PROPOSAL: PHÁT HIỆN BỆNH CÂY TRÊN THIẾT BỊ DI ĐỘNG

Tên đề tài (Tiếng Việt): Nghiên cứu tối ưu hóa Mạng nơ-ron Tích chập Hạng nhẹ (Lightweight CNNs) kết hợp Kỹ thuật Lượng tử hóa (Quantization) để phát hiện sớm bệnh cây trồng trên thiết bị di động ngoại tuyến (Offline Edge Devices).

Tên đề tài (Tiếng Anh - Suggested): *On-device Early Plant Disease Detection using Lightweight Convolutional Neural Networks and Quantization-aware Transfer Learning under Unconstrained Field Conditions.*

1. Đặt vấn đề (Problem Statement)

- **Thực trạng:** Sâu bệnh gây thiệt hại 20-40% năng suất cây trồng toàn cầu. Việc chẩn đoán thủ công dựa vào kinh nghiệm nông dân thường chậm trễ và sai sót.
- **Hạn chế công nghệ:**
    - Các mô hình Deep Learning chính xác (như VGG16, ResNet-50) thường có kích thước lớn (>100MB) và tốn tài nguyên, không thể chạy trên điện thoại tầm trung.
    - Các giải pháp dùng Cloud API (chụp ảnh -> gửi lên server -> trả kết quả) phụ thuộc vào Internet, vốn chập chờn ở vùng nông thôn, và có độ trễ cao.
- **Câu hỏi nghiên cứu (Research Question):** Làm thế nào để nén một mô hình Deep Learning phức tạp xuống kích thước siêu nhỏ (< 10MB) để chạy trực tiếp trên CPU di động (On-device inference) mà vẫn giữ được độ chính xác > 90% đối với các triệu chứng bệnh giai đoạn sớm (kích thước vết bệnh nhỏ)?

2. Mục tiêu nghiên cứu (Research Objectives)

1. **Xây dựng mô hình CNN hạng nhẹ (Lightweight Architecture):** Ứng dụng và cải tiến các kiến trúc Mobile-friendly như **MobileNetV3** hoặc **EfficientNet-Lite** để làm nền tảng (Backbone).
2. **Tối ưu hóa & Nén mô hình (Model Compression):** Áp dụng kỹ thuật **Transfer Learning** kết hợp với **Post-training Quantization** (chuyển đổi trọng số từ 32-bit float xuống 8-bit integer) để giảm kích thước và tăng tốc độ xử lý gấp 3-4 lần.
3. **Tăng cường dữ liệu thực tế (In-the-wild Robustness):** Giải quyết vấn đề nhiễu nền (background noise), thay đổi ánh sáng và bóng đổ lá cây để mô hình hoạt động tốt ngoài đồng ruộng.

3. Tổng quan tài liệu & Khoảng trống nghiên cứu (Literature Review)

3.1. Nhận diện bệnh cây truyền thống:

- Các phương pháp trước đây sử dụng SVM, Random Forest dựa trên đặc trưng màu sắc/kết cấu thủ công (Hand-crafted features). Độ chính xác thấp khi gặp điều kiện ánh sáng thay đổi [1].

3.2. Deep Learning trên thiết bị di động (Mobile Deep Learning):

- **MobileNet (Google):** Sử dụng *Depthwise Separable Convolutions* để giảm số lượng tham số. MobileNetV3 (2019) tích hợp thêm cơ chế *Attention (SE Block)* là chuẩn mực hiện nay.
- **EfficientNet (2019) & GhostNet (2020):** Các kiến trúc mới sử dụng "Ghost module" để tạo ra nhiều feature map từ ít phép tính hơn. Đây là các ứng viên sáng giá thay thế MobileNet [2].
- **Knowledge Distillation (Chưng cất tri thức):** Dạy một mạng nhỏ (Student) học theo mạng lớn (Teacher) để đạt hiệu năng cao.

3.3. Khoảng trống nghiên cứu (The Gap):

- **Dữ liệu phòng Lab vs. Thực tế:** 90% các bài báo hiện nay train trên bộ dữ liệu **PlantVillage** (lá cây đặt trên nền đen hoặc trắng). Khi mang ra ruộng (nền đất, cỏ, trời), mô hình thất bại hoàn toàn.
- **Vấn đề "Bệnh sớm" (Early Stage):** Hầu hết các nghiên cứu tập trung vào bệnh đã rõ ràng (toàn bộ lá bị vàng). Rất ít nghiên cứu tập trung vào phát hiện các đốm bệnh nhỏ li ti - giai đoạn vàng để chữa trị. Đây là điểm đề tài cần xoáy sâu để tạo tính mới.

4. Tài liệu tham khảo minh chứng (References)

Chọn lọc các bài báo uy tín (2022-2024) về AgriTech và Edge AI:

1. **Về Lightweight Models trong Nông nghiệp:**
   - *Paper:* "A Lightweight CNN Model for Detecting Plant Diseases in Mobile Environments" (Computers and Electronics in Agriculture, 2023). *Tạp chí Q1 uy tín nhất ngành AgriTech.*
   - *Paper:* "Plant Disease Detection using EfficientNet and Transfer Learning on Edge Devices" (IEEE Access, 2024).
2. **Về kỹ thuật nén và lượng tử hóa:**
   - *Paper:* "Quantization-Aware Training for Mobile Plant Disease Recognition" (Frontiers in Plant Science, 2023).
3. **Về dữ liệu thực tế (In-the-wild):**
   - *Dataset Paper:* "PlantDoc: A Dataset for Visual Plant Disease Detection in Laboratory and Field Conditions" (Dùng để chứng minh sự cần thiết của dữ liệu thực tế).

5. Tài nguyên bước đầu (Resources Part 1)

A. Datasets (Dữ liệu):

1. **PlantVillage:** (Kinh điển) Chứa 54,000 ảnh của 14 loại cây trồng và 26 loại bệnh. Dùng để *Pre-train* mô hình.
2. **PlantDoc:** (Quan trọng) Chứa ảnh chụp ngoài đồng ruộng với nhiễu nền. Dùng để *Fine-tune* và kiểm thử độ bền vững (Robustness).
3. **AI Challenger (Agricultural Disease):** Bộ dữ liệu khổng lồ từ Trung Quốc, chất lượng ảnh rất cao.

B. Sách nền tảng:

- *Deep Learning for Computer Vision with Python (Vol 3: Practitioner Bundle)* - Adrian Rosebrock. (Chương về MobileNets và training trên Android/iOS).
- *Tinyml: Machine Learning with Tensorflow Lite on Arduino and Ultra-Low-Power Microcontrollers* - Pete Warden (O'Reilly). (Cuốn sách gối đầu giường về tối ưu hóa model nhỏ).

5. Phương pháp nghiên cứu & Kiến trúc đề xuất (Methodology)

Tôi đề xuất quy trình **"Quantization-Aware Transfer Learning Framework"** (Khung làm việc học chuyển giao nhận thức lượng tử hóa). Quy trình gồm 3 giai đoạn:

Giai đoạn 1: Tăng cường dữ liệu thích ứng (Adaptive Data Augmentation)

Dữ liệu PlantVillage (nền đen/trắng) quá sạch so với thực tế. Nếu train trực tiếp, model sẽ bị "Overfitting vào nền" (tức là model học cái nền đen chứ không học vết bệnh).

- **Giải pháp:** Sử dụng kỹ thuật **On-the-fly Background Replacement**.
    - Tách lá cây ra khỏi nền cũ (dùng ngưỡng màu HSV hoặc thuật toán GrabCut).
    - Ghép lá cây vào các ảnh nền ngẫu nhiên (đất, cỏ, bầu trời, tay người cầm).
    - Áp dụng **CutMix** hoặc **Mosaic Augmentation** (kỹ thuật từ YOLOv4) để model học cách nhận diện khi lá bị che khuất một phần.

Giai đoạn 2: Kiến trúc mạng Lightweight (Lightweight Architecture Design)

Thay vì tự thiết kế mạng từ đầu (rất khó hội tụ), ta sử dụng **MobileNetV3-Small** làm backbone.

- **Cơ chế Attention (SE-Block):** MobileNetV3 tích hợp sẵn Squeeze-and-Excitation. Nó giúp mạng tự động gán trọng số cao cho các kênh (channels) chứa thông tin vết bệnh (màu vàng/nâu) và giảm trọng số của vùng lá xanh khỏe mạnh.
- **Thay đổi lớp Classifier:**
    - Bỏ lớp Fully Connected (FC) cuối cùng (1000 lớp của ImageNet).

- Thêm vào: GlobalAveragePooling -> Dropout (0.2) -> Dense (N_classes, activation='softmax').

Giai đoạn 3: Tối ưu hóa & Lượng tử hóa (Optimization & Quantization - Key Contribution)

Đây là bước biến model từ 20MB xuống dưới 5MB để chạy offline.

- **Quantization-Aware Training (QAT):**
  - Thay vì train xong mới nén (Post-training Quantization) - thường làm giảm độ chính xác đột ngột, ta giả lập việc lượng tử hóa *ngay trong quá trình training*.
  - Mạng sẽ học cách thích nghi với sai số khi trọng số bị làm tròn từ số thực 32-bit (float32) xuống số nguyên 8-bit (int8).
  - **Công thức lượng tử hóa:**

$$Q(x, scale, zero_point) = round\left(\frac{x}{scale} + zero_point\right)$$

  - Kết quả: Model int8 chạy nhanh hơn 3-4 lần trên CPU ARM của điện thoại nhờ tính toán số nguyên.

6. Kế hoạch thực nghiệm (Implementation Plan)

6.1. Thiết lập môi trường (Setup)

- **Framework:** TensorFlow 2.x (Keras) & TensorFlow Lite.
- **Hardware:** Train trên Google Colab (GPU T4). Test trên điện thoại Android tầm trung (ví dụ: Samsung A50, Xiaomi Redmi Note) để chứng minh tính thực tiễn.

6.2. Kịch bản đánh giá (Evaluation Strategy)

Sinh viên cần so sánh 3 phiên bản model để làm nổi bật đóng góp:

1. **Baseline:** VGG16 (Model nặng, độ chính xác cao).
2. **Lite Model:** MobileNetV3 (Model nhẹ, chưa nén).
3. **Proposed Model:** MobileNetV3 + QAT (Model nhẹ, đã nén 8-bit).

Các chỉ số đo lường (Metrics):

- **Accuracy & F1-Score:** Trên tập dữ liệu Test (PlantDoc - ảnh thực tế).
- **Model Size (MB):** Kích thước file .tflite. Mục tiêu: < 5MB.
- **Inference Time (ms):** Thời gian xử lý 1 ảnh trên điện thoại. Mục tiêu: < 50ms (Real-time cảm giác).
- **Battery Consumption (Joule):** Đo mức tiêu hao pin khi chạy liên tục 1000 ảnh (Optional - Điểm cộng lớn).

6.3. Ứng dụng di động (Mobile App Development)

Không cần viết app quá phức tạp. Sử dụng **Flutter** hoặc **Android Native** tích hợp **TFLite Interpreter**.

- **Luồng xử lý:** Camera Preview -> Lấy frame -> Resize (224x224) -> Normalize -> TFLite Model -> Kết quả (Tên bệnh + Độ tin cậy) -> Hiển thị lên màn hình.
- **Tính năng Offline:** Nhấn mạnh rằng app không cần internet sau khi cài đặt.

7. Tài nguyên kỹ thuật & Mã nguồn (Resources)

A. GitHub Repositories (Code mẫu):

1. **MobileNet-v3 Keras:** github.com/Xiaochus/MobileNetV3
   - Code cài đặt kiến trúc mạng MobileNetV3 sạch và dễ hiểu.
2. **TensorFlow Lite Plant Disease:** github.com/tensorflow/examples/tree/master/lite/examples/image_classification
   - Ví dụ chính chủ của Google về cách đưa model phân loại ảnh lên Android/iOS.
3. **Augmentation for Agriculture:** github.com/albumentations-team/albumentations
   - Thư viện tăng cường dữ liệu tốt nhất hiện nay.

B. Công cụ chuyển đổi:

- **Netron:** netron.app (Công cụ visualize file .tflite để xem cấu trúc model sau khi nén, rất cần thiết để chụp ảnh đưa vào báo cáo).

Lời khuyên:

1. **Xử lý "Negative Class" (Lá không phải mục tiêu):**
   - Khi nông dân chụp nhầm vào cục đất hay cái quần, model không được phán "Bệnh đốm lá".
   - *Giải pháp:* Thêm một lớp "Unknown" hoặc "Background" vào tập train. Hoặc thiết lập ngưỡng (Threshold): Nếu xác suất cao nhất < 0.7 thì báo "Không nhận diện được".
2. **Giải thích kết quả (Heatmap):**
   - Sử dụng **Grad-CAM** để hiển thị vùng nóng (vùng model nhìn vào).
   - *Giá trị:* Chứng minh model thực sự nhìn vào "vết bệnh" chứ không phải nhìn vào "cái tay nông dân" hay "nền đất". Đây là bằng chứng thép về độ tin cậy.