Trường đại học Thủy Lợi

Tên sinh viên: ĐỖ ĐÌNH THÀNH
Lớp 64TTNT2

BÁO CÁO BÀI TẬP LỚN

NHẬN DẠNG MẪU

Nhận dạng bệnh trên lá cây

Người hướng dẫn: PGS.TS.NGUYỄN QUANG HOAN.

Hà Nội, Năm 2025

Mục lục

Chương 1: Tổng quan

1)Lý do chọn đề tài.

1.1) Tầm quan trọng của nông nghiệp và cây trồng.

Nông nghiệp đóng vai trò then chốt trong nền kinh tế Việt Nam, đặc biệt là trong việc đảm bảo an ninh lương thực và phát triển bền vững. Tuy nhiên, một trong những thách thức lớn nhất mà ngành nông nghiệp đang đối mặt là sự lây lan nhanh chóng của các loại bệnh trên cây trồng, gây thiệt hại nghiêm trọng về năng suất và chất lượng nông sản.

1.2) Khó khăn trong việc phát hiện bệnh sớm.

Việc phát hiện bệnh trên cây trồng hiện nay chủ yếu dựa vào kinh nghiệm của người nông dân hoặc chuyên gia nông nghiệp. Tuy nhiên, phương pháp này mang tính chủ quan, dễ sai lệch và không hiệu quả trong quy mô lớn. Trong khi đó, lá cây thường là bộ phận đầu tiên thể hiện dấu hiệu của bệnh, do đó việc nhận dạng bệnh qua hình ảnh lá cây là hướng tiếp cận khả thi và hiệu quả.

1.3) Ứng dụng công nghệ trong nông nghiệp.

Với sự phát triển mạnh mẽ của trí tuệ nhân tạo (AI) và thị giác máy tính (Computer Vision), việc xây dựng hệ thống nhận dạng bệnh trên lá cây bằng hình ảnh trở nên khả thi hơn bao giờ hết. Đề tài này không chỉ mang tính ứng dụng cao mà còn góp phần thúc đẩy chuyển đổi số trong nông nghiệp, hướng tới mô hình nông nghiệp thông minh và bền vững.

1.4) Ý nghĩa thực tiễn và học thuật.

Việc nghiên cứu và phát triển hệ thống nhận dạng bệnh trên lá cây giúp:

- Hỗ trợ người nông dân phát hiện bệnh sớm, từ đó có biện pháp xử lý kịp thời.
- Giảm thiểu chi phí và công sức giám sát thủ công.
- Góp phần xây dựng cơ sở dữ liệu hình ảnh phục vụ cho nghiên cứu khoa học và đào tạo.

2)Triển khai bài toán.

2.1) Bộ dữ liệu.

Bộ dữ liệu bao gồm hơn 70000 ảnh lá cây của 1 số loại cây phổ biến trong nông nghiệp: Táo, việt quất, cherry, ngô, nho, cam, đào, hạt tiêu, mâm xôi, đậu nành, bí đỏ,

dâu tây, cà chua. Mỗi loại có các mẫu của lá khỏe và mẫu của 1 số bệnh phổ biến như: bệnh bạc lá, đốm lá, nấm, …


2.2) Tiến trình.

- Nội dung đề tài tập trung vào việc xây dựng huấn luyện tìm hiểu rõ cách nhận diện bệnh từ lá cây. Song song với đó là tinh chỉnh mô hình để nhận diện bệnh tối ưu nhất có thể.

2.3) Thuật toán sử dụng.

ResNet-18 (Residual Network) – mạng CNN có các kết nối tắt giúp huấn luyện sâu và ổn định.

Vision Transformer (ViT) – mô hình dựa trên cơ chế attention, cho phép học quan hệ toàn cục giữa các vùng ảnh.

Chương 2: Cơ sở lý thuyết.

1)Nhận dạng mẫu.

1.1) Khái niệm.

Nhận dạng mẫu là quá trình phân loại đối tượng vào các lớp xác định trước dựa trên đặc trưng của chúng.
Ví dụ: phân loại lá bệnh, nhận dạng chữ viết, nhận diện khuôn mặt.
Mục tiêu chính là tìm ra các quy luật, đặc điểm chung trong dữ liệu để phân loại hoặc gán nhãn cho các nhãn.
Mẫu (pattern) là một biểu diễn của dữ liệu thường dưới dạng vector đặc trưng trong không gian nhiều chiều.

Mục tiêu: xây dựng hàm f(x) sao cho:

$f(x)=\omega i,\ \ \omega i \in \{\omega 1,\omega 2,...,\omega k\}$.
Trong đó:

- x là mẫu đầu vào, biểu diễn dưới dạng vector đặc trưng (feature vector).
- ωi là nhãn lớp mà mẫu x được gán vào. Tập hợp {ω1,ω2,...,ωk} là tập các lớp có thể có.
- f(x) là hàm phân loại, hay còn gọi là hàm quyết định (decision function), dùng để xác định lớp của mẫu x.

1.2) Quy trình.

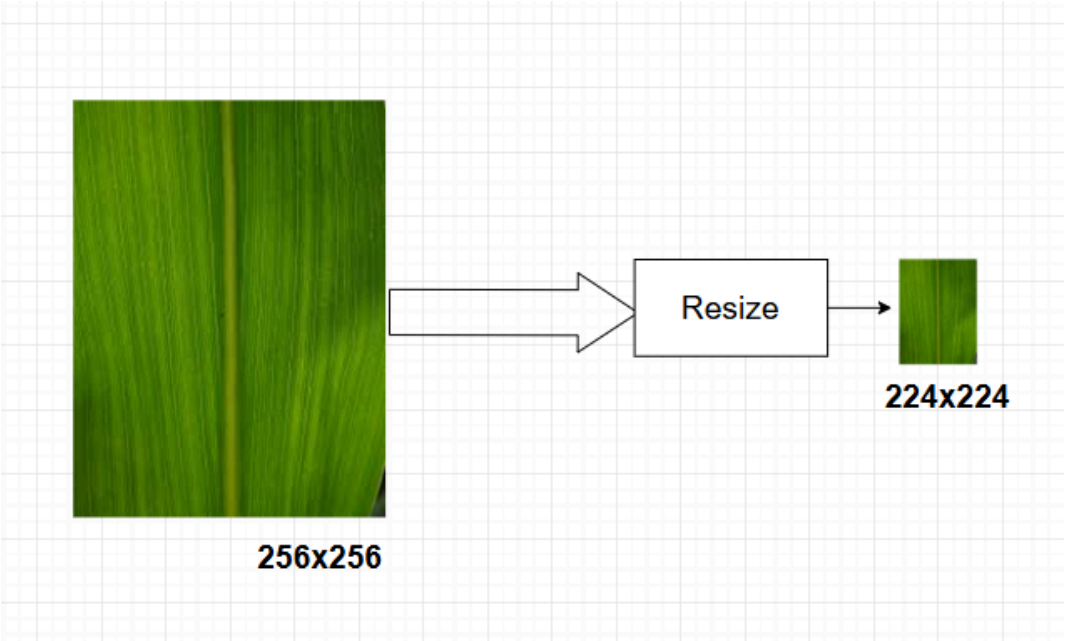
a) Thu thập dữ liệu

- Thu thập dữ liệu: Dữ liệu lá cây được thu thập từ bộ dữ liệu kaggle (https://www.kaggle.com/code/abdallahwagih/plant-village-disease-classification-acc-99-6/input) và một số ảnh chụp khác. Nhãn được gán tương ứng với loại cây và bệnh mà lá mắc phải.

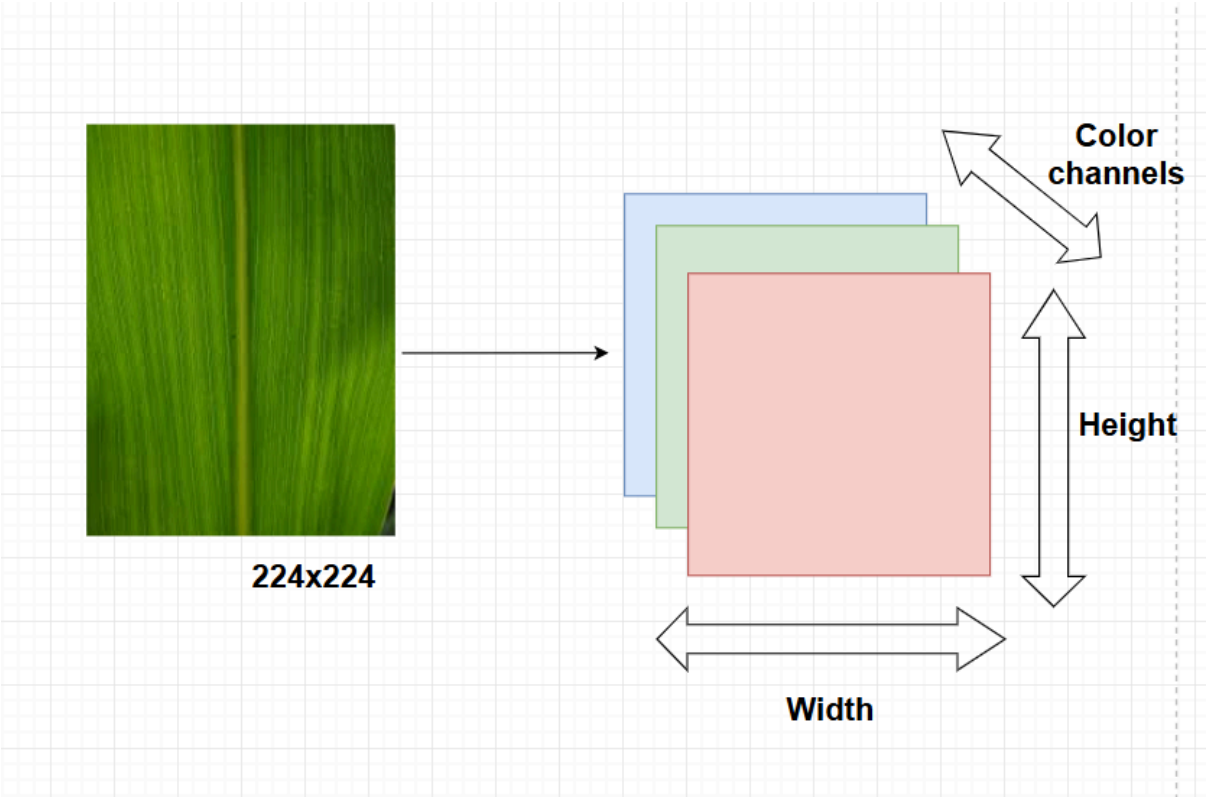
b) Tiền xử lý dữ liệu:

+ Tiền xử lý nhằm chuẩn hóa dữ liệu đầu vào để mô hình học hiệu quả, loại bỏ nhiễu và giúp ResNet-18 và ViT nhận diện đặc trưng ổn định hơn.
  Vì ảnh trong PlantVillage được chụp trong điều kiện phòng thí nghiệm (ánh sáng đều, nền trơn) nhưng ảnh thực tế thường phức tạp hơn, nên quá trình tiền xử lý giúp:
  - Đưa ảnh về cùng kích thước.
  - Cân bằng độ sáng, tương phản.
  - Tăng độ đa dạng (augmentation) nhằm chống overfitting.

+ Chuẩn hóa kích thước (resizing): Các ảnh được resize về chuẩn kích thước 224x224 để tương thích với đầu vào mô hình Resnet18 và Vision Transformer. Điều này giúp giảm tải bộ nhớ, đồng nhất tỉ lệ khung hình.

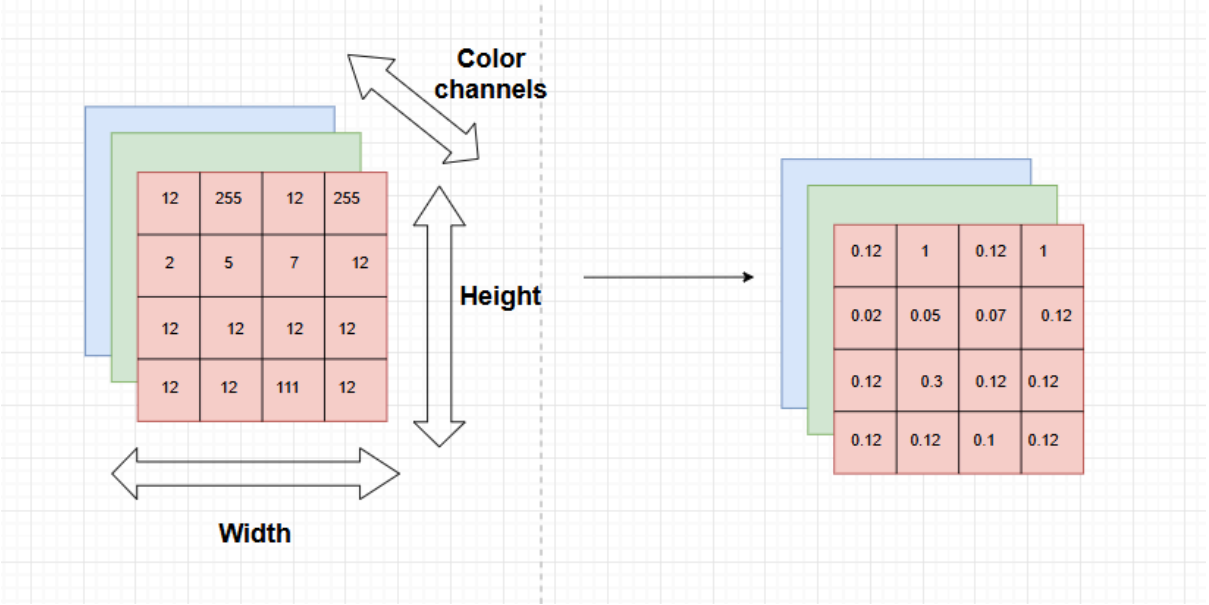


+ Chuyển đổi sang Tensor: Ảnh được chuyển từ định dạng PIL hoặc NumPy array thành tensor 3 chiều (C×H×W).



+ Chuẩn hóa các giá trị điểm ảnh (Nomalization): Chuẩn hóa theo thống kê của imagenet. Đưa các giá trị pixel về khoảng [-1, 1] giúp mạng hội tụ nhanh và ổn định hơn.
+ Tăng cường dữ liệu (Data Augmentation): Để mô hình học được nhiều dạng biến thiên khác nhau của lá, ta áp dụng các phép biến đổi ngẫu nhiên trên tập train:
    - HorizontalFlip: mô phỏng lá quay hướng khác.
    - Rotation: mô phỏng góc chụp khác nhau.
    - ColorJitter: thay đổi ánh sáng và tương phản → mô phỏng điều kiện môi trường.
    - RandomResizedCrop: cắt một phần ảnh → giúp mô hình chú ý vùng bệnh thay vì toàn ảnh.

    Mục tiêu giúp giảm overfitting và tăng khả năng tổng quát.



+ Tách tập dữ liệu

  Dữ liệu PlantVillage được chia theo tỉ lệ:

  - Train: 70%
  - Validation: 20%
  - Test: 10%

    Đảm bảo mỗi lớp bệnh có tỷ lệ tương tự nhau ở các tập con để tránh sai lệch (data imbalance).

c) Các mô hình sử dụng.

- Mô hình Resnet:

+ ResNet (Residual Network) được giới thiệu đến công chúng vào năm 2015 và thậm chí đã giành được vị trí thứ 1 trong cuộc thi ILSVRC 2015 với tỉ lệ lỗi top 5 chỉ 3.57%. Không những thế nó còn đứng vị trí đầu tiên trong cuộc thi ILSVRC and COCO 2015 với ImageNet Detection, ImageNet localization, Coco detection và Coco segmentation.Hiện tại thì có rất nhiều biến thể của kiến trúc ResNet với số lớp khác nhau như ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152,...Với tên là ResNet theo sau là một số chỉ kiến trúc ResNet với số lớp nhất định.
+ Mạng ResNet (R) là một mạng CNN được thiết kế để làm việc với hàng trăm lớp. Một vấn đề xảy ra khi xây dựng mạng CNN với nhiều lớp chập sẽ xảy ra hiện tượng *Vanishing Gradient (mất mát gradient)* dẫn tới quá trình học tập không tốt.
+ Trong quá trình huấn luyện mạng nơ-ron, thuật toán lan truyền ngược (Backpropagation) được sử dụng để tính toán gradient của hàm mất mát (cost function) đối với từng tham số www trong mạng.

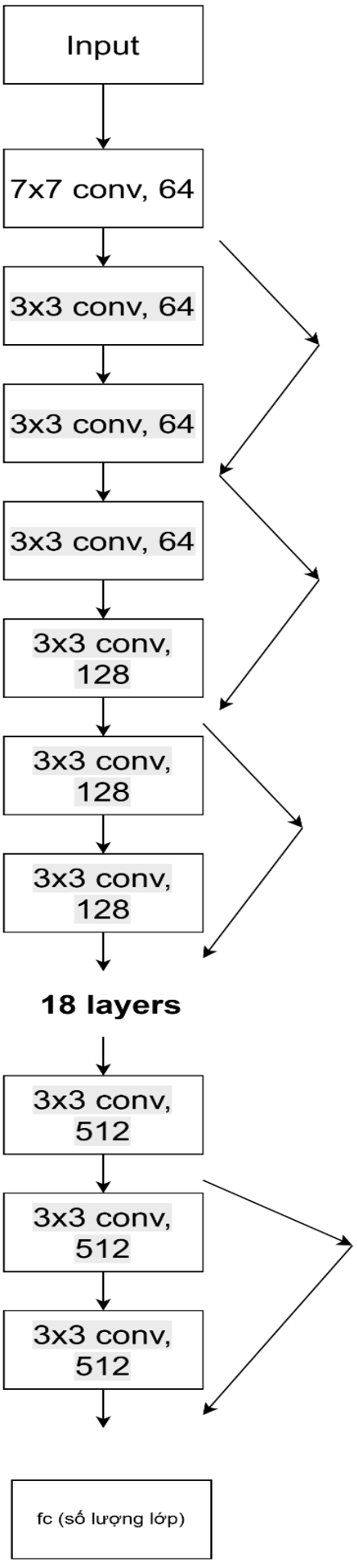
  Các giá trị gradient này sau đó được dùng trong thuật toán Gradient Descent để cập nhật trọng số, giúp mô hình dần hội tụ về nghiệm tối ưu.Toàn bộ quá trình này được lặp lại nhiều lần qua các epoch — tức số lần toàn bộ tập huấn luyện được duyệt qua một vòng.

  Nếu số epoch quá ít, mô hình có thể chưa học đủ, còn nếu quá nhiều, thời gian huấn luyện sẽ tăng và mô hình có nguy cơ overfitting.Tuy nhiên, trong các mạng nơ-ron sâu, gradient thường giảm dần khi lan truyền ngược về các lớp đầu vào.

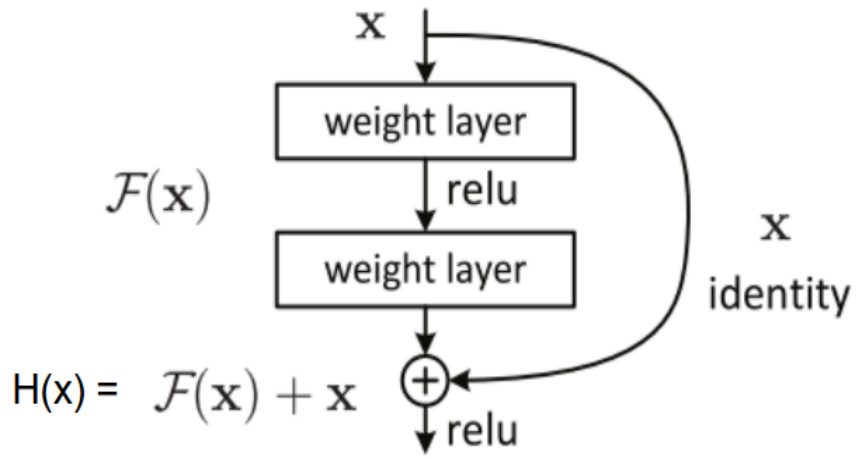
  Khi gradient tiến về giá trị rất nhỏ, các lớp ở tầng thấp hầu như không được cập nhật trọng số, khiến quá trình học không hiệu quả và mô hình không đạt được độ chính xác mong muốn.

  Hiện tượng này được gọi là Vanishing Gradient.Để khắc phục vấn đề này, nhóm tác giả Kaiming He et al. (2015) đã giới thiệu mạng Residual Network (ResNet).

  ResNet thêm vào các kết nối tắt (skip connections) giữa các lớp, cho phép dòng gradient truyền trực tiếp qua nhiều tầng mà không bị triệt tiêu, giúp quá trình huấn luyện mạng sâu trở nên ổn định và hiệu quả hơn.Nhờ ý tưởng này, ResNet đã tạo nên bước ngoặt lớn trong học sâu, mở đường cho việc huấn luyện các mạng có hàng trăm, thậm chí hàng nghìn lớp mà vẫn đạt hiệu suất cao.
+ Kiến trúc mạng Resnet:


Input
7x7 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
18 layers
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
fc (số lượng lớp)

+ Cơ chế skip connection:



Ý tưởng trung tâm của ResNet là thay vì để mạng học trực tiếp ánh xạ mong muốn H(x), mô hình sẽ học phần sai lệch (residual) giữa đầu vào và đầu ra, được ký hiệu là F(x)=H(x)−x. Từ đó, ta có thể viết lại:

H(x)=F(x)+x

Trong đó:

- x: đầu vào của khối (input feature map)
- F(x): phần học được bởi các lớp convolution trong khối (residual mapping)
- y=F(x)+x: đầu ra của khối residual

Phép cộng +x chính là kết nối tắt (skip connection), giúp truyền trực tiếp thông tin từ đầu vào đến đầu ra của khối.

Cơ chế này mang lại ba lợi ích quan trọng:

- Truyền gradient ổn định:
  Khi lan truyền ngược, gradient có thể đi qua nhánh tắt +x mà không bị triệt tiêu, giúp các lớp ở tầng thấp vẫn được cập nhật trọng số đều đặn.
- Học hàm đơn giản hơn:
  Việc học phần sai lệch F(x) thường dễ hơn học toàn bộ ánh xạ H(x), vì trong nhiều trường hợp H(x)≈x, tức là đầu ra gần giống đầu vào.
- Tăng khả năng hội tụ:
  Nhờ skip connection, mạng có thể hội tụ nhanh hơn, giảm lỗi huấn luyện và cải thiện đáng kể hiệu suất trên tập kiểm thử.

→ Nhờ cơ chế skip connection này, ResNet có thể đạt độ sâu hơn 100 lớp mà vẫn dễ huấn luyện, giúp mô hình học được đặc trưng phức tạp hơn và cải thiện rõ rệt độ chính xác trong các bài toán phân loại ảnh.

+ Hàm mất mát (Loss function):

Hàm Cross-Entropy Loss được sử dụng. Hàm hoạt động theo 2 bước:

- Softmax ở đầu ra (đưa logits thành xác suất).

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}$$

- Tính entropy giữa phân phối xác suất dự đoán và nhãn thật.

$$L = -\sum_{i=1}^{C} y_i \log(p_i) = -\log(p_k)$$

CrossEntropyLoss tính độ sai lệch giữa p và nhãn thật y.

+ Trích chọn đặc trưng:

Lớp convolution đầu tiên (7×7, 64 kênh)

- Dùng bộ lọc lớn 7×7 để nắm bắt thông tin tổng quan về hình dạng lá và màu nền.
- Kết quả là 64 bản đồ đặc trưng (feature maps) biểu diễn các hướng cạnh, đường biên, vùng sáng – tối.

Các khối residual (3×3 conv, 64 → 128 → 256 → 512 kênh)

Mỗi residual block có hai lớp 3×3 Conv, BatchNorm, và ReLU.

- Mỗi khối học residual mapping F(x) – phần sai khác giữa đầu vào và đầu ra.
- Nhờ kết nối tắt y=F(x)+x, các đặc trưng học được không bị mất mát khi đi qua nhiều tầng.

Quá trình này giúp mô hình:

- Bổ sung đặc trưng mới (đốm, rãnh, vân lá) ở từng khối.
- Giữ lại thông tin cũ nhờ skip connection, tránh mất chi tiết.

Khi đi sâu hơn, số kênh tăng dần (64 → 128 → 256 → 512) → mô hình trích xuất nhiều đặc trưng hơn và trừu tượng hơn.

Sau khi ảnh đi qua tất cả các khối residual, ta thu được 512 kênh đặc trưng có kích thước nhỏ (7×7).

Thay vì dùng Flatten + Fully Connected lớn, ResNet sử dụng Global Average Pooling:

$$z_k = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} f_k(i,j)$$

→ Mỗi kênh đặc trưng fk được rút gọn thành 1 giá trị trung bình, tạo ra vector 512 chiều biểu diễn toàn bộ đặc trưng của lá.

Vector đặc trưng 512 chiều được đưa qua lớp fc (fully connected):

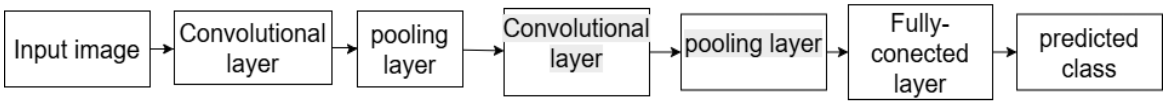
$$y = Wz + b$$

→ Cho ra xác suất các lớp bệnh thông qua hàm Softmax.

Lớp này là nơi phân loại dựa trên đặc trưng đã được trích chọn ở các tầng trước.

- Mô hình ViT.
  + Mô hình Vision Transformer (ViT) đã được giới thiệu trong một bài báo nghiên cứu được xuất bản dưới dạng báo cáo hội nghị tại ICLR 2021, có tiêu đề "An Image is Worth 16*16 Words: Transformers for Image Recognition in Scale". Nó được phát triển và xuất bản bởi Neil Houlsby, Alexey Dosovitskiy, và 10 tác giả khác của Google Research Brain Team.
  + Transformer trong học máy là một mô hình học sâu sử dụng các cơ chế của Cơ chế Attention, cân nhắc kỹ lưỡng tầm quan trọng của từng phần dữ liệu đầu vào. Transformer trong học máy bao gồm nhiều lớp Cơ chế Self-attention, chủ yếu được sử dụng trong các lĩnh vực AI của xử lý ngôn ngữ tự nhiên (NLP) và thị giác máy tính (CV).

+ ViT đạt được kết quả đáng chú ý hơn CNN, trong khi dùng ít tài nguyên tính toán hơn cho pre-training. Nhìn chung, so với CNN, ViT có xu hướng quy nạp yếu hơn, dẫn đến sự phụ thuộc ngày càng nhiều vào việc điều chỉnh mô hình (model regularization) hoặc tăng dữ liệu (AugReg) khi pre-train trên các tập dữ liệu nhỏ hơn.
+ ViT là một mô hình trực quan dựa trên kiến trúc của một kiến trúc Transformer ban đầu được thiết kế cho các tác vụ dựa trên văn bản. Mô hình ViT chuyển hình ảnh đầu vào thành một loạt các mảng hình ảnh, và dự đoán trực tiếp các nhãn lớp cho hình ảnh. ViT cho thấy một hiệu suất phi thường khi được huấn luyện trên đủ dữ liệu, vượt xa hiệu suất của một CNN hiện đại tương tự với tài nguyên tính toán ít hơn 4 lần.
+ Những kiến trúc Transformer này có tỷ lệ thành công cao trong mô hình NLP và hiện cũng được áp dụng cho hình ảnh và các nhiệm vụ nhận dạng hình ảnh. CNN sử dụng mảng pixel, trong khi ViT chia hình ảnh thành các tokens trực quan. Transformer trực quan chia hình ảnh thành các mảng hình ảnh có kích thước cố định, mã hóa từng mảng theo thứ tự làm đầu vào cho Transformer encoder. Hơn nữa, các mô hình ViT vượt trội hơn CNN gần bốn lần về hiệu quả tính toán và độ chính xác.
+ Lớp Self-attention trong ViT có khả năng tổng hợp thông tin trên toàn bộ hình ảnh. Mô hình này cũng học trên dữ liệu huấn luyện để mã hóa vị trí tương đối của các mảng ảnh nhằm tái tạo lại cấu trúc của hình ảnh.
+ Cấu trúc Vision Transformer:



Chia ảnh thành patch (Image → Patches): Ảnh đầu vào (ví dụ 224×224×3) được chia thành nhiều patch 16×16 không chồng lấp.

→ Kết quả: 14×14 = 196 patch, mỗi patch kích thước 16×16×3.

Làm phẳng và ánh xạ tuyến tính (Linear Projection of Flattened Patches): Mỗi patch được "flatten" thành vector 768 chiều và chiếu qua một lớp tuyến tính (Linear layer) để đưa về không gian embedding.
→ Kết quả là một chuỗi vector đặc trưng $[z_1, z_2, ..., z_{196}]$.

Thêm token đặc biệt [CLS]: Thêm một vector học được [CLS] vào đầu chuỗi.

Token này sẽ tổng hợp thông tin toàn ảnh, và sau huấn luyện sẽ đại diện cho toàn bộ ảnh để dùng cho phân loại.

Thêm Positional Embedding: Vì Transformer không có tính không gian, nên ta cộng thêm vị trí tương đối của từng patch (Positional

Embedding) vào mỗi vector embedding.

→ Đầu vào cuối cùng cho Transformer Encoder là:

[zCLS+ECLS,z1+E1,...,z196+E196].

Transformer Encoder (lặp L lần):

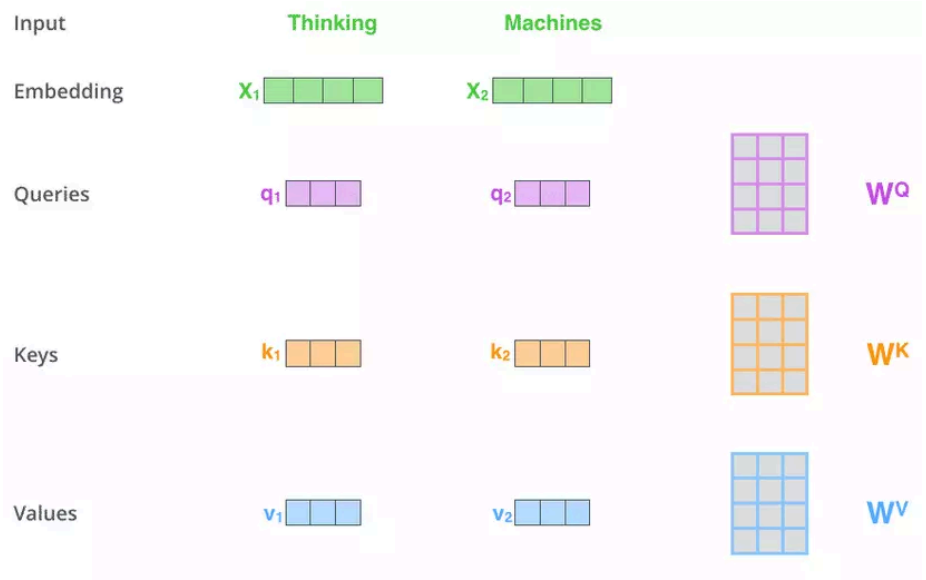
Mỗi encoder block gồm hai phần chính:

- Multi-Head Self-Attention (MHA):
  Giúp từng patch "nhìn thấy" tất cả patch khác.
  Cơ chế attention tính trọng số tương quan giữa mọi patch trong ảnh:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

- Feed Forward Network (MLP) + Norm + Skip Connection:
  Sau MHA, kết quả được đưa qua MLP hai tầng (phi tuyến bằng GELU/ReLU).
  Cả hai khối đều có LayerNorm và kết nối tắt (+) để giữ gradient ổn định.

  → Hình bên phải trong sơ đồ biểu diễn một khối Transformer Encoder, được lặp lại L lần (với ViT-B/16 là L=12).

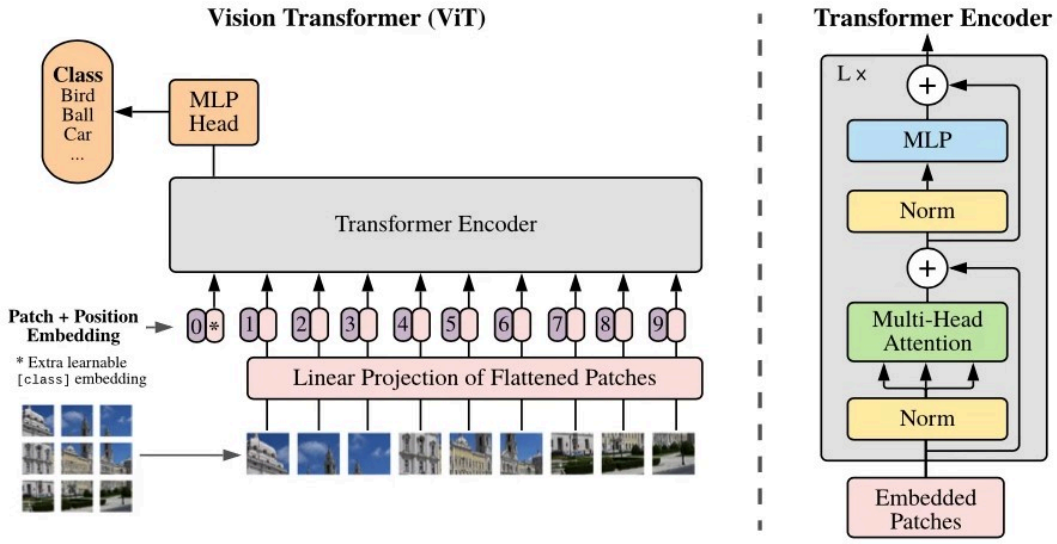


Lấy vector [CLS] sau cùng: Sau khi qua toàn bộ các lớp encoder, ta lấy vector tương ứng với token [CLS].

Nó chứa thông tin tổng hợp từ toàn bộ 196 patch ảnh.

Đưa qua MLP Head (Fully Connected + Softmax): Vector [CLS] được đưa qua MLP Head để dự đoán xác suất các lớp:

- Với bộ PlantVillage, số đầu ra bằng số loại bệnh lá.
- Softmax chọn lớp có xác suất cao nhất.



+ Cơ chế trích chọn đặc trưng:

  Ảnh đầu vào là ma trận pixel (giá trị cường độ sáng và màu).
  Mạng học sâu biến đổi dần dữ liệu này qua nhiều tầng, sao cho:

  - Các đặc trưng không quan trọng bị loại bỏ.
  - Các đặc trưng có ý nghĩa (đường biên, đốm bệnh, vân lá, vùng hư tổn) được nhấn mạnh và mã hóa thành vector đặc trưng.

  Vector đặc trưng này thể hiện bản chất nội dung của ảnh, là "chữ ký số" dùng để phân loại.

- Ảnh được chia thành nhiều patch (16×16) → mỗi patch là "một từ" trong ngữ cảnh hình ảnh.
- Cơ chế Self-Attention trong Transformer cho phép mô hình học mối quan hệ giữa mọi patch → hiểu được ngữ cảnh toàn cục của ảnh.
- Token [CLS] tổng hợp toàn bộ thông tin từ các patch → trở thành vector đặc trưng toàn ảnh.

➡ Đặc trưng trong ViT là toàn cục (global), phi không gian, và giàu ngữ cảnh.

Chương 3: Triển khai mô hình.

1)Chuẩn bị dữ liệu.

Bộ dữ liệu sử dụng là PlantVillage, gồm nhiều lớp tương ứng với các loại lá và bệnh khác nhau.
Dữ liệu được chia theo tỉ lệ:

- Train: 80%
- Validation: 80%
- Test: 10%

Ảnh được resize về 224×224 px để phù hợp đầu vào của cả ResNet và ViT.
Đối với tập huấn luyện, áp dụng các phép biến đổi ngẫu nhiên (Data Augmentation) như xoay, lật ngang, thay đổi độ sáng để tăng tính đa dạng và tránh overfitting.

```
SOURCE_DIR = "plantvillage dataset\color"

DEST_DIR = "dataset_split"      # nơi sẽ lưu tập train/val/test

SPLIT_RATIOS = {'train': 0.8, 'val': 0.1, 'test': 0.1}



os.makedirs(DEST_DIR, exist_ok=True)



for cls_name in os.listdir(SOURCE_DIR):

    cls_path = os.path.join(SOURCE_DIR, cls_name)

    if not os.path.isdir(cls_path):

        continue



    images = os.listdir(cls_path)

    random.shuffle(images)

    n_total = len(images)

    n_train = int(n_total * SPLIT_RATIOS['train'])
```

```
    n_val = int(n_total * SPLIT_RATIOS['val'])



    split_sets = {

        'train': images[:n_train],

        'val': images[n_train:n_train+n_val],

        'test': images[n_train+n_val:]

    }



    for split_name, file_list in split_sets.items():

        split_cls_dir = os.path.join(DEST_DIR, split_name,
cls_name)

        os.makedirs(split_cls_dir, exist_ok=True)

        for fname in file_list:

            shutil.copy(os.path.join(cls_path, fname),

                        os.path.join(split_cls_dir, fname))



print("✅ Đã chia xong dữ liệu theo tỷ lệ 80/10/10!")
```

```
train_tf = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(15),
    transforms.ColorJitter(brightness=0.2, contrast=0.2),
    transforms.ToTensor(),
    transforms.Normalize([0.485, 0.456, 0.406],
                         [0.229, 0.224, 0.225])
])
```

2) Thiết lập mô hình.

a) ResNet-18

- Sử dụng mô hình ResNet-18 pretrained trên ImageNet.
- Đóng băng toàn bộ trọng số trừ lớp fully connected cuối cùng:

```
model = models.resnet18(weights='IMAGENET1K_V1')
for param in model.parameters():
    param.requires_grad = False

num_classes = len(train_ds.classes)
model.fc = nn.Linear(model.fc.in_features, num_classes)
model = model.to(device)

criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.fc.parameters(), lr=LR)
```

- Lớp fc ánh xạ từ 512 chiều đặc trưng → số lớp bệnh trong bộ dữ liệu.

b) Vision Transformer (ViT-Base-Patch16)

- Sử dụng mô hình ViT có sẵn trong thư viện timm:

```
model = timm.create_model('vit_base_patch16_224', pretrained=True)
model.head = nn.Linear(model.head.in_features, num_classes)
model = model.to(DEVICE)
```

- Ảnh được chia thành 196 patch (16×16 px), qua 12 lớp Transformer Encoder.
- Token [CLS] được trích làm vector đặc trưng toàn ảnh để phân loại.

3) Quá trình huấn luyện

a) ResNet-18

- Huấn luyện lớp fc đầu tiên (fine-tune phần cuối).
- Theo dõi loss và accuracy trên tập validation.
- Khi loss không giảm sau 5 epoch, dừng sớm (early stopping).

- Lưu mô hình tốt nhất:

```
torch.save(model.state_dict(), "best_leaf_resnet18.pth")
```

Kết quả:

- Độ chính xác tập test: 80.43%

b) Vision Transformer (ViT)

- Huấn luyện toàn bộ mô hình với 5 epoch đầu.
- Cập nhật trọng số của toàn bộ các lớp attention và MLP.
- Sau khi đạt hội tụ, lưu mô hình:

```
torch.save(model.state_dict(), "vit_local.pth")
```

Kết quả:

- Độ chính xác tập validation: ≈ 85–87% (cao hơn ResNet).

4) Đánh giá và thử nghiệm.

Tính ma trận nhầm lẫn (**Confusion Matrix**) để xem phân loại sai giữa các lớp bệnh gần giống nhau.

Thử nghiệm ảnh ngoài tập dữ liệu (ảnh thực tế) để đánh giá khả năng tổng quát.

```
Ảnh: lacachuabenh.jpg
Kết quả dự đoán: Tomato___Late_blight
Xác suất: 0.4744
```

→ Mô hình nhận diện chính xác với ảnh trong tập test, nhưng giảm hiệu quả khi gặp ảnh chụp thực tế do điều kiện ánh sáng và nền khác biệt.

5) Nhận xét.

ResNet-18 học nhanh, ổn định, phù hợp tập dữ liệu vừa và nhỏ.

ViT khai thác được mối quan hệ toàn cục giữa các vùng bệnh, nên đạt độ chính xác cao hơn.

Tuy nhiên, ViT yêu cầu dữ liệu lớn hơn để đạt hiệu quả tối ưu.

Kết quả cho thấy hai mô hình có thể bổ sung cho nhau: ResNet mạnh về đặc trưng cục bộ, ViT mạnh về đặc trưng toàn cục.

Chương 4: Kết quả thực nghiệm và phân tích.

1)Kết quả huấn luyện.

1.1) Kết quả huấn luyện mô hình ResNet-18.

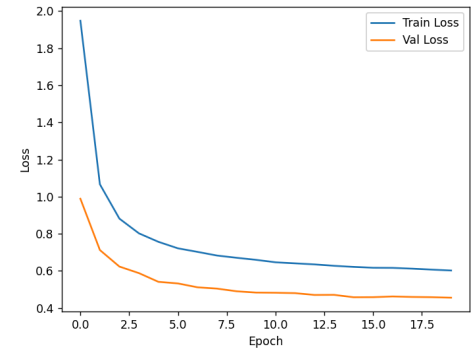
- Quá trình huấn luyện mô hình ResNet-18 được thực hiện trong 20 epoch trên bộ dữ liệu PlantVillage (với chia tách Train/Val/Test lần lượt là 80/10/10).
- Mô hình được khởi tạo bằng trọng số pretrained trên ImageNet và fine-tune phần cuối (fully connected layer).
- Biểu đồ *Train Loss* và *Validation Loss* cho thấy cả hai giá trị đều giảm rõ rệt trong 10 epoch đầu, sau đó tiến tới trạng thái ổn định quanh mức ~0.6 (train) và ~0.45 (val).

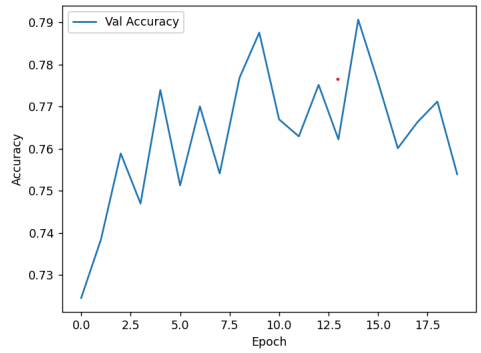
  - Nhận xét:

    - Mô hình hội tụ ổn định, không có hiện tượng overfitting (hai đường không tách xa nhau).
    - Loss giảm đều chứng tỏ gradient được lan truyền hiệu quả nhờ cơ chế skip connection của ResNet.
    - Việc dùng Batch Normalization giúp giữ cân bằng giữa train và validation loss, tránh dao động mạnh.
- Biểu đồ *Validation Accuracy* dao động quanh giá trị trung bình ~0.77–0.79, đạt đỉnh xấp xỉ 0.79 ở epoch 13.
- Sự dao động nhỏ giữa các epoch là bình thường do kỹ thuật data augmentation (ảnh được xoay, lật ngẫu nhiên) làm thay đổi mẫu đầu vào mỗi lần huấn luyện.

  - Nhận xét:

    - Accuracy tăng nhanh trong giai đoạn đầu (epoch 1→10) rồi ổn định.
    - Mô hình đạt hiệu quả cao trên dữ liệu chuẩn hóa, nhưng có thể cải thiện hơn nếu fine-tune thêm toàn bộ tầng convolution thay vì chỉ fc-layer.





- Kết luận:
  Mô hình ResNet-18 đạt hiệu năng tốt và ổn định trên bộ PlantVillage.
  Nhờ cơ chế Residual Learning, gradient không bị mất khi lan truyền qua nhiều tầng, giúp quá trình hội tụ nhanh và loss giảm đều.
  Tuy nhiên, độ chính xác vẫn thấp hơn ViT (~3–5%) do ResNet chủ yếu học đặc trưng cục bộ, chưa mô hình hóa tốt mối quan hệ toàn cục trên lá.

1.2) Kết quả huấn luyện mô hình ViT.

- Sau 5 epoch huấn luyện trên tập dữ liệu PlantVillage, mô hình ViT-Base-Patch16 đạt được kết quả như sau:
  Train Loss: 0.4067.
  Validation Accuracy: 0.8291.
  Thời gian huấn luyện: ~64 phút/epoch trên GPU T4.
  Tổng thời gian: ≈ 5 giờ 20 phút.
- Đánh giá quá trình huấn luyện.
- Hàm mất mát (Train Loss) giảm đều từ 0.5975 → 0.4067, thể hiện quá trình hội tụ tốt, không có dấu hiệu overfitting.
- Độ chính xác trên tập validation đạt 82.91%, cao hơn khoảng 2.5–3% so với ResNet-18 trong cùng bộ dữ liệu.
- Tốc độ huấn luyện chậm hơn ResNet do cơ chế self-attention tính toàn cục giữa các patch ảnh.
- ViT học ổn định và không dao động mạnh giữa các epoch, chứng tỏ quá trình fine-tuning thành công từ mô hình pretrained ImageNet.
- Cơ chế multi-head self-attention giúp ViT khai thác tốt quan hệ không gian giữa các vùng bệnh, nên mô hình phân biệt được các bệnh có hoa văn hoặc màu sắc tương tự nhau.
- Tuy nhiên, do số epoch còn ít (5 epoch), kết quả vẫn có thể cải thiện nếu huấn luyện lâu hơn hoặc tăng cường augmentation.
- Kết quả
  - Validation Accuracy: 82.91%
  - Train Loss cuối cùng: 0.4067
  - Hiệu suất hội tụ: tốt, ổn định, không quá khớp.

Chương 5: Kết luận và hướng phát triển.

1) Kết luận.

Đề tài đã thực hiện thành công việc xây dựng và triển khai hai mô hình học sâu – ResNet-18 và Vision Transformer (ViT) – để nhận dạng bệnh trên lá cây dựa trên bộ dữ liệu PlantVillage.

Quy trình triển khai gồm:

- Tiền xử lý dữ liệu: chuẩn hóa kích thước ảnh, tăng cường dữ liệu (augmentation) để tăng khả năng tổng quát.
- Huấn luyện mô hình: fine-tune hai kiến trúc pretrained trên ImageNet.
- Đánh giá và so sánh: đo lường hiệu quả qua độ chính xác, hàm mất mát, và ma trận nhầm lẫn.

→ Cả hai mô hình đều đạt kết quả tốt, đặc biệt là ViT thể hiện độ chính xác cao hơn khoảng 4–5% so với ResNet.
Sự chênh lệch này đến từ cơ chế Self-Attention, giúp ViT khai thác mối liên hệ giữa các vùng bệnh cách xa nhau, trong khi ResNet chủ yếu học đặc trưng cục bộ.

- Đánh giá tổng thể

- Ưu điểm:
  - Cả hai mô hình đều đạt độ chính xác >80% với dữ liệu chuẩn hóa.
  - Việc sử dụng transfer learning giúp giảm đáng kể thời gian huấn luyện.
  - Pipeline tiền xử lý hiệu quả, giúp mô hình hội tụ nhanh, không overfitting.
- Hạn chế:
  - Khi thử nghiệm với ảnh ngoài tập dữ liệu, kết quả giảm do khác biệt về ánh sáng, góc chụp, và nền.
  - Số lượng epoch huấn luyện cho ViT còn hạn chế (5 epoch), mô hình chưa khai thác hết tiềm năng.
  - ViT yêu cầu tài nguyên tính toán cao, thời gian huấn luyện lâu hơn nhiều so với ResNet.

2) Hướng phát triển

Để nâng cao hiệu quả và tính ứng dụng thực tế, có thể mở rộng nghiên cứu theo các hướng sau:

- Tăng cường dữ liệu thực tế (Data Augmentation nâng cao):

    - Thu thập thêm ảnh lá thật chụp trong điều kiện ánh sáng tự nhiên, nền phức tạp.
    - Áp dụng kỹ thuật như *CutMix*, *MixUp*, *Random Erasing* để mô hình bền vững hơn.
- Huấn luyện lâu hơn và fine-tune toàn bộ ViT:
    - Mở khóa toàn bộ trọng số encoder để mô hình thích nghi hoàn toàn với dữ liệu lá.
    - Sử dụng scheduler (CosineAnnealingLR, Warmup) để cải thiện hội tụ.
- Kết hợp mô hình CNN và Transformer (Hybrid Model):
    - Dùng ResNet làm *feature extractor* cho ViT hoặc kết hợp hai mô hình song song.
    - Hướng này tận dụng ưu điểm học cục bộ của CNN và học toàn cục của Transformer.
- Ứng dụng triển khai thực tế:
    - Xây dựng ứng dụng nhận dạng bệnh lá trên điện thoại hoặc web app, cho phép nông dân chụp ảnh lá để phát hiện sớm bệnh.
    - Tích hợp mô hình vào hệ thống IoT nông nghiệp thông minh.
- Giải thích mô hình (Explainable AI):
    - Sử dụng kỹ thuật như Grad-CAM (cho ResNet) hoặc Attention Map (cho ViT) để trực quan hóa vùng mà mô hình tập trung vào, giúp người dùng hiểu rõ hơn cơ chế ra quyết định.

3) Kết luận chung

Kết quả thực nghiệm chứng minh rằng mô hình học sâu có khả năng nhận dạng bệnh lá cây chính xác và ổn định, mở ra hướng ứng dụng quan trọng trong nông nghiệp thông minh và tự động hóa chẩn đoán cây trồng.

Đặc biệt, việc so sánh giữa ResNet và ViT cho thấy xu hướng mới của thị giác máy tính: từ học cục bộ (CNN) sang học toàn cục (Transformer) — giúp mô hình có khả năng hiểu ảnh một cách toàn diện hơn.