KẾ HOẠCH CHI TIẾT ĐỒ ÁN PHÁT HIỆN BỆNH LÁ SẦU RIÊNG

GIAI ĐOẠN 1: CHUẨN BỊ DỮ LIỆU (1-2 tuần)

1.1 Thu Thập và Khảo Sát Dataset

Mục tiêu: Hiểu rõ dữ liệu đang làm việc

- Xác định nguồn dataset (Kaggle, tự thu thập, hoặc từ đối tác nông nghiệp)
- Kiểm tra số lượng ảnh trong mỗi class bệnh lá (Lá khỏe mạnh, Bệnh đốm lá, Bệnh thán thư, các bệnh khác)
- Phân tích phân bố dữ liệu: Có mất cân bằng (imbalanced) giữa các class không?
- Ghi chép lại kích thước ảnh gốc, định dạng file, độ phân giải
- Quan sát trực quan các mẫu ảnh để hiểu đặc điểm của từng loại bệnh

1.2 Phân Tích Chất Lượng Dữ Liệu

Mục tiêu: Đánh giá và lọc dữ liệu chất lượng

- Kiểm tra ảnh bị lỗi, corrupt, hoặc không mở được
- Xác định ảnh có nhãn sai hoặc mơ hồ (cần expert kiểm tra)
- Đánh giá điều kiện chụp ảnh: ánh sáng, góc chụp, background
- Xác định các trường hợp biên (edge cases) khó phân loại
- Quyết định tiêu chí loại bỏ hoặc giữ lại ảnh

1.3 Tổ Chức và Phân Chia Dataset

Mục tiêu: Chuẩn bị cấu trúc thư mục chuẩn

- Tạo cấu trúc thư mục theo format ImageFolder của PyTorch (hoặc tương tự)
  - train/class_name/
  - val/class_name/
  - test/class_name/
- Phân chia dữ liệu theo tỷ lệ: 70% train, 15% validation, 15% test (có thể điều chỉnh)
- Đảm bảo phân chia ngẫu nhiên nhưng có stratified (giữ tỷ lệ các class)
- Lưu lại random seed để có thể reproduce kết quả
- Tạo file metadata chứa thông tin về từng ảnh (optional nhưng hữu ích)

1.4 Data Augmentation Planning

Mục tiêu: Lên kế hoạch tăng cường dữ liệu

- Xác định các phép biến đổi phù hợp với bài toán:
  - Random Horizontal Flip: phù hợp vì lá có thể lật ngang
  - Random Rotation (10-15 độ): mô phỏng góc chụp khác nhau
  - Random Brightness/Contrast: mô phỏng điều kiện ánh sáng
  - Random Crop và Resize: tăng tính đa dạng vị trí
  - Color Jitter: thay đổi màu sắc nhẹ
- Quyết định augmentation nào chỉ áp dụng cho training
- Validation và Test set không augment (giữ nguyên để đánh giá công bằng)
- Cân nhắc augmentation mạnh hơn cho class có ít mẫu (nếu imbalanced)

GIAI ĐOẠN 2: XÂY DỰNG BASELINE MODEL - ResNet18 (1-2 tuần)

2.1 Setup Môi Trường Phát Triển

Mục tiêu: Chuẩn bị environment làm việc

- Cài đặt các thư viện cần thiết: PyTorch, torchvision, numpy, matplotlib, seaborn, scikit-learn
- Kiểm tra GPU availability (sử dụng CUDA nếu có)
- Thiết lập Jupyter Notebook hoặc VS Code
- Tạo requirements.txt để quản lý dependencies
- Setup version control với Git

2.2 Xây Dựng Data Pipeline

Mục tiêu: Tạo data loader hiệu quả

- Tạo DataLoader cho train/val/test với batch size phù hợp (16, 32, hoặc 64)
- Implement data augmentation transforms cho training set
- Tạo transforms cho validation/test (chỉ resize và normalize)
- Tính toán mean và std của dataset để normalize đúng cách
- Viết code visualize một batch để kiểm tra data pipeline hoạt động đúng

2.3 Load và Fine-tune ResNet18 Pretrained

Mục tiêu: Tận dụng transfer learning

- Load ResNet18 pretrained trên ImageNet từ torchvision
- Quyết định chiến lược fine-tuning:
  - Option 1: Freeze các layer đầu, chỉ train classifier cuối
  - Option 2: Train toàn bộ model với learning rate nhỏ
  - Option 3: Unfreeze dần các layer sau mỗi vài epoch
- Thay thế fully connected layer cuối để match số class của bài toán
- Khởi tạo classifier layer mới với phân phối phù hợp

2.4 Thiết Lập Training Loop

Mục tiêu: Chuẩn bị quy trình training

- Chọn loss function: CrossEntropyLoss (có thể thêm class weights nếu imbalanced)
- Chọn optimizer: Adam hoặc SGD với momentum
- Thiết lập learning rate schedule:
  - StepLR: giảm LR sau mỗi X epoch
  - ReduceLROnPlateau: giảm LR khi validation loss không cải thiện
- Quyết định số epoch training (30-50 epoch ban đầu)
- Implement early stopping để tránh overfitting
- Thiết lập checkpoint saving: lưu best model dựa trên validation accuracy

2.5 Training và Monitoring

Mục tiêu: Train model và theo dõi kết quả

- Bắt đầu training với cấu hình ban đầu
- Log metrics sau mỗi epoch:
  - Training loss, Training accuracy
  - Validation loss, Validation accuracy
  - Learning rate hiện tại
- Vẽ learning curves (loss và accuracy qua các epoch)
- Quan sát xem có overfitting không (train acc cao, val acc thấp)
- Điều chỉnh hyperparameters nếu cần:
  - Tăng/giảm learning rate
  - Thêm/bớt regularization (weight decay)
  - Điều chỉnh batch size

- Mục tiêu: đạt ~82-88% validation accuracy

2.6 Evaluation trên Test Set

Mục tiêu: Đánh giá hiệu suất thực tế

- Load best checkpoint model
- Chạy inference trên test set (không train nữa)
- Tính toán các metrics:
  - Overall Accuracy
  - Per-class Accuracy
  - Precision, Recall, F1-score cho từng class
- Tạo Confusion Matrix để visualize
- Phân tích các trường hợp model dự đoán sai:
  - Class nào hay bị nhầm với class nào?
  - Có pattern nào trong các ảnh bị sai không?

GIAI ĐOẠN 3: CẢI TIẾN VỚI ResNet34 (1 tuần)

3.1 Phân Tích Kết Quả ResNet18

Mục tiêu: Hiểu điểm mạnh/yếu của baseline

- Review confusion matrix: class nào perform kém nhất?
- Xem xét false positives và false negatives nguy hiểm
- Đánh giá xem model có đang underfit hay overfit không
- Quyết định điểm cần cải thiện

3.2 Training ResNet34

Mục tiêu: Tăng capacity của model

- Load ResNet34 pretrained (mô hình sâu hơn, nhiều tham số hơn)
- Sử dụng configuration tương tự ResNet18 làm baseline
- Training với cùng data pipeline
- So sánh training time và resource usage
- Monitor validation accuracy, mục tiêu: 88-92%

3.3 Hyperparameter Tuning

Mục tiêu: Tối ưu hóa performance

- Thử nghiệm với các learning rate khác nhau (learning rate sweep)
- Điều chỉnh weight decay để cải thiện generalization
- Thử các augmentation mạnh hơn nếu cần
- Cân nhắc mixup hoặc cutmix nếu muốn advanced technique
- Có thể thử test-time augmentation (TTA) để tăng accuracy

GIAI ĐOẠN 4: MÔ HÌNH TỐI ƯU - ResNet50 (1 tuần)

4.1 Training ResNet50

Mục tiêu: Đạt performance tốt nhất

- Load ResNet50 pretrained (model sâu và mạnh nhất trong 3 model)
- Training với best configuration từ ResNet34
- Có thể cần tăng batch size hoặc gradient accumulation nếu GPU memory đủ
- Mục tiêu: đạt trên 92% validation accuracy

4.2 Ensemble Methods (Optional)

Mục tiêu: Tăng độ chính xác thêm nữa

- Kết hợp predictions từ ResNet18, ResNet34, ResNet50
- Thử voting schemes: majority voting, weighted averaging
- Đánh giá xem ensemble có cải thiện không

4.3 Model Analysis

Mục tiêu: Hiểu model đã học được gì

- Visualize filters/activations của các layer đầu
- Phân tích feature maps ở các stage khác nhau
- Xác định model đang focus vào đâu trên ảnh lá

GIAI ĐOẠN 5: INTERPRETABILITY - GRAD-CAM (3-5 ngày)

5.1 Implement Grad-CAM

Mục tiêu: Giải thích quyết định của model

- Hiểu lý thuyết Grad-CAM: sử dụng gradient để tìm vùng quan trọng
- Implement Grad-CAM cho ResNet architecture
- Chọn target layer phù hợp (thường là layer cuối của feature extractor)
- Test Grad-CAM trên một vài ảnh mẫu

5.2 Visualization và Analysis

Mục tiêu: Tạo heatmap trực quan

- Overlay Grad-CAM heatmap lên ảnh gốc
- Tạo visualization cho cả dự đoán đúng và sai
- Phân tích xem model có thực sự nhìn vào vùng bệnh hay không
- Tạo function để generate Grad-CAM cho bất kỳ ảnh nào

GIAI ĐOẠN 6: XÂY DỰNG ỨNG DỤNG DEMO (1 tuần)

6.1 Thiết Kế Interface

Mục tiêu: Tạo giao diện user-friendly trong Jupyter Notebook

- Sử dụng ipywidgets để tạo interactive widgets
- Thiết kế layout gồm:
  - Upload button để chọn ảnh
  - Display area để hiển thị ảnh đã upload
  - Prediction button để chạy model
  - Results area để hiển thị kết quả và confidence score
  - Grad-CAM visualization area

6.2 Implement Upload và Prediction

Mục tiêu: Xử lý upload và inference

- Tạo function upload ảnh từ local
- Preprocessing ảnh đúng format (resize, normalize)
- Load best model checkpoint
- Chạy inference và lấy prediction
- Tính confidence scores cho tất cả các classes
- Hiển thị top-3 predictions với xác suất

6.3 Grad-CAM Integration

Mục tiêu: Thêm khả năng giải thích

- Tích hợp Grad-CAM vào demo
- Tự động generate Grad-CAM heatmap sau khi predict
- Hiển thị heatmap cạnh ảnh gốc
- Thêm toggle để bật/tắt Grad-CAM visualization

6.4 Polish và User Experience

Mục tiêu: Hoàn thiện trải nghiệm

- Thêm loading indicator khi đang xử lý
- Error handling cho các trường hợp: ảnh không hợp lệ, format sai, etc.
- Thêm instructions/tooltips để hướng dẫn người dùng
- Styling với CSS trong notebook để đẹp hơn
- Thêm option để save kết quả

GIAI ĐOẠN 7: ĐÁNH GIÁ TOÀN DIỆN (3-5 ngày)

7.1 Model Performance Report

Mục tiêu: Tổng hợp kết quả của tất cả models

- Tạo bảng so sánh ResNet18, ResNet34, ResNet50:
  - Test Accuracy
  - Precision/Recall/F1 cho từng class
  - Inference time
  - Model size
  - Training time
- Vẽ confusion matrices cho cả 3 models
- Phân tích trade-off giữa accuracy và efficiency

7.2 Per-Class Analysis

Mục tiêu: Hiểu performance chi tiết từng loại bệnh

- Tạo report cho từng class bệnh:
  - Số lượng mẫu trong train/val/test

    - Accuracy, Precision, Recall riêng
    - Các class hay bị nhầm lẫn
- Tìm class khó nhất và dễ nhất để phân loại
- Đề xuất cải thiện cho các class perform kém

7.3 Error Analysis

Mục tiêu: Học từ những sai lầm

- Lấy tất cả ảnh bị predict sai
- Phân loại các loại lỗi:
    - Lỗi do ảnh chất lượng thấp
    - Lỗi do nhãn gốc sai
    - Lỗi do trường hợp biên (nhiều bệnh cùng lúc)
    - Lỗi do thiếu context
- Visualize các ảnh sai tiêu biểu
- Đưa ra khuyến nghị để cải thiện

7.4 Practical Considerations

Mục tiêu: Đánh giá tính ứng dụng thực tế

- Test với ảnh chụp từ môi trường thực tế (nếu có)
- Đánh giá robustness với:
    - Ảnh có background phức tạp
    - Ảnh chụp trong điều kiện ánh sáng khác nhau
    - Ảnh có nhiễu, mờ
- Đề xuất cách cải thiện model cho production

GIAI ĐOẠN 8: DOCUMENTATION VÀ DEPLOYMENT (3-5 ngày)

8.1 Code Documentation

Mục tiêu: Code dễ hiểu và maintain

- Thêm docstrings cho tất cả functions và classes
- Comment các phần code phức tạp
- Tạo README.md hướng dẫn cài đặt và chạy

- Tổ chức code thành modules rõ ràng:
  - data_processing.py
  - model.py
  - train.py
  - evaluate.py
  - visualize.py
  - demo.py

8.2 Technical Report

Mục tiêu: Viết báo cáo đầy đủ

- Giới thiệu bài toán và động lực
- Mô tả dataset và preprocessing
- Giải thích kiến trúc models
- Trình bày kết quả thực nghiệm
- So sánh các approaches
- Discussion và Conclusion
- Future work và limitations

8.3 Presentation Materials

Mục tiêu: Chuẩn bị demo cho giảng viên

- Tạo slides PowerPoint/Google Slides
- Chuẩn bị demo video hoặc live demo
- Tạo poster nếu cần
- Prepare Q&A về technical choices

8.4 Deployment Preparation (Optional)

Mục tiêu: Sẵn sàng cho ứng dụng thực tế

- Export model sang ONNX hoặc TorchScript để optimize
- Tạo API endpoint với Flask/FastAPI (nếu muốn web app)
- Dockerize application để dễ deploy
- Cân nhắc model compression/quantization nếu cần mobile deployment

---

GIAI ĐOẠN 9: TESTING VÀ VALIDATION (2-3 ngày)

9.1 Unit Testing

Mục tiêu: Đảm bảo code chạy đúng

- Viết tests cho data pipeline
- Test model inference với dummy data
- Test Grad-CAM implementation
- Test preprocessing functions

9.2 Integration Testing

Mục tiêu: Test end-to-end workflow

- Test toàn bộ quy trình từ upload ảnh đến kết quả
- Test với nhiều loại ảnh khác nhau
- Test edge cases và error handling
- Đảm bảo demo notebook chạy smoothly từ đầu đến cuối

CHECKLIST CUỐI CÙNG

Before Submission

- [ ] Code chạy được trên máy khác (test với bạn bè)
- [ ] Requirements.txt đầy đủ và cập nhật
- [ ] README.md hướng dẫn chi tiết
- [ ] All notebooks có outputs (đừng clear)
- [ ] Model checkpoints được lưu và có thể load
- [ ] Demo notebook chạy mượt mà
- [ ] Báo cáo kỹ thuật hoàn chỉnh
- [ ] Presentation slides sẵn sàng
- [ ] Dataset organized và có thể reproduce
- [ ] Git repository clean và organized

Documentation Checklist

- [ ] Mô tả rõ dataset source và licenses
- [ ] Giải thích tất cả hyperparameters
- [ ] Screenshots của results
- [ ] Confusion matrices cho tất cả models
- [ ] Learning curves
- [ ] Grad-CAM examples

- [ ] Comparison tables
- [ ] Error analysis với examples

TIPS ĐỂ THÀNH CÔNG

Technical Tips

1. **Version Control:** Commit thường xuyên, mỗi feature một branch
2. **Experiment Tracking:** Dùng TensorBoard hoặc Weights & Biases để log experiments
3. **Save Everything:** Lưu checkpoints, configs, và random seeds
4. **Reproducibility:** Document mọi thứ để có thể chạy lại

Time Management

1. **Ưu tiên baseline:** Có model chạy được trước, optimize sau
2. **Don't Perfectionism:** 85% accuracy với code clean tốt hơn 90% accuracy với code mess
3. **Buffer Time:** Luôn dành thời gian dự phòng cho bugs và issues

Academic Tips

1. **Literature Review:** Đọc papers về disease detection và plant pathology
2. **Cite Sources:**