

Detection and Classification of Apple Diseases using ML Techniques: A Leakage-Free Deep-Learning Framework with Honest Cross-Dataset Evaluation

Marko A. Malak ^{1,*}, Mostafa Thabet ¹, Alber S. Aziz ², Ayman A. Alhelbawy ¹

¹*Faculty of Computers and Artificial Intelligence, Fayoum University, Fayoum, Egypt*

²*Faculty of Computer Science and Information System, October 6th University, Giza, Egypt*

Abstract Many published apple-leaf disease systems report near-perfect accuracy on the PlantVillage collection although augmented copies of the same physical leaf are spread across the training and test partitions, so that the model is tested on images. This study rebuilds the four-class apple subset as a de-duplicated, leakage-free benchmark and measures what the leak actually changes. Retraining the same ResNet50 on the original, leaky split shows that the leak is invisible in the laboratory metric (99.98% versus 99.66% accuracy, and 100% even on test images whose leaf was never seen in training) but not in the field, where the leaky model is less accurate on the external FGVC8 dataset (39.0% versus 43.9% over three seeds). Five ImageNet-pretrained backbones fine-tuned under one protocol all reach 99.35–99.80% in-domain accuracy over five seeds, a statistical tie, so cost rather than accuracy decides deployment. External zero-shot accuracy on two real-field datasets is low and varies between training runs (PlantDoc $59.5 \pm 5.7\%$, FGVC8 $43.9 \pm 0.5\%$); most errors are healthy orchard leaves predicted as diseased, and geometric augmentation is the only augmentation family that measurably narrows the gap, whereas GAN-based oversampling of the minority class widens it. Unsupervised domain-adversarial training raises FGVC8 accuracy from 37.2% to 54.4% without field labels, but fine-tuning on only 68 labelled field images reaches 76.7%. A data-efficiency study with three backbones and two fine-tuning budgets shows that leakage-free laboratory pre-training makes adaptation faster (+0.21 macro-F1 with 80 field images under a short 15-epoch budget) but does not raise the accuracy reached once fine-tuning converges. Grad-CAM places 86% of its activation inside the leaf, with the class ordering confirmed by an independent GrabCut segmentation, and a YOLO11 detector localizes diseased leaves in the field at 93.5% mAP@0.5, ahead of YOLOv8 in a matched three-seed comparison (92.9% versus 88.1%).

Keywords Convolutional neural networks, Near-duplicate contamination, Data efficiency, Domain adaptation, Grad-CAM localization

AMS 2010 subject classifications 68T07, 68T45

DOI: 10.19139/soic-2310-5070-4492

1. Introduction

Apple is one of the most widely grown temperate fruit crops, and its yield and market value are strongly affected by foliar fungal diseases such as apple scab (*Venturia inaequalis*), black rot (*Botryosphaeria obtusa*, anamorph *Diplodia seriata*), and cedar apple rust (*Gymnosporangium juniperi-virginianae*) [7]. Left unchecked, these diseases reduce fruit quality, lower yield, and push growers toward heavier pesticide use, which raises cost and carries an environmental burden [1]. In most orchards, diagnosis still depends on visual inspection by trained staff, which is slow, subjective, and hard to scale [8].

*Correspondence to: Marko A. Malak (Email: ma6281@fayoum.edu.eg). Faculty of Computers and Artificial Intelligence, Fayoum University, Fayoum, Egypt.

Computer vision offers a practical alternative. Early work relied on hand-crafted color and texture features fed to classical classifiers such as support vector machines or k -nearest neighbours [9, 10], but these pipelines were brittle under changes in illumination and background [8]. Convolutional neural networks (CNNs) removed the need for manual feature design and now dominate plant-disease recognition [2, 11], with transfer learning from ImageNet [12, 13] allowing accurate models to be trained on relatively small agricultural datasets [1, 14]. This paradigm has been applied successfully to a range of crops and backbones [15, 16], and to apple diseases in particular [7]. More recently, vision transformers have begun to complement CNN backbones for plant-disease recognition, including lightweight mobile variants [34], and end-to-end detection transformers such as RT-DETR have started to challenge YOLO-family detectors on real-time object detection [35]. This work deliberately retains CNN backbones and a YOLO detector, which remain the pragmatic choice for the small, imbalanced field datasets used here; the leakage-free evaluation protocol proposed below is architecture-agnostic and applies equally to transformer-based pipelines, which is a natural direction for future work. Rigorous, honestly evaluated classification pipelines of this kind are also an active theme in this journal, spanning hybrid deep-learning/machine-learning fusion for ECG image interpretation [40], orthogonal-moment descriptors for image classification and object recognition [41], and multimodal fusion for skin-lesion diagnosis [42]; the present study extends this theme to plant pathology, with a specific emphasis on leakage-free evaluation.

A large share of the published results in this area, however, are obtained on the PlantVillage collection [2, 3], a laboratory dataset in which each leaf is photographed against a uniform background. Two problems recur. First, many studies expand the data by writing rotated, flipped, and color-shifted copies of each image to disk *before* splitting into train, validation, and test sets [17]. Because several copies of the same physical leaf then end up in different partitions, the test set is contaminated with near-duplicates of the training images, and the reported accuracy measures memorization rather than generalization; PlantVillage is also known to contain acquisition shortcuts that let a classifier score far above chance from background pixels alone [43], near-duplicate leakage of exactly this kind has been shown to inflate test accuracy substantially in medical imaging [44], and leakage has been identified as a systemic threat to reproducibility across machine-learning-based science generally [33]. Second, in-domain accuracy on such a clean dataset is rarely challenged against real-field imagery, so claims of “deployment readiness” are seldom substantiated [5].

This is not a minor technicality. Because leakage inflates every downstream comparison built on top of it, an unrecognized leak undermines the evidentiary basis on which the field selects backbones, tunes augmentation policies, and sets deployment thresholds; a ranking of models obtained on a leaky split need not survive contact with a clean one. This is why the present study treats leakage removal not as a preliminary data-cleaning step but as the central methodological contribution around which every subsequent experiment, including the leaky-versus-leakage-free comparison of Section 3.4 and the data-efficiency analysis of Section 3.7, is organized.

This work re-examines the four-class apple problem with those two issues placed at the center of the design. The dataset is rebuilt so that exactly one image represents each leaf, guaranteeing disjoint splits, and five modern CNN backbones [18, 19, 20, 21, 22] are evaluated with a single, consistent protocol that reports accuracy together with precision, recall, F1, ROC-AUC, and the practical costs of parameters, latency, and FLOPs. Pixel-level segmentation is deliberately avoided because the lesion masks distributed with the source release do not reliably overlap the visible lesions; disease regions are instead localized with Grad-CAM [4], which needs no mask supervision. Finally, the gap between laboratory and field conditions is quantified using the PlantDoc dataset [5], keeping a clear separation between an external zero-shot test and a cross-validation in which field images are added to training. The headline contribution is a controlled measurement of what leakage changes, obtained by training the same network with the same code on the leaky and on the leakage-free split; the remaining contributions build the reproducible pipeline around it and quantify how the lab–field gap can be closed. In detail, the main contributions are:

- (headline) a controlled leaky-versus-leakage-free experiment with identical training code, showing that leakage leaves the laboratory accuracy unchanged (it is invisible there, even on test leaves never seen in training) but lowers accuracy on an independent real-field dataset;
- a de-duplicated, leakage-free four-class apple benchmark with fixed, reproducible splits;

- a fair multi-backbone comparison under one training and evaluation protocol, including efficiency metrics, with all classification results reported as means over three seeds;
- an honest augmentation ablation on the clean split, isolating the true contribution of augmentation;
- mask-free lesion localization with Grad-CAM and a simple quantitative localization score, reported per disease class and validated against an independent segmentation;
- a cross-dataset study on two independent real-field datasets (PlantDoc and FGVC8) that separates the external zero-shot result from the cross-validated result and reports both openly, together with an analysis of where and why zero-shot predictions fail;
- a data-efficiency study on the field data, for three backbones and two fine-tuning budgets, showing that leakage-free laboratory pre-training makes adaptation to the field faster but does not raise the accuracy reached once fine-tuning converges;
- controlled baselines that position these results: unsupervised domain adaptation (CORAL and DANN), GAN-based oversampling of the minority class with an explicit near-duplicate filter, an augmentation ablation by transform family, a feature-space check of the frog-eye-to-black-rot mapping, a matched YOLOv8-versus-YOLOv11 comparison, and a five-seed repetition of the in-domain sweep; and
- an explicit object-detection component, in which a YOLOv11 detector trained on the PlantDoc apple bounding boxes localizes and classifies diseased leaves directly in field images.

We flag explicitly how this list should be read, since referees of an earlier version of this manuscript correctly observed that no new network architecture, loss function, or training algorithm is proposed here. The novelty claimed is methodological and empirical rather than architectural: the backbones, the detector, and the training recipe are all standard. What is new is the discipline of building the benchmark leakage-free *before* any model is trained, and then measuring, rather than asserting, what that discipline changes. In the course of this revision we also re-examined the data-efficiency result of the original submission, which reported that a leakage-free laboratory backbone needs less field data than an ImageNet backbone. Repeating the study for three backbones and two fine-tuning budgets showed that this advantage depends on how long the network is fine-tuned: it is large under the short, fixed schedule used originally and disappears when fine-tuning is run to convergence. We therefore report both settings side by side (Section 3.7) and state the finding in its narrower, correct form: leakage-free pre-training makes adaptation to field data faster and cheaper, not ultimately more accurate.

2. Materials and Methods

2.1. Dataset and the data-leakage problem

The raw images are drawn from the public PlantVillage repository [2, 3], restricted to the four apple categories: Apple Scab, Black Rot, Cedar Apple Rust, and Healthy. In a common but flawed preparation, the training set is enlarged by saving many augmented versions of each leaf to disk and only afterwards partitioning the files [17]. Rotations and flips of a single leaf are visually near-identical, so this ordering leaks the test set into training. All on-disk augmented copies are therefore discarded and the collection is reduced to one image per physical leaf before any split is made. Augmentation is reintroduced later, but only at training time and only on the training partition (Section 2.3), which keeps the evaluation honest.

2.2. Leakage-free splits

The de-duplicated apple set is divided into training, validation, and test partitions at roughly 70/15/15, stratified by class. Because de-duplication guarantees that no leaf appears twice, the partitions are disjoint by construction; this was verified explicitly, and zero overlapping image paths were found between any pair of splits. Table 1 lists the resulting distribution. The data are naturally imbalanced, with healthy leaves forming the majority and cedar apple rust the minority, so no rebalancing by duplication is applied (which would reintroduce leakage); imbalance is instead handled with class-weighted training and macro-averaged reporting [11]. The inverse-frequency class

weights are 0.48 (healthy), 1.29 (apple scab), 1.31 (black rot), and 2.63 (cedar apple rust). Figure 1 shows the class distribution across splits.

A deliberately leaky split for comparison. To measure what leakage actually changes, the original, non-duplicated folder (9,819 files, including the on-disk rotated, flipped and colour-shifted copies, which trace back to 3,277 physical leaves) is also split at random at the *image* level into 70/15/15% partitions, reproducing the flawed practice described above. A physical-leaf identifier is recovered for every file by stripping the augmentation suffixes from its file name, so each test image can be labelled as *contaminated* (another copy of the same leaf is in the training split) or *clean*. ResNet50 is then trained with identical code on this leaky split and on the leakage-free split, three seeds each, and both models are evaluated in-domain and, zero-shot, on the two field datasets (Section 3.4).

Table 1. Leakage-free apple-leaf dataset distribution (one image per leaf).

Class	Train	Validation	Test	Total
Healthy	1,180	253	254	1,687
Apple Scab	441	94	95	630
Black Rot	434	93	94	621
Cedar Apple Rust	216	46	47	309
Total (PlantVillage)	2,271	486	490	3,247
PlantDoc (real-field, 3 classes; for cross-validation)				287
FGVC8 (real-field, 4 classes; for cross-validation)				2,800

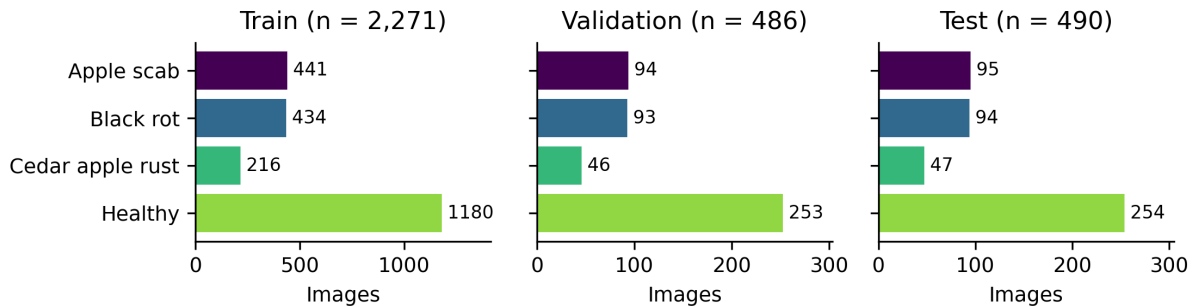


Figure 1. Class distribution of the leakage-free apple set across the training, validation, and test partitions.

2.3. Preprocessing and augmentation

All images are resized to the input size expected by each backbone (299×299 for InceptionV3, 224×224 for the others) and normalized with the ImageNet mean and standard deviation [13]. Validation and test images receive resize and normalize only. During training, the classification branch applies a domain-oriented augmentation policy: random resized cropping (scale 0.6–1.0), horizontal and vertical flips, rotation up to $\pm 35^\circ$, brightness/contrast and hue/saturation jitter, small RGB shifts, mild Gaussian blur, coarse dropout, and small shift–scale–rotate perturbations [17]. These transforms emulate the illumination, orientation, and occlusion variability of real orchards without ever being written to disk, so the before/after augmentation comparison in Section 3.3 is meaningful.

2.4. Classification models and training

Five ImageNet-pretrained CNN backbones are fine-tuned: MobileNetV2 [21], DenseNet121 [20], ResNet50 [19], VGG16 [18], and InceptionV3 [22]. Each backbone is followed by global average pooling (flattening for VGG16) and a small classification head, summarized in Table 2. The final layer has four outputs and is trained with class-weighted cross-entropy.

Table 2. Classification heads and input sizes used for each backbone.

Model	Head	Input
MobileNetV2	GAP $\rightarrow$ 128 $\rightarrow$ Dropout 0.2 $\rightarrow$ 4	224
DenseNet121	GAP $\rightarrow$ 256 $\rightarrow$ Dropout 0.2 $\rightarrow$ 4	224
ResNet50	GAP $\rightarrow$ 4	224
VGG16	Flatten $\rightarrow$ 512 $\rightarrow$ Dropout 0.5 $\rightarrow$ 4	224
InceptionV3	GAP $\rightarrow$ 256 $\rightarrow$ Dropout 0.3 $\rightarrow$ 4	299

Training follows a two-phase transfer-learning schedule [12, 14]. In phase one the backbone is frozen and only the head is trained for five epochs at a learning rate of 10^{-3} , which lets the new head adapt without disturbing the pretrained features. In phase two the whole network is unfrozen and fine-tuned for up to 25 epochs at 10^{-4} , with early stopping (patience 8) on validation accuracy and restoration of the best checkpoint. The optimizer is AdamW [23] with weight decay 10^{-4} ; the batch size is 16; mixed-precision arithmetic is used on the GPU. To characterize run-to-run variability rather than report a single lucky run, the full training and evaluation of all five backbones is repeated over three random seeds, and the in-domain metrics in Table 3 are reported as mean $\pm$ standard deviation over those seeds. Because three seeds is fewer than the five or more often recommended for comparative claims, the full in-domain sweep was repeated for this revision with five seeds (Section 3.1, Table 4), and the comparative data-efficiency claims are tested with a paired non-parametric test across the (fraction, seed) grid rather than by comparing means alone. Experiments were run in PyTorch 2.6 [24] on an NVIDIA RTX 5000 Ada GPU. The additional experiments of this revision were run with PyTorch 2.x on an NVIDIA T4 GPU (Google Colab) and on a laptop NVIDIA GeForce 930MX, from a single released notebook that is resumable after every epoch. In that notebook the augmentation policy of Section 2.3 is re-implemented with torchvision transforms in three families: geometric (random resized crop with scale 0.6–1.0, horizontal and vertical flips, rotation up to $\pm 35^\circ$), photometric (brightness, contrast and saturation jitter of 0.3 and hue jitter of 0.05) and occlusion (random erasing of 2–10% of the image with probability 0.2); all models retrained for the revision use this implementation.

2.5. Weakly-supervised lesion localization

Pixel-level segmentation was considered but ultimately rejected: the binary lesion masks bundled with the source release are unreliable, often failing to overlap the visible lesion, and a standard color-based lesion detector agrees with them at only about 0.17 IoU. We tested this claim under conditions deliberately favorable to the source masks: from the source release we curated a 20-images-per-class “best-case” subset by ranking masks on a blob-concentration/solidity heuristic and keeping only the least fragmented, highest-quality examples per diseased class, so that the source annotations were given their best possible chance. Even on this favorable subset, the same color-based detector agrees with the curated source masks at only $\text{IoU} = 0.168 \pm 0.175$ and $\text{Dice} = 0.252 \pm 0.232$ (mean $\pm$ SD). We report this as a comparison between an independent automatic method and the dataset’s own provided masks, not as a comparison against expert-drawn ground truth, since no independently, manually re-annotated lesion masks are available to us; producing such an independent gold-standard annotation for a validation subset is a concrete, feasible piece of future work that a subsequent revision of the pipeline could add. The low and highly variable agreement even in this favorable setting is what leads us to treat the source masks as unreliable for training or evaluation rather than as noisy-but-usable labels. Training a segmentation model against such labels would optimize toward noise. Instead, disease regions are localized without any mask supervision using gradient-weighted class activation mapping (Grad-CAM) [4] on the best classifier. Grad-CAM highlights the image regions

most responsible for the predicted class. To turn this into a quantitative check, a leaf mask is computed by simple saturation/value thresholding with morphological cleanup, and the fraction of Grad-CAM activation energy that falls inside the leaf is reported as

$$\text{CAM-in-leaf} = \frac{\sum_{x,y} C(x,y)L(x,y)}{\sum_{x,y} C(x,y)}, \quad (1)$$

where $C(x,y)$ is the Grad-CAM heat and $L(x,y)$ the binary leaf mask. A high value indicates that the classifier attends to leaf tissue rather than background; it does not, by itself, establish that attention is centered on the lesion. Because the score is only as good as the leaf mask, it is validated in this revision with a second, independent mask obtained by GrabCut [45] initialised from a centred rectangle. The score is reported per disease class and for both masks over all 490 test images, and every test image with an HSV score below 0.60 is inspected: a low score is attributed to a mask failure when the GrabCut score of the same image is at least 0.80, and to off-leaf attention when it is also below 0.60 (Section 3.10).

2.6. Cross-dataset evaluation protocol

To gauge field readiness, two independent real-field datasets are used. PlantDoc [5] shares three apple classes with the present set (it contains no black rot). Plant Pathology 2021 / FGVC8 [6] is a large field dataset of about 23,000 leaves; after keeping only single-label images of scab, cedar apple rust, frog-eye leaf spot, and healthy, mapping frog-eye leaf spot to black rot (frog-eye leaf spot is the foliar symptom phase of apple black rot: both are caused by the same fungus, *Botryosphaeria obtusa*, which produces frog-eye lesions on leaves and the black-rot rot on fruit), removing duplicates, and capping at 700 images per class, it yields a four-class field set of 2,800 images that, unlike PlantDoc, includes black rot. This equivalence is documented in extension plant-pathology literature [36]: the fruit-rot phase begins as a light-brown spot that develops concentric brown-and-black bands, while the leaf phase forms round-to-irregular spots with a light brown-to-gray center, one or more dark-brown concentric rings, and a purple margin, the two being different tissue-specific expressions of one infection rather than visually unrelated conditions. In this revision we also examine the mapping directly, both visually and in the feature space of the trained classifier (Section 3.6). Because every classification metric for this class is numerically identical whichever name is used (Section 4 revisits this point), the mapping is a labeling decision rather than a modeling choice, and a reader who treats frog-eye leaf spot and black rot as distinct can mentally relabel this single class without affecting any other result in the paper.

For each dataset two clearly distinct numbers are reported and must not be conflated. The external cross-dataset (“zero-shot”) evaluation applies the PlantVillage-only model directly to all field images; no field image is ever used for training, so this is the only genuine measure of out-of-distribution generalization. We retain the term “zero-shot” throughout for continuity with the rest of the paper, but we use it in the narrow sense that no field *image* is seen during training, not in the stricter sense from the zero-shot-learning literature in which the *classes* themselves are unseen; the class labels here are shared between the lab and field datasets. Where precision matters we therefore qualify it as external zero-shot / cross-dataset generalization evaluation, and we thank the referees for pointing out that the bare term risks being read as the stricter setting. The five-fold cross-validation splits the field dataset into five stratified folds; each fold is held out for testing in turn while the other four are added to the PlantVillage training data. Because field images are used in training in this second protocol, its score is an in-sample adaptation result, not a zero-shot generalization figure; it is reported as the mean $\pm$ standard deviation over folds. Crucially, the field datasets are used without any on-disk augmentation, so fold membership is defined at the level of the original photographs and the near-duplicate leakage that motivates this study cannot arise within the field cross-validation itself. We note one honest caveat about these folds: they are constructed with a standard stratified split over the image table, not grouped by leaf or photograph source, because neither PlantDoc nor FGVC8 publishes a leaf-identity or photographer field that would support such grouping. Since these field images are used exactly once each, unaugmented, and are not the on-disk near-duplicates that motivate this study, we do not believe this construction reintroduces the leakage problem the paper is built around; but if any single physical leaf happens to have been photographed more than once by the original dataset curators, our protocol would not detect or separate

those photographs, and we flag this as a limitation of the field cross-validation rather than of the leakage-free lab benchmark itself.

A related, and in our view under-appreciated, source of the lab-to-field accuracy drop is that “real-field” datasets are themselves heterogeneous in ways that go beyond background clutter. Manual inspection of a sample of PlantDoc images (which, like FGVC8, is assembled by scraping publicly available images rather than by a single controlled acquisition protocol) turned up genuine hand-held orchard photographs alongside studio-style photographs of detached leaves on a plain card and cropped extension-service reference images with a printed caption overlay, all under the same class label (Figure 2); a small number of images we found in the apple-scab class depict scab lesions on fruit rather than on foliage. None of this is a criticism of the original dataset curators, whose goal was breadth of coverage rather than acquisition uniformity, but it does mean that some of the external zero-shot gap reflects a mismatch in image *provenance* and occasional label noise, not solely a mismatch in illumination or background between a laboratory leaf and an orchard leaf. We report this as a qualitative observation to contextualize the zero-shot numbers rather than as a quantified error decomposition, since attributing individual misclassifications to a specific cause would require a labeled taxonomy of image provenance that neither field dataset currently provides; building and releasing such a taxonomy is future work that would also let this heterogeneity be measured rather than only illustrated.

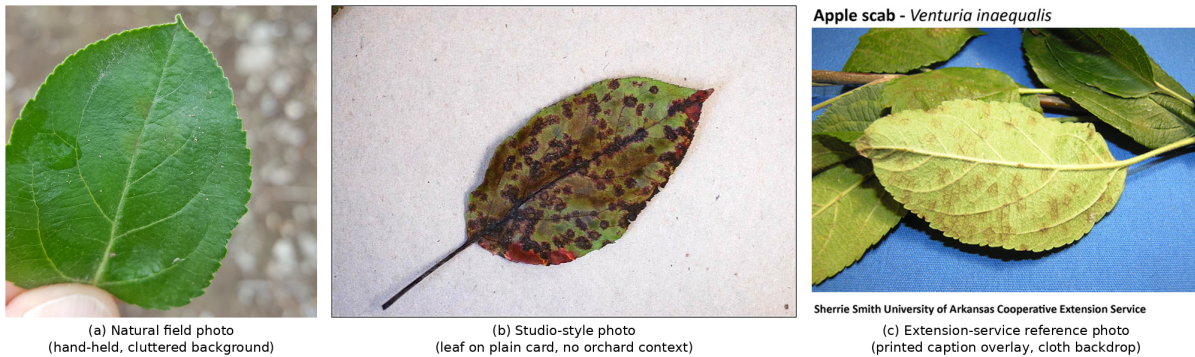


Figure 2. Three PlantDoc apple-scab images illustrating image-provenance heterogeneity within a single class label: (a) a natural, hand-held field photograph with a cluttered background; (b) a studio-style photograph of a detached leaf on a plain card, with no orchard context; (c) a cropped extension-service reference photograph with a printed caption overlay and cloth backdrop (image credit: Sherrie Smith, University of Arkansas Cooperative Extension Service, reproduced under the source’s educational-use terms). These are representative examples selected during manual inspection, not a systematic or exhaustive audit, and are not tied to specific model predictions.

2.7. Object detection of diseased leaves

Object detection for plant disease has progressed from two-stage detectors through the single-stage YOLO family to, most recently, end-to-end transformer detectors. Prior apple-specific work includes an improved single-shot detector for real-time field detection [27]; more broadly, successive YOLO releases have been applied to a range of crops, and detection transformers such as RT-DETR now report accuracy competitive with YOLO at comparable latency on general benchmarks [35]. This background motivates, but does not by itself decide, the detector choice made below.

Whole-image classification answers which disease is present but not where the affected leaf lies in a cluttered scene. To add an explicit detection capability, a one-stage YOLOv11 detector [25] is trained on the object-detection release of PlantDoc [5], which provides bounding boxes for three apple classes: apple scab, apple rust, and healthy apple leaf (the release contains no apple black rot boxes). The Pascal-VOC annotations are converted to YOLO format and restricted to these three classes, and the official PlantDoc split is preserved, giving 242 training images (557 boxes) and 29 test images (34 boxes). The small YOLOv11s variant (about 9 M parameters) is initialized from COCO-pretrained weights and fine-tuned for 100 epochs at an input resolution of 640×640 with a batch

size of 16, a cosine learning-rate schedule, and the built-in mosaic and HSV augmentations; the model size is deliberately kept small to match the limited number of field boxes and avoid over-fitting. Detection quality is reported with the standard mean average precision at an IoU threshold of 0.5 (mAP@0.5) and averaged over IoU 0.5–0.95 (mAP@0.5:0.95), together with precision and recall, and test-time augmentation is applied at inference.

YOLO11 was originally selected as the current Ultralytics release. To test whether the architecture itself matters on this small data set, YOLOv8s [37] and YOLO11s were additionally trained under an identical, shorter schedule (30 epochs, 640 pixels, COCO initialization, the same augmentations and the same official split) with three seeds each, and evaluated with and without test-time augmentation (Section 3.11).

The absence of an apple black-rot box class is a substantive limitation for deployment, not merely a dataset gap to note in passing: a detector that cannot localize black rot would miss one of the three diseases this paper otherwise classifies, in exactly the scenario, an automated field scouting tool, where localization matters most. Two mitigations seem practical. First, the trained classifier together with Grad-CAM (Section 2.5) already produces an approximate location for black rot within a whole-image crop, which could seed weak (pseudo-label) bounding boxes for a black-rot detector without new manual annotation, at the cost of the same weak-localization caveats already discussed for Grad-CAM. Second, transfer learning from the fruit-rot phase of black rot, which is visually distinctive and better represented in some public fruit-disease collections, combined with a small number of manually boxed black-rot leaves, could bootstrap a fourth detection class more cheaply than annotating from scratch. We have not implemented either mitigation in this revision, since both require new annotation or pseudo-labeling work beyond the scope of a text and analysis revision, and we present them as concrete next steps rather than as completed experiments.

2.8. Data-efficiency protocol

The cross-dataset results establish that a large lab–field gap exists; a more useful question for practitioners is how that gap can be closed, and how much field data and training closing it requires. This is quantified on the four-class FGVC8 field set, balanced at 500 images per class. A fixed, stratified 20% of the images (400) is held out as a field test set and never used for training. A classifier is then trained on increasing stratified fractions of the remaining field images, 5, 10, 25, 50, and 100% (80 to 1,600 images), from three initializations: generic ImageNet weights, the leakage-free PlantVillage model of the same architecture, and, for ResNet50, the model trained on the leaky split (Section 2.2). To avoid any label-order mismatch between the lab and field class spaces, only the backbone is transferred and the classifier head is reinitialized in all cases, so the comparison isolates the value of the pre-trained features. Each (initialization, fraction) cell is repeated over three random seeds and reported as the mean $\pm$ standard deviation of the macro-F1 on the held-out field test set.

Because the outcome turned out to depend on how long the network is fine-tuned, every configuration is run under two fine-tuning budgets. The *short budget* trains all layers for a fixed 15 epochs with Adam (learning rate 10^{-4} , cosine decay, batch size 32, unweighted cross-entropy) on the whole subset; this is the schedule behind the data-efficiency results of the original submission, and at the 5% fraction it amounts to only about 30 optimizer steps. The *full budget* uses the in-domain training protocol of Section 2.4: five frozen-backbone epochs followed by up to 25 fine-tuning epochs with early stopping on a 15% validation split carved from the subset, so that each run trains to convergence (the training set is correspondingly 15% smaller). The study is run for ResNet50, MobileNetV2 and DenseNet121, to check that the conclusions are not specific to one backbone, and repeated for ResNet50 on the three-class PlantDoc set (fixed 20% test split of 58 images, fractions of 10–100%) under the short budget.

2.9. Additional baselines and analyses

Unsupervised domain adaptation. To position fine-tuning against methods that need no field labels, ResNet50 (ImageNet initialization) is trained on the labelled PlantVillage training split together with the 1,600 FGVC8 pool images used *without* labels, with three objectives: source-only (classification loss only; the zero-shot reference with this code), correlation alignment (Deep CORAL [38], matching the covariance of the pooled features of lab and field batches), and domain-adversarial training (DANN [39], a domain discriminator behind a gradient-reversal layer with the standard $2/(1 + e^{-10p}) - 1$ schedule). Each is trained for 20 epochs with AdamW (learning rate 10^{-4}) and three seeds. Because no labelled field validation set exists in this setting, the final epoch is evaluated, on

the same 400-image FGVC8 test split used for the data-efficiency study, so that the results compare directly with fine-tuning on 68–1,360 labelled field images.

GAN-based oversampling. A DCGAN [46] (latent size 100, 128×128 output) is trained on the cedar-apple-rust training images only, and 300 images are generated. A synthetic image is kept only if it is not a near-duplicate of any real training or test image (cosine similarity below 0.98 in ResNet50 feature space) and the leakage-free classifier assigns it to cedar apple rust with probability at least 0.90, so that off-class samples do not add label noise. ResNet50 is then retrained with class weighting alone and with class weighting plus the kept synthetic images, three seeds each, and tested in-domain and zero-shot on both field sets.

Augmentation ablation by family. ResNet50 is trained on the leakage-free split with no augmentation, with each augmentation family alone (geometric, photometric, occlusion), and with the full policy. Every arm uses the same schedule, class weights and early stopping, so the no-augmentation baseline is tuned exactly as carefully as the others, and each model is evaluated in-domain and zero-shot on both field sets.

Feature-space check of the frogeye mapping. Global-average-pooled features of the leakage-free ResNet50 are extracted for the PlantVillage test images and for FGVC8 images. For every FGVC8 class, the mean cosine similarity to each PlantVillage class centroid, the nearest-centroid class, and the majority class among the ten nearest PlantVillage neighbours are computed, and the embeddings are visualised with t-SNE [47]. The same analysis with ImageNet-only features serves as a reference.

Uncertainty and error analysis. Zero-shot accuracy and macro-F1 are accompanied by 95% percentile bootstrap confidence intervals over 2,000 resamples of the test images [48], and zero-shot errors are related to simple image properties (mean brightness, sharpness as the variance of the Laplacian, and the fraction of vegetation pixels) with Mann–Whitney tests.

2.10. Evaluation metrics

Classification is assessed with accuracy, macro-averaged precision, recall, and F1, and one-vs-rest ROC-AUC; per-class F1 and confusion matrices are also reported because the data are imbalanced. Deployment cost is captured by the number of parameters, single-image inference latency, and FLOPs. The latency figures in Table 3 were measured for the original submission on a single NVIDIA RTX 5000 (Ada) GPU at batch size 1 with mixed precision: after three discarded warm-up passes, 20 timed forward passes on a fixed input tensor already resident on the GPU are averaged, with `torch.cuda.synchronize()` called before each timestamp, so they exclude image loading and preprocessing. For this revision latency was re-measured for all five models on one low-end laptop GPU (NVIDIA GeForce 930MX, 2 GB; Intel Core i-series CPU with two threads; PyTorch 2.1, CUDA 11.8; batch size 1; mixed precision; the same warm-up and timing protocol), both for the forward pass alone and including JPEG decoding and resizing (Table 4). Localization quality is measured by the CAM-in-leaf score of Equation (1), and detection quality by mean average precision as defined above. All classification metrics are computed with standard scikit-learn implementations [26].

3. Results

3.1. In-domain classification

Table 3 reports the test-set performance of all five backbones on the leakage-free split, together with their size, latency, and compute. To avoid drawing conclusions from a single lucky run, every accuracy figure is the mean over three random seeds, reported with its standard deviation. Every model exceeds 99% accuracy, and all reach a one-vs-rest ROC-AUC of 1.000, confirming that the four apple classes are well separated once evaluation is honest. ResNet50 [19] and VGG16 [18] are marginally the strongest at 99.86% accuracy, followed by DenseNet121 [20] and InceptionV3 [22] at 99.73% and MobileNetV2 [21] at 99.12%. The entire spread is under three quarters of a point and is of the same order as the per-model standard deviations (0.10–0.35 points), so the top of the ranking is a statistical tie rather than a meaningful ordering: the near-perfect scores should be read as evidence that the task is easy on clean, leakage-free data, not as separation between architectures. The one-vs-rest ROC-AUC of 1.000 (with zero variance across seeds) should be read in the same light, as a consequence of the classes being

well separated on this small clean set. Importantly, the lightweight MobileNetV2 attains 99.12% accuracy with only 2.39 M parameters and 0.60 GFLOPs, roughly a tenth of ResNet50’s parameters, making it the most attractive option for on-device or edge deployment. Figure 3 visualizes the comparison. ResNet50, one of the two most accurate models, is carried forward as the primary model for the per-class, localization, and detection analyses.

Table 3. Test-set classification performance on the leakage-free split, with efficiency metrics. Best accuracy in bold.

Model	Acc. (%)	Macro-F1	ROC-AUC	Params (M)	Latency (ms)	FLOPs (G)
ResNet50	99.86 ± 0.19	0.9974 ± 0.0037	1.000	23.52	4.01	8.17
VGG16	99.86 ± 0.10	0.9988 ± 0.0009	1.000	27.56	0.99	30.72
DenseNet121	99.73 ± 0.10	0.9961 ± 0.0016	1.000	7.22	9.49	5.67
InceptionV3	99.73 ± 0.25	0.9948 ± 0.0049	1.000	22.31	7.73	11.42
MobileNetV2	99.12 ± 0.35	0.9857 ± 0.0057	1.000	2.39	3.49	0.60

Accuracy and macro-F1 are mean ± standard deviation over three random seeds; ROC-AUC is 1.000 (±0.000) for all. Parameter, latency, and FLOP counts are for the deployed models.

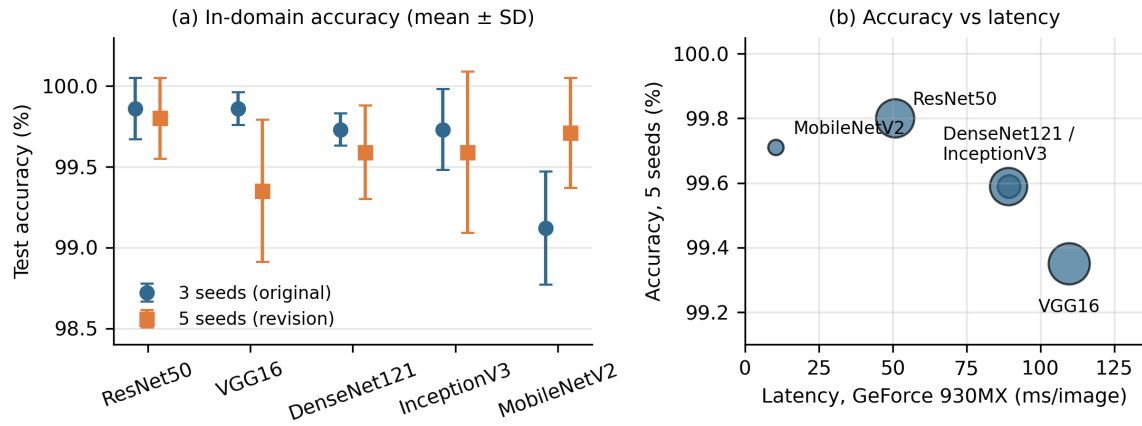


Figure 3. (a) In-domain test accuracy of the five backbones on the leakage-free split, mean ± SD over three seeds (original submission) and five seeds (this revision); the between-model differences are within the run-to-run spread and the ordering changes between the two sweeps. (b) Five-seed accuracy against single-image latency on a laptop GPU; marker area is proportional to the number of parameters. ROC-AUC is 1.000 for all models and is not shown.

Five seeds. Repeating the sweep with five seeds (Table 4) confirms this reading. All five backbones fall between 99.35% and 99.80% accuracy, the standard deviations (0.25–0.50 points) overlap for every pair, and the order among the models changes relative to the three-seed run (MobileNetV2 is now second at 99.71% and VGG16 last at 99.35%). The in-domain ranking is therefore not stable and should not be used to select a backbone. What does separate the models is cost: on the laptop GPU MobileNetV2 needs 10.4 ms per image, against 50.8 ms for ResNet50 and 109.7 ms for VGG16, while being as accurate in-domain and, as Sections 3.5 and 3.7 show, at least as good in the field and in adaptation.

3.2. Per-class performance

Table 5 gives per-class F1 for every model. Cedar apple rust, despite being the smallest class, is recognized perfectly by all five backbones, which suggests its visual signature is distinctive. The only occasional confusions involve black rot, whose dark necrotic lesions can resemble scab; even so, the weakest per-class F1 across all models is 0.979. The confusion matrix and ROC curves of the best model, ResNet50, are shown in Figure 4 for a

Table 4. Five-seed repetition of the in-domain sweep on the leakage-free split (mean $\pm$ SD over seeds 0–4) and latency re-measured on a laptop GPU (NVIDIA GeForce 930MX, batch size 1, mean $\pm$ SD over 20 timed passes after 3 warm-up passes).

Model	Acc. (%)	Macro-F1	Params (M)	Latency, forward only (ms)	Latency, incl. load + resize (ms)
ResNet50	99.80 $\pm$ 0.25	0.9969 $\pm$ 0.0040	23.52	50.8 $\pm$ 15.2	53.2 $\pm$ 10.4
VGG16	99.35 $\pm$ 0.44	0.9898 $\pm$ 0.0071	27.56	109.7 $\pm$ 0.3	113.2 $\pm$ 0.7
DenseNet121	99.59 $\pm$ 0.29	0.9938 $\pm$ 0.0048	7.22	89.4 $\pm$ 3.3	98.6 $\pm$ 21.1
InceptionV3	99.59 $\pm$ 0.50	0.9929 $\pm$ 0.0090	22.31	89.2 $\pm$ 0.2	93.7 $\pm$ 0.9
MobileNetV2	99.71 $\pm$ 0.34	0.9966 $\pm$ 0.0039	2.39	10.4 $\pm$ 0.3	14.9 $\pm$ 1.4

representative run: almost every one of the 490 test images falls on the diagonal, with at most one or two errors depending on the seed.

Table 5. Per-class F1 score on the leakage-free test set.

Model	Apple Scab	Black Rot	Cedar Apple Rust	Healthy
ResNet50	1.000	1.000	1.000	1.000
InceptionV3	1.000	1.000	1.000	1.000
MobileNetV2	1.000	0.995	1.000	0.998
VGG16	1.000	0.995	1.000	0.998
DenseNet121	0.995	1.000	0.989	0.996

3.3. Honest effect of augmentation

A frequent claim in the literature is that augmentation yields several accuracy points [17]. On a leaky split this is an artifact: augmentation multiplies the near-duplicates that already bridge train and test [44]. To measure the real contribution, the best model was trained twice on the clean split, once with augmentation disabled and once enabled, and test accuracy was compared. Table 6 shows that, in a single controlled run, augmentation improves accuracy by only about 0.20 points (99.80% to 100.0%). Augmentation is still worthwhile, but its true effect on a clean, in-domain split is modest; its larger value appears under distribution shift, as the next section shows.

Table 6. Honest augmentation ablation for the best model (ResNet50) on the leakage-free test set.

Setting	Test accuracy
Without augmentation	99.80%
With augmentation	100.00%
Honest gain	+0.20 points

Which augmentations matter. The ablation above answers whether augmentation helps in-domain; it does not say which transforms help in the field. Table 7 trains ResNet50 with each augmentation family separately, under the same schedule and early stopping as the full policy. In-domain, all five arms lie between 99.39% and 100% and cannot be distinguished. In the field they differ. Without augmentation, FGVC8 zero-shot accuracy is 36.5%; geometric transforms alone raise it to 43.5%, as much as the full policy (43.6%), whereas photometric (41.9%) and occlusion (41.0%) transforms help less. On PlantDoc geometric augmentation again gives the highest accuracy (64.8%) and photometric augmentation alone the lowest (51.9%), but these PlantDoc differences are within the run-to-run spread of ± 5.7 points measured for the full policy, and each arm was run with a single seed, so only

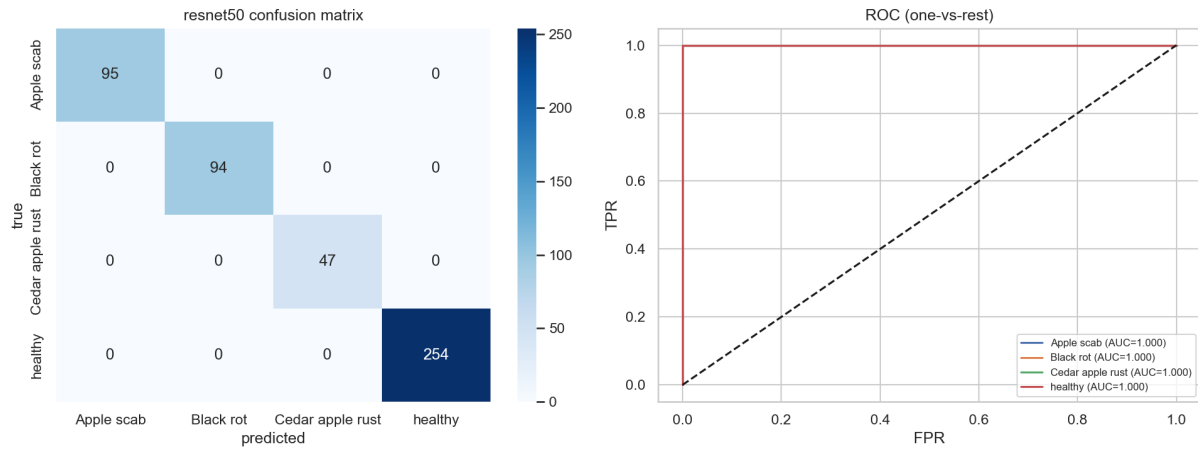


Figure 4. Confusion matrix (left) and one-vs-rest ROC curves (right) for the best model, ResNet50, on the leakage-free test set.

the FGVC8 ordering, where the three-seed spread is 0.5 points, should be read as evidence. Geometric variation of scale, crop and orientation is thus the part of the policy that transfers; the photometric jitter used here does not emulate orchard illumination well enough to close the gap on its own.

Table 7. Augmentation ablation by transform family for ResNet50 on the leakage-free split (one seed per arm): in-domain test accuracy and external zero-shot accuracy on both field datasets.

Augmentation policy	PlantVillage acc. (%)	PlantDoc zero-shot (%)	FGVC8 zero-shot (%)
None	100.00	56.79	36.50
Geometric only	99.59	64.81	43.54
Photometric only	99.80	51.92	41.89
Occlusion only	99.80	55.75	40.96
Full policy	99.39	62.02	43.61

3.4. What leakage changes: leaky versus leakage-free training

Table 8 compares ResNet50 trained with identical code on the leaky, image-level split and on the leakage-free split (Section 2.2). On the leaky split, 84.9% of the test images have a rotated, flipped, or colour-shifted copy of the same physical leaf in the training set, and the reported test accuracy is 99.98%. The more telling number is the accuracy on the remaining test images, whose leaf was never seen during training: it is also 100%. On this saturated, single-background task the leak therefore does not inflate the laboratory metric at all, which is exactly why it cannot be detected from the headline accuracy and why a correct split has to be enforced by construction rather than inferred from the results.

The cost of the leak appears outside the laboratory. Evaluated zero-shot on the external FGVC8 set, the leaky model is less accurate than the leakage-free one ($39.0 \pm 1.9\%$ versus $43.9 \pm 0.5\%$; macro-F1 0.36 versus 0.42; Welch t -test $p = 0.04$ over three seeds per arm). On PlantDoc the leaky model is also lower on average (56.8% versus 59.5%), but there the difference is well within the large run-to-run spread of the leakage-free models and is not significant ($p = 0.50$). Training on several near-identical copies of each leaf enlarges the training set threefold in files but not in information, and appears to fit the laboratory appearance more tightly without adding anything that transfers. We note that the leaky training set is larger than the leakage-free one in number of files; this is not a

confound to be removed but precisely the practice under evaluation, in which on-disk copies are created to enlarge the data before splitting.

Table 8. Leaky versus leakage-free training of ResNet50 with identical code (mean $\pm$ SD over three seeds). “Contaminated” test images have another copy of the same physical leaf in the training split; “clean” test images do not. Zero-shot (ZS) columns use no field image in training; FGVC8 is the fixed 2,800-image evaluation sample (700 per class) and PlantDoc all 287 images.

Training split	Reported test acc. (%)	Contaminated test images (%)	Acc. on contaminated (%)	Acc. on clean (%)	PlantDoc ZS acc. (%)	FGVC8 ZS acc. (%)	FGVC8 ZS macro-F1
Leaky (image-level)	99.98 $\pm$ 0.04	84.9 $\pm$ 0.9	99.97 $\pm$ 0.05	100.0 $\pm$ 0.0	56.8 $\pm$ 1.6	39.0 $\pm$ 1.9	0.362 $\pm$ 0.035
Leakage-free	99.66 $\pm$ 0.24	0	–	99.66 $\pm$ 0.24	59.5 $\pm$ 5.7	43.9 $\pm$ 0.5	0.421 $\pm$ 0.010

3.5. External zero-shot evaluation and field cross-validation on two datasets

In-domain accuracy on PlantVillage says little about orchard performance, because PlantVillage leaves are photographed against a uniform background [5]. Two clearly separated measurements are reported for each dataset in Table 9, and they answer different questions. The *external zero-shot evaluation* applies the PlantVillage-only model to the field images without ever training on any of them, and is the only genuine test of generalization; here accuracy collapses to 59.5 $\pm$ 5.7% on PlantDoc and to 43.9 $\pm$ 0.5% on the harder four-class FGVC8 (three ResNet50 models retrained on the leakage-free split), a stark reminder that laboratory accuracy does not transfer directly to the field. These zero-shot numbers are also unstable from one training run to the next: the released ResNet50 checkpoint reaches 62.4% on PlantDoc but only 33.4% on FGVC8, and the run behind the original submission gave 41.5% and 32.9%. Table 11 shows the same spread across the five released backbones (PlantDoc 38.0–62.4%, FGVC8 29.6–38.4%). In-domain accuracy is identical to within a fraction of a point for all of these models, so the laboratory benchmark cannot predict which of them will transfer best; the lightweight MobileNetV2 in fact gives the highest FGVC8 zero-shot accuracy. The *five-fold cross-validation*, in which field images are mixed into training, recovers to 81.9% $\pm$ 6.6% on PlantDoc and to 94.1% $\pm$ 1.3% on FGVC8; the recovery is far stronger and tighter on FGVC8 because it is much larger and supplies more field examples per fold. Because field images are seen during training here, these figures reflect in-sample adaptation and are deliberately *not* presented as generalization results. On the four-class FGVC8 set (Table 10, Figure 6) all classes including black rot are recovered well; apple scab remains the hardest field class on both datasets. The essential message is that field performance is governed by the availability of field training data, not by the choice of architecture.

Where zero-shot predictions fail. The zero-shot errors are not random (Figure 7). On FGVC8 the leakage-free ResNet50 assigns 70% of all field images to apple scab and labels only 4 of the 700 healthy field leaves as healthy: to a model that has only seen single, clean leaves on a grey background, almost every orchard photograph, with its texture, shadows, and several overlapping leaves, looks diseased. On PlantDoc, whose images are closer to single-leaf close-ups, the diagonal is much stronger; the errors are spread between predictions of apple scab and of black rot, a class that PlantDoc does not contain, so a model restricted to the three PlantDoc classes would already score higher. We also tested whether the errors track simple image properties. On FGVC8, images with a larger fraction of vegetation pixels (closer, leaf-filling shots) are classified correctly more often (43.5% in the top tercile versus 27.3% in the bottom; Mann–Whitney $p < 10^{-12}$), and sharper images less often, consistent with fine background texture being mistaken for lesions; on PlantDoc the association with vegetation fraction runs the other way and brighter images are recognized more often. No single property explains the gap across both datasets, which supports the conclusion that the failure is a broad shift in image provenance and composition (Figure 2) rather than one nuisance factor such as illumination or blur.

Figure 5 shows the sixteen most confident FGVC8 errors. Almost all are healthy leaves photographed in the orchard, sharp, well lit and filling the frame, which the model labels as apple scab with probability close to 1; the two black-rot (frog-eye) errors carry only a few small lesions, and the single cedar-rust error a single lesion. The failure is therefore not caused by blur, darkness or occlusion, but by a change in what the image looks like as a whole: fine vein texture, specular highlights and a green, cluttered background are cues that, in PlantVillage, only

occur together with scab lesions. For the released ResNet50, zero-shot accuracy is 62.4% [95% bootstrap CI 56.4, 67.9] on PlantDoc (macro-F1 0.676 [0.617, 0.725]) and 33.4% [31.6, 35.1] on FGVC8 (macro-F1 0.271 [0.255, 0.286]); the PlantDoc interval is wide because only 287 images are available. Restricting the prediction to the three classes that PlantDoc contains raises PlantDoc accuracy to 54–73% across the five backbones.

fgvc8: 16 most confident zero-shot errors (ResNet50, leakage-free)



Figure 5. The sixteen most confident zero-shot errors of the leakage-free ResNet50 on FGVC8, with true label, predicted label and predicted probability. Most are healthy, sharp, well-lit orchard leaves predicted as apple scab.

3.6. Is frogeye leaf spot a valid stand-in for black rot?

Figure 8 places PlantVillage black-rot leaves next to FGVC8 frogeye leaves. The PlantVillage “black rot” images show the foliar frogeye symptom, small round brown spots with a darker margin, not the fruit rot, so the two classes depict the same leaf lesion; what differs is the rest of the image, a detached leaf on grey card versus an attached leaf in the orchard, and many large lesions in PlantVillage versus few, often early-stage lesions in FGVC8. The feature-space check (Figure 9) shows how this plays out in the classifier. The four PlantVillage classes form four tight, well-separated clusters, whereas all FGVC8 images, frogeye and scab alike, fall into one separate cluster:

Table 9. Field evaluation on two independent real-field datasets. The zero-shot column uses no field image in training; the cross-validation column mixes field images into training and is therefore an adaptation, not a generalization, result. Zero-shot: mean $\pm$ SD of three ResNet50 models retrained on the leakage-free split, evaluated on all 287 PlantDoc images and on a fixed 2,800-image FGVC8 sample (700 per class); five-fold CV: ResNet50, as in the original submission.

Field dataset (classes)	External zero-shot	Five-fold CV
PlantDoc (3 classes)	59.47% $\pm$ 5.68%	81.90% $\pm$ 6.56%
FGVC8 (4 classes)	43.94% $\pm$ 0.46%	94.07% $\pm$ 1.26%

Table 10. Per-class performance under five-fold cross-validation on the four-class FGVC8 field dataset.

Class	Precision	Recall	F1	Support
Apple Scab	0.90	0.92	0.91	700
Black Rot	0.94	0.96	0.95	700
Cedar Apple Rust	0.99	0.95	0.97	700
Healthy	0.93	0.94	0.94	700
Weighted avg	0.94	0.94	0.94	2,800

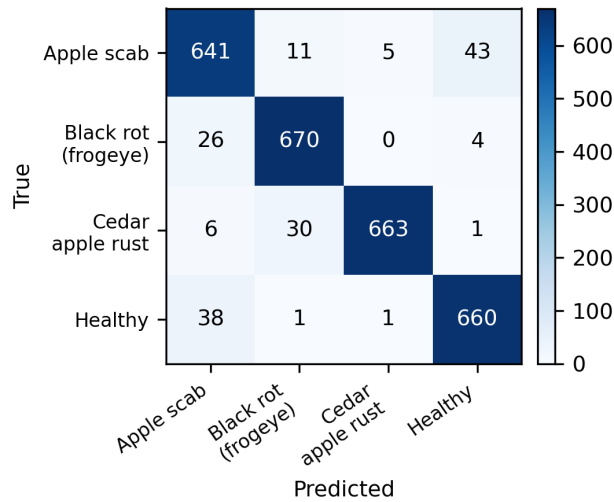


Figure 6. Aggregated confusion matrix over the five FGVC8 cross-validation folds (four classes; each image tested once).

Table 11. External zero-shot accuracy of the five released leakage-free checkpoints. In-domain test accuracy is within half a point for all five, yet field accuracy differs widely, so the laboratory benchmark does not predict transfer.

Backbone	PlantVillage test acc. (%)	PlantDoc zero-shot (%)	FGVC8 zero-shot (%)
ResNet50	99.80	62.37	33.36
VGG16	99.80	43.90	32.39
DenseNet121	99.59	47.04	29.57
InceptionV3	99.39	49.48	33.29
MobileNetV2	99.80	37.98	38.36

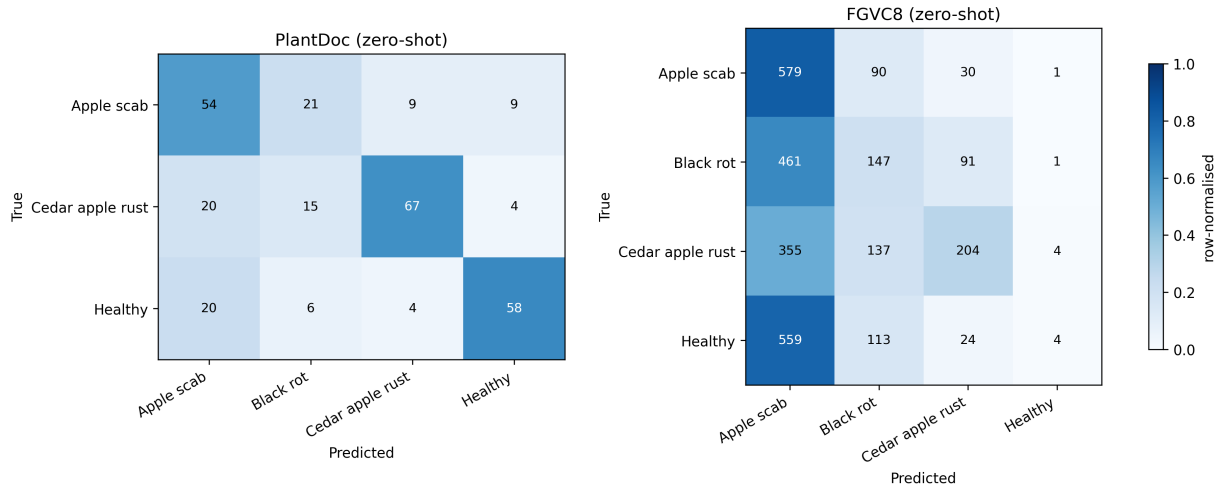


Figure 7. Zero-shot confusion matrices of the released leakage-free ResNet50 on PlantDoc (left, three classes present) and FGVC8 (right). Cell labels are image counts; colour is row-normalized. On FGVC8 most field images, including healthy leaves, are predicted as apple scab.

the dominant factor in feature space is the domain, not the disease. Within that field cluster the frog-eye images are closer to the PlantVillage black-rot centroid (mean cosine similarity 0.782) than FGVC8 scab or healthy images are (0.744 and 0.750), and closer to it than to the cedar-rust (0.699) or healthy (0.590) centroids, but they are closer still to the scab centroid (0.804), to which the model assigns most of the field set. For 38% of the frog-eye images black rot is the nearest centroid and for 42% it is the majority among the ten nearest PlantVillage neighbours, against 0% and 2% for FGVC8 healthy leaves; with ImageNet-only features the nearest-centroid figure is 4%, so the partial alignment is learned from the laboratory black-rot class. We conclude that the mapping is biologically and visually justified and is partly reflected in the learned features, but that its effect on the zero-shot numbers cannot be separated from the general domain shift, which affects every class. Once field images are seen in training, frog-eye leaves are recognised as a coherent class (recall 0.96 in cross-validation, Table 10), so the mapping does not introduce label noise that penalizes the adapted model. Readers who prefer to treat frog-eye leaf spot as a distinct disease can read the FGVC8 black-rot results as “frog-eye leaf spot” without any other result changing.

3.7. Data efficiency and the value of leakage-free pre-training

The cross-validation above shows that field accuracy can be recovered by adding field data, but not how much data and training are needed, nor whether the leakage-free lab model is a better starting point than generic ImageNet features. Table 13 and Figure 10 answer these questions for ResNet50, and Table 14 for two further backbones. The answer depends on the fine-tuning budget.

Short budget. With the fixed 15-epoch schedule, the leakage-free lab backbone is clearly the better starting point when field data is scarce: with only 80 field images (5%) it reaches a macro-F1 of 0.808 ± 0.021 against 0.599 ± 0.047 for ImageNet initialization, a gain of +0.21 that closely reproduces the +0.17 reported in the original submission with a different checkpoint and field sample. The gain shrinks as field data grows (+0.08 at 10%, +0.02 at 25%) and vanishes at 100%. Across all fractions and seeds the lab initialization is better in 13 of 15 paired runs (one-sided Wilcoxon signed-rank $p = 2 \times 10^{-4}$). The same pattern holds for MobileNetV2 (+0.045 at 5%, 13 of 15 paired runs, $p = 0.004$) and DenseNet121 (+0.066 at 5%, 12 of 15, $p = 0.005$).

Full budget. When each run is instead fine-tuned to convergence with the two-phase schedule and early stopping, the advantage disappears for all three backbones: at 5% ResNet50 reaches 0.762 ± 0.030 from the lab initialization and 0.751 ± 0.082 from ImageNet, and the lab initialization wins only 6 of 15 paired runs ($p = 0.66$; MobileNetV2 $p = 0.93$, DenseNet121 $p = 0.68$). The ImageNet-initialized network simply needs more optimization steps to

PlantVillage black rot



FGVC8 frogeye leaf spot



PlantVillage scab



FGVC8 scab



Figure 8. Random examples of PlantVillage black rot, FGVC8 frogeye leaf spot, and, for comparison, PlantVillage and FGVC8 apple scab. The PlantVillage black-rot leaves show the same round, brown, dark-margined leaf spots as the FGVC8 frogeye leaves; the main difference is the acquisition setting.

reach the same point; given them, it does. At the smallest budget the full protocol is also noisier (SD up to 0.08), because early stopping then relies on a validation split of only 12 images.

This refines, and partly corrects, the claim made in the original submission. Leakage-free laboratory pre-training does not reduce the amount of field data needed to reach a given accuracy once training is run to convergence; what it does is make adaptation faster and cheaper, so that a usable field model can be obtained from a few dozen labelled field images with a short, fixed fine-tuning run. That is a practical advantage when retraining must be quick, frequent, or performed on modest hardware, but it is a statement about training cost rather than about data requirements, and we now state it as such.

Leaky versus leakage-free initialization. Starting from the backbone trained on the leaky split (Section 3.4) tests the referees' question whether a clean lab model adapts better than a leaky one. Under the short budget it does at the smallest data budget: with 80 field images the leakage-free initialization reaches 0.808 against 0.665 ± 0.053 for the leaky one (+0.14, better in all three seeds). From 10% of the field data onward the two are equivalent or the leaky initialization is marginally higher, and over all fractions the difference is not significant (lab better in 4 of 15 paired runs); under the full budget the three initializations cannot be distinguished. The leak thus matters most exactly where laboratory pre-training is supposed to help, when field data and training are both scarce, while its effect washes out once enough field data is available.

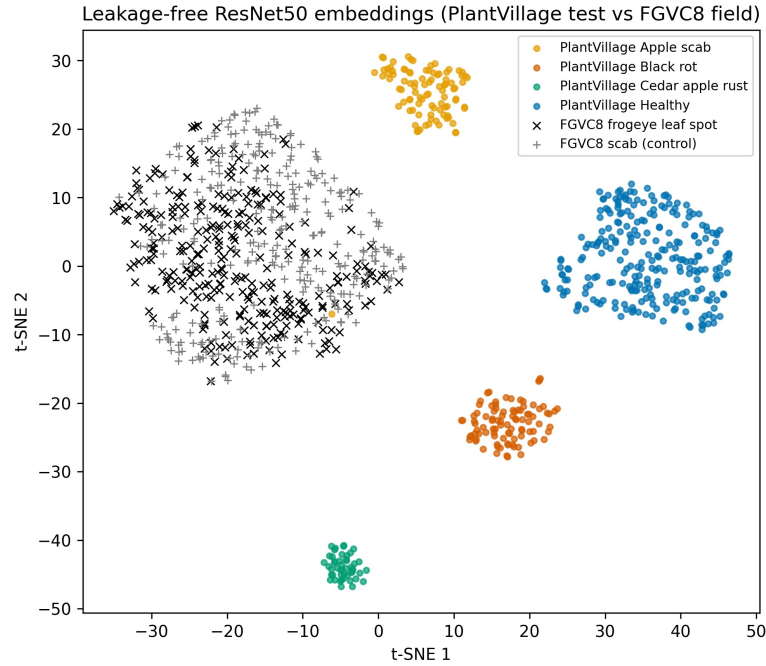


Figure 9. t-SNE of global-average-pooled features of the leakage-free ResNet50 for the four PlantVillage test classes, FGVC8 frog eye leaf spot, and FGVC8 scab as a control. The field images form one cluster separate from all laboratory classes; the domain dominates the disease.

PlantDoc. On the three-class PlantDoc set the short-budget study reproduces the FGVC8 pattern (Table 12). With 22 or 57 field images the leakage-free lab initialization is ahead by about $+0.19$ macro-F1 (0.697 versus 0.510 and 0.733 versus 0.539), the gain shrinks to $+0.06$ with 114 images, and with all 229 training images ImageNet initialization is better (0.886 versus 0.839); across all paired runs the lab initialization wins 8 of 11 (one-sided Wilcoxon $p = 0.007$). This table replaces Table 8 of the original submission. These results carry large uncertainty: the fixed test split contains only 58 images, so one image changes accuracy by 1.7 points, and the standard deviations reach 0.08. We therefore read them only as qualitative confirmation of the FGVC8 findings, and did not repeat the full-budget protocol on PlantDoc, where the 15% early-stopping split of the smallest fraction would contain three images.

Does the lighter model adapt as well? MobileNetV2 adapts to the field at least as efficiently as the heavier backbones (Table 14). Under the short budget and from ImageNet initialization it reaches 0.739 macro-F1 with 80 field images, against 0.599 for ResNet50 and 0.666 for DenseNet121, and under the full budget the three backbones are within 0.03 of each other at every fraction (0.75–0.79 at 5%, 0.94–0.97 at 100%). MobileNetV2 also had the highest FGVC8 zero-shot accuracy of the released checkpoints (Table 11). Since it is about five times faster than ResNet50 and ten times faster than VGG16 on the laptop GPU (Table 4), nothing in these results argues for the heavier models in a field deployment.

3.8. Unsupervised domain adaptation versus fine-tuning

Table 15 compares the fine-tuning route of Section 3.7 with two standard unsupervised domain-adaptation methods that use the FGVC8 pool images without labels. Correlation alignment does not help ($38.6 \pm 5.5\%$ versus $37.2 \pm 3.0\%$ for the source-only model trained with the same code). Domain-adversarial training does: DANN raises field accuracy to $54.4 \pm 2.7\%$ (macro-F1 0.525) while keeping in-domain accuracy at 99.5%, the largest improvement obtained in this study without any field label. It remains far below fine-tuning, however: with only 68 labelled field images the fine-tuned model reaches 76.7% from the leakage-free initialization and 75.7% from

Table 12. Data efficiency of ResNet50 on the three-class PlantDoc field set under the short budget: macro-F1 (mean $\pm$ SD over three seeds) on a fixed 58-image test split. This table replaces Table 8 of the original submission.

Field data (training images)	ImageNet	Lab (leak-free)	Gain
10% (22)	0.510 ± 0.077	0.697 ± 0.023	+0.187
25% (57)	0.539 ± 0.032	0.733 ± 0.039	+0.194
50% (114)	0.753 ± 0.049	0.812 ± 0.036	+0.058
100% (229)	0.886 ± 0.019	$0.839 \pm 0.012^\dagger$	-0.047

† Two seeds. Lab vs ImageNet over the 11 paired runs: 8 wins, one-sided Wilcoxon $p = 0.007$.

Table 13. Data efficiency of ResNet50 on the four-class FGVC8 field set under two fine-tuning budgets: macro-F1 (mean $\pm$ SD over three seeds) on the fixed 400-image field test split, for ImageNet, leakage-free lab, and leaky lab initialization. Short: fixed 15 epochs on the whole subset; full: two-phase training with early stopping, 85% of the subset used for training.

Field data	Short budget (15 epochs)			Full budget (to convergence)		
	ImageNet	Lab (leak-free)	Lab (leaky)	ImageNet	Lab (leak-free)	Lab (leaky)
5% (80 img)	0.599 ± 0.047	0.808 ± 0.021	0.665 ± 0.053	0.751 ± 0.082	0.762 ± 0.030	0.708 ± 0.103
10% (160)	0.810 ± 0.014	0.895 ± 0.018	0.893 ± 0.015	0.869 ± 0.008	0.868 ± 0.029	0.877 ± 0.030
25% (400)	0.912 ± 0.007	0.932 ± 0.007	0.943 ± 0.004	0.918 ± 0.030	0.916 ± 0.014	0.913 ± 0.035
50% (800)	0.935 ± 0.008	0.947 ± 0.018	0.963 ± 0.012	0.941 ± 0.013	0.941 ± 0.016	0.943 ± 0.012
100% (1,600)	0.959 ± 0.005	0.961 ± 0.001	0.964 ± 0.001	0.969 ± 0.005	0.949 ± 0.014	0.964 ± 0.010

Lab (leak-free) vs ImageNet, paired over fractions $\times$ seeds: short 13/15 wins, $p = 2 \times 10^{-4}$; full 6/15 wins, $p = 0.66$ (one-sided Wilcoxon).

Table 14. Data efficiency of two further backbones on FGVC8: macro-F1 (mean over three seeds) for ImageNet / leakage-free lab initialization, and the gain of the lab initialization, under both fine-tuning budgets.

Backbone	Budget	5%	10%	25%	50%	100%
MobileNetV2	short	$0.739 / 0.784 (+0.045)$	$0.873 / 0.891 (+0.017)$	$0.922 / 0.935 (+0.014)$	$0.941 / 0.945 (+0.003)$	$0.957 / 0.952 (-0.005)$
	full	$0.759 / 0.769 (+0.009)$	$0.841 / 0.810 (-0.032)$	$0.922 / 0.916 (-0.007)$	$0.935 / 0.928 (-0.007)$	$0.962 / 0.941 (-0.021)$
DenseNet121	short	$0.666 / 0.732 (+0.066)$	$0.848 / 0.875 (+0.027)$	$0.935 / 0.935 (+0.001)$	$0.950 / 0.955 (+0.004)$	$0.955 / 0.958 (+0.003)$
	full	$0.780 / 0.788 (+0.009)$	$0.856 / 0.827 (-0.029)$	$0.917 / 0.919 (+0.002)$	$0.938 / 0.946 (+0.008)$	$0.963 / 0.957 (-0.006)$

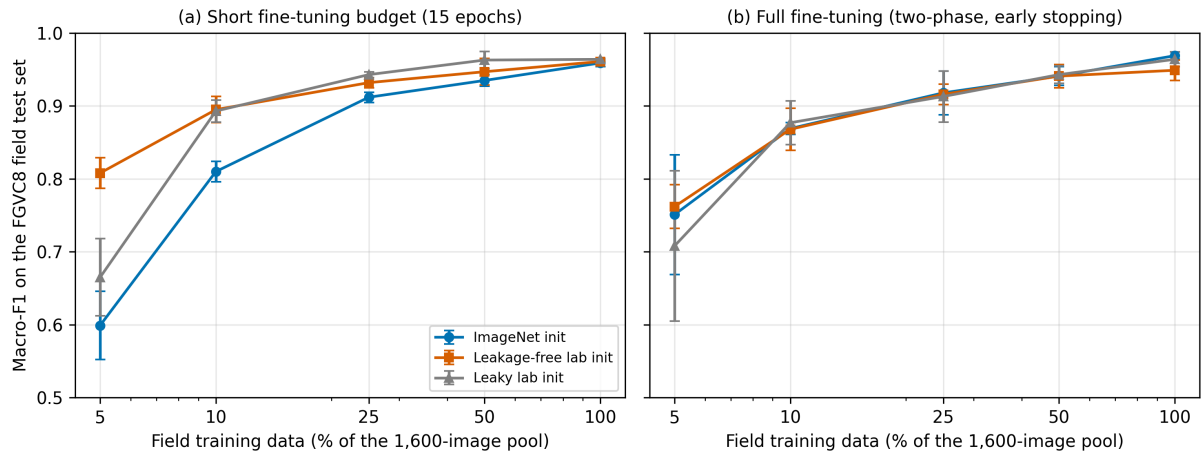


Figure 10. Data-efficiency curves of ResNet50 on the FGVC8 field set (mean $\pm$ SD over three seeds) for ImageNet, leakage-free lab, and leaky lab initialization. (a) With a short, fixed fine-tuning budget the leakage-free lab initialization is far ahead when field data is scarce. (b) With fine-tuning run to convergence the three initializations reach the same accuracy.

ImageNet, and with 1,360 labelled images 95–97%. For a practitioner the implication is direct: labelling a few dozen field images is worth more than the best unlabelled adaptation method tested here, and domain adaptation is the fallback when no field label can be obtained. Neither alignment method improves PlantDoc accuracy beyond the run-to-run spread (55–61%, SD 8–9 points), consistent with the adaptation being specific to the FGVC8 distribution it was aligned to.

Table 15. Unsupervised domain adaptation (no field labels) versus fine-tuning with labelled field images, ResNet50, mean $\pm$ SD over three seeds. All rows are evaluated on the same 400-image FGVC8 field test split. Fine-tuning rows use the full budget of Section 2.8.

Method	Labelled field images	FGVC8 acc. (%)	FGVC8 macro-F1	PlantVillage acc. (%)	PlantDoc acc. (%)
Source only	0	37.17 $\pm$ 2.98	0.337 $\pm$ 0.022	99.93 $\pm$ 0.12	55.40 $\pm$ 9.04
Deep CORAL	0 (1,600 unlabelled)	38.58 $\pm$ 5.48	0.346 $\pm$ 0.065	99.05 $\pm$ 0.62	60.74 $\pm$ 9.09
DANN	0 (1,600 unlabelled)	54.42 $\pm$ 2.74	0.525 $\pm$ 0.021	99.52 $\pm$ 0.51	59.70 $\pm$ 7.51
Fine-tune, ImageNet init.	68	75.67 $\pm$ 7.67	0.751 $\pm$ 0.082	–	–
Fine-tune, lab init.	68	76.67 $\pm$ 3.26	0.762 $\pm$ 0.030	–	–
Fine-tune, ImageNet init.	1,360	96.92 $\pm$ 0.52	0.969 $\pm$ 0.005	–	–
Fine-tune, lab init.	1,360	94.92 $\pm$ 1.42	0.949 $\pm$ 0.014	–	–

3.9. Generative oversampling of the minority class

The near-duplicate filter flagged none of the 300 generated cedar-rust images (the highest cosine similarity to any real image was 0.906, median 0.857, well below the 0.98 threshold), and the classifier assigned 299 of them to cedar apple rust with $p \geq 0.90$, so 299 synthetic images were added to the 216 real cedar-rust training images. Table 16 shows the outcome. In-domain, the synthetic images slightly improve accuracy (99.93% versus 99.66%) and make cedar-rust F1 perfect in all three seeds. In the field they are harmful: FGVC8 zero-shot accuracy drops from 43.9% to 39.7%, and cedar-rust recall falls from 0.33 to 0.16 on FGVC8 and from 0.55 to 0.43 on PlantDoc. The generator reproduces the laboratory appearance of cedar rust, uniform background included, so its samples reinforce the lab-specific cues that fail to transfer. GAN oversampling therefore looks beneficial on the in-domain benchmark while making the field model worse, a small-scale version of the leakage effect of Section 3.4, and supports keeping class weighting, which has no such side effect, as the imbalance strategy.

Table 16. Class weighting alone versus class weighting plus 299 filtered GAN-generated cedar-apple-rust images, ResNet50, mean $\pm$ SD over three seeds. ZS: zero-shot.

Training data	PV acc. (%)	PV cedar F1	FGVC8 ZS acc. (%)	FGVC8 cedar recall	PlantDoc ZS acc. (%)	PlantDoc cedar recall
Class weighting	99.66 $\pm$ 0.24	0.990 $\pm$ 0.010	43.94 $\pm$ 0.46	0.328 $\pm$ 0.115	59.47 $\pm$ 5.68	0.550 $\pm$ 0.162
+ 299 GAN images	99.93 $\pm$ 0.12	1.000 $\pm$ 0.000	39.65 $\pm$ 2.98	0.158 $\pm$ 0.050	58.77 $\pm$ 3.44	0.431 $\pm$ 0.070

3.10. Lesion localization with Grad-CAM

Without any lesion masks, Grad-CAM [4] on the best model, ResNet50, places 86.2% $\pm$ 15.1% of its activation energy inside the HSV leaf mask over all 490 test images (89.4% on the smaller sample reported in the original submission); Table 17 breaks this down by class and repeats it with the independent GrabCut mask. Qualitatively, the heatmaps tend to fall on leaf tissue, often near visible lesions for diseased leaves and on the leaf body for healthy ones (Figures 11 and 12). This provides interpretable, image-level localization that can draw a grower’s attention to affected regions, and it avoids the false precision of training against unreliable pixel masks; it should be read as suggestive, weak-localization evidence rather than a claim of pixel-accurate lesion detection.

The per-class breakdown explains the large pooled standard deviation. Apple scab (96.9%) and black rot (91.4%) score high and uniformly; cedar apple rust scores lowest (67.7% $\pm$ 14.7%) and healthy leaves are intermediate but variable (83.8% $\pm$ 15.6%). The GrabCut mask, which follows the leaf outline more tightly and excludes cast shadows (IoU 0.61 with the HSV mask), gives lower absolute values (69.4% pooled) but the same class ordering,

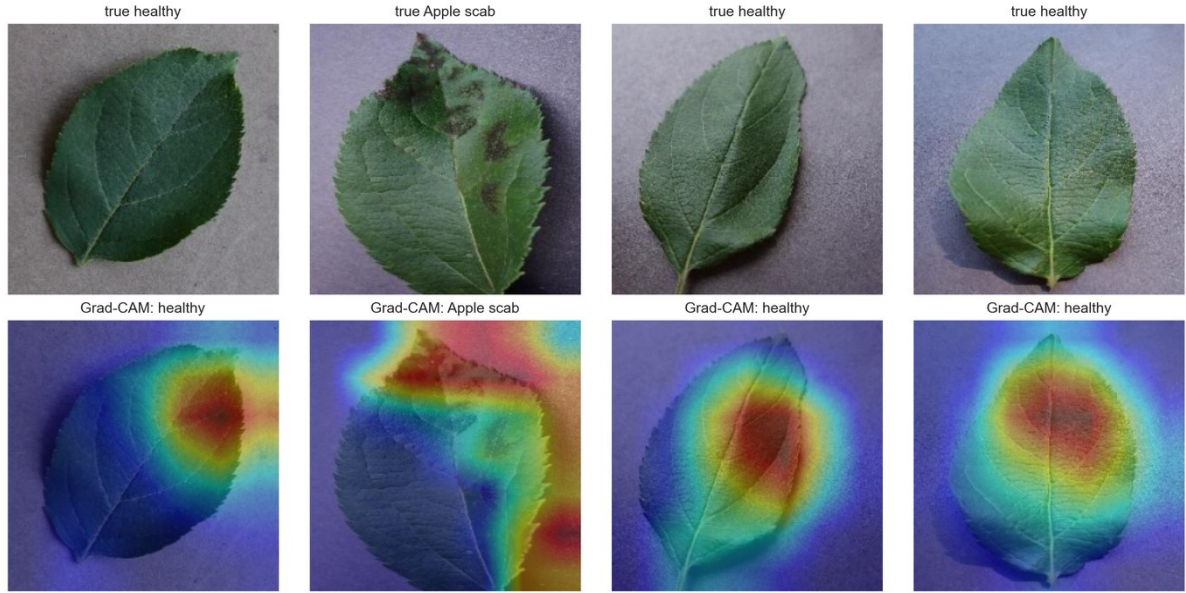


Figure 11. Grad-CAM localization on test leaves (top: input; bottom: overlay). Heat concentrates on leaf tissue, not the background.

with cedar apple rust again lowest (57.3%), so the ordering is not an artefact of the HSV threshold. Of the 28 test images with an HSV score below 0.60, all are cedar rust (12) or healthy (16) leaves; none is a mask failure in the sense defined in Section 2.5, and 23 are also low under GrabCut. Inspection of these images (Figure 13) shows two situations. Most are small or curled leaves occupying a minor part of the frame: the heatmap peak still lies on the leaf, and in cedar rust usually on the orange lesions, but the Grad-CAM map, computed on a 7×7 grid and upsampled, spreads much of its energy onto the surrounding background. For several dark or glossy healthy leaves the HSV mask misses part of the leaf, while the heat is centred on the leaf. Only in a minority of cases does the peak sit on the leaf edge and its shadow. Low scores therefore mostly reflect the coarse resolution of Grad-CAM relative to small leaves and, for healthy leaves, an imperfect HSV mask, rather than a model that attends to the background. The score is a useful check that attention falls on foliage, not a lesion-level localization accuracy; validating it against expert lesion annotations remains future work.

Table 17. CAM-in-leaf (%) per class on all 490 leakage-free test images (mean $\pm$ SD), with the HSV leaf mask used in the paper and an independent GrabCut mask. Low: HSV score < 0.60 ; mask failure: low under HSV but GrabCut score ≥ 0.80 ; off-leaf: low under both masks.

Class	n	CAM-in-leaf, HSV	CAM-in-leaf, GrabCut	IoU(HSV, GrabCut)	Low	Mask failure	Off-leaf
Apple scab	95	96.9 ± 4.7	74.2 ± 6.4	0.55 ± 0.12	0	0	0
Black rot	94	91.4 ± 8.4	69.7 ± 6.5	0.58 ± 0.13	0	0	0
Cedar apple rust	47	67.7 ± 14.7	57.3 ± 8.3	0.76 ± 0.14	12	0	12
Healthy	254	83.8 ± 15.6	69.7 ± 7.5	0.62 ± 0.21	16	0	11
All (pooled)	490	86.2 ± 15.1	69.4 ± 8.4	0.61 ± 0.18	28	0	23

3.11. Object detection with YOLO

Beyond image-level recognition, the YOLOv11s detector localizes and classifies diseased apple leaves directly in field images, using the official PlantDoc detection split, whose train and test partitions are disjoint by construction, so the detection evaluation is free of the leakage that motivates the rest of the paper. On the held-out PlantDoc apple

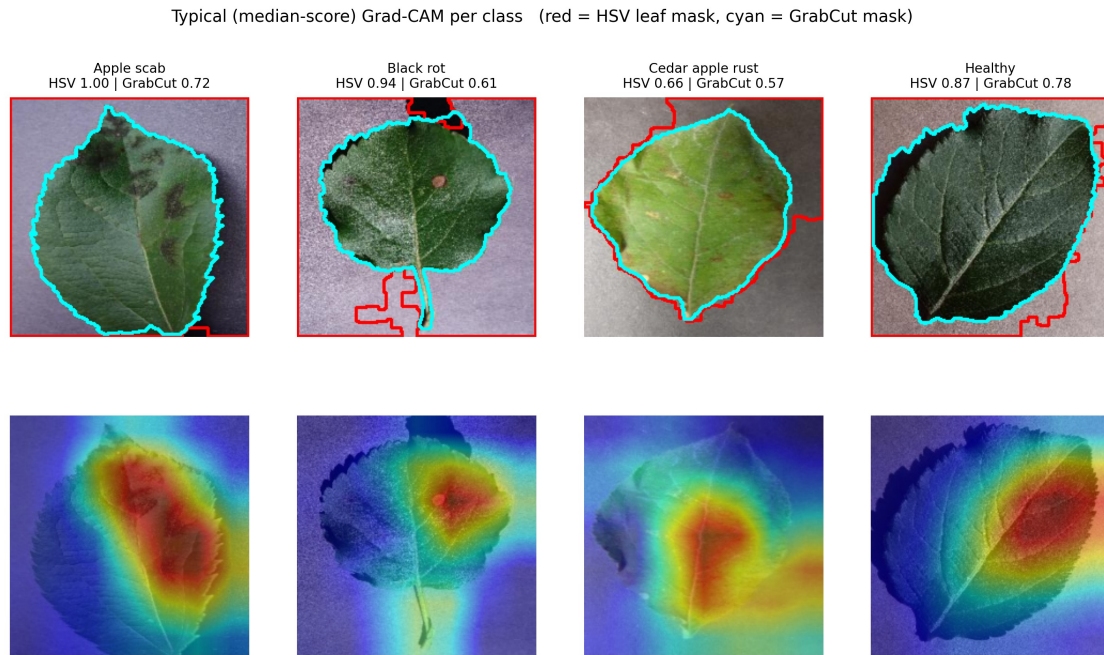


Figure 12. Typical (median-score) Grad-CAM example for each class, with the HSV leaf mask (red) and the independent GrabCut mask (cyan) outlined on the input (top) and the Grad-CAM overlay (bottom).

test split it attains a mean average precision of 92.7% at IoU 0.5 and 71.3% averaged over IoU 0.5–0.95 without test-time augmentation, rising to 93.5% and 73.1% respectively when test-time augmentation is applied (Table 19). Per class, the healthy and rust leaves are detected most reliably (mAP@0.5 of 0.978 and 0.943), while apple scab is the hardest (0.859), mirroring the classification results, where scab is likewise the most confusable field class; its lower recall reflects the small, scattered scab lesions and the modest number of training boxes. Qualitatively, the detector places tight, correctly labeled boxes on leaves photographed against complex backgrounds, at hand-held scale, and under uneven lighting (Figure 14). Because only a few hundred field boxes are available, a small backbone and test-time augmentation are used to keep the detector from over-fitting. These detection scores must, however, be read with caution: the PlantDoc apple test split contains only 29 images and 34 boxes, so the per-class figures in particular carry wide confidence intervals and should be treated as a promising proof of concept for the detection component rather than as a definitive benchmark. Table 18 reports the matched comparison with YOLOv8s (30 epochs, three seeds each). YOLO11s is ahead on every metric: mAP@0.5 of 0.929 ± 0.005 versus 0.881 ± 0.009 with test-time augmentation and 0.929 ± 0.017 versus 0.879 ± 0.004 without, and mAP@0.5:0.95 of 0.726 versus 0.693. The gap of about five points in mAP@0.5 is larger than the seed-to-seed spread of either model, so on this data set the architecture upgrade brings a consistent gain; with 34 test boxes its exact size should nevertheless be taken as indicative. The 100-epoch single-seed model of Table 19 (0.935 with test-time augmentation) agrees with the three-seed 30-epoch result.

Table 18. Matched comparison of YOLOv8s and YOLO11s on the PlantDoc apple test split (29 images, 34 boxes): identical 30-epoch schedule, data and augmentation, mean $\pm$ SD over three seeds; TTA: test-time augmentation.

Detector	Inference	Precision	Recall	mAP@0.5	mAP@0.5:0.95
YOLOv8s	base	0.802 ± 0.030	0.853 ± 0.016	0.879 ± 0.004	0.688 ± 0.012
YOLOv8s	TTA	0.837 ± 0.066	0.859 ± 0.031	0.881 ± 0.009	0.693 ± 0.028
YOLO11s	base	0.839 ± 0.082	0.873 ± 0.041	0.929 ± 0.017	0.722 ± 0.015
YOLO11s	TTA	0.847 ± 0.025	0.903 ± 0.023	0.929 ± 0.005	0.726 ± 0.019

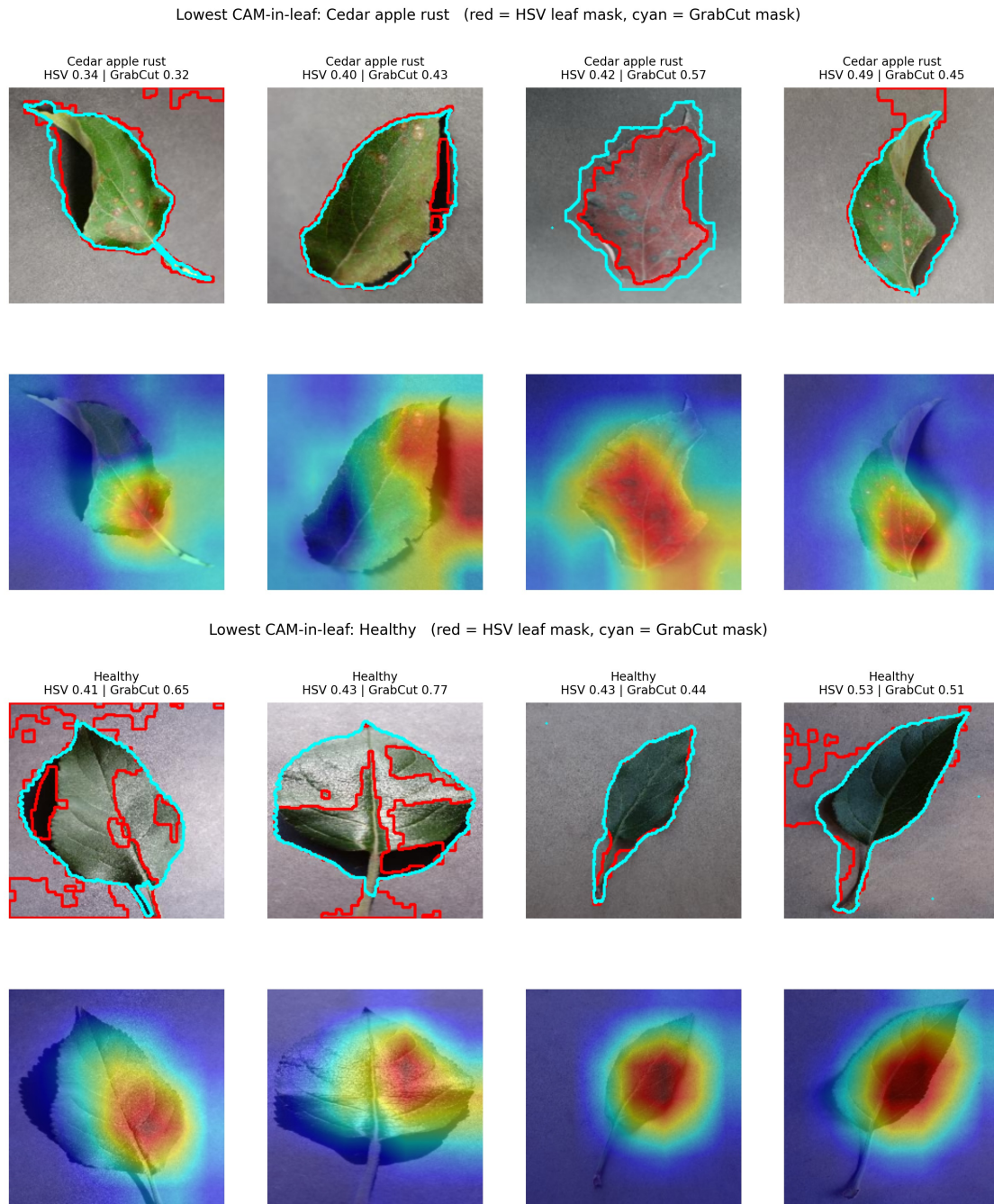


Figure 13. The four lowest-scoring cedar-apple-rust (top pair of rows) and healthy (bottom pair of rows) test images, with HSV (red) and GrabCut (cyan) masks and the Grad-CAM overlay. The heat peak lies on the leaf in most cases; the low scores come from the coarse map spreading beyond small leaves and, for dark healthy leaves, from gaps in the HSV mask.

3.12. Comparison with prior work

Table 20 places these results next to representative studies on PlantVillage and apple-leaf data. The comparison is deliberately cautious: the figures are those reported by the original authors on their own datasets, class counts,

Table 19. YOLOv11s detection performance on the PlantDoc apple test split (per-class and overall without test-time augmentation). Test-time augmentation raises the overall scores to precision 0.896, recall 0.847, mAP@0.5 0.935, and mAP@0.5:0.95 0.731.

Class	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Apple scab	0.881	0.692	0.859	0.679
Apple rust	0.822	0.844	0.943	0.739
Healthy leaf	0.871	0.900	0.978	0.722
All (base)	0.858	0.812	0.927	0.713
All (with TTA)	0.896	0.847	0.935	0.731

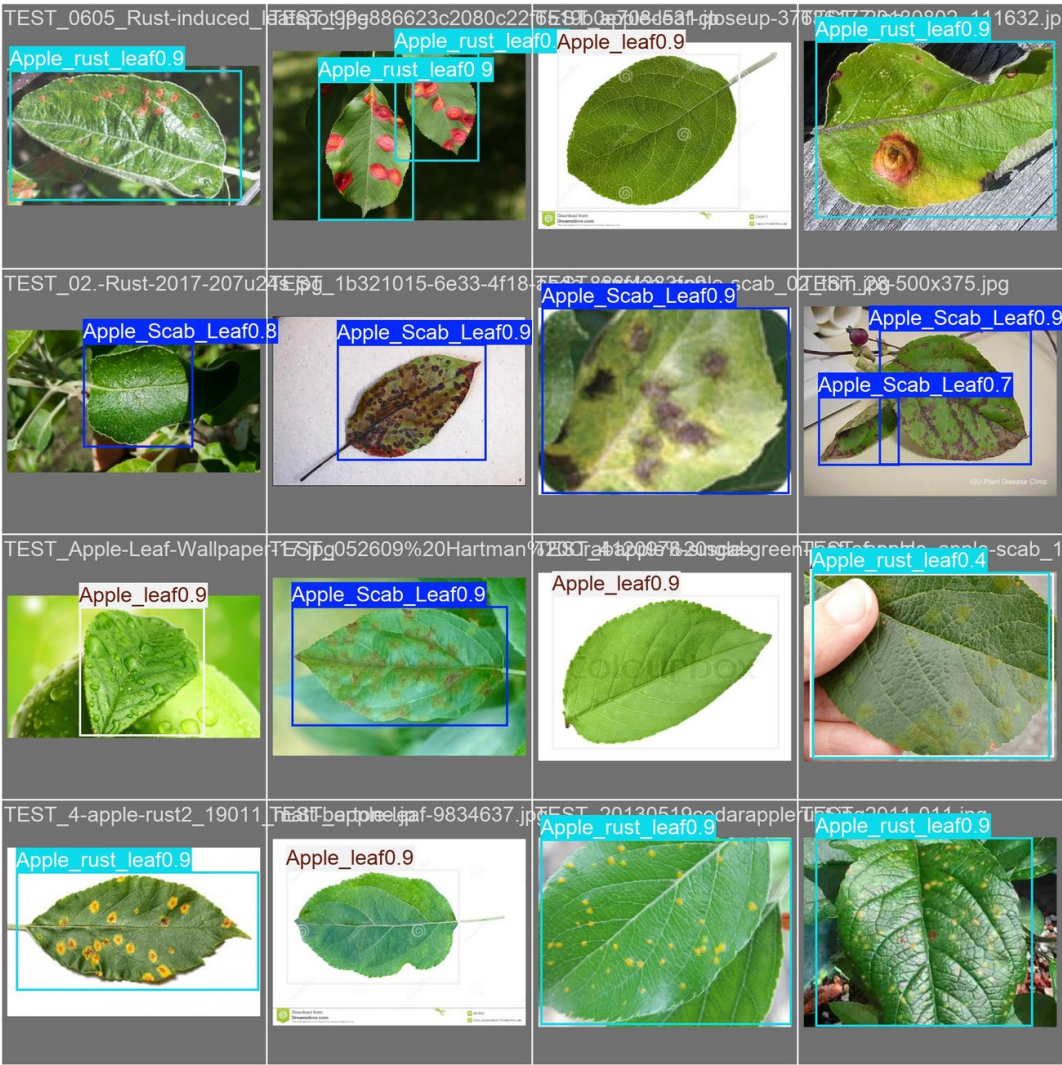


Figure 14. YOLOv11s detections on PlantDoc apple test images: bounding boxes with predicted class and confidence for scab, rust, and healthy leaves in field conditions.

and splits, and are *not* the product of a common re-evaluation, so the table is a map of the landscape rather than a like-for-like ranking. Two patterns stand out. First, the studies on clean PlantVillage imagery cluster between about

96% and 99.8% accuracy, the same range as the present in-domain result, which is expected because all measure performance on the same easy, single-background data. Second, and more tellingly, almost none of these studies verifies a leaf-level disjoint split or reports an independent field evaluation, and the one apple study framed as field detection, Jiang et al. [27], reports a markedly lower 78.8% mAP, closely echoing the drop this work observes once imagery becomes realistic. The contribution here is therefore not a higher in-domain number, which would be meaningless given the saturation and the confidence intervals above, but the leakage-free protocol, the separated zero-shot and cross-validated field results on two datasets, and the added detection component.

Table 20. Representative studies on PlantVillage and apple-leaf data with author-reported accuracy. Figures are on different datasets, class counts, and splits and are not directly comparable; the last two columns name whether a leakage-free (leaf-disjoint) split was reported and the limitation most relevant to this work. “Not reported” means the cited paper does not state whether near-duplicate leaf overlap between splits was checked or prevented, not that it necessarily leaked; only Too2019 and Geetharamani2019 explicitly describe augmenting before splitting, which is why they are marked “No”.

Study	Dataset	Cls.	Model	Acc.	Leak-free?	Limitation vs. this work
Mohanty et al. 2016 [2]	PlantVillage	38	GoogLeNet	99.35%	Not reported	Authors note field collapse; no leakage control
Ferentinos 2018 [28]	Open plant DB	58	VGG	99.53%	Not reported	Lab images; drops on field photos
Too et al. 2019 [15]	PlantVillage	38	DenseNet121	99.75%	No	Single dataset; on-disk augmentation
Geetharamani & Pandian 2019 [29]	PlantVillage	39	9-layer CNN	96.46%	No	Heavy on-disk augmentation before split
Liu et al. 2018 [30]	Own apple set	4	AlexNet-like	97.62%	Not reported	Small custom set; no cross-dataset test
Zhong & Zhao 2020 [31]	AI Challenger apple	6	DenseNet121	93.71%	Not reported	Single dataset; no field test
Bansal et al. 2021 [32]	Plant Pathology 2020	4	DenseNet + EffNet	96.25%	Not reported	Ensemble cost; no field test
Jiang et al. 2019 [27]	ALDD apple (field)	5	INAR-SSD	78.8% mAP	N/A (detect.)	Field detection; lower, realistic score
This work (in-domain)	PlantVillage apple, leak-free	4	ResNet50	99.86%	Yes	Verified disjoint splits; 3-seed mean
This work (zero-shot)	PlantDoc / FGVC8 (field)	3/4	ResNet50	59.5% / 43.9%	Yes	Only genuine field-generalization figures
This work (detection)	PlantDoc apple (field)	3	YOLOv11s	92.7% mAP	Yes	Explicit field detection

4. Discussion

The central message of this study is methodological. When the apple subset of PlantVillage is prepared correctly, using one image per leaf with augmentation applied only at training time, modern CNNs separate the four classes almost perfectly [2, 15], and the choice of backbone matters less than is often implied. ResNet50 and VGG16 lead by a few hundredths of a point, but MobileNetV2 matches them for practical purposes at a fraction of the cost, which is the more relevant fact for field deployment [21]. The near-perfect in-domain scores, whose between-backbone spread is within the run-to-run standard deviation, should therefore be read not as evidence of a breakthrough architecture but as confirmation that the task is easy on clean laboratory data, and that most of the difficulty in prior work came from the evaluation setup rather than the disease-recognition problem itself [43].

Two of the results push back directly on common claims. First, the augmentation ablation shows a true in-domain gain of less than half a point, not the multi-point improvements sometimes reported [17]; those larger figures are consistent with augmentation amplifying leakage rather than improving generalization [44]. Second, and more

importantly, the external zero-shot evaluation, now on two independent field datasets, exposes the gap between laboratory and field performance: a model that is flawless on PlantVillage is correct on only about 40–60% of real-field images when no field data are used in training [5, 6]. The recovery under cross-validation, to 82% on PlantDoc and 94% on the larger FGVC8, is achieved only by adding field images to training, so it is an adaptation result and not evidence that the laboratory-trained model generalizes on its own; that the recovery is much stronger on FGVC8 is itself the clearest evidence that field data, not architecture, is the binding constraint. This distinction is precisely the information a practitioner needs, and it is routinely omitted from single-dataset studies.

The decision to localize disease with Grad-CAM instead of pixel-level segmentation is likewise driven by the data. Reporting a high segmentation IoU against masks that are themselves only about 0.17 IoU consistent with the visible lesions would be misleading. Grad-CAM offers weak but honest localization with no additional labels [4], and the CAM-in-leaf score of 86%, whose class ordering is confirmed by an independent GrabCut mask, is consistent with the classifier attending to leaf tissue rather than background, without being taken as proof of lesion-level focus. Its large standard deviation is explained by class, with cedar apple rust and some healthy leaves scoring lower, and by the coarse resolution of Grad-CAM on small leaves, not by the model attending to the background (Section 3.10).

A note on the frog-eye-to-black-rot mapping is warranted. Because FGVC8 contains no separately labeled apple black rot, the “black rot” field result is obtained entirely from frog-eye-leaf-spot images; the mapping is therefore a naming decision, and every classification metric is numerically identical whether the class is called “frog-eye leaf spot” or “black rot”. We report it transparently so that readers who treat the two as distinct can reinterpret that single class accordingly; Section 2.6 gives the symptomatic justification, and Section 3.6 shows that the mapping is visually supported and partly reflected in feature space, but that its effect cannot be separated from the general domain shift. The framework itself is crop-agnostic, and applying the same leakage-free, honestly-evaluated protocol to other crops and to larger multi-disease field collections is a natural direction for future work.

Two methodological choices that referees asked about can now be defended with data rather than argument. First, class imbalance is handled with inverse-frequency class weighting (Section 2.2) rather than oversampling by duplication, which would re-create the near-duplicates this paper removes. Generative oversampling avoids literal duplicates, and our filter confirmed that none of the synthetic cedar-rust images was a near-copy of a real one, yet it still made the field model worse (Section 3.9): synthetic images inherit the laboratory appearance of their training set, so they add in-domain accuracy that does not transfer. Focal loss was not tested; with in-domain accuracy saturated and the minority class already recognised almost perfectly, there is no room for any imbalance strategy to improve the laboratory metric, and the relevant question, as the GAN result shows, is how a strategy affects the field. Second, fine-tuning on field data and unsupervised domain adaptation are complementary, since the latter needs no field labels. On FGVC8, of the roughly 62 points lost between the laboratory and the field, domain-adversarial training recovers about 17, correlation alignment none, and fine-tuning on 68 labelled images about 40 (Section 3.8).

Guidelines for practitioners. Taken together, the results suggest a short checklist for building a field-ready disease classifier: (i) split by physical leaf before any augmented copy is written to disk, because a leak cannot be detected from the laboratory metric; (ii) treat in-domain accuracy on a clean laboratory set as a sanity check rather than a model-selection criterion, and choose the backbone by cost, since MobileNetV2 is as accurate, adapts as well, and runs five to ten times faster than the larger models; (iii) report zero-shot field accuracy over several training runs with confidence intervals, since it varied by about 20 points on PlantDoc between runs of the same architecture; (iv) favour geometric augmentation, the family that transfers; (v) label a few dozen field images and fine-tune, starting from the leakage-free laboratory model when the fine-tuning budget is short, and fall back on adversarial adaptation only when no field label can be obtained; and (vi) distrust any change, such as synthetic oversampling, that improves the laboratory metric without a field test, since it can make the field model worse.

The study has clear limitations. The clean four-class set is modest in size, especially for cedar apple rust, so the in-domain accuracies carry the run-to-run standard deviations reported in Section 3.1 and the top of the ranking is a statistical tie, even with five seeds; one field dataset (PlantDoc) covers only three classes, but the second (FGVC8) covers all four including black rot via the frog-eye-leaf-spot mapping, whose effect on the zero-shot numbers cannot be fully separated from the general domain shift (Section 3.6); the detection component is trained and tested on

only a few hundred and a few dozen PlantDoc apple boxes respectively, so its mAP is a proof of concept with wide intervals, even though the YOLO11-versus-YOLOv8 difference is consistent across seeds, and it does not localize black rot at all (Section 2.7); and the localization is image-level rather than pixel-accurate, validated against an independent automatic segmentation but not against expert lesion annotation. Both field datasets are assembled from heterogeneous sources, which introduces label and provenance noise that we illustrate qualitatively (Section 2.6) but do not quantify, and we did not test the sensitivity of the models to deliberately injected label errors. The augmentation-family ablation used one seed per arm, the PlantDoc data-efficiency study was run only under the short budget with a 58-image test split, and the whole study concerns one crop: the protocol is crop-agnostic, but we have not demonstrated it on a second crop. None of these undermine the main conclusions, but they mark the road ahead: an expert-annotated validation subset for Grad-CAM; a black-rot detection class, whether via classifier-seeded pseudo-labels or new annotation, and a larger field detection set; a provenance and label-noise audit of the field datasets; applying the leakage-free protocol to the other PlantVillage crops, where the same on-disk augmentation practice is common; and larger and more diverse field collections. Disease severity, the fraction of leaf area affected, is what growers ultimately need for treatment decisions. The leaf masks and Grad-CAM maps computed here could provide a weak, annotation-free proxy for affected area, but reliable severity estimation requires lesion-level annotation; should trustworthy annotations become available, a return to genuine pixel-level severity estimation would also let disease progression, not just presence, be tracked over a growing season.

5. Conclusions

This paper presented an apple-leaf disease framework whose contribution is as much about how models are evaluated as about the models themselves. Building a de-duplicated, leakage-free benchmark removes the near-duplicate contamination found in much of the prior literature [33], and a controlled comparison with identical code shows what that contamination does: it leaves the laboratory accuracy untouched, even on test leaves never seen in training, so it cannot be detected from the headline number, yet it lowers accuracy on an independent field dataset (39.0% versus 43.9% on FGVC8). Under the honest protocol five CNN backbones all reach 99.35–99.80% accuracy over five seeds, a tie that makes cost the deciding factor; MobileNetV2 is five to ten times faster than the larger models while adapting to field data at least as well. External zero-shot accuracy on two field datasets is low and unstable across training runs ($59.5 \pm 5.7\%$ on PlantDoc, $43.9 \pm 0.5\%$ on FGVC8), with most field images, including healthy ones, predicted as diseased; geometric augmentation is the only augmentation family that measurably narrows this gap, and synthetic minority-class images from a GAN widen it. Unsupervised domain-adversarial training recovers part of the gap without field labels (54.4% on FGVC8), but fine-tuning on only 68 labelled field images recovers far more (76.7%), and five-fold cross-validation with field data reaches 81.9% and 94.1%. Leakage-free laboratory pre-training makes this adaptation much faster (+0.21 macro-F1 with 80 field images under a short 15-epoch budget, consistently across three backbones and on PlantDoc) but does not raise the accuracy reached once fine-tuning converges. Grad-CAM attention falls on the leaf for all four classes, with the class ordering confirmed by an independent segmentation, and a YOLO11 detector localizes diseased leaves in the field at 93.5% mAP@0.5, ahead of YOLOv8 in a matched three-seed comparison. Together these results provide a reproducible, honestly evaluated baseline for apple-leaf disease detection and recognition, and they argue that future work in this area should report leakage-free splits, clearly labelled cross-dataset performance, and variability across training runs as a matter of course.

Author Contributions

M. A. Malak: conceptualization, methodology, software, data curation, formal analysis, investigation, writing (original draft), visualization. **M. Thabet:** conceptualization, methodology, supervision, writing (review and editing). **A. S. Aziz:** methodology, validation, writing (review and editing). **A. A. Alhelbawy:** conceptualization, validation, resources, supervision, project administration, writing (review and editing). All authors read and approved the final manuscript.

Funding

This research received no external funding.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data Availability

The PlantVillage, PlantDoc, and Plant Pathology 2021 (FGVC8) datasets used in this study are publicly available. To support reproducibility, the de-duplicated, leakage-free split files (explicit per-partition image lists), the training and evaluation code, the notebooks, and the YOLO data-preparation scripts are released publicly at <https://github.com/MarkoAMalak/apple-leaf-disease-framework> and archived on Zenodo at <https://doi.org/10.5281/zenodo.21336556> (concept DOI, which always resolves to the latest version). The code, exact split files, per-image predictions and a summary of every number reported in this revised version are in the `revision_2026` folder of the repository (release v2.0.0).

Acknowledgement

The authors would like to thank Fayoum University for supporting the publication of this work.

REFERENCES

1. A. Kamilaris, and F.X. Prenafeta-Boldú, *Deep learning in agriculture: a survey*, Computers and Electronics in Agriculture, vol. 147, pp. 70–90, 2018.
2. S.P. Mohanty, D.P. Hughes, and M. Salathé, *Using deep learning for image-based plant disease detection*, Frontiers in Plant Science, vol. 7, 1419, 2016.
3. D.P. Hughes, and M. Salathé, *An open access repository of images on plant health to enable the development of mobile disease diagnostics*, arXiv:1511.08060, 2015.
4. R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, *Grad-CAM: visual explanations from deep networks via gradient-based localization*, In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 618–626, 2017.
5. D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, and N. Batra, *PlantDoc: a dataset for visual plant disease detection*, In Proceedings of the 7th ACM IKDD CoDS and 25th COMAD, pp. 249–253, 2020.
6. R. Thapa, K. Zhang, N. Snavely, S. Belongie, and A. Khan, *The Plant Pathology Challenge 2020 data set to classify foliar disease of apples*, Applications in Plant Sciences, vol. 8, no. 9, e11390, 2020.
7. A.I. Khan, S.M.K. Quadri, and S. Banday, *Deep learning for apple diseases: classification and identification*, International Journal of Computational Intelligence Studies, vol. 10, pp. 1–12, 2021.
8. J.G.A. Barbedo, *A review on the main challenges in automatic plant disease identification based on visible range images*, Biosystems Engineering, vol. 144, pp. 52–60, 2016.
9. S.S. Harakannanavar, J.M. Rudagi, V.I. Puranikmath, A. Siddiqua, and R. Pramodhini, *Plant leaf disease detection using computer vision and machine learning algorithms*, Global Transitions Proceedings, vol. 3, pp. 305–310, 2022.
10. S. Kaur, S. Pandey, and S. Goel, *Plants disease identification and classification through leaf images: a survey*, Archives of Computational Methods in Engineering, vol. 26, pp. 507–530, 2019.
11. M.H. Saleem, J. Potgieter, and K.M. Arif, *Plant disease detection and classification by deep learning*, Plants, vol. 8, 468, 2019.
12. S.J. Pan, and Q. Yang, *A survey on transfer learning*, IEEE Transactions on Knowledge and Data Engineering, vol. 22, pp. 1345–1359, 2010.
13. O. Russakovsky et al., *ImageNet large scale visual recognition challenge*, International Journal of Computer Vision, vol. 115, pp. 211–252, 2015.
14. J. Chen, J. Chen, D. Zhang, Y. Sun, and Y.A. Nanekharan, *Using deep transfer learning for image-based plant disease identification*, Computers and Electronics in Agriculture, vol. 173, 105393, 2020.
15. E.C. Too, L. Yujian, S. Njuki, and L. Yingchun, *A comparative study of fine-tuning deep learning models for plant disease identification*, Computers and Electronics in Agriculture, vol. 161, pp. 272–279, 2019.

16. J. Amara, B. Bouaziz, and A. Algergawy, *A deep learning-based approach for banana leaf diseases classification*, In BTW (Workshops), pp. 79–88, 2017.
17. C. Shorten, and T.M. Khoshgoftaar, *A survey on image data augmentation for deep learning*, Journal of Big Data, vol. 6, 60, 2019.
18. K. Simonyan, and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, In International Conference on Learning Representations (ICLR), 2015.
19. K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.
20. G. Huang, Z. Liu, L. van der Maaten, and K.Q. Weinberger, *Densely connected convolutional networks*, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4700–4708, 2017.
21. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, *MobileNetV2: inverted residuals and linear bottlenecks*, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4510–4520, 2018.
22. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, *Rethinking the Inception architecture for computer vision*, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2818–2826, 2016.
23. I. Loshchilov, and F. Hutter, *Decoupled weight decay regularization*, In International Conference on Learning Representations (ICLR), 2019.
24. A. Paszke et al., *PyTorch: an imperative style, high-performance deep learning library*, In Advances in Neural Information Processing Systems (NeurIPS), pp. 8024–8035, 2019.
25. G. Jocher, and J. Qiu, *Ultralytics YOLO11*, 2024. Available: <https://github.com/ultralytics/ultralytics>.
26. F. Pedregosa et al., *Scikit-learn: machine learning in Python*, Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
27. P. Jiang, Y. Chen, B. Liu, D. He, and C. Liang, *Real-time detection of apple leaf diseases using an improved convolutional neural network*, IEEE Access, vol. 7, pp. 59069–59080, 2019.
28. K.P. Ferentinos, *Deep learning models for plant disease detection and diagnosis*, Computers and Electronics in Agriculture, vol. 145, pp. 311–318, 2018.
29. G. Geetharamani, and A. Pandian, *Identification of plant leaf diseases using a nine-layer deep convolutional neural network*, Computers & Electrical Engineering, vol. 76, pp. 323–338, 2019.
30. B. Liu, Y. Zhang, D. He, and Y. Li, *Identification of apple leaf diseases based on deep convolutional neural networks*, Symmetry, vol. 10, 11, 2018.
31. Z. Zhong, and M. Zhao, *Research on deep learning in apple leaf disease recognition*, Computers and Electronics in Agriculture, vol. 168, 105146, 2020.
32. P. Bansal, R. Kumar, and S. Kumar, *Disease detection in apple leaves using deep convolutional neural network*, Agriculture, vol. 11, no. 7, 617, 2021.
33. S. Kapoor, and A. Narayanan, *Leakage and the reproducibility crisis in machine-learning-based science*, Patterns, vol. 4, 100804, 2023.
34. G. Li, Y. Wang, Q. Zhao, P. Yuan, and B. Chang, *PMVT: a lightweight vision transformer for plant disease identification on mobile devices*, Frontiers in Plant Science, vol. 14, 1256773, 2023.
35. W. Lv, S. Xu, Y. Zhao, G. Wang, J. Wei, C. Cui, Y. Du, Q. Dang, and Y. Liu, *DETRs beat YOLOs on real-time object detection*, In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024.
36. M.A. Ellis, *Black rot and frog-eye leaf spot of apple*, Ohio State University Extension Factsheet PLPATH-FRU-07, 2016.
37. G. Jocher, A. Chaurasia, and J. Qiu, *Ultralytics YOLOv8*, 2023. Available: <https://github.com/ultralytics/ultralytics>.
38. B. Sun, and K. Saenko, *Deep CORAL: correlation alignment for deep domain adaptation*, In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, pp. 443–450, 2016.
39. Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, *Domain-adversarial training of neural networks*, Journal of Machine Learning Research, vol. 17, no. 1, pp. 2096–2130, 2016.
40. O.H.M. Assass, B.S. Tawfik, M.M. El-Gazzar, and A.M.M. Elnawsany, *ECG image classification using hybrid machine learning and deep learning techniques*, Statistics, Optimization and Information Computing, vol. 16, no. 3, pp. 2311–2330, 2026.
41. A. Bourzik, B. Bouikhalene, J. El-Mekkaoui, and A. Hjouji, *Discrete orthogonal moment descriptors applied in image classification and object recognition enhanced by machine learning methods*, Statistics, Optimization and Information Computing, vol. 15, no. 2, pp. 955–974, 2025.
42. A.S. Abd El Aziz, M.M. El-Gazzar, and M.M. Saeid, *A hybrid fusion approach for skin cancer detection using deep learning on clinical images and machine learning on patient metadata*, Statistics, Optimization and Information Computing, vol. 14, pp. 2091–2103, 2025.
43. M.A. Noyan, *Uncovering bias in the PlantVillage dataset*, arXiv:2206.04374, 2022.
44. I.E. Tampu, A. Eklund, and N. Haj-Hosseini, *Inflation of test accuracy due to data leakage in deep learning-based classification of OCT images*, Scientific Data, vol. 9, 580, 2022.
45. C. Rother, V. Kolmogorov, and A. Blake, *“GrabCut”: interactive foreground extraction using iterated graph cuts*, ACM Transactions on Graphics, vol. 23, no. 3, pp. 309–314, 2004.
46. A. Radford, L. Metz, and S. Chintala, *Unsupervised representation learning with deep convolutional generative adversarial networks*, In International Conference on Learning Representations (ICLR), 2016.
47. L. van der Maaten, and G. Hinton, *Visualizing data using t-SNE*, Journal of Machine Learning Research, vol. 9, pp. 2579–2605, 2008.
48. B. Efron, and R.J. Tibshirani, *An Introduction to the Bootstrap*, Chapman & Hall/CRC, New York, 1993.