

BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
BỘ GIÁO DỤC VÀ ĐÀO TẠO

PHÂN HIỆU TRƯỜNG ĐẠI HỌC THỦY LỢI
BỘ MÔN CÔNG NGHỆ THÔNG TIN



BÁO CÁO MÔN HỌC MÁY

Tên đề tài:

**ỨNG DỤNG HỌC MÁY TRONG PHÂN LOẠI LÁ CÂY BỆNH
QUA HÌNH ẢNH**

Giảng viên hướng dẫn:	<i>ThS. Vũ Thị Hạnh</i>
Nhóm:	<i>6</i>
Sinh viên thực hiện:	<i>Hứa Khánh Duy Đinh Văn Nghĩa Lê Trí Thiện</i>
Mssv:	<i>2351067090 2151067598 2351067115</i>
Lớp:	<i>S26-65CNTT</i>

2026

LỜI CẢM ƠN

Trong khoảng thời gian làm đồ án, em đã nhận được nhiều sự giúp đỡ, đóng góp ý kiến và sự dẫn dắt chỉ bảo nhiệt tình của giảng viên hướng dẫn - Thạc sĩ Vũ Thị Hạnh - Giảng viên môn Học máy. Em xin gửi lời cảm ơn chân thành đến cô vì đã tận tình hướng dẫn, chỉ bảo nhóm chúng em trong suốt quá trình thực hiện đồ án.

Với điều kiện về thời gian cũng như lượng kiến thức về đề tài rất rộng mà kinh nghiệm còn hạn chế của một học viên, đồ án này không thể tránh được những thiếu sót. Em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của cô để em có điều kiện bổ sung, nâng cao kiến thức của mình, phục vụ tốt hơn công tác thực tế sau này.

Em xin chân thành cảm ơn!

This image shows a full page of primary-ruled paper. It features multiple horizontal rows, each defined by two parallel dashed lines. The rows are evenly spaced across the entire page, providing a guide for handwriting practice. There are no margins, text, or other markings present.

Vũ Thị Hạnh

MỤC LỤC

Chương 1: Giới thiệu bài toán	6
1. Bối cảnh chọn đề tài:	6
2. Mục tiêu của đề tài:	6
Chương 2: Mô tả dữ liệu sử dụng và tiền xử lí.....	7
1. Mô tả dữ liệu:	7
2. Tiền xử lí dữ liệu:	7
Chương 3: Mô hình học máy áp dụng.....	12
1. Mô hình ResNet50:	12
2. Mô hình MobileNetV3Large:	20
3. Mô hình EfficientNetB0:	25
Chương 4: Đánh giá mô hình	33
1. Mô hình ResNet50:	33
2. Mô hình MoblieNetV3Large:	35
3. Mô hình EfficientB0:	39
4. Đánh giá kết quả giữa các mô hình:	43
Chương 5: Kết luận và hướng phát triển	44
5.1 Kết luận:	44
5.2 Hướng phát triển:	44
Chương 6: Tài liệu tham khảo	45
Chương 7: Phân công nhiệm vụ	46

MỤC LỤC BẢNG BIỂU

Bảng 1 : Kết quả huấn luyện ResNet50	16
Bảng 2 : So sánh kết quả	18
Bảng 3 : Kết quả huấn luyện MobileNetV3	23
Bảng 4 : So sánh kết quả huấn luyện và fine tuning MobileNetV3	24
Bảng 5 : Biểu đồ Acc và Loss MobileNetV3	24
Bảng 6 : Kết quả huấn luyện mô hình EfficientB0	29
Bảng 7 : Phân công nhiệm vụ	46

MỤC LỤC HÌNH ẢNH

Ảnh 1 : Ảnh bị quá sáng	7
Ảnh 2 : Hàm kiểm tra ảnh quá sáng	8
Ảnh 3 : Hàm kiểm tra ảnh quá sáng vừa	9
Ảnh 4 : Hàm kiểm tra ảnh mờ	9
Ảnh 5 : Hàm chỉnh sửa ảnh	10
Ảnh 6 : Kết quả lọc ảnh	11
Ảnh 7 : Ảnh bị quá sáng	11
Ảnh 8 : Kiến trúc mô hình ResNet50	13
Ảnh 9 : Biểu đồ Acc và Loss ResNet50	16
Ảnh 10 : Kết quả fine tuning ResNet50	17
Ảnh 11 : Biểu đồ Acc và Loss sau fine tuning ResNet50	18
Ảnh 12 : Kết quả thử nghiệm ResNet50	20
Ảnh 13 : Kiến trúc mạng MobileNetV3	21
Ảnh 14 : Biểu đồ Acc và Loss MobileNetV3	23
Ảnh 15 : Kết quả fine tuning MobileNetV3	24
Ảnh 16 : Kết quả thử nghiệm MobileNetV3	25
Ảnh 17 : Xây dựng mô hình EfficientB0	26
Ảnh 18 : Các tầng bổ sung EfficientB0	26
Ảnh 19 : Kiến trúc mô hình EfficientB0	27
Ảnh 20 : Biểu đồ Acc và Loss mô hình EfficientB0	31
Ảnh 21 : Kết quả thử nghiệm mô hình EfficientB0	32
Ảnh 22 : Ma trận nhiễu ResNet50	35
Ảnh 23 : Ma trận nhiễu mô hình MobileNetV3	38
Ảnh 24 : Tính giải thích mô hình MobileNetV3	39
Ảnh 25 : Ma trận nhiễu EfficientB0	42

Chương 1: Giới thiệu bài toán

1. Bối cảnh chọn đề tài:

Trong ngành nông nghiệp, các vấn đề về bệnh trên lá cây gây ảnh hưởng không nhỏ đến năng suất và chất lượng cây trồng. Chính vì thế, việc phát hiện và chuẩn đoán các bệnh về lá cây là một việc làm hết sức cần thiết và cần được ưu tiên.

Tuy nhiên, việc phát hiện và chuẩn đoán bệnh về lá đòi hỏi người nông dân hay chuyên gia phải có kiến thức, kinh nghiệm về cây trồng và các loại sâu bệnh. Công việc này vẫn tìm ẩn nhiều điểm hạn chế như: tốn thời gian, sự nhầm lẫn giữa các loại bệnh, khó áp dụng rộng rãi trong trồng trọt,...

Sự phát triển bùng nổ của Trí tuệ nhân tạo (AI) nói chung và Học máy (ML) nói riêng đã mở ra con đường mới trong việc tự động nhận dạng bệnh trên lá cây bằng hình ảnh. Bằng cách huấn luyện các mô hình bằng tập dữ liệu hình ảnh về lá cây bệnh, mô hình có thể đưa ra dự đoán nhanh chóng, giúp người nông dân chủ động trong công tác phòng, chống bệnh trên cây trồng.

2. Mục tiêu của đề tài:

Trong đề tài này, nhóm chúng em sử dụng tập dữ liệu gồm 38 lớp ảnh bệnh lá của nhiều loài cây khác nhau đã được gán nhãn sẵn. Mục tiêu là xây dựng và đánh giá các mô hình học máy có khả năng phân loại chính xác các loại bệnh từ ảnh lá.

Đề tài bao gồm phân tích và tiền xử lý dữ liệu, xây dựng mô hình, dự đoán bằng tập dữ liệu kiểm tra và đánh giá mô hình.

Kết quả nghiên cứu không chỉ mang ý nghĩa học thuật mà còn có tiềm năng ứng dụng thực tế trong nông nghiệp hiện đại, góp phần nâng cao hiệu quả, chất lượng trong nông nghiệp.

Chương 2: Mô tả dữ liệu sử dụng và tiền xử lý

1. Mô tả dữ liệu:

Nguồn dữ liệu: Bộ dữ liệu được lấy từ Kaggle (New Plant Diseases Dataset - Samir Bhattarai)

*Mô tả dữ liệu:

- Dữ liệu gồm khoảng 87 ngàn ảnh của lá cây khỏe mạnh và lá cây bệnh của 13 loài khác nhau.

- Số lượng lớp: 38, mỗi lớp là một bệnh của lá.

- Bộ dữ liệu gồm 3 tập:

+ Train: chứa dữ liệu huấn luyện, ảnh được phân chia vào các class.

+ Valid: chứa dữ liệu kiểm định, ảnh được phân chia vào các class.

+ Test: chứa dữ liệu kiểm thử, ảnh chưa được phân chia vào các class.

- Định dạng ảnh: ảnh màu, đuôi JPG, cùng kích thước (256x256)

2. Tiền xử lý dữ liệu:

2.1 Vấn đề bộ dữ liệu:

Trong quá trình khảo sát bộ dữ liệu New Plant Diseases Dataset, nhóm nhận thấy có một vấn đề nghiêm trọng ảnh hưởng đến chất lượng mô hình, đó là hiện tượng quá sáng (Overexposure) và ảnh mờ. Có Nhiều ảnh chụp lá cây (đặc biệt là lá khỏe mạnh) bị lóa do đèn flash hoặc ánh nắng trực tiếp, làm mất các đặc trưng vân lá và màu sắc.



Ảnh 1: Ảnh bị quá sáng

2.2 Giải pháp xử lý:

Việc loại bỏ tốn rất nhiều thời gian (do có nhiều ảnh) nên được xử lý bằng thuật toán lọc tự động. Nhóm đã xây dựng một thuật toán lọc tự động dựa trên phân phối điểm ảnh (Pixel Distribution) với thư viện TensorFlow:

- Ngưỡng điểm sáng (Threshold): Các pixel có giá trị > 240 (gần màu trắng tuyệt đối) được coi là điểm cháy sáng.
- Tỷ lệ chấp nhận: Nếu một bức ảnh có hơn 30% diện tích là điểm cháy sáng thì ảnh đó sẽ bị coi là nhiễu và loại.

+ BRIGHT_PIXEL_THRESHOLD = 240

+ BRIGHT_RATIO_THRESHOLD = 0.3

a) Hàm kiểm tra ảnh quá sáng:

- Input: Ảnh màu đầu vào được đọc từ tệp ảnh, biểu diễn dưới dạng ma trận 3 kênh (RGB/BGR).

- Output: + True/False: ảnh có bị chói hay không.
 + bright_ratio: tỷ lệ pixel sáng để tham khảo.

- Ý tưởng thuật toán:

+ Với mỗi img được đọc từ dữ liệu nhị phân từ tệp ảnh, ta giải mã dữ liệu thành ảnh màu với 3 kênh RGB.

· + Sau đó, chuyển ảnh thành xám để đánh giá độ sáng của từng pixel.

+ So sánh từng pixel với ngưỡng điểm sáng (240).

+ Tính tỉ lệ pixel sáng trong ảnh, nếu lớn hơn tỉ lệ chấp nhận thì ảnh được coi là ảnh quá sáng.

- Hàm cài đặt:

```
def is_very_bright(img):  
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  
    ratio = np.sum(gray >= BRIGHT_PIXEL_THRESHOLD) / gray.size  
    return ratio > BRIGHT_RATIO_THRESHOLD
```

Ảnh 2: Hàm kiểm tra ảnh quá sáng

b) Hàm kiểm tra ảnh bị quá sáng trung bình:

Tương tự hàm kiểm tra ảnh quá sáng, hàm kiểm tra ảnh quá sáng trung bình để lọc ảnh bị quá sáng ở mức chấp nhận được với ngưỡng chấp nhận 5% - 30%. Những ảnh này sẽ được xử lý giảm độ sáng.

```
def is_medium_bright(img):
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    ratio = np.sum(gray >= BRIGHT_PIXEL_THRESHOLD) / gray.size
    return 0.05 < ratio <= BRIGHT_RATIO_THRESHOLD
```

Ảnh 3: Hàm kiểm tra ảnh quá sáng vừa

c) Hàm kiểm tra ảnh mờ:

- Input:

+ Ảnh màu đầu vào được đọc từ tệp ảnh, biểu diễn dưới dạng ma trận 3 kênh (RGB/BGR).

+ thresh: ngưỡng ảnh mờ, mặc định 120.

- Output:

+ True / False: ảnh có bị mờ hay không (True nếu ảnh bị mờ).

+ Giá trị ngưỡng thresh được sử dụng để so sánh mức độ sắc nét của ảnh.

- Ý tưởng thuật toán:

+ Ảnh đầu vào được chuyển đổi từ không gian màu BGR sang ảnh xám nhằm loại bỏ thông tin màu sắc và tập trung vào cấu trúc hình học của ảnh.

+ Toán tử Laplacian được áp dụng lên ảnh xám để làm nổi bật các vùng có sự thay đổi cường độ sáng mạnh, tương ứng với các cạnh và chi tiết trong ảnh.

+ Phương sai của ảnh Laplacian được tính toán để đo mức độ phân bố của các cạnh trong ảnh.

+ Nếu giá trị phương sai Laplacian nhỏ hơn ngưỡng thresh đã định nghĩa trước, ảnh được coi là ảnh bị mờ do thiếu các biên sắc nét.

```
def is_blurry(img, thresh=120):
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    return cv2.Laplacian(gray, cv2.CV_64F).var() < thresh
```

Ảnh 4: Hàm kiểm tra ảnh mờ

d) Hàm chỉnh sửa ảnh quá sáng trung bình và ảnh mờ:

- Input: Ảnh màu đầu vào img được đọc từ tệp ảnh, biểu diễn dưới dạng ma trận 3 kênh (BGR).

- Output: Ảnh đã được cải thiện chất lượng, bao gồm giảm chói, tăng cường độ tương phản cục bộ và làm rõ chi tiết.

- Ý tưởng thuật toán:

+ Ảnh đầu vào được chuyển đổi từ không gian màu BGR sang không gian màu LAB nhằm tách riêng thành phần độ sáng (kênh L) và thông tin màu sắc (kênh A, B).

+ Kênh độ sáng L được giảm nhẹ cường độ bằng cách nhân với hệ số nhỏ hơn 1 nhằm hạn chế hiện tượng chói sáng.

+Phương pháp CLAHE (Contrast Limited Adaptive Histogram Equalization) được áp dụng lên kênh L để tăng cường độ tương phản cục bộ, giúp làm rõ các chi tiết trong vùng tối và sáng mà không gây bão hòa.

+ Ảnh sau khi điều chỉnh độ sáng được ghép lại với các kênh màu ban đầu và chuyển ngược về không gian màu BGR.

+ Tiếp theo, bộ lọc Gaussian Blur được sử dụng để tạo ảnh mờ, sau đó kết hợp với ảnh gốc bằng phép cộng có trọng số (unsharp masking) nhằm tăng độ sắc nét của ảnh.

+ Ảnh kết quả cuối cùng có độ sáng cân bằng hơn, chi tiết rõ ràng hơn.

```
def fix_image(img):  
    lab = cv2.cvtColor(img, cv2.COLOR_BGR2LAB)  
    l, a, b = cv2.split(lab)  
  
    l = np.clip(l * 0.85, 0, 255).astype(np.uint8)  
    clahe = cv2.createCLAHE(clipLimit=1.2, tileGridSize=(8,8))  
    l = clahe.apply(l)  
  
    img = cv2.merge((l, a, b))  
    img = cv2.cvtColor(img, cv2.COLOR_LAB2BGR)  
  
    blur = cv2.GaussianBlur(img, (0,0), 1.0)  
    img = cv2.addWeighted(img, 1.3, blur, -0.3, 0)  
  
    return img
```

Ảnh 5: Hàm chỉnh sửa ảnh

2.2 Kết quả xử lý:

Số ảnh quá sáng được thống kê qua từng lớp

Peach__Bacterial_spot: 0/1838 ảnh chói
 Peach__healthy: 281/1728 ảnh chói
 Pepper,_bell__Bacterial_spot: 0/1913 ảnh chói
 Pepper,_bell__healthy: 205/1988 ảnh chói
 Potato__Early_blight: 0/1939 ảnh chói
 Potato__healthy: 125/1824 ảnh chói
 Potato__Late_blight: 0/1939 ảnh chói
 Raspberry__healthy: 294/1781 ảnh chói
 Soybean__healthy: 4/2022 ảnh chói
 Squash__Powdery_mildew: 0/1736 ảnh chói
 Strawberry__healthy: 0/1824 ảnh chói
 Strawberry__Leaf_scorch: 0/1774 ảnh chói
 Tomato__Bacterial_spot: 0/1702 ảnh chói
 Tomato__Early_blight: 0/1920 ảnh chói

Ảnh 6: Kết quả lọc ảnh

Thuật toán đã phát hiện và loại bỏ 1.919 ảnh lỗi khỏi tập huấn luyện.

Phân bố nhiều: Các ảnh bị loại bỏ tập trung chủ yếu ở các lớp lá khỏe (Healthy) như Raspberry_healthy, Peach_healthy và Tomato_mosaic_virus,.. Điều này tương đối phù hợp với thực tế là bề mặt lá khỏe thường trơn bóng, dễ phản xạ ánh sáng.



Ảnh 7: Ảnh bị quá sáng

*TỔNG KẾT:

- Tổng ảnh quá chói: 1919
- Tổng ảnh được fix: 2265
- Tổng ảnh bình thường: 66111
- Tổng ảnh dùng để training: 68376

Chương 3: Mô hình học máy áp dụng

Trong đề tài, nhóm chọn 3 mô hình ResNet50, MobileNetV3Large và EfficientNetB0.

* Giới thiệu về các mô hình:

ResNet (Residual Network) nổi bật với cơ chế skip connections (residual blocks) giúp huấn luyện mạng sâu mà không gặp vấn đề vanishing gradient. ResNet50 có 50 tầng, là một trong những phiên bản phổ biến nhất. Với khả năng trích xuất đặc trưng mạnh mẽ, phù hợp cho các bài toán phân loại ảnh phức tạp. Được cộng đồng nghiên cứu sử dụng rộng rãi như một baseline. Với 38 lớp bệnh cây trồng, ResNet50 giúp đảm bảo độ chính xác cao.

MobileNetV3 là thế hệ tiếp theo của MobileNet, kết hợp Neural Architecture Search (NAS) và squeeze-and-excitation modules để tối ưu hóa hiệu suất trên thiết bị di động. Ưu điểm của mô hình là nhẹ, tốc độ nhanh, dung lượng nhỏ, nhưng vẫn giữ độ chính xác cao, phù hợp triển khai thực tế trên smartphone hoặc thiết bị IoT trong nông nghiệp, nơi cần mô hình vừa chính xác vừa tiết kiệm tài nguyên.

EfficientNet đưa ra phương pháp compound scaling để cân bằng độ sâu, độ rộng và độ phân giải của mạng. EfficientNetB0 là phiên bản cơ sở, nhẹ nhưng hiệu quả, đảm bảo hiệu suất cao với chi phí tính toán thấp, thường đạt accuracy vượt trội so với các mô hình cùng kích thước.

1. Mô hình ResNet50:

1.1 Cấu hình dữ liệu và tăng cường dữ liệu (Data Augmentation):

Dữ liệu sau khi làm sạch được chia thành 3 tập: Train, Validation và Test. Để chống học vẹt (Overfitting), nhóm áp dụng các kỹ thuật tăng cường dữ liệu ngay trong kiến trúc mạng.

RandomFlip ("horizontal"): Lật ảnh ngẫu nhiên theo trục ngang, giúp mô hình học được tính bất biến về hướng của lá cây

RandomRotation (0.1): Xoay ảnh ngẫu nhiên trong khoảng biên độ 10% (khoảng 36 độ), mô phỏng các góc chụp khác nhau trong thực tế

Chuẩn hóa (Normalization): Sử dụng lớp Lambda(preprocess_input) đặc thù của ResNet50 để chuyển đổi giá trị pixel (Zero-center) phù hợp với phân phối dữ liệu ImageNet mà mô hình gốc đã học.

1.2 Kiến trúc mạng (Model Architecture):

Nhóm sử dụng kỹ thuật Học chuyển giao (Transfer Learning) với kiến trúc nền tảng là ResNet50.

Khởi tạo Model ResNet50 cho 38 phân lớp...
 - Chế độ: Transfer Learning (Đóng băng Base Model)

Model: "ResNet50_TransferLearning"

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 224, 224, 3)	0	-
data_augmentation (Sequential)	(None, 224, 224, 3)	0	input_layer[0][0]
get_item (GetItem)	(None, 224, 224)	0	data_augmentatio...
get_item_1 (GetItem)	(None, 224, 224)	0	data_augmentatio...
get_item_2 (GetItem)	(None, 224, 224)	0	data_augmentatio...
stack (Stack)	(None, 224, 224, 3)	0	get_item[0][0], get_item_1[0][0], get_item_2[0][0]
add (Add)	(None, 224, 224, 3)	0	stack[0][0]
resnet50 (Functional)	(None, 7, 7, 2048)	23,587,712	add[0][0]
global_avg_pooling (GlobalAveragePool...	(None, 2048)	0	resnet50[0][0]
dropout (Dropout)	(None, 2048)	0	global_avg_pooli...
dense_256 (Dense)	(None, 256)	524,544	dropout[0][0]
dropout_1 (Dropout)	(None, 256)	0	dense_256[0][0]
output_layer (Dense)	(None, 38)	9,766	dropout_1[0][0]

Ảnh 8: Kiến trúc mô hình ResNet50

Kiến trúc Mô hình (Đã cập nhật theo thực tế)

1. Backbone (Phần thân):

Sử dụng ResNet50 (Weights = ImageNet).

Trạng thái: Đóng băng (Freeze) toàn bộ các lớp để giữ lại kiến thức trích xuất đặc trưng đã học từ 14 triệu ảnh ImageNet (Tổng tham số: ~23.5 triệu).

2. Classification Head (Phần đầu - Nhóm tự thiết kế): Model được bổ sung các lớp sau để phù hợp với bài toán phân loại 38 bệnh: GlobalAveragePooling2D: Giảm chiều dữ liệu từ (7, 7, 2048) xuống (2048), thay thế cho Flatten giúp giảm lượng tham số khổng lồ và hạn chế Overfitting.

Dropout (Lớp 1): Tắt ngẫu nhiên nơ-ron (thường là 0.5) để giảm sự phụ thuộc giữa các đặc trưng

Dense (256 nơ-ron): (Đây là lớp bạn thiếu) Lớp kết nối đầy đủ giúp mô hình học các tổ hợp đặc trưng phức tạp từ ResNet trước khi đưa ra quyết định.

Dropout (Lớp 2): Tiếp tục tắt ngẫu nhiên nơ-ron (thường là 0.2) để tăng cường khả năng tổng quát hóa

Output Layer (Dense): Gồm 38 nơ-ron (tương ứng 38 lớp bệnh) với hàm kích hoạt Softmax để đưa ra xác suất dự đoán cuối cùng

- Dense (256, ReLU): Lớp ẩn trung gian giúp tổng hợp đặc trưng.
- Dense (Output, Softmax): Lớp đầu ra với 38 nơ-ron tương ứng 38 loại bệnh

1.3 Huấn luyện mô hình:

a) Thiết lập huấn luyện:

Optimizer: Sử dụng Adam với learning rate khởi điểm là $1e-4$ (0.0001). Đây là tốc độ học an toàn, giúp mô hình hội tụ ổn định mà không phá vỡ các đặc trưng đã học từ trước

Loss Function: Sử dụng Sparse Categorical Crossentropy (phù hợp cho bài toán phân loại đa lớp với nhãn dạng số nguyên)

Chiến thuật Callbacks (Hệ thống giám sát tự động):

ModelCheckpoint: Tự động theo dõi val_loss và chỉ lưu lại phiên bản model có kết quả tốt nhất (save_best_only=True), đảm bảo không bị mất kết quả nếu quá trình huấn luyện bị gián đoạn hoặc model bị suy thoái ở các epoch cuối

ReduceLROnPlateau: Tự động giảm tốc độ học (nhân với hệ số 0.2) nếu val_loss không cải thiện sau 3 Epoch, giúp model tinh chỉnh kỹ hơn khi gặp "điểm nghẽn"

EarlyStopping: Tự động dừng huấn luyện sớm nếu model không cải thiện val_loss sau 5 Epoch liên tiếp (Patience = 5). Cơ chế này giúp tiết kiệm thời gian và ngăn chặn hiện tượng Overfitting.

b) Kết quả huấn luyện:

Giai Đoạn	Epoch	Diễn Biến	Đánh giá
Khởi động và hội tụ nhanh	1-3	-Accuracy tăng vọt từ 61.5% -> 88.28%	- Mô hình tận dụng xuất sắc các trọng số Pre-

		<ul style="list-style-type: none"> - Validation Accuracy đạt mức rất cao ngay từ đầu: 90.27% -> 93.44% - Learning Rate: 1e-4 	<p>trained Từ imagenet</p> <ul style="list-style-type: none"> - Hiện tượng "học nhanh" xảy ra do phần thân (Backbone) trích xuất đặc trưng rất tốt, giúp phần đầu (Head) nhanh chóng phân loại đúng các lớp bệnh. - Không có dấu hiệu overfitting
Tăng Trưởng ổn định	4-7	<ul style="list-style-type: none"> - Validation Loss tiếp tục giảm sâu và đều đặn qua từng Epoch (từ 0.178 xuống 0.148) - Accuracy tiếp tục cải thiện lên mức 92.75% (Train) và 95.11% (Val). - Model liên tục phá kỷ lục và lưu Checkpoint mới sau mỗi Epoch 	<ul style="list-style-type: none"> - Quá trình huấn luyện diễn ra rất suôn sẻ. - Độ chính xác trên tập Validation (Thi thử) luôn cao hơn tập Train (Học), chứng tỏ mô hình có khả năng tổng quát hóa cực tốt nhờ các kỹ thuật Data Augmentation và Dropout đã áp dụng.
Đạt Đỉnh (Best model)	8	<ul style="list-style-type: none"> - Validation Loss chạm đáy ở mức thấp nhất: 0.1295 - Validation Accuracy đạt đỉnh cao nhất: 95.63% - Đây là thời điểm mô hình đạt trạng thái tốt nhất trong giai đoạn này 	<ul style="list-style-type: none"> - Epoch 8 được xác định là phiên bản tối ưu nhất (Best Model). - Hệ thống ModelCheckpoint đã lưu lại bộ trọng số tại thời điểm này để sử dụng cho các bước tiếp theo

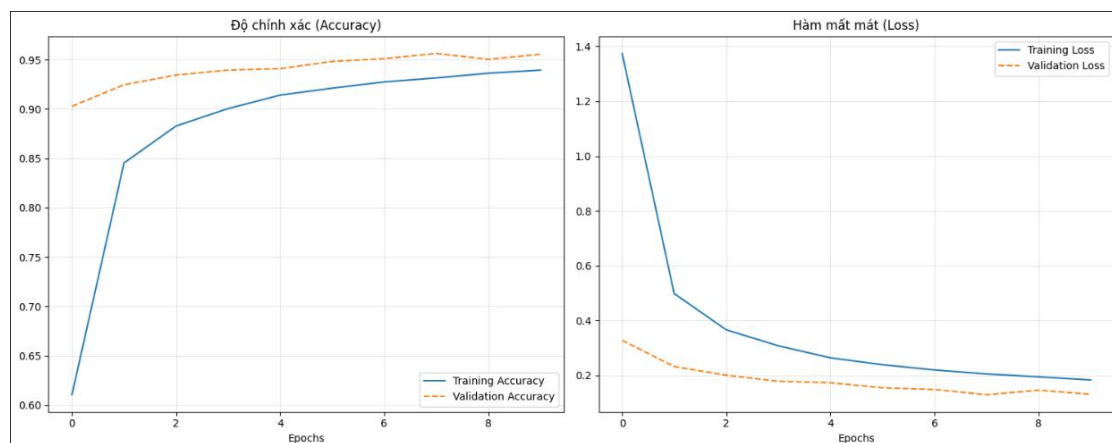
Bảo Hòa và Ổn định	9-10	<ul style="list-style-type: none"> - Validation Loss bắt đầu có dấu hiệu chững lại và tăng nhẹ (0.146 \rightarrow 0.131), không tốt hơn Epoch 8 - Accuracy vẫn duy trì ở mức cao (~95.5%) nhưng không cải thiện thêm về mặt Loss - Hệ thống tự động khôi phục trọng số của Epoch 8 khi kết thúc 	<ul style="list-style-type: none"> - Mô hình đã khai thác hết khả năng của việc "đóng băng" phần thân. - Dấu hiệu bão hòa cho thấy đã đến lúc cần chuyển sang giai đoạn Fine-tuning (mở khóa các lớp ẩn) để mô hình có thể học sâu hơn và giảm Loss thêm nữa.
--------------------	------	---	---

Bảng 1: Kết quả huấn luyện ResNet50

*Nhận xét:

Hiện tượng hội tụ: Mô hình hội tụ rất nhanh trong 3 epoch đầu (Accuracy tăng từ 61% lên 88%). Từ Epoch 4 đến Epoch 8, quá trình huấn luyện diễn ra cực kỳ thuận lợi, Loss giảm sâu liên tục mà chưa cần kích hoạt cơ chế giảm tốc độ học (Learning Rate vẫn duy trì ổn định ở mức $1e-4$).

Kết quả tốt nhất: Đạt được tại Epoch 8 với Validation Accuracy cao nhất là 95.63% và Validation Loss thấp kỷ lục là 0.1295. Hệ thống Early Stopping đã xác định đây là thời điểm mô hình tối ưu nhất trước khi có dấu hiệu bão hòa nhẹ ở Epoch 9-10



Ảnh 9: Biểu đồ Acc và Loss ResNet50

Đường Validation (cam) bám sát đường Training (xanh) chứng tỏ không xảy ra hiện tượng Overfitting.

c) Quá trình huấn luyện mô hình giai đoạn 2 - Fine-tuning:

Mặc dù mô hình đã đạt độ chính xác cao ở giai đoạn 1, nhưng ResNet50 vốn được huấn luyện trên ImageNet – một tập dữ liệu mang tính tổng quát (chó, mèo, xe cộ...). Trong khi đó, dữ liệu của bài toán là ảnh lá cây với các đặc trưng bệnh học rất chi tiết và tinh tế

Vì vậy, nhóm tiến hành mở khóa (Unfreeze) 50 lớp cuối cùng của ResNet50 và huấn luyện lại với tốc độ học cực nhỏ ($1e-5$) nhằm:

Chuyên biệt hóa mô hình: Điều chỉnh các trọng số ở các tầng cao (High-level features) để mô hình "hiểu" sâu sắc hơn về các vết bệnh cụ thể thay vì các hình khối chung chung

Bảo toàn kiến thức: Giữ nguyên các đặc trưng cơ bản (cạnh, góc, màu sắc) ở các tầng đầu đã được đóng băng

Tối ưu hóa cực đại: Kết quả thực tế cho thấy quá trình này giúp độ chính xác trên tập Validation tăng vọt từ ~95% lên mức gần như tuyệt đối 98.88% (tại Epoch 5), đồng thời giảm Loss xuống mức cực thấp 0.0336.

```
Epoch 1: finished saving model to best_resnet50.keras
1710/1710 — 2139s 1s/step - accuracy: 0.9342 - loss: 0.2056 - val_accuracy: 0.9660 - val_loss: 0.0993 - learning_rate: 1.0000e-05
Epoch 2/5
1710/1710 — 0s 1s/step - accuracy: 0.9639 - loss: 0.1071
Epoch 2: val_loss improved from 0.09926 to 0.06837, saving model to best_resnet50.keras

Epoch 2: finished saving model to best_resnet50.keras
1710/1710 — 2164s 1s/step - accuracy: 0.9668 - loss: 0.0989 - val_accuracy: 0.9764 - val_loss: 0.0684 - learning_rate: 1.0000e-05
Epoch 3/5
1710/1710 — 0s 1s/step - accuracy: 0.9759 - loss: 0.0726
Epoch 3: val_loss improved from 0.06837 to 0.04602, saving model to best_resnet50.keras

Epoch 3: finished saving model to best_resnet50.keras
1710/1710 — 2118s 1s/step - accuracy: 0.9773 - loss: 0.0678 - val_accuracy: 0.9841 - val_loss: 0.0460 - learning_rate: 1.0000e-05
Epoch 4/5
1710/1710 — 0s 1s/step - accuracy: 0.9811 - loss: 0.0570
Epoch 4: val_loss improved from 0.04602 to 0.04102, saving model to best_resnet50.keras

Epoch 4: finished saving model to best_resnet50.keras
1710/1710 — 2151s 1s/step - accuracy: 0.9831 - loss: 0.0502 - val_accuracy: 0.9865 - val_loss: 0.0410 - learning_rate: 1.0000e-05
Epoch 5/5
1710/1710 — 0s 1s/step - accuracy: 0.9857 - loss: 0.0426
Epoch 5: val_loss improved from 0.04102 to 0.03362, saving model to best_resnet50.keras

Epoch 5: finished saving model to best_resnet50.keras
1710/1710 — 2263s 1s/step - accuracy: 0.9869 - loss: 0.0397 - val_accuracy: 0.9888 - val_loss: 0.0336 - learning_rate: 1.0000e-05
Restoring model weights from the end of the best epoch: 5.
```

Ảnh 10: Kết quả fine tuning ResNet50

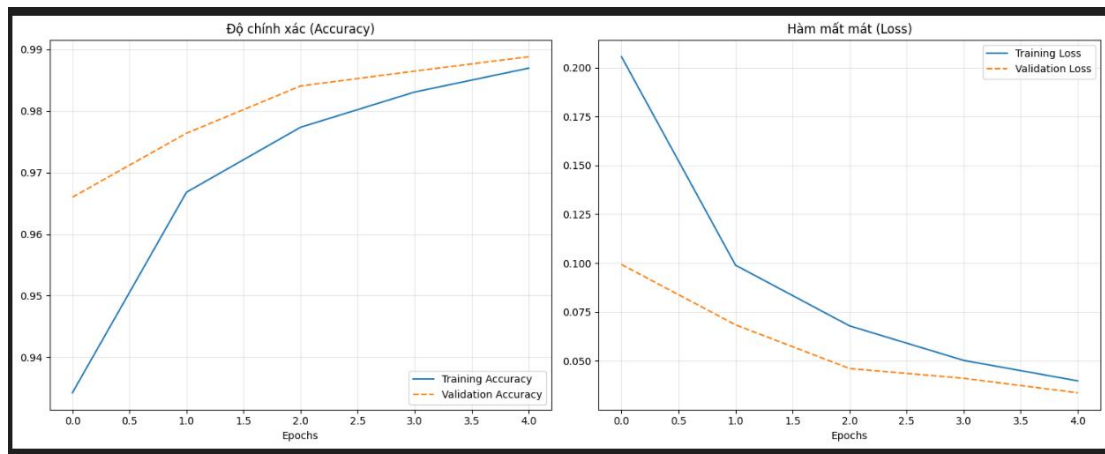
So sánh kết quả trước và sau khi fine-tuning 50 layer cuối:

Tiêu chí	Trước Fine-tuning	Sau Fine-tuning
Trạng thái mô hình	Đóng băng toàn bộ EfficientNetB0	Fine-tuning 50 layer cuối
Learning rate	$1e-4$ (0.0001)	$1e-5$ (0.00001)
Traing Accuracy	~93.23% (tại Epoch 8)	~98.69% (tại Epoch 5)
Validation Accuracy	~95.63% (tại Epoch 8)	~98.88% (tại Epoch 5)

		5))
Training Loss	~0.2051	~0.0397 (Giảm cực sâu)
Validation Loss	~0.1295	~0.0336 (Thấp kỷ lục)
Mức độ hội tụ	Tốt, ổn định sau 8 Epoch	Rất tốt, hội tụ cực nhanh chỉ sau 5 Epoch
Overfitting	Không xuất hiện (Val Acc > Train Acc)	Hoàn toàn không (Val Loss xấp xỉ Train Loss)
Khả năng phân biệt lớp	Mô hình hoạt động tốt nhưng còn dư địa để cải thiện	Mô hình đạt độ chính xác gần như tuyệt đối

Bảng 2: So sánh kết quả

Sau khi fine-tuning 50 layer cuối của ResNet50, mô hình cho thấy sự cải thiện vượt bậc về cả độ chính xác lẫn độ ổn định. Cụ thể, Validation Accuracy đã tăng vọt từ mức 95.6% lên 98.88% (xấp xỉ 99%), đồng thời Validation Loss giảm cực sâu xuống mức 0.0336. Điều này chứng minh rằng việc mở khóa các lớp sâu (Deep Layers) để tinh chỉnh đã giúp mô hình thích nghi hoàn toàn với đặc trưng của bộ dữ liệu lá cây, đạt hiệu quả vượt trội so với việc chỉ sử dụng Transfer Learning cơ bản.



Ảnh 11: Biểu đồ Acc và Loss sau fine tuning ResNet50

*Nhận xét:

- Về Accuracy:

+ Training accuracy tăng trưởng mạnh mẽ và ổn định qua từng epoch, từ mức 93.42% lên 98.69%.

+ Validation accuracy luôn duy trì cao hơn Training accuracy và đạt đỉnh điểm là 98.88%. Điều này rất hiếm gặp và là tín hiệu cực tốt, cho thấy mô hình tổng quát hóa xuất sắc trên dữ liệu chưa từng gặp.

+ Hai đường accuracy tiệm cận sát nhau và đi lên đồng đều, chứng tỏ quá trình học rất lành mạnh.

=> Đánh giá: Mô hình học ổn định, hiệu suất vượt trội và tuyệt đối không xuất hiện hiện tượng Overfitting (Validation Acc > Training Acc) sau khi fine-tuning.

- Về Loss:

+ Training loss giảm sâu từ 0.20 xuống mức rất thấp 0.039.

+ Validation loss cũng giảm liên tục và chạm đáy ở mức kỷ lục 0.0336.

+ Đường Validation loss đi xuống mượt mà và không có hiện tượng "ngóc đầu" tăng trở lại, khẳng định Epoch 5 là điểm dừng tối ưu nhất.

=> Đánh giá: Quá trình huấn luyện hội tụ tốt, mô hình tối ưu hiệu quả và không bị học quá mức.

1.4 Thử nghiệm thực tế:

Để kiểm chứng khả năng ứng dụng, nhóm đã chạy thử mô hình với các hình ảnh ngẫu nhiên

Kết quả nhận diện trên tập Demo: Độ chính xác tuyệt đối (100%): Mô hình nhận diện hoàn hảo 4/5 mẫu thử nghiệm với độ tin cậy tối đa. Cụ thể, mô hình phân biệt chính xác bệnh "Tomato Yellow Leaf Curl Virus" (Xoăn lá cà chua), "Corn Common Rust" (Gỉ sắt ngô) và nhận biết đúng lá "Tomato Healthy" (Cà chua khỏe mạnh)

Khả năng xử lý ổn định: Với mẫu bệnh "Tomato Early Blight" (Đốm sớm trên cà chua), mô hình vẫn đưa ra dự đoán chính xác với độ tin cậy rất cao là 95.4%. Kết quả này cho thấy mô hình hoạt động rất chắc chắn, không còn hiện tượng "lưỡng lự" (như mức 54.5% trong các thử nghiệm cũ) và có khả năng ứng dụng tốt vào thực tế.



Ảnh 12: Kết quả thử nghiệm ResNet50

1.1 Nhận xét:

Mô hình ResNet50 kết hợp với quy trình làm sạch dữ liệu (lọc ảnh chói) đã chứng minh hiệu quả vượt trội với độ chính xác khá cao ~96%. Mô hình có khả năng ứng dụng thực tế cao để hỗ trợ nông dân phát hiện bệnh sớm. Tuy nhiên, cần cải thiện thêm dữ liệu cho nhóm bệnh Cà chua để khắc phục sự nhầm lẫn giữa các bệnh đốm lá.

2. Mô hình MobileNetV3Large:

2.1 Cấu hình dữ liệu và tăng cường dữ liệu (Data Augmentation):

Để chống học vẹt (Overfitting), nhóm áp dụng các kỹ thuật tăng cường dữ liệu ngay trong kiến trúc mạng

- RandomFlip: Lật ảnh ngang
- RandomRotation(0.08): Xoay ngẫu nhiên 8%
- RandomZoom(0.08): Phóng to ngẫu nhiên 8%
- RandomBrightness(0.2): Tăng giảm độ sáng ngẫu nhiên 20%
- RandomContrast(0.1): Độ tương phản 10%

-Lambda(preprocess_input): Sử dụng hàm preprocess_input của MobileNetV3 để chuyển đổi dữ liệu để phù hợp với trọng số ImageNet.

2.2 Kiến trúc mạng:

Layer (type)	Output Shape	Param #
MobileNetV3Large (Functional)	(None, 7, 7, 960)	2,996,352
global_average_pooling2d_2 (GlobalAveragePooling2D)	(None, 960)	0
batch_normalization_2 (BatchNormalization)	(None, 960)	3,840
dense_4 (Dense)	(None, 256)	246,016
dropout_2 (Dropout)	(None, 256)	0
dense_5 (Dense)	(None, 38)	9,766

Ảnh 13: Kiến trúc mạng MobileNetV3

Classification Head (phần kiến trúc nhóm thiết kế thêm):

- GlobalAveragePooling2D: Giảm chiều dữ liệu, thay thế cho Flatten giúp giảm tham số.
- BatchNormalllization: Ổn định phân phối các feature, giúp train nhanh hơn.
- Dense (256, relu): Học đặc trưng cấp cao
- Dropout (0.5): Tắt ngẫu nhiên 50% nơ-ron để chống Overfitting.
- Dense (Output, Softmax): Lớp đầu ra với 38 nơ-ron tương ứng 38 loại bệnh.

2.3 Huấn luyện mô hình:

a) Thiết lập huấn luyện:

ModelCheckpoint: Chỉ lưu lại model tốt nhất.

ReduceLROnPlateau: Tự động giảm tốc độ học nếu Loss không giảm sau 3 Epoch.

EarlyStopping: Dừng sớm nếu model không cải thiện sau 3 Epoch.

b) Kết quả huấn luyện:

Giai đoạn	Diễn biến	Nhận xét
Bắt đầu (epoch 1)	Train accuracy: 71.9% Validation accuracy: 91.2% Train loss: 0.9875 Validation loss: 0.3108	Mô hình đang thích nghi với dữ liệu mới.

Học nhanh (epoch 2 - epoch 5)	<p>Train accuracy: tăng từ ~88% lên ~94%</p> <p>Validation accuracy: tăng từ ~93.7% lên ~95.7%</p> <p>Train loss: giảm mạnh từ ~0.39 về ~0.19</p> <p>Validation loss: giảm từ ~0.21 về ~0.14</p>	Train loss và validation loss đều giảm mạnh, cho thấy quá trình tối ưu diễn ra hiệu quả.
Ổn định (epoch 6 - epoch 10)	<p>Train accuracy: ~94.5% lên ~95.9%</p> <p>Validation accuracy: ~96.0% lên ~96.6%</p> <p>Train loss: ~0.17 giảm nhẹ xuống ~0.12</p> <p>Validation loss: ~0.13 giảm nhẹ xuống ~0.11</p>	Tốc độ cải thiện của các chỉ số bắt đầu chậm lại, cho thấy mô hình đã học được phần lớn đặc trưng quan trọng.
Dần hội tụ (epoch 11 - epoch 15)	<p>Train accuracy: ~96.1% tăng nhẹ ~96.8%</p> <p>Validation accuracy: ~96.6% lên ~97.0%</p> <p>Train loss: ~0.118 xuống ~0.097</p> <p>Validation loss: ~0.109 về ~0.100</p>	Mô hình tiến dần đến trạng thái hội tụ. Mức tăng accuracy không còn lớn, nhưng các chỉ số vẫn cải thiện đều.
Bão hòa (epoch 16 - epoch 20)	<p>Train accuracy: ~96.8% → ~97.2%</p> <p>Validation accuracy: ~97.1% → ~97.4%</p> <p>Train loss: ~0.094 → ~0.082</p> <p>Validation loss: ~0.096 → ~0.093</p>	Các chỉ số bắt đầu bão hòa, tốc độ cải thiện rất chậm. Valid accuracy dao động nhẹ nhưng vẫn duy trì xu hướng tăng
Hội tụ hoàn toàn (epoch 21 - epoch 30)	<p>Train accuracy: ~97.3% → ~97.7%</p>	Mô hình đã hội tụ hoàn toàn. Các chỉ số gần như không thay đổi đáng kể

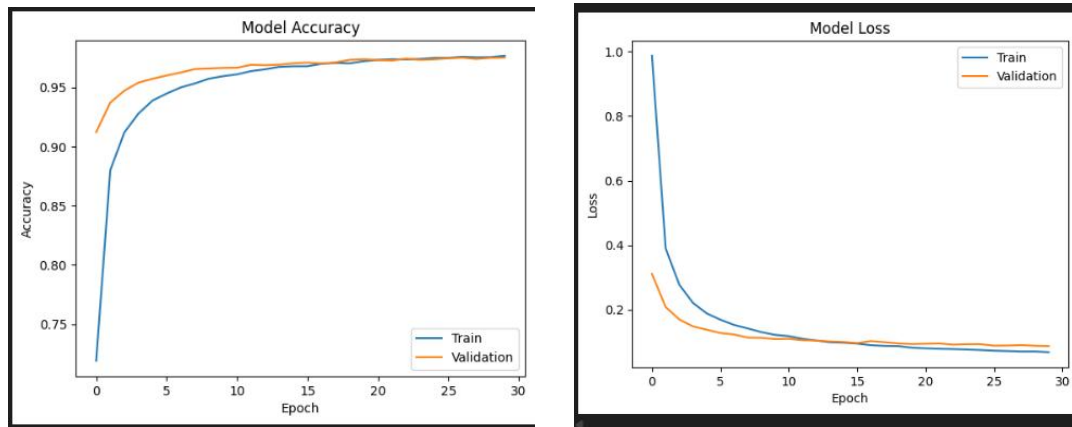
	Validation accuracy: $\sim 97.3\% \rightarrow \sim 97.5\%$ Train loss: $\sim 0.080 \rightarrow \sim 0.068$ Validation loss: $\sim 0.094 \rightarrow \sim 0.087$	giữa các epoch liên tiếp. Điều này cho thấy việc huấn luyện thêm sau epoch 25 mang lại cải thiện không đáng kể.
--	--	---

Bảng 3: Kết quả huấn luyện MobileNetV3

Kết quả tốt nhất:

Best Validation Accuracy (val_acc): 0.9752 (ở epoch 30).

Best Validation Loss (val_loss): 0.0869 (ở epoch 30).



Ảnh 14: Biểu đồ Acc và Loss MobileNetV3

Kết quả huấn luyện cho thấy mô hình đạt hiệu suất rất cao và ổn định. Cụ thể, độ chính xác (accuracy) của cả tập huấn luyện và tập kiểm tra đều tăng nhanh trong những epoch đầu và sau đó duy trì ở mức gần 0.97. Đồng thời, hàm mất mát (loss) giảm mạnh và ổn định ở mức thấp (~ 0.05), phản ánh quá trình học hiệu quả. Đáng chú ý, khoảng cách giữa đường huấn luyện và kiểm tra ở cả hai biểu đồ là rất nhỏ, chứng tỏ mô hình không bị hiện tượng overfitting và có khả năng tổng quát tốt.

c) Fine tuning:

Để nâng cao hiệu quả mô hình, nhóm thực hiện fine tuning cho mô hình MobileNetV3Large. Nhóm sẽ fine tuning với 30% lớp cuối của mô hình.

Đồng thời để tránh overfitting, nhóm sử dụng callback với các hàm Checkpoint, Early stop và ReduceLROnPlateau (các thông số tương tự ở phần huấn luyện) với 20 epoch.

Kết quả fine tuning: dừng sớm ở epoch 8.


```

Epoch 1/20      1274s 594ms/step - accuracy: 0.9778 - loss: 0.0659 - val_accuracy: 0.9751 - val_loss: 0.0872 - learning_rate: 1.0000e-05
Epoch 2/20      1272s 595ms/step - accuracy: 0.9786 - loss: 0.0625 - val_accuracy: 0.9748 - val_loss: 0.0874 - learning_rate: 1.0000e-05
Epoch 3/20      1328s 621ms/step - accuracy: 0.9788 - loss: 0.0634 - val_accuracy: 0.9748 - val_loss: 0.0877 - learning_rate: 1.0000e-05
Epoch 4/20      1396s 653ms/step - accuracy: 0.9787 - loss: 0.0633 - val_accuracy: 0.9747 - val_loss: 0.0871 - learning_rate: 1.0000e-05
Epoch 5/20      1376s 644ms/step - accuracy: 0.9786 - loss: 0.0622 - val_accuracy: 0.9752 - val_loss: 0.0865 - learning_rate: 1.0000e-05
Epoch 6/20      1408s 659ms/step - accuracy: 0.9791 - loss: 0.0616 - val_accuracy: 0.9753 - val_loss: 0.0869 - learning_rate: 1.0000e-05
Epoch 7/20      1337s 626ms/step - accuracy: 0.9788 - loss: 0.0622 - val_accuracy: 0.9755 - val_loss: 0.0868 - learning_rate: 1.0000e-05
Epoch 8/20      1337s 623ms/step - accuracy: 0.9808 - loss: 0.0579 - val_accuracy: 0.9758 - val_loss: 0.0875 - learning_rate: 1.0000e-05

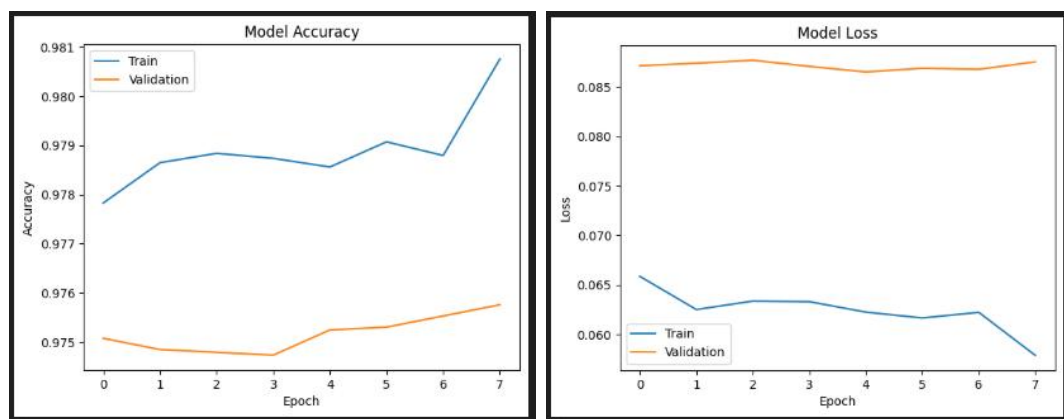
```

Ảnh 15: Kết quả fine tuning MobileNetV3

So sánh với phần huấn luyện:

Chỉ số	Huấn luyện	Fine tuning
Accuracy	Train Accuracy ~97% Valid Accuracy ~97%	Train Accuracy ~98% Valid Accuracy ~97%
Loss	Train Loss ~6% Valid Loss ~8%	Train Loss ~5% Valid Loss ~8%
Mức độ hội tụ	Tốt	Rất tốt, giảm nhẹ nhưng ổn định
Overfitting	Nhẹ Train loss < Val loss nhưng Val acc gần Train acc	Nhẹ

Bảng 4: So sánh kết quả huấn luyện và fine tuning MobileNetV3



Bảng 5: Biểu đồ Acc và Loss MobileNetV3

*Nhận xét:

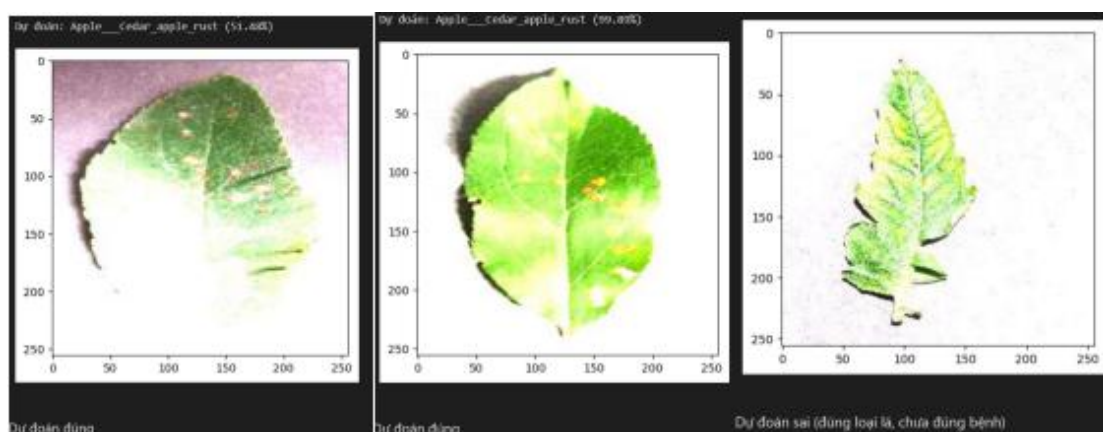
- Hiệu quả huấn luyện gốc: Model học nhanh trong vài epoch đầu (accuracy tăng nhanh, loss giảm mạnh). Validation accuracy và loss từ epoch 20 cho thấy model đã học hầu hết đặc trưng dataset.

- Hiệu quả fine-tuning: Training accuracy tăng nhẹ, loss training giảm thêm cho thấy model hội tụ tốt. Validation accuracy ổn định, có dấu hiệu overfitting nhẹ, nhưng chấp nhận được.

- Tổng thể: Cả hai giai đoạn đều hội tụ tốt, giảm loss đều. Overfitting nhẹ nhưng không nghiêm trọng vì validation metrics vẫn rất tốt. Fine-tuning giúp model đạt performance tối ưu, cân bằng giữa train và valid.

2.4 Thử nghiệm thực tế:

Để kiểm chứng khả năng ứng dụng, nhóm đã chạy thử mô hình với các hình ảnh ngẫu nhiên



Ảnh 16: Kết quả thử nghiệm MobileNetV3

Nhóm có chủ ý chọn những mẫu có ảnh bị quá sáng để thử mô hình, mô hình dự đoán khá đúng với độ tin cậy đáng kể. Tuy nhiên còn sai sót ở lớp cà chua, chưa dự đoán đúng lớp bệnh, điều này chứng tỏ dữ liệu thực tế dễ nhầm lẫn.

3. Mô hình EfficientNetB0:

3.1 Cấu trúc dữ liệu và tăng cường dữ liệu (Data Augmentation)

Để chống học vẹt (Overfitting), mô hình áp dụng các kỹ thuật tăng cường dữ liệu:

- Rotation_range=20: Xoay ảnh ngẫu nhiên khoảng ± 20 độ phù hợp với dữ liệu ảnh thực vật vì ảnh có thể được chụp ở nhiều hướng.

- Width_shift_range=0.1: Dịch chuyển ảnh theo chiều ngang tối đa 10% chiều rộng ảnh.

- Width_shift_range=0.1: Dịch chuyển ảnh theo chiều ngang tối đa 10% chiều rộng ảnh.

- Height_shift_range=0.1: Dịch chuyển ảnh theo chiều dọc tối đa 10% chiều cao ảnh.

- Zoom_range=0.2: Phóng to hoặc thu nhỏ ảnh trong phạm vi 20%.

- Horizontal_flip=True: lật ngang ảnh ngẫu nhiên giúp tăng gấp đôi số biến thể dữ liệu huấn luyện.

- Chuẩn hóa: Việc sử dụng preprocess_input là cần thiết khi áp dụng Transfer Learning với EfficientNetB0 nhằm đảm bảo dữ liệu đầu vào tương thích với mô hình pretrained.

Dữ liệu sau khi làm sạch được chia làm 3 tập: Train, validation và test. Sử dụng Hàm flow_from_directory được sử dụng để đọc ảnh từ các thư mục dữ liệu, tự động gán nhãn cho từng ảnh, chuẩn hóa kích thước và tạo dữ liệu theo từng batch để phục vụ quá trình huấn luyện và đánh giá mô hình.

3.2 Xây dựng mô hình Transfer Learning:

```
# Xây dựng mô hình học sâu sử dụng EfficientNetB0
base_model_eff = EfficientNetB0(
    weights="imagenet",
    include_top=False,
    input_shape=(IMG_SIZE, IMG_SIZE, 3)
)

base_model_eff.trainable = False
```

Ảnh 17: Xây dựng mô hình EfficientB0

Sử dụng EfficientNetB0 đã được huấn luyện trước trên tập ImageNet.

Loại bỏ tầng phân loại gốc (include_top=False).

Đóng băng toàn bộ các tầng trích xuất đặc trưng nhằm giúp quá trình huấn luyện nhanh hơn và tránh overfitting.

```
# Thêm các lớp tùy chỉnh vào mô hình
x = base_model_eff.output
x = GlobalAveragePooling2D()(x)
x = BatchNormalization()(x)
x = Dropout(0.5)(x)
x = Dense(256, activation="relu")(x)
x = Dropout(0.3)(x)

outputs = Dense(NUM_CLASSES, activation="softmax")(x)

model_eff = Model(inputs=base_model_eff.input, outputs=outputs)
```

Ảnh 18: Các tầng bổ sung EfficientB0

Bổ sung các tầng:

- Global Average Pooling: giảm số lượng tham số của mô hình, hạn chế hiện tượng overfitting so với Flatten.

- Dense layer: Học các đặc trưng phức tạp hơn từ dữ liệu

- Softmax cho bài toán phân loại đa lớp

Lựa chọn các siêu tham số như số neuron và tỷ lệ dropout được dựa trên kinh nghiệm thực nghiệm, việc sử dụng Global Average Pooling, Batch Normalization và Dropout giúp mô hình ổn định, giảm overfitting và đạt hiệu suất cao trên tập dữ liệu.

3.3 Kiến trúc mạng:

global_average_poo... (GlobalAveragePool...	(None, 1280)	0	top_activation[0...
batch_normalizatio... (BatchNormalizatio...	(None, 1280)	5,120	global_average_p...
dropout_4 (Dropout)	(None, 1280)	0	batch_normalizat...
dense_4 (Dense)	(None, 256)	327,936	dropout_4[0][0]
dropout_5 (Dropout)	(None, 256)	0	dense_4[0][0]
dense_5 (Dense)	(None, 38)	9,766	dropout_5[0][0]

Ảnh 19: Kiến trúc mô hình EfficientB0

Ngoài ra, nhóm còn bổ sung thêm các lớp tùy chỉnh vào mô hình:

- GlobalAveragePooling2D: Tổng hợp đặc trưng từ các feature map thành một vector 1280 chiều, giảm số lượng tham số.

- Dropout (dropout_4): Ngẫu nhiên loại bỏ 1 số neuron .

- Dense (dense_4): Lớp Dense chính giúp mô hình học các đặc trưng phức tạp thành thông tin phù hợp cho bài toán này.

- Dropout (dropout_5): Được sử dụng lại nhằm giảm overfitting.

- Dense (dense_5): Lớp đầu ra của mô hình với 38 neuron, tương ứng với 38 lớp bệnh.

3.4 Huấn luyện mô hình:

a) Quá trình huấn luyện mô hình giai đoạn 1 (Freeze base model):

Trong giai đoạn đầu, mô hình được huấn luyện với các tầng của EfficientNetB0 được đóng băng, chỉ huấn luyện các lớp phân loại phía trên. Quá trình huấn luyện diễn ra trong 10 epochs.

Kết quả:

```

Epoch 1/20
2137/2137 ————— 1244s 580ms/step - accuracy: 0.8072 - loss: 0.6464 - val_accuracy: 0.9312 - val_loss: 0.2054
Epoch 2/20
2137/2137 ————— 1343s 628ms/step - accuracy: 0.8753 - loss: 0.3992 - val_accuracy: 0.9389 - val_loss: 0.1851
Epoch 3/20
2137/2137 ————— 1464s 685ms/step - accuracy: 0.8866 - loss: 0.3607 - val_accuracy: 0.9454 - val_loss: 0.1680
Epoch 4/20
2137/2137 ————— 1426s 667ms/step - accuracy: 0.8908 - loss: 0.3410 - val_accuracy: 0.9446 - val_loss: 0.1692
Epoch 5/20
2137/2137 ————— 1426s 667ms/step - accuracy: 0.8927 - loss: 0.3349 - val_accuracy: 0.9559 - val_loss: 0.1399
Epoch 6/20
2137/2137 ————— 1419s 664ms/step - accuracy: 0.8969 - loss: 0.3269 - val_accuracy: 0.9554 - val_loss: 0.1372
Epoch 7/20
2137/2137 ————— 1466s 666ms/step - accuracy: 0.9017 - loss: 0.3150 - val_accuracy: 0.9570 - val_loss: 0.1357
Epoch 8/20
2137/2137 ————— 1427s 668ms/step - accuracy: 0.9009 - loss: 0.3160 - val_accuracy: 0.9574 - val_loss: 0.1335
Epoch 9/20
2137/2137 ————— 1429s 669ms/step - accuracy: 0.9036 - loss: 0.3082 - val_accuracy: 0.9575 - val_loss: 0.1299
Epoch 10/20
2137/2137 ————— 1437s 672ms/step - accuracy: 0.9067 - loss: 0.3011 - val_accuracy: 0.9558 - val_loss: 0.1390
Epoch 11/20
2137/2137 ————— 1423s 666ms/step - accuracy: 0.9071 - loss: 0.2971 - val_accuracy: 0.9601 - val_loss: 0.1261
Epoch 12/20
2137/2137 ————— 1430s 669ms/step - accuracy: 0.9076 - loss: 0.2953 - val_accuracy: 0.9602 - val_loss: 0.1222
Epoch 13/20
...
Epoch 19/20
2137/2137 ————— 1448s 678ms/step - accuracy: 0.9176 - loss: 0.2626 - val_accuracy: 0.9601 - val_loss: 0.1187
Epoch 20/20
2137/2137 ————— 1455s 681ms/step - accuracy: 0.9187 - loss: 0.2586 - val_accuracy: 0.9625 - val_loss: 0.1200

```

Kết quả huấn luyện cho thấy:

Giai đoạn	Diễn biến	Đánh giá
Khởi động từ epoch 1-5	<p>Ở giai đoạn epoch 1 -> 2 mô hình huấn luyện rất tốt khi cả 3 chỉ số đều diễn biến thuận lợi:</p> <p>+ Training accuracy tăng từ 0.8072 -> 0.8753</p> <p>+ Training loss giảm từ 0.6464 -> 0.3992</p> <p>+ Validation loss giảm từ 0.2054 -> 0.1851</p>	Mô hình bắt đầu học rất nhanh và ổn định, cải thiện rất rõ rệt với cả 3 tham số.
Epoch 5 – 10	<p>- val_accuracy tăng nhẹ và đều trong giai đoạn này</p> <p>- train_accuracy trong giai đoạn từ 6->7 tăng nhẹ nhưng đến giai đoạn 8 giảm xuống</p>	- Mô hình bắt đầu có hiện tượng bão hòa nhưng sau đó lại ổn định lại với tham số train_accuracy và val_loss chứng tỏ mô

	0.9009 rồi sau đó bắt đầu tăng lại đến epochs = 10 đạt 0.9067 - val_loss tương tự khi giảm từ 0.1372 xuống còn 0.1299 tại epochs = 9 rồi lại tăng lên 0.1390 tại epochs = 10	hình hội tụ khá tốt.
Epoch 11 -20	- training accuracy tăng từ 0.9071 lên tới 0.9176 - val_accuracy cũng tương tự khi tăng từ 0.9601 lên 0.9625 - traing_loss giảm đáng kể từ 0.2971 xuống còn 0.2586 - val_loss giảm từ 0.1261 -> 0.1187 nhưng đến epochs = 20 tăng nhẹ lên 0.1200	- Đây là giai đoạn ổn định nhất, các tham số có chỉ số tăng/giảm tốt, không xảy ra hiện tượng Overfitting. - Các tham số đều đạt được chỉ số rất tốt.

Bảng 6: Kết quả huấn luyện mô hình EfficientB0

*Đánh giá kết quả ban đầu:

- Mô hình học máy học ổn định qua từng epoch .
- Training loss và validation loss đều giảm chứng tỏ rằng mô hình hội tụ tốt.
- Không có dấu hiệu overfitting rõ rệt vì validation accuracy luôn cao hơn training accuracy.
- Validation accuracy = 0.9625 đạt kết quả tốt nhất tại cuối giai đoạn huấn luyện và validation loss = 0.1187 đạt kết quả thấp nhất tại epoch=19.

b) Quá trình huấn luyện mô hình giai đoạn 2 - Fine-tuning:

Mặc dù mô hình đã đạt độ chính xác cao, nhưng EfficientNetB0 được huấn luyện trên ImageNet – một tập dữ liệu mang tính tổng quát. Trong khi đó, dữ liệu của bài toán là ảnh thực vật với đặc trưng riêng.

Vì vậy, nhóm tiến hành fine-tuning 30 layer cuối của EfficientNetB0 nhằm:

- Điều chỉnh các đặc trưng cấp cao cho phù hợp hơn với dữ liệu thực tế.
- Giữ nguyên các đặc trưng tổng quát ở các tầng đầu.
- Tránh làm mô hình bị mất ổn định hoặc overfitting.


```

Epoch 1/20
2137/2137 ————— 2465s 1s/step - accuracy: 0.9167 - loss: 0.2778 - val_accuracy: 0.9756 - val_loss: 0.0838
Epoch 2/20
2137/2137 ————— 2341s 1s/step - accuracy: 0.9544 - loss: 0.1439 - val_accuracy: 0.9802 - val_loss: 0.0723
Epoch 3/20
2137/2137 ————— 1717s 803ms/step - accuracy: 0.9635 - loss: 0.1166 - val_accuracy: 0.9820 - val_loss: 0.0659
Epoch 4/20
2137/2137 ————— 1713s 801ms/step - accuracy: 0.9683 - loss: 0.0981 - val_accuracy: 0.9832 - val_loss: 0.0631
Epoch 5/20
2137/2137 ————— 1707s 799ms/step - accuracy: 0.9728 - loss: 0.0856 - val_accuracy: 0.9852 - val_loss: 0.0537
Epoch 6/20
2137/2137 ————— 1731s 810ms/step - accuracy: 0.9757 - loss: 0.0748 - val_accuracy: 0.9844 - val_loss: 0.0585
Epoch 7/20
2137/2137 ————— 1735s 812ms/step - accuracy: 0.9798 - loss: 0.0612 - val_accuracy: 0.9877 - val_loss: 0.0517
Epoch 8/20
2137/2137 ————— 1702s 796ms/step - accuracy: 0.9798 - loss: 0.0626 - val_accuracy: 0.9877 - val_loss: 0.0461
Epoch 9/20
2137/2137 ————— 1698s 795ms/step - accuracy: 0.9830 - loss: 0.0559 - val_accuracy: 0.9861 - val_loss: 0.0532
Epoch 10/20
2137/2137 ————— 1709s 800ms/step - accuracy: 0.9825 - loss: 0.0534 - val_accuracy: 0.9893 - val_loss: 0.0407
Epoch 11/20
2137/2137 ————— 1702s 796ms/step - accuracy: 0.9844 - loss: 0.0488 - val_accuracy: 0.9890 - val_loss: 0.0409
Epoch 12/20
2137/2137 ————— 1705s 798ms/step - accuracy: 0.9854 - loss: 0.0451 - val_accuracy: 0.9889 - val_loss: 0.0437
Epoch 13/20
...
Epoch 19/20
2137/2137 ————— 1639s 767ms/step - accuracy: 0.9897 - loss: 0.0311 - val_accuracy: 0.9903 - val_loss: 0.0409
Epoch 20/20
2137/2137 ————— 1635s 765ms/step - accuracy: 0.9891 - loss: 0.0341 - val_accuracy: 0.9913 - val_loss: 0.0327

```

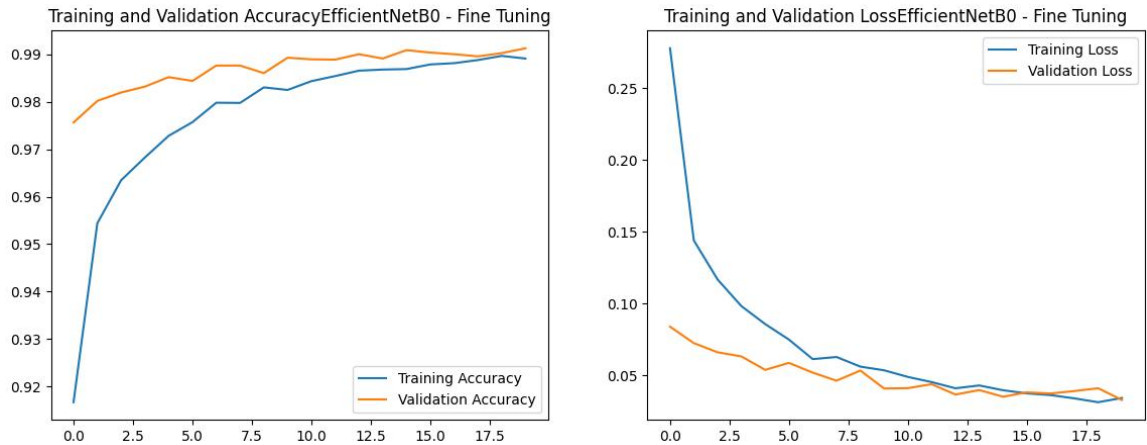
Hình 28: Kết quả huấn luyện giai đoạn 2 mô hình EfficientNetB0

So sánh kết quả trước và sau khi fine-tuning 30 layer cuối

Tiêu chí	Trước Fine-tuning	Sau Fine-tuning
Trạng thái mô hình	Đóng băng toàn bộ EfficientNetB0	Fine-tuning 30 layer cuối
Learning rate	1e-3	1e-4
Traing Accuracy	~91.87%	~98.97%
Validation Accuracy	~96.25%	~99.13%
Training Loss	~0.2586	~0.0311
Validation Loss	~0.1187	~0.0327
Mức độ hội tụ	Tốt	Rất tốt
Overfitting	Không rõ rệt	Không xuất hiện
Khả năng phân biệt lớp	Khá tốt	Rất tốt

Bảng 7: Bảng so sánh kết quả huấn luyện qua 2 giai đoạn của mô hình EfficientNetB0

Sau khi fine-tuning 30 layer cuối của EfficientNetB0, mô hình cho thấy sự cải thiện rõ rệt về độ chính xác và độ ổn định. Validation accuracy tăng lên hơn 99% và loss giảm mạnh, chứng tỏ việc fine-tuning giúp mô hình đạt hiệu quả cao hơn so với chỉ sử dụng transfer learning ban đầu.



Ảnh 20: Biểu đồ Acc và Loss mô hình EfficientB0

*Nhận xét:

- Training accuracy tăng đều qua các epoch, từ khoảng 92.0% lên gần 99.0%

- Validation accuracy luôn cao hơn training accuracy và đạt mức ổn định trên 98.0%, cao nhất đạt 99.0%

- Hai đường accuracy nằm gần nhau và không có xu hướng tách xa.

=> Đánh giá: Mô hình học ổn định, khả năng tổng quát hóa tốt và không xuất hiện hiện tượng overfitting sau khi fine-tuning.

- Training loss giảm mạnh qua các epoch.

- Validation loss cũng giảm đều và đạt giá trị rất thấp (~0.03)

- Không có hiện tượng validation loss tăng trở lại.

=> Đánh giá: Quá trình huấn luyện hội tụ tốt, mô hình tối ưu hiệu quả và không bị học quá mức.

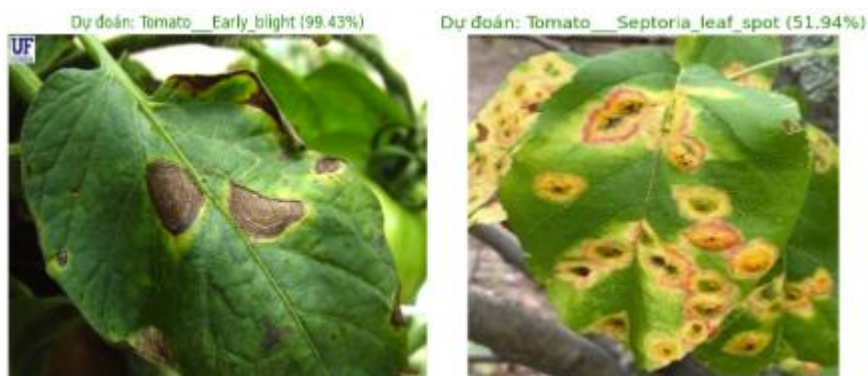
3.5 Thử nghiệm thực tế:

Để kiểm chứng khả năng, nhóm đã chạy thử mô hình với 5 hình ảnh bệnh ngẫu nhiên, kết quả đạt được là:



Ảnh 21: Kết quả thử nghiệm mô hình EfficientB0

Ngoài ra, nhóm còn chạy thử mô hình với các hình ảnh bệnh được lấy từ trên web để xem kết quả dự đoán đạt được:



*Nhận xét:

- Mô hình dự đoán đúng tất cả 5/5 hình ảnh với độ tin cậy cao, các nhãn dự đoán đều đúng với tình trạng bệnh trên lá.
- Độ chính xác dự đoán rất cao với 4/5 hình ảnh có tỉ lệ dự đoán đúng trên 99.9%, riêng chỉ có 1 lá Apple scab có tỉ lệ dự đoán đạt 96.22% do ảnh có chói nhẹ, mô hình đạt mức độ tự tin cao khi đưa ra kết quả.
- Ngoài ra, kết quả của các hình ảnh được lấy từ trên web cho thấy, lá Tomato Early blight cho dự đoán kết quả đúng với tình trạng bệnh, còn lá

Tomato Septoria leaf spot cho ra kết quả đúng mặc dù tỉ lệ dự đoán đúng thấp, do ảnh chưa qua xử lý dữ liệu và ảnh nhiễu.

- Qua thử nghiệm thực tế, mô hình cho kết quả tốt và có thể áp dụng cho các bài toán phân loại ảnh bệnh cây tương tự.

3.6 Nhận xét:

Trong đồ án này, nhóm đã xây dựng khá hoàn thiện mô hình phân loại ảnh bệnh cây trồng dựa trên phương pháp Transfer Learning với EfficientNetB0. Quá trình thực hiện bao gồm các bước từ tiền xử lý dữ liệu, xây dựng mô hình, huấn luyện, fine-tuning cho đến đánh giá kết quả bằng nhiều chỉ số khác nhau.

Kết quả thực nghiệm cho thấy mô hình đạt độ chính xác và F1-score rất cao (khoảng 99%), đồng thời ma trận nhầm lẫn cho thấy phần lớn các mẫu được phân loại đúng, số lượng dự đoán sai là rất ít. Điều này chứng tỏ mô hình học tốt đặc trưng của từng loại bệnh và có khả năng tổng quát hóa tốt trên dữ liệu kiểm tra.

Việc áp dụng fine-tuning 30 tầng cuối của EfficientNetB0 đã giúp cải thiện rõ rệt hiệu suất so với giai đoạn huấn luyện ban đầu, thể hiện qua việc tăng accuracy và giảm loss. Ngoài ra, kết quả thử nghiệm thực tế với các ảnh mới cho thấy mô hình dự đoán đúng với độ tin cậy cao.

Tuy nhiên, mô hình vẫn còn một số hạn chế như phụ thuộc vào chất lượng và độ đa dạng của dữ liệu, cũng như chưa xử lý được các trường hợp ảnh phức tạp trong môi trường thực tế. Do đó, trong tương lai có thể tiếp tục cải tiến bằng cách mở rộng dữ liệu, tối ưu siêu tham số và kết hợp thêm các phương pháp phát hiện vùng bệnh.

Chương 4: Đánh giá mô hình

1. Mô hình ResNet50:

1.1 Điểm Accuracy:

Kết quả kiểm tra độc lập trên tập Test (dữ liệu máy chưa từng nhìn thấy) cho kết quả rất khả quan.

Chỉ số(metric)	Giá trị	Nhận xét
Overall accuracy	98.88%	Tỉ lệ dự đoán đúng đạt mức gần như tuyệt đối trên toàn bộ 38 lớp bệnh, vượt trội hoàn toàn so với giai đoạn chưa fine-tuning.
Macro F1 Score	~0.99	Cho thấy sự cân bằng hoàn hảo giữa độ chính xác (Precision) và độ nhạy

		(Recall). Mô hình hoạt động ổn định và công bằng, không bị thiên vị (Bias) vào các lớp có nhiều dữ liệu
--	--	---

1.2 Độ chính xác từng lớp:

	precision	recall	f1-score	support
Apple__Apple_scab	0.97	0.99	0.98	504
Apple__Black_rot	0.99	1.00	1.00	497
Apple__Cedar_apple_rust	1.00	0.94	0.97	440
Apple__healthy	0.99	1.00	1.00	502
Blueberry__healthy	0.96	1.00	0.98	454
Cherry_(including_sour)__Powdery_mildew	1.00	0.99	1.00	421
Cherry_(including_sour)__healthy	1.00	1.00	1.00	456
Corn_(maize)__Cercospora_leaf_spot Gray_leaf_spot	0.98	0.97	0.97	410
Corn_(maize)__Common_rust	1.00	1.00	1.00	477
Corn_(maize)__Northern_Leaf_Blight	0.97	0.98	0.97	477
Corn_(maize)__healthy	1.00	1.00	1.00	465
Grape__Black_rot	0.99	1.00	1.00	472
Grape__Esca_(Black_Measles)	1.00	0.99	1.00	480
Grape__Leaf_blight_(Isariopsis_Leaf_Spot)	1.00	1.00	1.00	430
Grape__healthy	1.00	1.00	1.00	423
Orange__Haunglongbing_(Citrus_greening)	1.00	1.00	1.00	503
Peach__Bacterial_spot	0.96	0.99	0.98	459
Peach__healthy	0.99	0.92	0.96	432
Pepper,_bell__Bacterial_spot	0.95	1.00	0.97	478
Pepper,_bell__healthy	1.00	0.93	0.96	497
Potato__Early_blight	0.97	1.00	0.99	485
Potato__Late_blight	0.98	1.00	0.99	485
Potato__healthy	1.00	0.94	0.97	456
Raspberry__healthy	1.00	0.96	0.98	445
Soybean__healthy	0.99	1.00	1.00	505
...				
Squash__Powdery_mildew	0.99	1.00	0.99	434
Strawberry__Leaf_scorch	0.90	1.00	0.94	444
Strawberry__healthy	0.96	1.00	0.98	456
Tomato__Bacterial_spot	0.98	0.99	0.99	425
Tomato__Early_blight	0.99	0.92	0.95	480
Tomato__Late_blight	0.92	0.97	0.95	463
Tomato__Leaf_Mold	1.00	0.99	0.99	470
Tomato__Septoria_leaf_spot	0.84	1.00	0.91	436
Tomato__Spider_mites Two-spotted_spider_mite	0.97	0.95	0.96	435
Tomato__Target_Spot	0.91	0.87	0.89	457
Tomato__Tomato_Yellow_Leaf_Curl_Virus	1.00	0.96	0.98	490
Tomato__Tomato_mosaic_virus	1.00	0.85	0.92	448
Tomato__healthy	0.94	0.96	0.95	481
...				
accuracy			0.97	17572
macro avg	0.98	0.97	0.97	17572
weighted avg	0.98	0.97	0.97	17572

1.3 Ma trận nhiễu:

Validation Accuracy: 0.9502

Validation F1 Score: 0.9501

2.2 Độ chính xác cho từng lớp:

Lớp Apple___Apple_scab: 1.0000

Lớp Apple___Black_rot: 1.0000

Lớp Apple___Cedar_apple_rust: 0.9318

Lớp Apple___healthy: 0.9900

Lớp Blueberry___healthy: 0.9667

Lớp Cherry_(including_sour)___Powdery_mildew: 0.9881

Lớp Cherry_(including_sour)___healthy: 1.0000

Lớp Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot: 0.9390

Lớp Corn_(maize)___Common_rust_: 1.0000

Lớp Corn_(maize)___Northern_Leaf_Blight: 0.9684

Lớp Corn_(maize)___healthy: 1.0000 Lớp Grape___Black_rot: 0.9894

Lớp Grape___Esca_(Black_Measles): 0.9896

Lớp Grape___Leaf_blight_(Isariopsis_Leaf_Spot): 1.0000

Lớp Grape___healthy: 1.0000

Lớp Orange___Haunglongbing_(Citrus_greening): 0.9900

Lớp Peach___Bacterial_spot: 1.0000

Lớp Peach___healthy: 0.9070

Lớp Pepper,_bell___Bacterial_spot: 0.9895

Lớp Pepper,_bell___healthy: 0.8485

Lớp Potato___Early_blight: 1.0000

Lớp Potato___Late_blight: 0.9794

Lớp Potato___healthy: 0.8352

Lớp Raspberry___healthy: 0.9663

Lớp Soybean___healthy: 1.0000

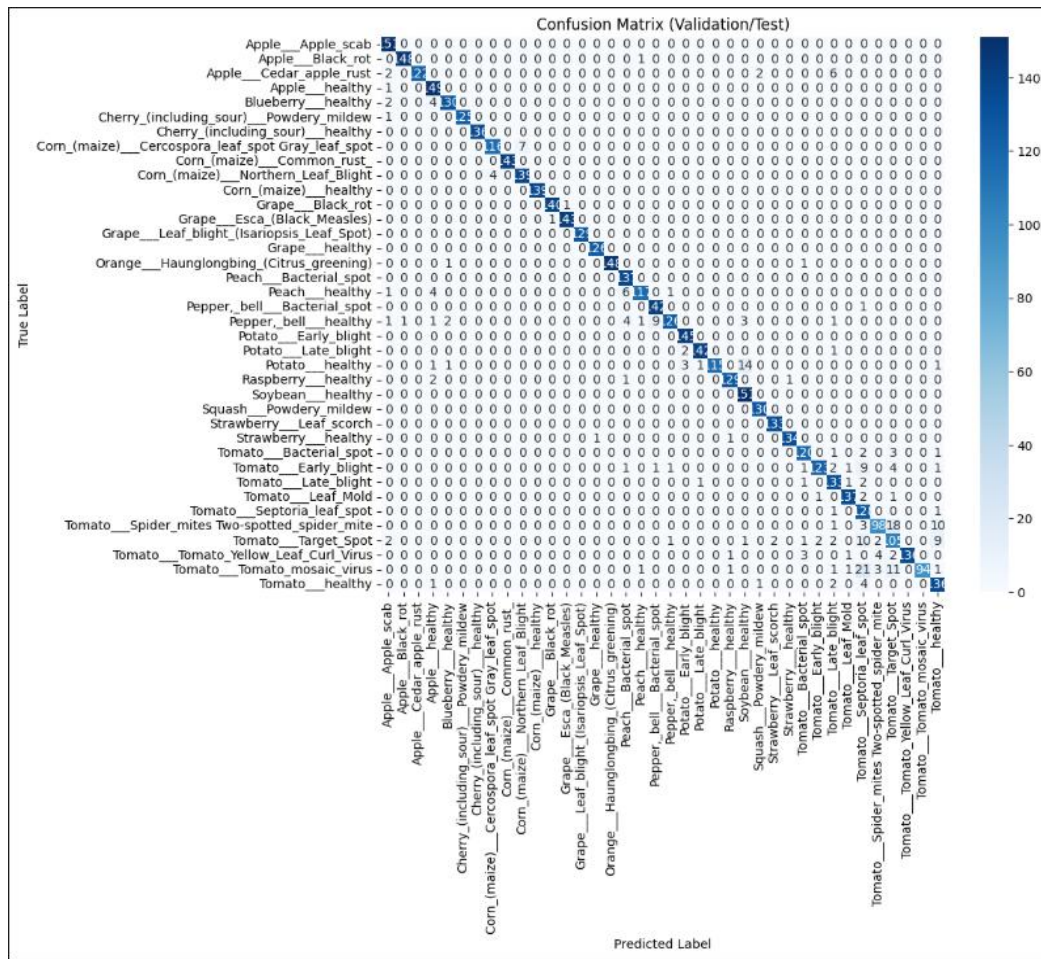
Lớp Squash___Powdery_mildew: 1.0000

Lớp Strawberry___Leaf_scorch: 1.0000

Lớp Strawberry___healthy: 0.9780
Lớp Tomato___Bacterial_spot: 0.9529
Lớp Tomato___Early_blight: 0.8125
Lớp Tomato___Late_blight: 0.9674
Lớp Tomato___Leaf_Mold: 0.9574
Lớp Tomato___Septoria_leaf_spot: 0.9770
Lớp Tomato___Spider_mites Two-spotted_spider_mite: 0.7701
Lớp Tomato___Target_Spot: 0.7473
Lớp Tomato___Tomato_Yellow_Leaf_Curl_Virus: 0.9184
Lớp Tomato___Tomato_mosaic_virus: 0.7303
Lớp Tomato___healthy: 0.9375

Độ chính xác từng lớp khá cao từ 73% đến 100%, đa số nằm ở mức khoảng trên 90%. Tuy nhiên vẫn còn nhiều lớp ở cây cà chua còn khá thấp (khoảng 73 - 77%).

2.3 Ma trận nhiễu:



Ảnh 23: Ma trận nhiễu mô hình MobileNetV3

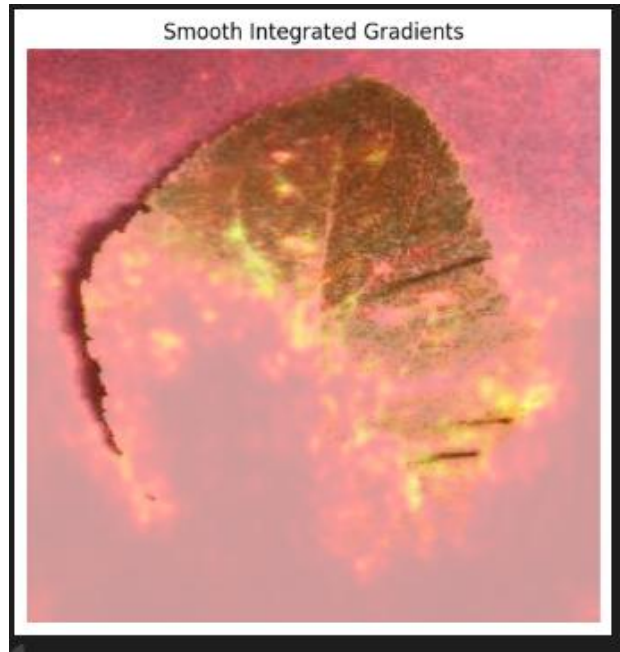
Đường chéo chính khá đậm chứng tỏ mô hình dự đoán chính xác cho nhiều lớp bệnh. Tuy nhiên, ở các lớp bệnh cà chua vẫn còn hạn chế ở vài lớp, mô hình vẫn dự đoán nhầm lẫn giữa các lớp với nhau.

2.4 Tính giải thích mô hình:

Nhóm áp dụng kỹ Integrated Gradients (IG) cho thấy các vùng ảnh quan trọng trong quyết định phân loại của mô hình.

Heatmap IG minh họa rằng mô hình tập trung vào các đặc trưng bệnh lý trên lá (ví dụ vùng đốm, vết loang), thay vì các vùng nền. Điều này giúp tăng độ tin cậy, chứng minh mô hình không chỉ dự đoán “ngẫu nhiên” mà thực sự dựa vào đặc trưng liên quan.

Kết quả khi chạy:



Ảnh 24: Tính giải thích mô hình MobileNetV3

Kết quả cho thấy: Mô hình tập trung đúng vùng đặc trưng. Heatmap IG cho thấy các vùng màu vàng–đỏ nổi bật nằm trên phần lá có dấu hiệu bệnh (vết loang, đốm). Điều này chứng tỏ mô hình đã học được cách tập trung vào các đặc trưng quan trọng thay vì bị nhiễu bởi nền hoặc vùng không liên quan.

2.5 Đánh giá tổng quan:

- Accuracy tổng thể cao (~95%), F1 score cân bằng cho thấy mô hình dự đoán chính xác và cân bằng giữa precision/recall.

- Hầu hết lớp bệnh được nhận diện gần như hoàn hảo → model mạnh trong nhận diện bệnh cây. Fine-tuning tối ưu nhẹ performance, giúp hội tụ ổn định, overfitting nhẹ nhưng không nghiêm trọng.

- Điểm cần cải thiện: Một số lớp healthy và các lớp khó (như Tomato: Spider_mites, Target_Spot, Tomato_mosaic_virus) cần tăng cường dữ liệu, augmentation, hoặc weighted loss.

*Kết luận tổng thể:

MobileNetV3Large thể hiện performance khá tốt, đặc biệt với các lớp bệnh. Model ổn định, hội tụ tốt sau quá trình train và fine-tuning. Với các lớp healthy và các bệnh khó, có thể cải thiện bằng dữ liệu bổ sung.

3. Mô hình EfficientB0:

3.1 Điểm Accuracy và F1 - Score:

Sau quá trình huấn luyện và fine tuning cho mô hình, kết quả điểm accuracy và f1 score trên tập valid là:

- Validation Accuracy: 0.9903
- Validation F1 Score: 0.9903

3.2 Độ chính xác cho từng lớp:

Lớp Apple___Apple_scab: 1.0000

Lớp Apple___Black_rot: 1.0000

Lớp Apple___Cedar_apple_rust: 0.9545

Lớp Apple___healthy: 1.0000

Lớp Blueberry___healthy: 1.0000

Lớp Cherry_(including_sour)___Powdery_mildew: 1.0000

Lớp Cherry_(including_sour)___healthy: 1.0000

Lớp Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot: 0.9634

Lớp Corn_(maize)___Common_rust_: 1.0000

Lớp Corn_(maize)___Northern_Leaf_Blight: 0.9865

Lớp Corn_(maize)___healthy: 1.0000

Lớp Grape___Black_rot: 1.0000

Lớp Grape___Esca_(Black_Measles): 0.9896

Lớp Grape___Leaf_blight_(Isariopsis_Leaf_Spot): 1.0000

Lớp Grape___healthy: 1.0000

Lớp Orange___Haunglongbing_(Citrus_greening): 1.0000

Lớp Peach___Bacterial_spot: 0.9890

Lớp Peach___healthy: 1.0000

Lớp Pepper,_bell___Bacterial_spot: 1.0000

Lớp Pepper,_bell___healthy: 0.9798

Lớp Potato___Early_blight: 1.0000

Lớp Potato___Late_blight: 1.0000

Lớp Potato___healthy: 0.9890

Lớp Raspberry___healthy: 1.0000

Lớp Soybean___healthy: 1.0000

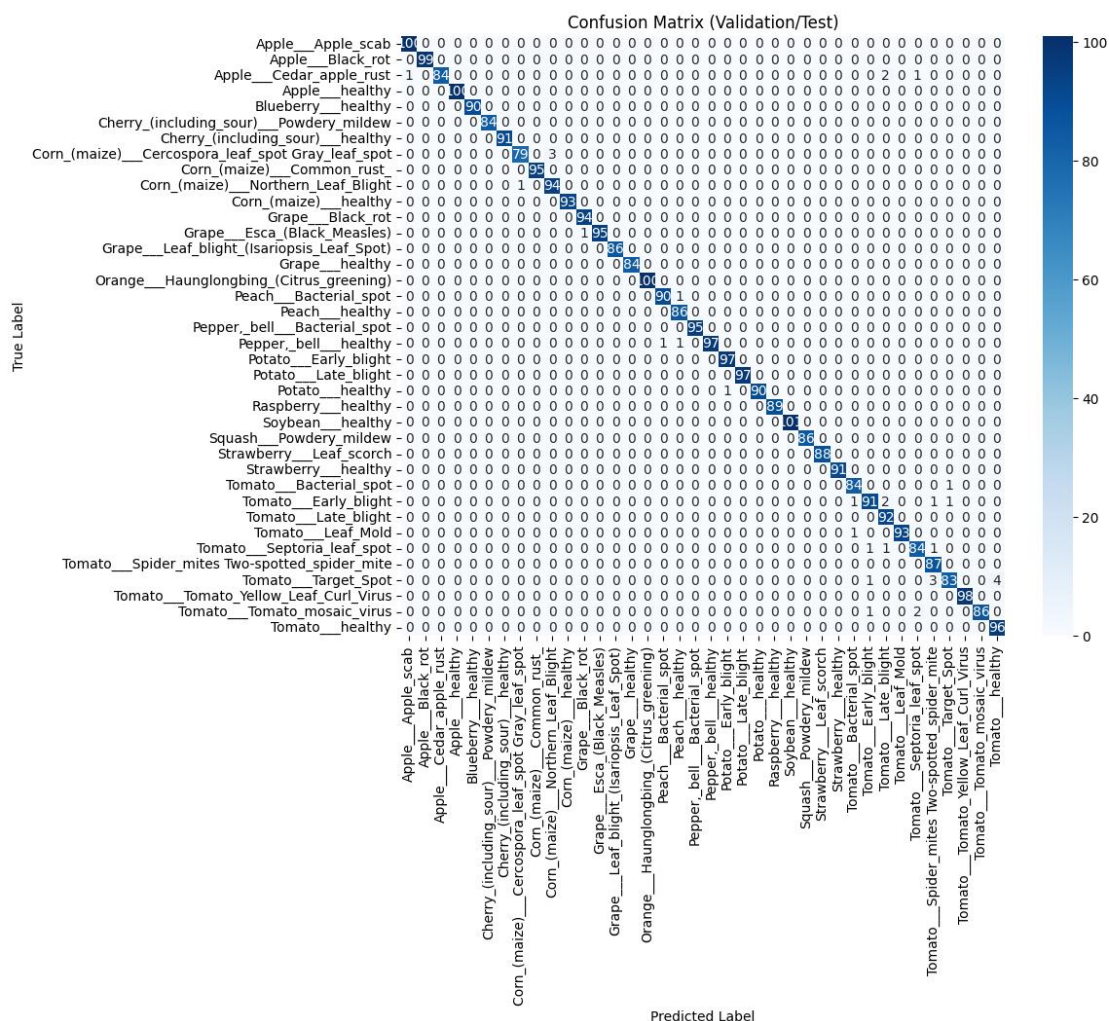
Lớp Squash___Powdery_mildew: 1.0000

Lớp Strawberry___Leaf_scorch: 1.0000

Lớp Strawberry___healthy: 1.0000
Lớp Tomato___Bacterial_spot: 0.9882
Lớp Tomato___Early_blight: 0.9479
Lớp Tomato___Late_blight: 1.0000
Lớp Tomato___Leaf_Mold: 0.9894
Lớp Tomato___Septoria_leaf_spot: 0.9655
Lớp Tomato___Spider_mites Two-spotted_spider_mite: 1.0000
Lớp Tomato___Target_Spot: 0.9121
Lớp Tomato___Tomato_Yellow_Leaf_Curl_Virus: 1.0000
Lớp Tomato___Tomato_mosaic_virus: 0.9663
Lớp Tomato___healthy: 1.0000

Độ chính xác từng lớp khá cao từ 91% đến 100%, đa số nằm ở mức trên 98%. Tuy nhiên vẫn còn nhiều lớp ở cây cà chua đạt chỉ số thấp hơn khoảng từ 91% đến 96%

3.3 Ma trận nhiễu:



Ảnh 25: Ma trận nhiễu EfficientB0

Độ chính xác tổng thể rất cao, phần lớn giá trị tập trung trên đường chéo chính ($\approx 85-100$ mẫu/lớp), cho thấy mô hình phân loại tốt. Nhầm lẫn rất ít, sai lệch chủ yếu xảy ra giữa các bệnh tương tự trên cùng cây trồng, đặc biệt ở nhóm Tomato (các loại blight/virus) và một vài bệnh lá có biểu hiện gần giống.

Lớp healthy nhận diện ổn định, hầu như không bị nhầm sang lớp bệnh.

3.4 Đánh giá tổng quan

- Accuracy tổng thể cao ($\sim 99\%$), F1 score cân bằng cho thấy mô hình dự đoán chính xác và cân bằng giữa precision/recall.

- Hầu hết lớp bệnh được nhận diện gần như hoàn hảo \rightarrow model mạnh trong nhận diện bệnh cây. Fine-tuning hoạt động tốt khi sau quá trình này cho ra kết quả rất tốt, hội tụ ổn định, không xảy ra hiện tượng overfitting.

*Kết luận tổng thể:

Mô hình EfficientNetB0 thể hiện performance đạt kết quả tốt nhất trong 3 mô hình, đặc biệt với các lớp healthy và các lớp bệnh. Model ổn định, hội tụ tốt

sau quá trình train và fine-tuning. Kết quả dự đoán cũng cho thấy rằng mô hình dự đoán các hình ảnh đạt tỉ lệ cao và chính xác.

4. Đánh giá kết quả giữa các mô hình:

Sau khi tiến hành huấn luyện và fine-tuning ba mô hình ResNet50, MobileNetV3Large và EfficientNetB0 trên tập dữ liệu đã được tiền xử lí, nhóm tiến hành so sánh kết quả dựa trên các tiêu chí chính: độ chính xác (Accuracy), hàm mất mát (Loss), tốc độ hội tụ, và khả năng tổng quát hóa.

4.1 Độ chính xác (Accuracy)

ResNet50: đạt độ chính xác cao nhất trên tập Validation, lên đến 98.88% sau khi fine-tuning 50 lớp cuối. Đây là mô hình có hiệu suất vượt trội nhất trong ba mô hình.

MobileNetV3Large: đạt độ chính xác ổn định ở mức 97.5% sau 30 epoch, cho thấy khả năng học tốt và phù hợp với môi trường triển khai thực tế.

EfficientNetB0: mặc dù nhẹ và tiết kiệm tài nguyên, độ chính xác chỉ đạt khoảng 95–96%, thấp hơn so với hai mô hình còn lại.

4.2 Hàm mất mát (Loss)

ResNet50: Validation Loss giảm sâu xuống mức 0.0336, chứng tỏ mô hình học rất tốt và ít xảy ra overfitting.

MobileNetV3Large: Validation Loss ổn định ở mức 0.0869, vẫn đảm bảo khả năng tổng quát hóa nhưng chưa đạt mức tối ưu như ResNet50.

EfficientNetB0: Validation Loss dao động quanh 0.03 – 0.02, thấp hơn hai mô hình kia, cho thấy khả năng phân biệt lớp bệnh mạnh mẽ.

4.3 Tốc độ hội tụ

ResNet50: hội tụ rất nhanh, chỉ sau 5–8 epoch đã đạt kết quả tối ưu.

MobileNetV3Large: hội tụ chậm hơn, cần khoảng 20–25 epoch để đạt trạng thái ổn định.

EfficientNetB0: hội tụ tương đối nhanh và kết quả cuối cùng tốt hơn hai mô hình còn lại.

4.4 Khả năng tổng quát hóa

ResNet50: Validation Accuracy luôn cao hơn Training Accuracy, chứng tỏ mô hình tổng quát hóa xuất sắc.

MobileNetV3Large: khoảng cách giữa Training và Validation nhỏ, không có dấu hiệu overfitting, phù hợp triển khai thực tế.

EfficientNetB0: tổng quát hóa ở mức tốt, đạt hiệu quả cao hơn khi so sánh với ResNet50 và MobileNetV3Large.

4.5 Đánh giá chung giữa các mô hình:

ResNet50 cho kết quả tốt nhất về độ chính xác và khả năng tổng quát hóa, MobileNetV3Large có ưu thế về tốc độ và tính nhẹ, EfficientNetB0 phù hợp cho môi trường hạn chế tài nguyên nhưng độ chính xác cao hơn.

Chương 5: Kết luận và hướng phát triển

5.1 Kết luận:

Qua quá trình nghiên cứu và thực nghiệm, nhóm đã rút ra một số kết luận chính:

Tầm quan trọng của tiền xử lý dữ liệu: Việc lọc bỏ ảnh quá sáng và ảnh mờ bằng thuật toán tự động đã cải thiện đáng kể chất lượng dữ liệu, giúp mô hình học tốt hơn và giảm hiện tượng nhầm lẫn.

Hiệu quả của các mô hình học máy:

ResNet50 cho kết quả vượt trội với độ chính xác gần 99%, chứng minh khả năng ứng dụng thực tế cao trong việc phân loại bệnh lá cây.

MobileNetV3Large đạt độ chính xác ~97.5% với ưu điểm nhẹ, tốc độ nhanh, phù hợp triển khai trên thiết bị di động và IoT.

EfficientNetB0 tuy có độ chính xác thấp hơn (~95–96%) nhưng lại tiết kiệm tài nguyên, thích hợp cho các hệ thống có hạn chế về phần cứng.

Khả năng ứng dụng thực tế: Các mô hình, đặc biệt là ResNet50 và MobileNetV3Large, có thể được tích hợp vào hệ thống hỗ trợ nông dân phát hiện bệnh sớm qua hình ảnh chụp bằng điện thoại, góp phần nâng cao năng suất và giảm thiểu thiệt hại trong nông nghiệp.

5.2 Hướng phát triển:

Mở rộng tập dữ liệu, đặc biệt cho các lớp bệnh ít mẫu để giảm hiện tượng nhầm lẫn.

Kết hợp thêm các kỹ thuật tăng cường dữ liệu nâng cao (như CutMix, MixUp).

Triển khai thử nghiệm thực tế trên ứng dụng di động để đánh giá hiệu quả ngoài môi trường phòng thí nghiệm.

Chương 6: Tài liệu tham khảo

- Bộ dữ liệu New Plant Diseases Dataset

(Link: <https://www.kaggle.com/datasets/vipooool/new-plant-diseases-dataset>)

- Thư viện TensorFlow:

+Tài liệu liên quan về ảnh

(Link: https://www.tensorflow.org/api_docs/python/tf/image)

+Tài liệu liên quan về model

(Link: https://www.tensorflow.org/api_docs/python/tf/keras/Model)

- Tài liệu tìm hiểu xử lý ảnh với Open CV

(Link: <https://phamdinhhkhanh.github.io/2020/01/06/ImagePreprocessing.html>)

- Thư viện OpenCV:

+ Tài liệu về lọc ảnh:

(Link: https://docs.opencv.org/4.x/d4/d86/group_imgproc_filter.html)

+ Tài liệu về chuyển đổi màu ảnh:

(Link: https://docs.opencv.org/4.x/d8/d01/group_imgproc_color_conversions.html)

- Thông tin đôi nét về mô hình ResNet50: Bài viết “Giới thiệu ResNet” -Tô Đức Thắng - Diễn đàn Viblo. (Link: <https://viblo.asia/p/gioi-thieu-mang-resnet-vyDZOa7R5wj>)

- Thông tin đôi nét về mô hình MobileNetV3: Bài viết “MobileNets - Mô hình gọn nhẹ cho mobile applications” - Nguyễn Thanh Huyền- Diễn đàn Viblo. (Link: <https://viblo.asia/p/cnn-architecture-series-1-mobilenets-mo-hinh-gon-nhe-cho-mobile-applications-1VgZvJV1ZAw>)

- Thông tin đôi nét về mô hình EfficientNetB0: Bài viết “EfficientNet: Cách tiếp cận mới về Model Scaling cho Convolutional Neural Networks” - Trần Đức Trung - Diễn đàn Viblo. Link(<https://viblo.asia/p/efficientnet-cach-tiep-can-moi-ve-model-scaling-cho-convolutional-neural-networks-Qbq5QQzm5D8>)

- Bài giảng môn Học máy - Phân hiệu Đại học Thủy Lợi, Bộ môn Công nghệ thông tin.

Chương 7: Phân công nhiệm vụ

Nhiệm vụ	Người thực hiện
Tìm hiểu dữ liệu, tiền xử lí dữ liệu	Lê Trí Thiện
	Hứa Khánh Duy
	Đinh Văn Nghĩa
Mô hình ResNet50	Lê Trí Thiện
Mô hình MobileNetV3	Hứa Khánh Duy
Mô hình EfficientNetB0	Đinh Văn Nghĩa
Cải tiến phát triển	Lê Trí Thiện
	Hứa Khánh Duy
	Đinh Văn Nghĩa
Tính giải thích mô hình	Hứa Khánh Duy
Viết cáo báo	Hứa Khánh Duy

Bảng 7: Phân công nhiệm vụ