanh và khả năng triển khai linh hoạt. So với các phiên bản trước, YOLOv8 cải tiến kiến trúc mạng, hỗ trợ tốt hơn cho các thiết bị edge và dễ dàng tùy chỉnh cho nhiều bài toán thực tế. Vì vậy, nhóm lựa chọn YOLOv8 nhằm đảm bảo hiệu quả cao nhất trong việc nhận diện nhanh và chính xác các loại trái cây qua hình ảnh.

1.5. phạm vi nghiên cứu

Đề tài tập trung vào việc triển khai mô hình YOLOv8 để nhận diện một số loại trái cây phổ biến như táo, chuối, cam, xoài... thông qua hình ảnh. Phạm vi nghiên cứu bao gồm: thu thập và gán nhãn dữ liệu hình ảnh, tiền xử lý ảnh, huấn luyện mô hình YOLOv8, đánh giá độ chính xác và thử nghiệm nhận diện trên ảnh tĩnh và video đơn giản. Đề tài không đi sâu vào tối ưu phần cứng, không so sánh với tất cả các mô hình khác và chưa triển khai trên thiết bị thực tế

1.6. Đối tượng và ý nghĩa nghiên cứu

1.6.1. Đối tượng:

- Đối tượng của đề tài là hình ảnh các loại trái cây và mô hình phát hiện vật thể YOLOv8 – một mô hình học sâu được ứng dụng để nhận diện và phân loại trái cây qua ảnh.

1.6.2. Ý nghĩa thực tiễn:

- Đề tài góp phần ứng dụng công nghệ thị giác máy tính vào lĩnh vực nông nghiệp thông minh, hỗ trợ tự động hóa quy trình phân loại sản phẩm. Đồng thời, giúp sinh viên tiếp cận thực tiễn việc xây dựng hệ thống xử lý ảnh, từ khâu dữ liệu đến huấn luyện và triển khai mô hình, phục vụ cho các ứng dụng trong công nghiệp, thương mại và giáo dục

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Nông nghiệp thông minh

Khái niệm:

Nông nghiệp thông minh (Smart Agriculture) là hệ thống canh tác ứng dụng các công nghệ kỹ thuật số để nâng cao hiệu quả sản xuất và khả năng ứng phó với biến đổi khí hậu, thiên tai và dịch bệnh.

Các công nghệ chính trong nông nghiệp thông minh:

- IoT (Internet of Things):
Cảm biến đo độ ẩm, nhiệt độ, pH đất; dữ liệu thu thập giúp tưới tiêu và bón phân tự động.
- Trí tuệ nhân tạo (AI):
AI hỗ trợ phân tích dữ liệu lớn, dự báo thời tiết, sâu bệnh và tối ưu hóa lịch trình canh tác.
- Drone và thiết bị bay không người lái:
Giám sát cây trồng, chụp ảnh định kỳ, phun thuốc bảo vệ thực vật chính xác.
- Big Data và dữ liệu vệ tinh:
Tổng hợp dữ liệu từ hàng triệu cảm biến giúp người nông dân ra quyết định kịp thời.
- Tự động hóa và robot:
Máy móc tự động gieo hạt, tưới nước, thu hoạch, giúp tiết kiệm chi phí và tăng hiệu suất.
- Lợi ích:
Tăng năng suất nông nghiệp lên 20–30%.

Giảm chi phí vận hành và nhân công.

Giảm thiểu lãng phí nước và phân bón.

Đáp ứng tiêu chuẩn nông sản sạch và an toàn.
- Ví dụ tại Việt Nam:
Vườn rau khí canh ở Hà Nội sử dụng IoT để kiểm soát chất lượng cây trồng.

Trang trại chăn nuôi ở Lâm Đồng ứng dụng AI để phát hiện bệnh vật nuôi sớm.

2.2. Nhận diện vật thể

Nhận diện vật thể (Object Detection) là một bước tiến cao hơn so với phân loại ảnh (Image Classification). Thay vì chỉ xác định một nhãn cho toàn ảnh, nhận diện vật thể cho biết có những đối tượng gì, ở vị trí nào trong ảnh.

So sánh các tác vụ trong thị giác máy tính:

Tác vụ	Đầu ra
Phân loại ảnh	Một nhãn cho toàn bộ ảnh
Nhận diện vật thể	Nhãn + vị trí (bounding box) của từng đối tượng trong ảnh
Phân đoạn ảnh	Gán nhãn cho từng pixel trong ảnh (các đối tượng tách biệt chính xác)

Một số mô hình phổ biến:

- CNN (Convolutional Neural Networks): Cơ sở cho hầu hết mô hình xử lý ảnh hiện nay.
- SSD (Single Shot Detector): Phát hiện vật thể nhanh, nhưng độ chính xác không cao bằng các mô hình mới hơn.
- Faster R-CNN: Độ chính xác cao, nhưng tốc độ chậm, khó dùng trong thời gian thực.
- YOLO (You Only Look Once): Mô hình nhận diện một bước (one-stage), nổi bật với tốc độ xử lý nhanh và độ chính xác tốt, đặc biệt phù hợp với ứng dụng thời gian thực như nhận diện hoa quả trong video.

2.3. Công nghệ sử dụng

YOLO là mô hình phát hiện vật thể một bước, nghĩa là chỉ cần một lần quét toàn bộ ảnh để dự đoán vị trí và nhãn của tất cả vật thể. Điều này giúp YOLO trở thành mô hình lý tưởng cho các ứng dụng cần tốc độ cao như xe tự lái, giám sát an ninh, và nhận diện sản phẩm.

Nguyên lý hoạt động:

YOLO (You Only Look Once) hoạt động dựa trên cách tiếp cận chia ảnh đầu vào thành một lưới (grid), thường là lưới $S \times S$. Mỗi ô lưới sẽ chịu trách nhiệm phát hiện các vật thể mà tâm

của vật thể nằm trong ô đó. Với mỗi ô, mô hình sẽ dự đoán một số lượng cố định các bounding boxes (hộp giới hạn), đồng thời dự đoán:

- Tọa độ của hộp (bao gồm x , y là tọa độ trung tâm của vật thể, và w , h là chiều rộng, chiều cao của hộp)
- Độ tin cậy (confidence score) của mỗi hộp: thể hiện xác suất có vật thể trong hộp và mức độ chính xác của dự đoán vị trí.
- Xác suất thuộc về từng lớp (class probabilities): xác suất vật thể thuộc vào mỗi loại nhãn (ví dụ: người, xe, chó, v.v.)

Các phiên bản chính của YOLO:

- YOLOv1 \rightarrow YOLOv4: Các phiên bản đầu giúp khẳng định tên tuổi YOLO với tốc độ xử lý cực nhanh.
- YOLOv5: Phổ biến nhất, dễ dùng với nhiều công cụ hỗ trợ, hỗ trợ triển khai linh hoạt.
- YOLOv7: Cải tiến mạnh về độ chính xác và tối ưu hóa tốt hơn cho GPU.
- YOLOv8: Phiên bản mới nhất, hỗ trợ tốt cho cả detection, segmentation và classification. Kiến trúc mạng nhẹ hơn, tốc độ nhanh hơn và chính xác cao hơn.

Ưu điểm của YOLOv8:

- Xử lý ảnh thời gian thực với tốc độ cao.
- Hiệu suất tốt trên ảnh độ phân giải thấp.
- Dễ triển khai, tích hợp vào các hệ thống sản xuất, bán hàng hoặc tự động hóa.

Hạn chế:

- Hiệu quả phát hiện vật thể nhỏ hoặc chồng chéo chưa tối ưu như một số mô hình hai bước.
- Phụ thuộc nhiều vào chất lượng dữ liệu và công cụ gán nhãn.

Các phiên bản chính của YOLO:

- YOLOv1 → YOLOv4: Các phiên bản đầu giúp khẳng định tên tuổi YOLO với tốc độ xử lý cực nhanh.
- YOLOv5: Phổ biến nhất, dễ dùng với nhiều công cụ hỗ trợ, hỗ trợ triển khai linh hoạt.
- YOLOv7: Cải tiến mạnh về độ chính xác và tối ưu hóa tốt hơn cho GPU.
- YOLOv8: Phiên bản mới nhất, hỗ trợ tốt cho cả detection, segmentation và classification. Kiến trúc mạng nhẹ hơn, tốc độ nhanh hơn và chính xác cao hơn.

Ưu điểm của YOLOv8:

- Xử lý ảnh thời gian thực với tốc độ cao.
- Hiệu suất tốt trên ảnh độ phân giải thấp.
- Dễ triển khai, tích hợp vào các hệ thống sản xuất, bán hàng hoặc tự động hóa.

Hạn chế:

- Hiệu quả phát hiện vật thể nhỏ hoặc chồng chéo chưa tối ưu như một số mô hình hai bước.
- Phụ thuộc nhiều vào chất lượng dữ liệu và công cụ gán nhãn.

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1. Kiến trúc hệ thống Hệ thống nhận diện hoa quả sử dụng mô hình YOLOv8 được thiết kế với các thành phần chính như sau:

- Bộ thu thập dữ liệu (Dataset Collector): Thu thập hình ảnh các loại trái cây từ nhiều nguồn (trực tuyến hoặc thực tế), đảm bảo đa dạng góc chụp, ánh sáng và bối cảnh.
- Bộ gán nhãn dữ liệu (Annotation Tool): Sử dụng công cụ như LabelImg hoặc Roboflow để gán nhãn cho từng vật thể trong ảnh (bounding box và class).
- Tiền xử lý dữ liệu (Preprocessing): Chuyển đổi kích thước ảnh về chuẩn (thường là 640x640), chuẩn hóa pixel, và chia tập dữ liệu thành train/val/test.
- Huấn luyện mô hình (Training): Sử dụng YOLOv8 với các siêu tham số phù hợp (batch size, learning rate, epochs...) để học từ dữ liệu.
- Kiểm thử và đánh giá (Evaluation): Sử dụng các chỉ số như mAP (mean Average Precision), Precision, Recall để đánh giá độ chính xác của mô hình.
- Triển khai mô hình (Deployment): Tích hợp vào ứng dụng nhận diện ảnh tĩnh hoặc video.

3.2. Quy trình thực hiện

1. Thu thập dữ liệu: Chụp hoặc tải hình ảnh hoa quả đa dạng (mỗi loại ~200–500 ảnh).
2. Gán nhãn: Gán nhãn cho mỗi đối tượng trong ảnh bằng bounding box và nhãn tương ứng.
3. Tiền xử lý: Chuyển ảnh về kích thước chuẩn, chia train/test, kiểm tra chất lượng dữ liệu.
4. Huấn luyện mô hình YOLOv8: Sử dụng framework như Ultralytics YOLO với file YAML cấu hình lớp và đường dẫn dữ liệu.
5. Đánh giá: Sử dụng tập test để đánh giá mô hình bằng chỉ số mAP@0.5, mAP@0.5:0.95.
6. Triển khai: Tạo giao diện demo hoặc ứng dụng thực tế để nhận diện hoa quả từ webcam hoặc video.

CHƯƠNG 4: THỰC NGHIỆM VÀ KẾT QUẢ

4.1. Dữ liệu huấn luyện

Sử dụng roboflow để gán nhãn và train dữ liệu bằng kaggle

```
1]: !pip install ultralytics
!pip install roboflow

import ultralytics
ultralytics.checks()

from roboflow import Roboflow
rf = Roboflow(api_key="vJYu0dNBMeQ0qrD1DHi9")
project = rf.workspace("yolo-klrym").project("object-detection-htfhu")
version = project.version(7)
dataset = version.download("yolov8-obb")

Ultralytics 8.3.128 Python-3.11.11 torch-2.5.1+cu124 CUDA:0 (Tesla T4, 15095MiB)
Setup complete (4 CPUs, 31.4 GB RAM, 6281.8/8062.4 GB disk)
loading Roboflow workspace...
loading Roboflow project...
Downloading Dataset Version Zip in Object-Detection-7 to yolov8-obb:: 100%| 263166/263166 [00:12<00:00, 21171.15it/s]

Extracting Dataset Version Zip to Object-Detection-7 in yolov8-obb:: 100%| 16362/16362 [00:01<00:00, 9070.32it/s]
```

Sử dụng dòng lệnh : "!yolo task=detect mode=train model=yolov8n.pt data=/kaggle/working/Object-Detection-7/data.yaml epochs=100 imgsz=640 save_period=20 #save sau 20 epoch" để train model.

Sau đó dùng Arduino để truyền dữ liệu vào trong cam Esp32 để nhận diện hoa quả và kết nối với flask để đẩy lên web để thống kê và tổng hợp số lượng quả hỏng hay tươi.

- **Code Arduino:**

```
#include "esp_camera.h"
```

```
#include <WiFi.h>
```

```
#include <WebServer.h>
```

```
#include "esp_http_server.h"
```

```

// □ Thay thông tin Wi-Fi tại đây

const char* ssid = "test123";

const char* password = "12345678";

// 🧠 Khai báo chân cho AI Thinker ESP32-CAM

#define PWDN_GPIO_NUM 32

#define RESET_GPIO_NUM -1

#define XCLK_GPIO_NUM 0

#define SIOD_GPIO_NUM 26

#define SIOC_GPIO_NUM 27

#define Y9_GPIO_NUM 35

#define Y8_GPIO_NUM 34

#define Y7_GPIO_NUM 39

#define Y6_GPIO_NUM 36

#define Y5_GPIO_NUM 21

#define Y4_GPIO_NUM 19

#define Y3_GPIO_NUM 18

#define Y2_GPIO_NUM 5

#define VSYNC_GPIO_NUM 25

#define HREF_GPIO_NUM 23

#define PCLK_GPIO_NUM 22

// ⚙️ Khởi tạo luồng MJPEG

esp_err_t stream_handler(httpd_req_t *req) {

    camera_fb_t *fb = NULL;

    esp_err_t res = ESP_OK;

```



```

res = httpd_resp_set_type(req, "multipart/x-mixed-replace; boundary=frame");

if (res != ESP_OK) return res;

while (true) {

    fb = esp_camera_fb_get();

    if (!fb) {

        Serial.println("Camera capture failed");

        continue;

    }

    char part_buf[64];

    size_t hlen = snprintf(part_buf, sizeof(part_buf),

        "--frame\r\nContent-Type: image/jpeg\r\nContent-Length: %u\r\n\r\n", fb->len);

    res = httpd_resp_send_chunk(req, part_buf, hlen);

    if (res == ESP_OK) res = httpd_resp_send_chunk(req, (const char *)fb->buf, fb->len);

    if (res == ESP_OK) res = httpd_resp_send_chunk(req, "\r\n", 2);

    esp_camera_fb_return(fb);

    if (res != ESP_OK) break;

}

return res;

}

void startCameraServer() {

    httpd_config_t config = HTTPD_DEFAULT_CONFIG();

    httpd_uri_t uri_handler = {

        .uri      = "/",

        .method    = HTTP_GET,

        .handler    = stream_handler,

```

```

    .user_ctx = NULL

};

httpd_handle_t server = NULL;

httpd_start(&server, &config);

httpd_register_uri_handler(server, &uri_handler);
}

void setup() {
    Serial.begin(115200);

    Serial.setDebugOutput(true); // Bật debug để xem chi tiết lỗi Wi-Fi

    // Kết nối Wi-Fi với timeout

    WiFi.begin(ssid, password);

    Serial.println("Connecting to Wi-Fi...");

    unsigned long startAttemptTime = millis();

    const unsigned long timeout = 20000; // Timeout sau 10 giây

    while (WiFi.status() != WL_CONNECTED && millis() - startAttemptTime < timeout) {

        delay(500);

        Serial.print(".");

    }

    // Kiểm tra kết nối Wi-Fi

    if (WiFi.status() != WL_CONNECTED) {

        Serial.println("\nFailed to connect to Wi-Fi. Check credentials, signal, or band (2.4GHz).");

        Serial.print("WiFi Status Code: ");

        Serial.println(WiFi.status());

        return; // Dừng chương trình nếu không kết nối được

    }
}

```

```

Serial.println("");

Serial.println("WiFi connected");

Serial.print("ESP32-CAM IP Address: http://");

Serial.println(WiFi.localIP());

// Cấu hình camera

camera_config_t config;

config.ledc_channel = LEDC_CHANNEL_0;

config.ledc_timer = LEDC_TIMER_0;

config.pin_d0    = Y2_GPIO_NUM;

config.pin_d1    = Y3_GPIO_NUM;

config.pin_d2    = Y4_GPIO_NUM;

config.pin_d3    = Y5_GPIO_NUM;

config.pin_d4    = Y6_GPIO_NUM;

config.pin_d5    = Y7_GPIO_NUM;

config.pin_d6    = Y8_GPIO_NUM;

config.pin_d7    = Y9_GPIO_NUM;

config.pin_xclk  = XCLK_GPIO_NUM;

config.pin_pclk  = PCLK_GPIO_NUM;

config.pin_vsync = VSYNC_GPIO_NUM;

config.pin_href  = HREF_GPIO_NUM;

config.pin_sscb_sda = SIOD_GPIO_NUM;

config.pin_sscb_scl = SIOC_GPIO_NUM;

config.pin_pwdn  = PWDN_GPIO_NUM;

config.pin_reset = RESET_GPIO_NUM;

config.xclk_freq_hz = 20000000;

```

```

config.pixel_format = PIXFORMAT_JPEG;

config.frame_size  = FRAMESIZE_QVGA;

config.jpeg_quality = 6;

config.fb_count    = 1;

// Khởi động camera

if (esp_camera_init(&config) != ESP_OK) {

    Serial.println("Camera init failed");

    return;

}

startCameraServer();

Serial.println("Camera Stream Ready!");

}

void loop() {

    // Không cần gì trong loop

}

```

- **Code app.py:**

```

# app.py (hoàn chỉnh với YOLO, IPFS, Blockchain, CSV Export)

from flask import Flask, request, render_template, send_from_directory, send_file

from ultralytics import YOLO

import os

from werkzeug.utils import secure_filename

import threading

import cv2

import uuid

import webbrowser

```

```

from web3 import Web3

import json

import requests

import datetime

import csv

# ===== Flask setup =====

app = Flask(__name__)

app.config['UPLOAD_FOLDER'] = 'uploads'

app.config['RESULT_FOLDER'] = 'runs/detect/predict'

app.config['DASHBOARD_FOLDER'] = 'dashboard'

os.makedirs(app.config['UPLOAD_FOLDER'], exist_ok=True)

os.makedirs(app.config['DASHBOARD_FOLDER'], exist_ok=True)

# ===== YOLO model =====

model = YOLO('weights/best.pt')

video_extensions = ['.mp4', '.avi', '.mov', '.mkv']

ESP32_STREAM_URL = "http://192.168.109.61/"

# ===== Web3 & Contract Setup =====

w3 = Web3(Web3.HTTPProvider("http://127.0.0.1:7545"))

account = w3.eth.accounts[0]

with open("abi.json") as f:

    abi = json.load(f)

if not os.path.exists("contract_address.txt"):

    raise FileNotFoundError("❌ Lỗi: Chưa tìm thấy file contract_address.txt. Hãy tạo file này với địa chỉ contract đã deploy trong Remix.")

with open("contract_address.txt") as f:

```

```

contract_address = f.read().strip()

contract = w3.eth.contract(address=contract_address, abi=abi)

# ===== IPFS Config =====

PINATA_API_KEY = '395644cd1afc16115605'

PINATA_SECRET_API_KEY =
'33f6e83cb40f066e4590f50a5a0733b0ccdc209c35cec8361352f26c5c2ab0f'

def upload_to_ipfs(file_path):

    url = "https://api.pinata.cloud/pinning/pinFileToIPFS"

    headers = {

        "pinata_api_key": PINATA_API_KEY,

        "pinata_secret_api_key": PINATA_SECRET_API_KEY

    }

    with open(file_path, 'rb') as file:

        response = requests.post(url, files={"file": file}, headers=headers)

    if response.status_code == 200:

        return response.json()['IpfsHash']

    else:

        print("IPFS upload error:", response.text)

        return None

# ===== Export to CSV =====

def export_to_csv(start_date=None, end_date=None):

    count = contract.functions.getRecordCount().call()

    with open("records.csv", "w", newline="") as f:

        writer = csv.writer(f)

        writer.writerow(["Timestamp", "DateTime", "Rotten", "Fresh"])

```

```

    for i in range(count):

        ts, r, fsh = contract.functions.getRecord(i).call()

        dt = datetime.datetime.fromtimestamp(ts)

        if start_date and end_date:

            if not (start_date <= dt.date() <= end_date):

                continue

            writer.writerow([ts, dt.strftime("%Y-%m-%d %H:%M:%S"), r, fsh])

@app.route('/')
def index():

    return render_template('index.html')

@app.route('/download-records')
def download_records():

    start = request.args.get('start')

    end = request.args.get('end')

    if start and end:

        start_date = datetime.datetime.strptime(start, "%Y-%m-%d").date()

        end_date = datetime.datetime.strptime(end, "%Y-%m-%d").date()

    else:

        start_date = end_date = None

    export_to_csv(start_date, end_date)

    return send_file("records.csv", as_attachment=True)

@app.route('/filter')
def filter_form():

    return render_template('filter_form.html')

@app.route('/esp32-live')

```

```

def esp32_live():

    threading.Thread(target=stream_yolo_from_esp32).start()

    webbrowser.open("http://127.0.0.1:5000/dashboard")

    return "\U0001f680 ESP32-CAM stream started."

def stream_yolo_from_esp32():

    cap = cv2.VideoCapture(ESP32_STREAM_URL)

    if not cap.isOpened():

        print("ESP32 not connected.")

        return

    while True:

        ret, frame = cap.read()

        if not ret:

            break

        frame = cv2.resize(frame, (320, 240))

        results = model(frame, conf=0.5)

        labels = []

        save_image = False

        if results[0].boxes.shape[0] > 0:

            for box in results[0].boxes:

                label = model.names[int(box.cls[0])]

                labels.append(label)

                if 'rotten' in label:

                    save_image = True

        if save_image:

            annotated_frame = results[0].plot()

```



```

count_rotten = sum(1 for l in labels if 'rotten' in l)

count_fresh = len(labels) - count_rotten

try:

    tx = contract.functions.updateCounts(count_rotten, count_fresh).transact({'from': account})

    w3.eth.wait_for_transaction_receipt(tx)

except Exception as e:

    print("Blockchain error:", e)

filename = f"{uuid.uuid4().hex}.jpg"

path = os.path.join(app.config['DASHBOARD_FOLDER'], filename)

cv2.imwrite(path, annotated_frame)

upload_to_ipfs(path)

display_frame = annotated_frame if save_image else frame

cv2.imshow("ESP32 + YOLO", display_frame)

if cv2.waitKey(1) == ord('q'):

    break

cap.release()

cv2.destroyAllWindows()

@app.route('/dashboard')

def dashboard():

    images = os.listdir(app.config['DASHBOARD_FOLDER'])

    images.sort(reverse=True)

    rotten_count = contract.functions.rottenCount().call()

    fresh_count = contract.functions.freshCount().call()

    return render_template('dashboard.html', images=images, rotten_count=rotten_count,
fresh_count=fresh_count)

```

```

@app.route('/dashboard_img/<filename>')

def dashboard_image(filename):

    return send_from_directory(app.config['DASHBOARD_FOLDER'], filename)

@app.route('/uploads/<filename>')

def uploaded_file(filename):

    return send_from_directory(app.config['UPLOAD_FOLDER'], filename)

@app.route('/result/<filename>')

def result_file(filename):

    return send_from_directory(app.config['RESULT_FOLDER'], filename)

# ===== Chạy Flask =====

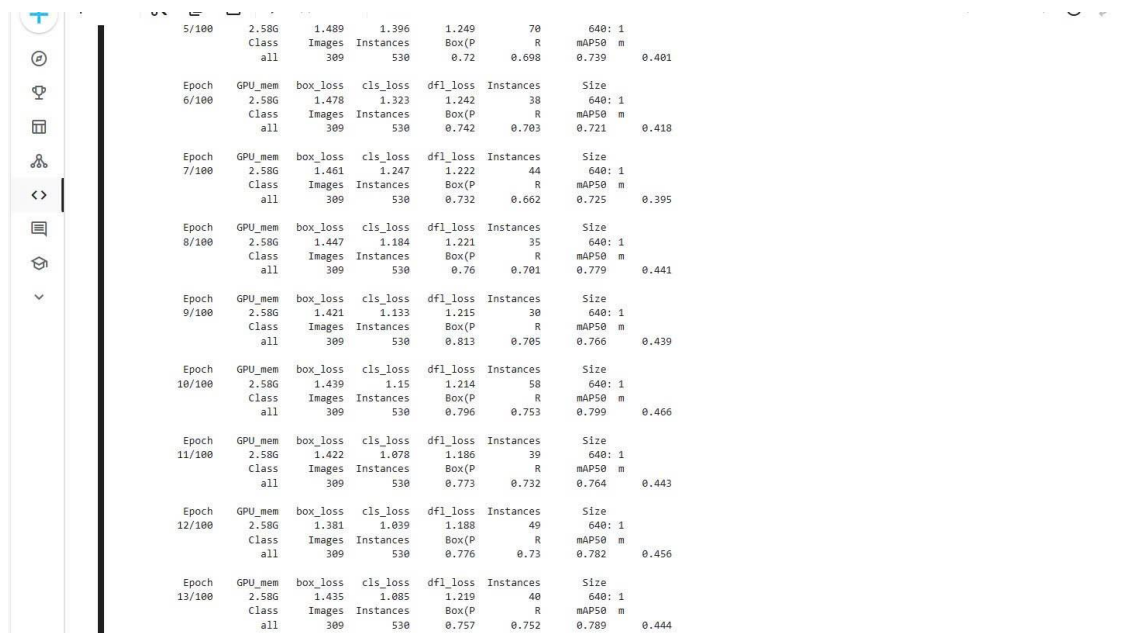
if __name__ == '__main__':

    app.run(host='127.0.0.1', port=5000, debug=True)

```

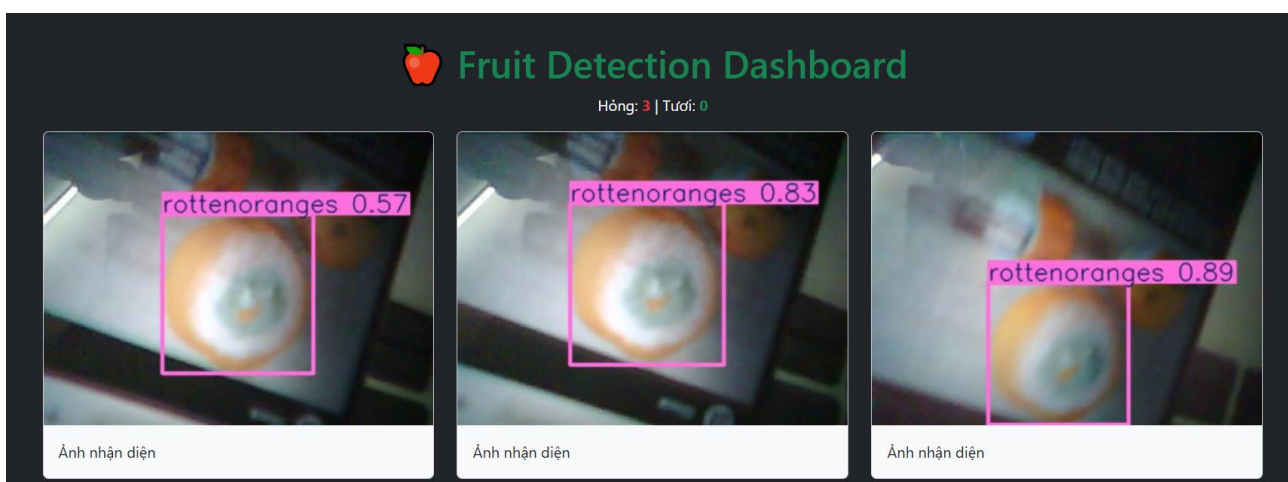
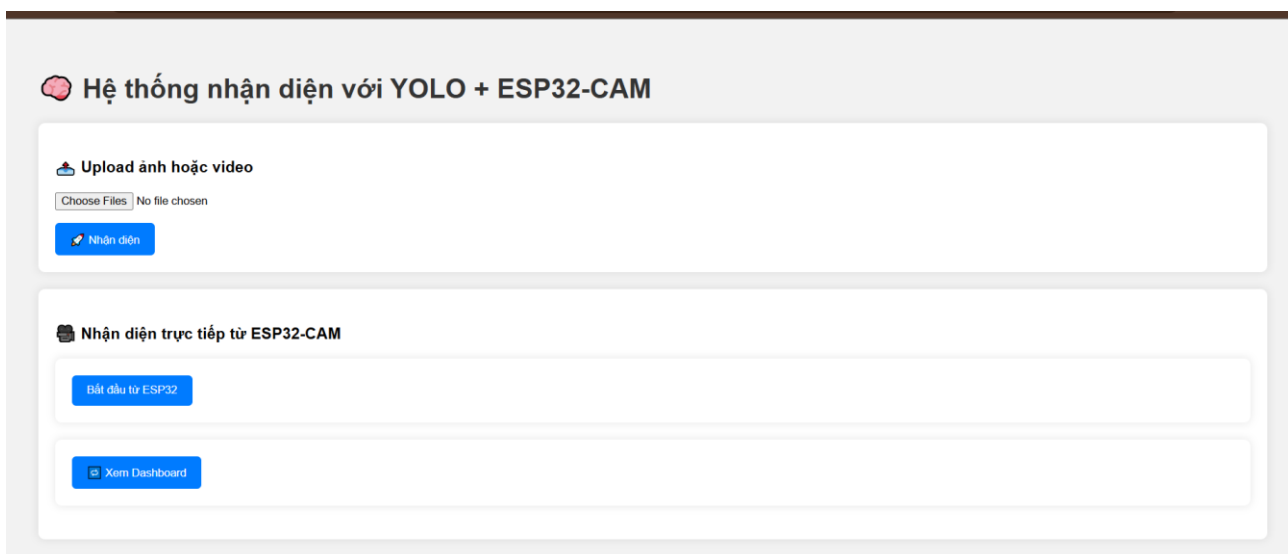
4.2. Thông số huấn luyện

Số liệu sau khi huấn luyện được các thông số dưới ảnh sau:



Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size	mAP50
5/100	2.586	1.489	1.396	1.249	70	640: 1	
Class		Images	Instances	Box(P	R	mAP50	m
all		309	530	0.72	0.698	0.739	0.401
Epoch 6/100	2.586	1.478	1.323	1.242	38	640: 1	
Class		Images	Instances	Box(P	R	mAP50	m
all		309	530	0.742	0.703	0.721	0.418
Epoch 7/100	2.586	1.461	1.247	1.222	44	640: 1	
Class		Images	Instances	Box(P	R	mAP50	m
all		309	530	0.732	0.662	0.725	0.395
Epoch 8/100	2.586	1.447	1.184	1.221	35	640: 1	
Class		Images	Instances	Box(P	R	mAP50	m
all		309	530	0.76	0.701	0.779	0.441
Epoch 9/100	2.586	1.421	1.133	1.215	30	640: 1	
Class		Images	Instances	Box(P	R	mAP50	m
all		309	530	0.813	0.705	0.766	0.439
Epoch 10/100	2.586	1.439	1.15	1.214	58	640: 1	
Class		Images	Instances	Box(P	R	mAP50	m
all		309	530	0.796	0.753	0.799	0.466
Epoch 11/100	2.586	1.422	1.078	1.186	39	640: 1	
Class		Images	Instances	Box(P	R	mAP50	m
all		309	530	0.773	0.732	0.764	0.443
Epoch 12/100	2.586	1.381	1.039	1.188	49	640: 1	
Class		Images	Instances	Box(P	R	mAP50	m
all		309	530	0.776	0.73	0.782	0.456
Epoch 13/100	2.586	1.435	1.085	1.219	40	640: 1	
Class		Images	Instances	Box(P	R	mAP50	m
all		309	530	0.757	0.752	0.789	0.444

4.3. Kết quả



4.4. Đánh giá kết quả

Kết quả nhận diện đúng nhưng vẫn sẽ có một vài yếu tố gây ảnh hưởng tới kết quả nhận diện như ánh sáng độ phân giải cam delay,.....

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

Đề tài được thực hiện nhằm mục đích cải thiện trong việc nhận diện hoa quả hỏng và tổng hợp lại nhằm mục đích tăng sự chính xác trong việc phân loại các loại sản phẩm hoa quả trong đời sống nhằm cải thiện đời sống.

5.2. Hướng phát triển

Tích hợp thêm đèn led và cảm biến báo động khi phát hiện ra hoa quả bị hỏng và tích hợp thêm blockchain để đẩy và lưu trữ dữ liệu vào các khối block nhằm tránh việc thay đổi kết quả và giúp cho trung thực hơn trong các hoạt động của các công ty.

KẾT LUẬN

Trong quá trình nghiên cứu và triển khai hệ thống ERP quản lý khách hàng trên nền tảng Odoo. Hệ thống không chỉ giúp tối ưu hóa quy trình quản lý thông tin khách hàng, hợp đồng, và các giao dịch kinh doanh mà còn tích hợp nhiều model hỗ trợ như phân tích khách hàng, ghi nhận phản hồi, theo dõi tương tác và quản lý cơ hội bán hàng. Qua đó, doanh nghiệp có thể nắm bắt được toàn bộ hành trình của khách hàng từ lúc tiếp cận đến chăm sóc, tạo điều kiện thuận lợi cho việc đưa ra các chiến lược kinh doanh chính xác.

Với khả năng phân tích sâu và báo cáo trực quan, ban lãnh đạo dễ dàng theo dõi tiến trình và đánh giá hiệu quả của từng bộ phận, từ đó có những điều chỉnh kịp thời để tối ưu hóa quy trình làm việc. Hơn nữa, còn tích hợp thêm các module quản lý nội bộ như nhân viên, nhiệm vụ và chiến lược marketing.

Tuy nhiên, hệ thống hiện tại vẫn tồn tại một số hạn chế cần được khắc phục. Một số module, đặc biệt là các chức năng phân tích khách hàng và dự đoán xu hướng, vẫn chủ yếu dựa trên dữ liệu định tính và chưa đạt được mức độ tự động hóa cao. Ngoài ra, giao diện người dùng mặc dù thân thiện nhưng vẫn có thể được cải thiện thêm về mặt trực quan để phù hợp hơn với các xu hướng thiết kế mới, nhằm nâng cao trải nghiệm của người dùng cuối.

Trong hướng phát triển tương lai, việc mở rộng tích hợp các công nghệ tiên tiến như trí tuệ nhân tạo (AI) và học máy (machine learning) sẽ là một bước đột phá quan trọng. AI có thể giúp dự đoán xu hướng, phân tích hành vi khách hàng một cách chính xác hơn, từ đó đưa ra các chiến lược marketing cá nhân hóa và nâng cao hiệu quả bán hàng. Bên cạnh đó, việc phát triển ứng dụng di động cho phép khách hàng và nhân viên truy cập hệ thống mọi lúc, mọi nơi sẽ góp phần cải thiện khả năng tương tác và nâng cao hiệu quả quản lý. Các module nội bộ khác cũng có thể được mở rộng, tích hợp với các công cụ quản lý dự án và CRM để tạo ra một hệ thống tổng thể hoàn thiện.

DANH MỤC TÀI LIỆU THAM KHẢO

- [1]. <https://www.odoo.com/documentation>
- [2]. <https://ubuntu.com/server/docs>
- [3]. Sumner, M. (2005). Enterprise Resource Planning (ERP): The Dynamics of Operations Management
- [4]. Bradford, M. (2010). Modern ERP: Select, Implement, and Use Today's Advanced Business Systems
- [5]. Leon, A. (2008). ERP Demystified
- [6]. Odoo Community Association (OCA)

