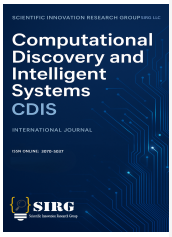


Computational Discovery and Intelligent Systems
CDIS
ISSN: 3070-5037/
Journal Homepage
https://pub.scientificirg.com/index.php/CDIS

Multimodal RGB–Thermal Deep Learning for Red Palm Weevil Damage Classification in Date Palms with a Tree-Level Evaluation of Six Backbone Architectures

Jumana Mohamed Ahmed[a] , **Mohammed Melhi**[b] , **and A. A. Somaie**[c,1]

[a] *Computer Science and Artificial Intelligence, Beni-Suef University, Beni-Suef City, Egypt; E-mail: jumanamohamed340@gmail.com*

[b] *PhD, University of Bradford, UK, Chief Executive Officer CEO of Rushd AI Company and Amatrix AI Company, Riyadh, KSA; E-mail: mmelhi@rushai.com, mmelhi@promatrix.ai*

[c] *PhD, University of Bradford, UK, PDF Post-Doctoral Fellow Research Associate, University of Calgary, Canada, Software Engineering Program SE, Faculty of Computer Science, October University for Modern Sciences & Arts MSA, 6 October, Giza, Egypt; E-mail: alibrahim@msa.edu.eg, aasomaie@gmail.com*

ABSTRACT

The red palm weevil (*Rhynchophorus ferrugineus*) is the most destructive pest of date palm (*Phoenix dactylifera* L.), feeding inside the trunk and destroying vascular tissue well before any external symptom becomes visible. Field surveys, acoustic sensing, and single-modality imaging each address part of this early-detection problem but none jointly exploits the complementary visual and thermal symptoms that a weevil infestation produces. This study introduces a dual-branch deep learning framework that fuses paired RGB and thermal photographs of individual date palms for four-class health classification (non-infected, infected, badly damaged, dead). Six convolutional and transformer backbones, namely ResNet-50, EfficientNet-B0, ConvNeXt-Tiny, Xception, ViT-Small, and ConvNeXt V2-Tiny, are trained under one identical protocol that combines a multi-scale feature-pyramid fusion neck, thermal-guided spatial attention, efficient channel attention, class-balanced focal loss, prototype-based contrastive regularization, curriculum-scheduled mixup and CutMix, and a stochastic weight averaging tail phase. Evaluation uses a leakage-free, tree-level split of 179 field-surveyed palms (125 train, 27 validation, 27 test trees) with 95% bootstrap confidence intervals and paired McNemar and DeLong significance testing. The strongest individual backbone, ConvNeXt-Tiny, reaches 74.6% test accuracy and a macro-F1 of 0.764; a calibration-weighted ensemble of all six backbones reaches 76.2% accuracy with a macro-F1 of 0.756 (95% bootstrap interval, 0.65 to 0.86 for accuracy). Ensembling yields a statistically significant gain over the two weakest backbones but not over the stronger individual models at this sample size, indicating that multimodal fusion and ensembling provide measurable, architecture-dependent gains that remain modest on a dataset of this scale.

PAPER INFORMATION

HISTORY
Received: 29 March 2026
Revised: 8 May 2026
Accepted: 19 June 2026
Online: 30 June 2026

MSC
68T07; 68R10; 94A60; 68M15

KEYWORDS
Multimodal Fusion;
Deep Learning;
Red Palm Weevil;
Damage Classification;
Ensemble Learning.

[1]Corresponding author at PhD, University of Bradford, UK, PDF Post-Doctoral Fellow Research Associate, University of Calgary, Canada, Software Engineering Program SE, Faculty of Computer Science, October University for Modern Sciences & Arts MSA, 6 October, Giza, Egypt; E-mail: alibrahim@msa.edu.eg, aasomaie@gmail.com

1 INTRODUCTION

Date palm (*Phoenix dactylifera* L.) cultivation supports the livelihoods of millions of smallholder farmers and agribusinesses across the Middle East, North Africa, and other arid and semi-arid regions, carrying substantial economic, nutritional, and cultural value in these areas. Among the threats facing date palm plantations, the red palm weevil (RPW), *Rhynchophorus ferrugineus*, is widely regarded as the most destructive, owing to a larval feeding habit that destroys the internal vascular tissue of the trunk before any external symptom becomes visible [1, 2]. By the time canopy discoloration, frond wilting, or structural collapse appears, the infestation has typically progressed beyond the point at which treatment can save the tree. Early, reliable, and scalable detection therefore remains the central unsolved problem in RPW management.

Conventional inspection methods, including manual visual surveys, acoustic resonance sensing, pheromone trapping, and chemical volatile detection, each carry limitations in labor cost, geographic coverage, or sensitivity to early-stage infestation [3, 4]. Remote sensing combined with deep learning has accordingly attracted interest as a more scalable monitoring approach. Unmanned aerial vehicles can capture RGB, multispectral, and thermal imagery at fine spatial resolution [5], and thermal imaging in particular offers a physically motivated signal: larval feeding activity and the resulting tissue decay can alter the trunk and crown temperature profile relative to a healthy tree, a cue that is largely invisible in the RGB channel alone. RGB imagery, conversely, captures the external morphological symptoms, including canopy discoloration, frond sparsity, and trunk surface damage, that accumulate as infestation progresses. The two modalities are therefore complementary in principle, yet most prior RPW classification studies use only one of them, leaving open the question of whether a learned fusion of the two yields a measurable and statistically defensible improvement over either modality alone.

This study addresses that question directly. A tree-level health classifier fuses paired RGB and thermal photographs of individual date palms acquired during field surveys. Rather than committing to a single fixed backbone, six widely used image classification architectures, spanning convolutional networks of different design families and a vision transformer, are trained under one identical dual-branch protocol so that the contribution of backbone choice can be isolated from the contribution of the fusion mechanism itself. The architecture combines independent feature extraction branches for each modality, a hierarchical multi-scale fusion neck inspired by feature pyramid networks, thermal-guided spatial attention, efficient channel attention, a class-balanced focal training objective, prototype-based contrastive regularization, curriculum-scheduled mixing augmentation, and a stochastic weight averaging tail phase. Performance is reported with tree-level data splitting to avoid the leakage that would otherwise occur if multiple photographs of the same tree were distributed across training and test sets, together with bootstrap confidence intervals, paired statistical significance testing, and a three-way ablation against single-modality and naively concatenated baselines. Every number reported in this manuscript is traceable to the execution log or to the figures generated during the corresponding run; no projected, assumed, or illustrative figures are reported.

The principal contributions of this study are summarized as follows.

1. A systematic, identical-protocol comparison of six backbone architectures within a dual-branch RGB–thermal fusion network for four-class date palm health classification, evaluated on a leakage-free, tree-level split with bootstrap confidence intervals and paired significance testing (Section 4).
2. A class-imbalance handling strategy combining effective number class balanced focal loss, inverse-frequency weighted sampling, curriculum-scheduled mixup and CutMix with minority-class-biased pairing, and domain-specific morphological augmentation that simulates RPW symptoms, with the imbalance explicitly quantified in **Table 2** and Section 3.3.
3. A transparent, code-verified account of where each architectural component does and does not apply; for example, the multi-scale fusion neck operates for only three of the six backbones because intermediate feature hooks are registered only for those architectures in the implementation actually executed, an asymmetry that is reported explicitly in Section 3.4 and revisited in Section 5.3 rather than left implicit.
4. A five-way comparison of ensembling strategies, unweighted soft voting, validation-macro-F1-weighted voting, inverse-calibration-error-weighted voting, and a cross-validated logistic-regression stacking meta-learner, supported by McNemar and DeLong significance tests that distinguish genuine gains from sampling noise at this test-set size.
5. A fully reconciled set of dataset, training, and evaluation statistics, with every reported number traceable to either the training log or to the figures generated during the run, replacing inconsistent or unverifiable figures from earlier drafts of this work.

The remainder of this paper is organized as follows. Section 2 reviews related work on RPW detection and on the supervised-learning techniques employed. Section 3 describes the dataset, preprocessing pipeline, network architecture, training procedure, and evaluation protocol. Section 4 reports experimental results, including ablation, ensembling, calibration, and stability analyses. Section 5 discusses limitations and threats to validity. Section 6 concludes with directions for future work.

2 RELATED WORK

2.1 Remote sensing and deep learning for RPW detection

Early work on automated RPW monitoring demonstrated that aerial and street-level imagery, processed with deep convolutional networks, could map palm tree locations and flag candidate infestations at urban and regional scale; Kagan et al. analyzed more than 100,000 aerial and street-view images to localize palm trees and surface potential RPW infestation hotspots without specialized sensing equipment [6]. Delalieux et al. used two consecutive seasons of unmanned-aerial-vehicle-acquired RGB, multispectral, and thermal imagery over an experimental field of infested and control palms and found that temporal changes in vegetation-index values and thermal canopy signatures could indicate infestation before visible crown symptoms developed, although the internal trunk-temperature anomaly associated with larval activity does not always propagate to a detectable external crown-temperature change [5]. More recent aerial detection work has shifted toward end-to-end object detectors for palm localization; Ashraf et al. reviewed several recent unmanned-aerial-vehicle- and satellite-oriented palm detection efforts within a broader pest-detection study [7].

Acoustic sensing represents a complementary, non-visual modality. Boulila et al. converted recorded RPW activity sounds into multi-feature image representations and applied deep convolutional classifiers to discriminate infested from healthy palms [3], while Ashry et al. combined optical-fiber distributed acoustic sensing with a convolutional classifier in a field deployment, illustrating that fiber-based sensing along existing cable infrastructure can supplement or replace discrete point sensors for early detection [4]. Sayed et al. proposed a modified ResNet-34 architecture for RPW life-stage classification from a moderate-sized image set [8], and Arasi et al. combined a bird-swarm metaheuristic with a deep network for RPW image classification [9]. A recent Internet-of-Things implementation embedded a compact convolutional network on a Raspberry Pi for field-deployable, thermal-image-based screening [10], illustrating continuing interest in low-cost, edge-deployable RPW screening systems. None of these studies combines paired RGB and thermal imagery within a single fusion network trained and evaluated under a leakage-free, tree-level protocol with multiple backbone architectures, which is the gap addressed by the present work.

The thermal-RGB paired dataset used in this study originates from Nadeem et al., who collected RGB and thermal images from 179 farmer-surveyed date palms in Khairpur, Sindh, Pakistan, with each tree labeled into one of four health categories (non-infected, infected, badly damaged, dead) based on farmer-reported assessment, and released the collection as an open dataset article [11]. The present study is an independent application study built on that released dataset rather than a re-description of the dataset itself; the dataset and the preprocessing actually consumed by the implementation are described in Section 3.1.

2.2 Supervised learning techniques employed

Several established techniques from the broader computer-vision literature are incorporated into the classification pipeline and are cited at the point of use in Section 3: class-balanced loss reweighting based on the effective number of samples [12], focal loss [13], mixup [14] and CutMix [15] data mixing, efficient channel attention [16], stochastic weight averaging [17], the one-cycle learning-rate policy [18], decoupled weight-decay regularization [19], prototype-based contrastive representation learning [20, 21], post-hoc logit adjustment for long-tailed recognition [22], AugMix data augmentation [23], Grad-CAM and Grad-CAM++ visual explanation [24, 25], integrated-gradients attribution [26], and the standard statistical comparison tools of the DeLong test [27] and the McNemar test [28]. The backbone architectures evaluated are ResNet-50 [29], EfficientNet-B0 [30], ConvNeXt-Tiny [31], Xception [32], ViT-Small [33], and ConvNeXt V2-Tiny [34].

2.3 Summary of related studies

Table 1 summarizes the methodological characteristics of the most closely related published studies, alongside the present work, on the dimensions that determine direct comparability: the modality used, and the task definition. The summary clarifies, in compact form, why a literal accuracy ranking across these studies would be misleading; the underlying differences are discussed in detail in Section 5.2.

3 MATERIALS AND METHODS

3.1 Dataset

The classification dataset originates from the publicly released collection of Nadeem et al. [11], comprising RGB and thermal images of date palm trunks and crowns collected during field surveys, with each tree categorized by farmer

Table 1: Summary of related RPW detection and classification studies

Study	Year	Modality	Dataset	Task	Model	Contribution	Limitation
Kagan et al. [6]	2021	RGB (aerial / street-view)	>100,000 images	Palm localization, infestation hotspot flagging	CNN detector	Large-scale, equipment-free localization	No health-severity labels; no thermal cue
Delalieux et al. [5]	2023	RGB + multispectral + thermal	Small experimental field, two seasons	Binary infested vs. control	Vegetation- and thermal-index analysis	Temporal pre-symptomatic thermal signal	No deep fusion model; small controlled field
Boulila et al. [3]	2023	Acoustic (image-encoded)	Moderate, not tree-level reported	Binary infested vs. healthy	CNN on acoustic features	Non-visual, cable-free sensing	Acoustic-only; no visual health grading
Ashry et al. [4]	2022	Fiber-optic acoustic	Field deployment	Binary infested vs. healthy	CNN classifier	Infrastructure-integrated sensing	Requires fiber installation; binary only
Sayed et al. [8]	2025	RGB	Moderate image set	RPW life-stage classification	Modified ResNet-34	Life-stage granularity	Image-level split; small set; single modality
Arasi et al. [9]	2023	RGB	Moderate image set	RPW image classification	Bird-swarm-optimized CNN	Metaheuristic-tuned architecture	Image-level split; single modality
Martin et al. [10]	2026	Thermal	Field deployment, edge device	Binary infestation screening	Compact CNN on Raspberry Pi	Low-cost, edge-deployable screening	Binary task; thermal-only; deployment-scale evaluation not specimen independent
Ashraf et al. (PalmNeXt) [7]	2026	RGB	Leaf-level pest images	Pest detection in leaf imagery	ConvNeXt-based classifier	ConvNeXt applied to a related pest task	Leaf-level, not whole-tree; single modality

assessment into one of four classes: non-infected, infected, badly damaged, and dead. After matching RGB and thermal images by palm identifier and discarding identifiers present in only one modality, 179 trees with both RGB and thermal coverage were retained. Each tree contributes between zero and four paired views, corresponding to different vantage points around the trunk and crown; after enforcing a maximum of four matched views per modality and tree, the implementation produced 423 paired RGB–thermal samples in total.

Table 2 reports the class-wise distribution of the dataset at the tree level, together with the distribution after splitting. The dataset is markedly imbalanced: the badly damaged class accounts for only 6.7% of trees (12 of 179), against 44.7% for the majority non-infected class, an imbalance ratio of approximately 6.7:1 between the largest and smallest classes. This imbalance, and the steps taken to address it, are discussed in Section 3.3.

Table 2: Class-wise distribution of the paired RGB–thermal dataset at the tree level, before and after the stratified, leakage-free 70/15/15 split (random seed 42)

Class	Full dataset (n, %)	Train (n, %)	Validation (n, %)	Test (n, %)
Non-infected	80 (44.7%)	56 (44.8%)	12 (44.4%)	12 (44.4%)
Infected	32 (17.9%)	22 (17.6%)	5 (18.5%)	5 (18.5%)
Dead	55 (30.7%)	39 (31.2%)	8 (29.6%)	8 (29.6%)
Badly damaged	12 (6.7%)	8 (6.4%)	2 (7.4%)	2 (7.4%)
Total trees	**179**	**125**	**27**	**27**
Total paired images	423	294	66	63

Splitting was performed at the tree level rather than at the image level, using stratified random sampling (70% train, 15% validation, 15% test, random seed 42), so that every paired view of a given tree belongs to a single split. A leakage check confirmed an empty intersection between the tree-identifier sets of the three splits. This protocol prevents the data leakage that would otherwise occur if multiple, near-duplicate photographs of the same tree were distributed across training and test sets, which would inflate reported accuracy. Splitting at the image level instead of the tree level would produce a nominal test set of 63 images that still corresponds to only 27 distinct trees; the protocol used throughout this study reports the true unit of independent observation, the tree, explicitly, while also reporting the number of correlated images per split for

completeness (**Table 2**).

Representative paired RGB and thermal samples for all four classes are shown in **Figure 1**; thermal images are visualized using a false-color (pseudo-color) palette.

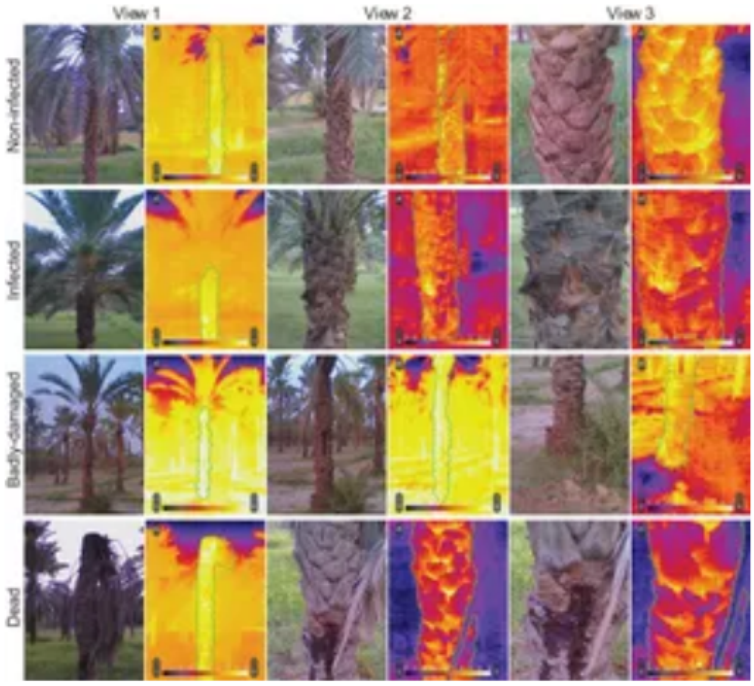


Figure 1: Representative paired RGB and thermal photographs from all four health classes

3.2 Preprocessing pipeline

Each RGB sample is processed through a sequential enhancement chain prior to network input: bilateral filtering for edge-preserving denoising, single-scale Retinex illumination normalization, gray-world white balancing, contrast-limited adaptive histogram equalization (CLAHE) applied in the L channel of the CIE-LAB color space, gamma correction ($\gamma = 1.15$), and unsharp masking. Each thermal sample is processed with a lighter chain consisting of an adaptive per-channel histogram stretch followed by CLAHE in LAB space, since thermal pseudo-color maps already encode a calibrated temperature-to-color mapping that aggressive photometric correction could distort. **Figure 2** shows the cumulative effect of each stage (original, bilateral filtering, Retinex normalization, white balance, CLAHE, gamma correction, unsharp masking) on representative RGB samples.

After photometric preprocessing, images are resized to 224×224 pixels. RGB training images receive geometric and photometric augmentation, including random resized cropping, horizontal and vertical flips, rotation up to $\pm 25°$, RandAugment, AugMix, TrivialAugment, color jitter, Gaussian blur, random erasing, and GridMask occlusion, together with a domain-specific morphological augmentation that synthesizes RPW-like symptoms, namely irregular brown trunk patches, vertical dark streaks, and crown sparsity masks, with probability 0.35, intended to encourage the network to learn symptom morphology rather than viewpoint-specific cues. Thermal training images receive only geometric augmentation (flips, rotation, resized cropping) to preserve the integrity of the temperature-encoded color mapping. Each modality is normalized separately prior to tensor conversion, as described next.

3.2.1 Thermal branch input and normalization

The thermal branch accepts each thermal file in its native three-channel, pseudo-colored (false-color) representation, identical in tensor shape to the RGB branch; the input channel count of the backbone is not modified. The two branches differ in how each modality is normalized prior to network input: RGB tensors are normalized with the standard ImageNet channel statistics (mean $[0.485, 0.456, 0.406]$, standard deviation $[0.229, 0.224, 0.225]$), consistent with the use of ImageNet-pretrained backbone weights for the RGB branch, while thermal tensors are normalized with a fixed mean and standard deviation of 0.5 per channel, mapping the pseudo-colored thermal pixel range to $[-1, 1]$ without assuming any natural-image color statistics. This choice reflects the fact that thermal pseudo-color images do not share the statistical properties of natural RGB photographs that motivate ImageNet normalization, while a fixed, symmetric normalization

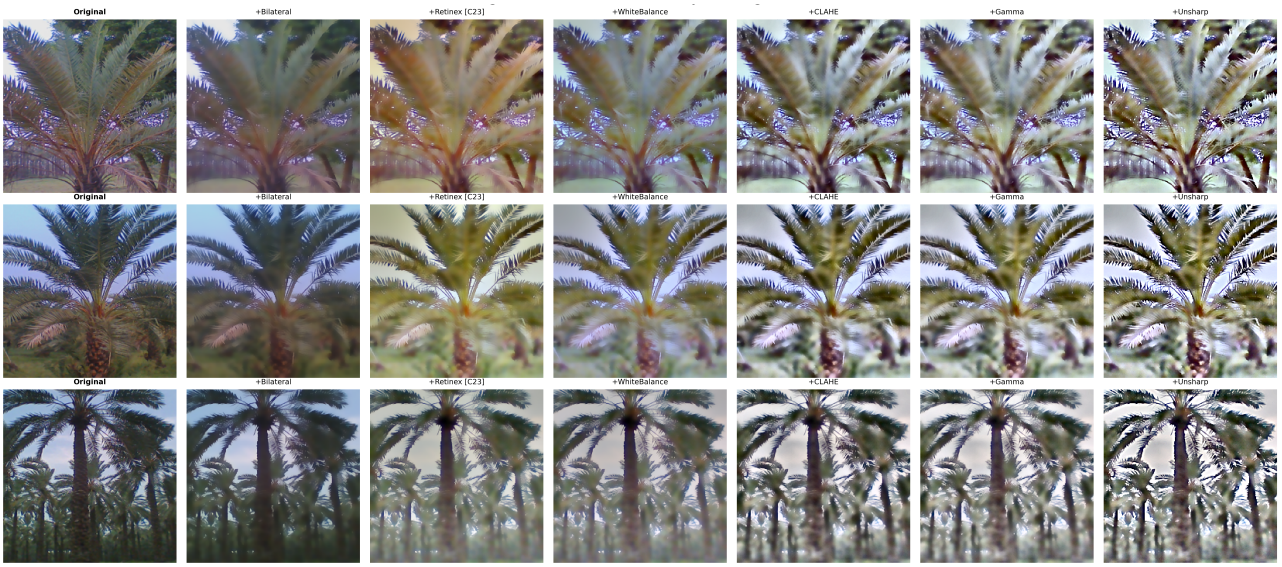


Figure 2: Cumulative effect of the RGB preprocessing chain on three representative training samples

avoids introducing an unjustified prior. The thermal backbone is initialized from the same ImageNet-pretrained weights as the RGB backbone purely as a convenient general-purpose feature initialization, not because the thermal pseudo-color images are treated as natural RGB photographs.

3.3 Class imbalance handling strategy

Given the class imbalance quantified in **Table 2** (an approximately 6.7:1 ratio between the majority non-infected class and the minority badly damaged class within the training split), five complementary mechanisms are combined, consistent with established practice for long-tailed visual recognition [12, 22].

First, per-class loss weights are computed from the effective number of samples [12], as given in **Equation 1**:

$$E_{n_c} = \frac{1-\beta^{n_c}}{1-\beta}, \qquad w_c = \frac{1/E_{n_c}}{\sum_{k=1}^{C} 1/E_{n_k}} \cdot C, \tag{1}$$

where n_c is the number of training trees in class c, $\beta = 0.9999$ is the effective-number hyperparameter, and $C = 4$ is the number of classes. With the training counts in **Table 2** ($n = 56, 22, 39, 8$ for non-infected, infected, dead, and badly damaged, respectively), this procedure assigns weights of 0.334, 0.850, 0.480, and 2.336 to the four classes, up-weighting the contribution of badly damaged samples to the loss by roughly a factor of seven relative to the majority class.

Second, these weights enter a class-balanced focal loss. With label smoothing parameter $\varepsilon = 0.1$, the smoothed soft target for true class y is given in **Equation 2**:

$$s_i = \begin{cases} 1-\varepsilon, & i = y, \\ \varepsilon/(C-1), & i \neq y, \end{cases} \tag{2}$$

and the per-sample class-balanced focal loss [13, 12] is then defined in **Equation 3**:

$$\mathcal{L}_{\mathrm{CBF}}(\mathbf{z}, y) = w_y \left(1 - p_t\right)^{\gamma} \mathrm{CE}(\mathbf{s}, \mathbf{z}), \qquad \mathrm{CE}(\mathbf{s}, \mathbf{z}) = -\sum_{i=1}^{C} s_i \log \mathrm{softmax}(\mathbf{z})_i, \qquad p_t = e^{-\mathrm{CE}(\mathbf{s},\mathbf{z})}, \tag{3}$$

with focusing parameter $\gamma = 2.0$, logits $\mathbf{z}$, and the per-class weight w_y indexed by the true (hard) label y. When a training sample is the product of mixup or CutMix blending two source labels y_a and y_b with mixing coefficient λ (**Equations 6** and **7** below), the class-balanced focal loss is computed once for each source label, using each label's own weight w_{y_a} or w_{y_b}, and combined as shown in **Equation 4**:

$$\mathcal{L}_{\mathrm{CBF}}^{\mathrm{mix}} = \lambda\, \mathcal{L}_{\mathrm{CBF}}(\mathbf{z}, y_a) + (1-\lambda)\, \mathcal{L}_{\mathrm{CBF}}(\mathbf{z}, y_b), \tag{4}$$

so that the class-balanced weighting is applied consistently to both constituent labels of a mixed sample rather than only to the dominant one; unmixed samples correspond to the special case $y_a = y_b = y$ and $\lambda = 1$.

Third, a weighted random sampler draws training batches with inverse class-frequency sampling probability, so that, in expectation, each class is seen with approximately equal frequency over an epoch regardless of its underlying count.

Fourth, mixup [14] and CutMix [15] partner selection is biased toward minority classes, and the per-class augmentation intensity is scaled upward for classes with fewer samples, from 1× to 4× relative augmentation strength, proportional to the inverse class frequency.

Fifth, an inference-time logit adjustment [22] is available as an optional, separately reported correction, given in **Equation 5**:

$$\mathbf{z}^{\text{adj}} = \mathbf{z} - \tau \cdot \log \hat{\boldsymbol{\pi}}, \tag{5}$$

where $\hat{\pi}_c$ is the empirical training class prior and $\tau = 1.0$. Logit-adjusted results are reported separately from the default-inference results and are not used to inflate the headline figures of this study.

No class is downsampled or discarded at any stage. Mixup and CutMix are formally defined as in **Equations 6** and **7**, respectively.

$$\tilde{\mathbf{x}}_{\text{rgb}} = \lambda\, \mathbf{x}^{(i)}_{\text{rgb}} + (1-\lambda)\, \mathbf{x}^{(j)}_{\text{rgb}}, \qquad \tilde{\mathbf{x}}_{\text{thm}} = \lambda\, \mathbf{x}^{(i)}_{\text{thm}} + (1-\lambda)\, \mathbf{x}^{(j)}_{\text{thm}}, \qquad \lambda \sim \text{Beta}(\alpha, \alpha), \tag{6}$$

$$\tilde{\mathbf{x}}_{\mathcal{B}} = \mathbf{x}^{(i)} \odot (1 - \mathbf{M}) + \mathbf{x}^{(j)} \odot \mathbf{M}, \qquad \lambda_{\text{adj}} = 1 - \frac{\text{area}(\mathbf{M})}{H \times W}, \tag{7}$$

where $\mathbf{M}$ is a randomly placed rectangular binary mask applied identically to both modalities, and λ_{adj} replaces λ as the mixing coefficient used in **Equation 4** for CutMix samples. The mixup interpolation parameter α follows a cosine curriculum, decaying from $\alpha = 1.0$ at the first epoch to $\alpha = 0.1$ at the final epoch, so that early training emphasizes strong regularization while later training emphasizes cleaner, less-blended samples. At each training step, mixing (mixup or CutMix, chosen with equal probability) is applied with overall probability 0.5; the remaining batches are trained on unmixed samples.

3.4 Network architecture

Figure 3 shows a schematic overview of the pipeline, from the paired RGB and thermal input through preprocessing, the dual-branch encoder, feature fusion, the classification head, and the evaluation protocol applied on the held-out test set. The diagram groups the per modality backbone family, the fusion stages, and the regularization and optimization mechanisms applied during training into one view; each block is described in full in the remainder of this section. One clarification on architectural scope is necessary here and is stated explicitly rather than left to the figure alone: the feature pyramid network fusion stage and its associated multi scale extraction at intermediate backbone depths, shown centrally in the diagram, are active only for the three backbones with intermediate hook layers registered in the implementation actually executed, namely ResNet 50, EfficientNet B0, and ConvNeXt Tiny, as detailed later in this subsection. For Xception, ViT Small, and ConvNeXt V2 Tiny, the pipeline proceeds directly from the dual branch encoder to cross modal attention and efficient channel attention without the intervening feature pyramid network and token mixing stages.

The classification network is a dual-branch architecture in which an RGB branch and a thermal branch independently extract features before late fusion and joint classification. Each branch is built from one of six pretrained backbones, ResNet-50 [29], EfficientNet-B0 [30], ConvNeXt-Tiny [31], Xception [32], ViT-Small/16 [33], or ConvNeXt V2-Tiny [34], with the earliest 50% of backbone stages frozen and the remainder fine-tuned at one-tenth of the classification-head learning rate. The two branches of a given run always share the same backbone family; results are reported separately for each of the six choices rather than mixing backbone families within a single fusion model.

The global feature vector from each branch is passed through a sigmoid-gated channel reweighting layer and an efficient channel attention (ECA) module [16]. ECA computes a one-dimensional convolution kernel size adaptively from the channel count C, as defined in **Equation 8**:

$$k(C) = \text{clip}\left(\left\lceil \frac{\log_2(C) + b}{\gamma} \right\rceil_{\text{odd}}, 3, 31\right), \qquad \gamma = 2,\ b = 1, \tag{8}$$

where the inner term is first truncated toward zero, then incremented by one if it is even, so that the resulting kernel size is odd, and the result is finally clipped to the range $[3, 31]$. This adaptive kernel is applied as a one-dimensional convolution across the channel dimension to obtain per-channel attention weights without channel dimensionality reduction. The gated RGB and thermal global features are then combined through a lightweight cross-modal attention layer in which the RGB feature attends to the thermal feature.

For backbones with explicitly registered intermediate hook layers, namely ResNet-50 (stages `layer2`, `layer3`, `layer4`; 512, 1024, 2048 channels), EfficientNet-B0 (`features.3`, `features.5`, `features.8`; 40, 112, 1280 channels), and

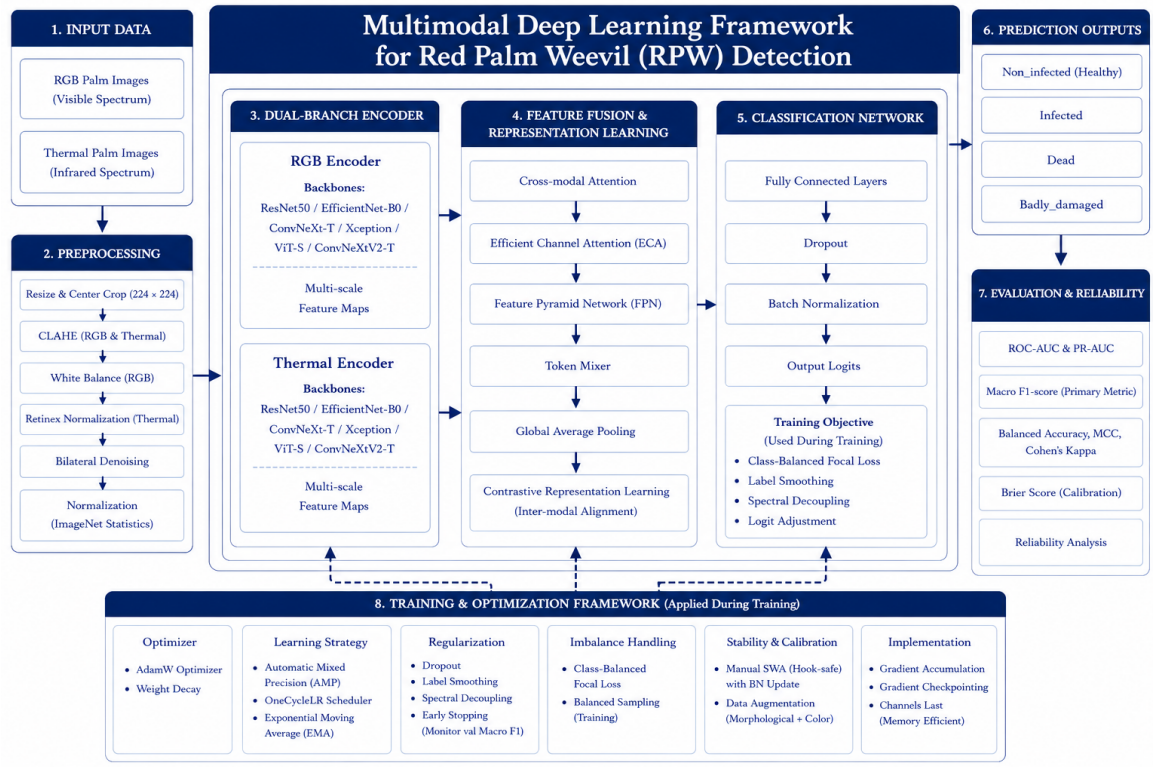


Figure 3: Schematic overview of the multimodal deep learning framework

ConvNeXt-Tiny (`features.3`, `features.5`, `features.7`; 192, 384, 768 channels), a hierarchical feature pyramid network (FPN) fusion neck additionally combines multi-scale features from both modalities. At each scale, a thermal-guided spatial attention module reweights the RGB feature map using a learned function of the co-located thermal feature map, as given in **Equation 9**:

$$\mathbf{A}_{\text{thm}} = \sigma\big(\text{Conv}_{1\times1}\big(\text{ReLU}\big(\text{BN}\big(\text{Conv}_{3\times3}(\mathbf{T})\big)\big)\big)\big), \qquad \mathbf{R}' = \mathbf{R} \odot \mathbf{A}_{\text{thm}}, \tag{9}$$

where $\mathbf{T}$ and $\mathbf{R}$ are the thermal and RGB feature maps at a given scale and $\sigma(\cdot)$ is the sigmoid function. Per-scale RGB and thermal feature vectors are pooled, projected to a common dimension, summed, optionally passed through a lightweight gated token-mixing block across the three scales, and combined with a learned, softmax-normalized scale-importance weighting before a final fusion projection. Xception, ViT-Small, and ConvNeXt V2-Tiny do not have intermediate hook layers registered in the implementation, so for these three backbones the FPN neck is inactive and the model relies on the global-feature cross-modal fusion path only. This asymmetry is a concrete architectural limitation of the present implementation rather than a deliberate ablation and is reported as such in Section 5.3.

The final feature representation, formed by concatenating the cross-modal-fused RGB feature, the FPN output where available, and the gated thermal feature, is passed through a two-layer multilayer perceptron classification head (512 and 256 hidden units, batch normalization, GELU activation, dropout rates of 0.5 and 0.3) to produce four-class logits. Two auxiliary objectives regularize the fusion model and are absent from the single-modality and naive-concatenation ablation baselines described in Section 4.3: a prototype-based contrastive loss, given in **Equation 10**:

$$\mathcal{L}_{\text{proto}} = -\log \frac{\exp(\mathbf{z}_i \cdot \mathbf{p}_{y_i}/\tau)}{\sum_{c=1}^{C} \exp(\mathbf{z}_i \cdot \mathbf{p}_c/\tau)}, \qquad \mathbf{p}_c \leftarrow \rho\, \mathbf{p}_c + (1-\rho)\, \bar{\mathbf{z}}_c, \tag{10}$$

where $\mathbf{z}_i$ is an ℓ_2-normalized projected feature, $\mathbf{p}_c$ is an exponential-moving-average class prototype updated with decay $\rho = 0.99$ after each batch, $\bar{\mathbf{z}}_c$ is the ℓ_2-normalized mean feature of class c within the current batch, and $\tau = 0.07$ [20, 21]; and a spectral decoupling regularizer that penalizes the squared error between a predicted and an observed low-frequency Fourier magnitude spectrum of the RGB input, intended to discourage reliance on high-frequency texture shortcuts. The total training objective is given in **Equation 11**:

$$\mathcal{L} = \mathcal{L}_{\text{CBF}} + 0.25\, \mathcal{L}_{\text{proto}} + 0.05\, \mathcal{L}_{\text{spectral}}. \tag{11}$$

3.5 Training procedure

Table 3 lists the training hyperparameters as configured and executed. All backbones are trained with the AdamW optimizer [19], using a separate learning rate for the classification head (1×10^{-4}) and for unfrozen backbone parameters (1×10^{-5}, one-tenth of the head rate). The default learning-rate schedule is the one-cycle policy [18], with a peak learning rate ten times the base rate reached after 10% of total optimization steps and a cosine anneal thereafter; a cosine-annealing-with-warm-restarts schedule is implemented as a configurable alternative but is not used for the headline results. Mixed-precision training, gradient clipping at a global norm of 1.0, and an exponential moving average (EMA) of model weights (decay 0.998) are used throughout. Gradient checkpointing is enabled for the three more memory-intensive backbones (ViT-Small, ConvNeXt-Tiny, ConvNeXt V2-Tiny) to reduce peak activation memory, and gradient accumulation is used so that every backbone trains at an equivalent effective batch size of 16 regardless of its per-step physical batch size.

Table 3: Training hyperparameters as configured in the executed implementation

Hyperparameter	Value
Optimizer	AdamW
Head learning rate	1×10^{-4}
Backbone learning rate	1×10^{-5} (0.1× head rate)
Learning-rate schedule	One-cycle (cosine anneal, 10% warmup); warm restarts available
Weight decay	1×10^{-4}
Effective batch size	16 (per-backbone physical batch size 8–16, via gradient accumulation)
Maximum epochs	90
Early-stopping criterion	Validation macro-F1, patience 25 epochs
Image input size	224×224
Mixed precision	Enabled
Exponential moving average decay	0.998
Backbone frozen fraction	50% of early stages
Gradient checkpointing	ViT-Small, ConvNeXt-Tiny, ConvNeXt V2-Tiny
Focal loss focusing parameter γ	2.0
Class-balanced loss β	0.9999
Label smoothing ε	0.1
Mixup/CutMix α (curriculum)	$1.0 \rightarrow 0.1$ (cosine)
Per-batch mixing probability	0.5
Contrastive loss weight / temperature	0.25 / 0.07
Spectral regularization weight	0.05
Stochastic weight averaging tail	8 epochs at 1×10^{-5}, batch-norm re-estimated over 50 batches
Continued-training stage	20 epochs at 0.1× head rate, pseudo-label confidence threshold 0.95
Random seed (primary run)	42
Stability-analysis seeds	42, 123, 2024
Bootstrap resamples	2000, 95% confidence interval

A stochastic weight averaging (SWA) tail phase [17] follows early stopping. After restoring the best validation-macro-F1 checkpoint and comparing it against the EMA shadow weights on the validation set, retaining whichever scores higher, training continues for 8 additional epochs at a fixed learning rate of 1×10^{-5}. The weights produced at each of these epochs are accumulated into a running average over model state dictionaries, as defined in **Equation 12**:

$$\boldsymbol{\theta}_{\mathrm{SWA}}^{(n)} = \boldsymbol{\theta}_{\mathrm{SWA}}^{(n-1)} \cdot \frac{n-1}{n} + \boldsymbol{\theta}_{\mathrm{epoch}} \cdot \frac{1}{n}, \qquad n = 1, \ldots, 8, \tag{12}$$

where $\boldsymbol{\theta}_{\mathrm{SWA}}^{(0)}$ is initialized from the pre-tail weights. The average is accumulated directly over the model's own state dictionary, in place, rather than through a deep-copied auxiliary model, because the multi-scale feature extractor registers forward hooks whose closures capture a reference to a specific Python dictionary object; deep-copying the model would silently detach those hooks from the copy and disable the FPN neck during SWA evaluation. After the SWA tail, batch-normalization running statistics are re-estimated over 50 training batches using both modalities, and the SWA-averaged weights are adopted as the final model only if their validation macro-F1 matches or exceeds the pre-SWA validation macro-F1; otherwise, the pre-SWA checkpoint is restored.

Algorithm 1 summarizes the per-backbone training procedure as executed.

Algorithm 1 Dual-branch RGB–thermal fusion training with class-balanced curriculum and stochastic weight averaging

Require: Tree-level dataset $\mathcal{D}$, backbone family $\mathcal{B}$, maximum epochs $E_{\max} = 90$, patience $P = 25$

Split $\mathcal{D}$ into $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{val}}, \mathcal{D}_{\text{test}}$ by tree identifier, stratified by class (70/15/15); verify an empty pairwise tree-identifier intersection
Compute class-balanced weights w_c from **Equation 1**
Build dual-branch model $M_{\mathcal{B}}$; freeze the earliest 50% of backbone stages in both branches
Initialize AdamW with head and backbone learning rates; initialize the one-cycle scheduler and the EMA shadow weights
$n_{\text{patience}} \leftarrow 0$, $F1^{*}_{\text{val}} \leftarrow 0$
for $e = 1$ **to** $E_{\max}$ **do**
    for each mini-batch $(\mathbf{x}_{\text{rgb}}, \mathbf{x}_{\text{thm}}, y)$ drawn from $\mathcal{D}_{\text{train}}$ via inverse-frequency sampling **do**
        Apply the RGB and thermal preprocessing chains (Section 3.2)
        Apply augmentation (RandAugment, AugMix, GridMask, morphological symptom synthesis)
        With probability 0.5, apply mixup (**Equation 6**) or CutMix (**Equation 7**), chosen with equal probability, using curriculum $\alpha(e)$ and minority-biased partner sampling
        $\mathbf{z}, \text{aux} \leftarrow M_{\mathcal{B}}(\mathbf{x}_{\text{rgb}}, \mathbf{x}_{\text{thm}})$
        Compute $\mathcal{L}$ from **Equation 11**; backpropagate; clip the global gradient norm to 1.0
        Update the optimizer, the scheduler, and the EMA weights
    end for
    Evaluate $M_{\mathcal{B}}$ on $\mathcal{D}_{\text{val}}$; compute the validation macro-F1, $F1_{\text{val}}$
    if $F1_{\text{val}} > F1^{*}_{\text{val}}$ **then**
        $F1^{*}_{\text{val}} \leftarrow F1_{\text{val}}$; checkpoint $M_{\mathcal{B}}$; $n_{\text{patience}} \leftarrow 0$
    else
        $n_{\text{patience}} \leftarrow n_{\text{patience}} + 1$
    end if
    if $n_{\text{patience}} \geq P$ **then break**
    end if
end for
Restore the best checkpoint; evaluate the EMA shadow weights on $\mathcal{D}_{\text{val}}$; retain whichever of the two scores higher on validation macro-F1
SWA tail: for 8 epochs at learning rate 1×10^{-5}, continue training and accumulate $\boldsymbol{\theta}_{\text{SWA}}$ via **Equation 12**, updating a state-dictionary average rather than a deep-copied model
Load $\boldsymbol{\theta}_{\text{SWA}}$ into $M_{\mathcal{B}}$; re-estimate batch-normalization statistics over 50 training batches by calling $M_{\mathcal{B}}(\mathbf{x}_{\text{rgb}}, \mathbf{x}_{\text{thm}})$
Evaluate $M_{\mathcal{B}}$ with $\boldsymbol{\theta}_{\text{SWA}}$ on $\mathcal{D}_{\text{val}}$; adopt $\boldsymbol{\theta}_{\text{SWA}}$ only if its validation macro-F1 is at least the pre-SWA best; otherwise restore the pre-SWA checkpoint
return the final model $M_{\mathcal{B}}$

3.6 Continued-training stage

A two-pass self-training stage was configured to fine-tune all six trained backbones using high-confidence pseudo-labels on the training set, with a confidence acceptance threshold of 0.95. In the executed run, zero of 294 training samples reached this threshold under the ensemble-averaged softmax confidence of the six trained models, so no pseudo-labels were accepted. The fine-tuning loader was nonetheless constructed from the union of the training and validation trees, using the original, farmer-assigned labels rather than any pseudo-label, since none were accepted, and each backbone was further trained for 20 epochs at a reduced learning rate. This stage is consequently better characterized as a continued-training experiment on an expanded train-plus-validation set than as genuine pseudo-label self-training, and its effect on each backbone was mixed: it improved some backbones, most notably ConvNeXt-Tiny, whose test macro-F1 rose from 0.581 to 0.764, while degrading others, most notably ResNet-50, whose test macro-F1 fell from 0.707 to 0.648, as reported in **Table 5**. Because the validation split was included in this fine-tuning loader, the post-continued-training checkpoints can no longer be regarded as having been selected using a strictly held-out validation set, and this limitation is carried forward explicitly into the results and discussion sections.

3.7 Evaluation protocol

Test-time prediction uses four-view test-time augmentation, namely identity, horizontal flip, vertical flip, and 90° rotation, with softmax-probability averaging across views. Reported metrics are accuracy, balanced accuracy, macro-averaged F1, weighted F1, Cohen's kappa, the Matthews correlation coefficient, the geometric mean of per-class recall, macro-averaged one-versus-rest area under the receiver operating characteristic curve, mean area under the precision-recall curve, and the

mean Brier score. Ninety-five percent confidence intervals are obtained by percentile bootstrap resampling of the test set with 2000 resamples. Pairwise statistical comparison between models uses the McNemar test [28] on paired correct and incorrect outcomes, applying a continuity-corrected chi-square statistic when the discordant-pair count $n_{01} + n_{10} \geq 25$ and an exact binomial test otherwise, and the DeLong test [27] for paired, per-class one-versus-rest area-under-the-curve comparison. A multi-seed stability analysis repeats training and evaluation under three random seeds (42, 123, 2024) for a selected backbone to characterize the sensitivity of test performance to initialization and data-ordering randomness, which is of particular relevance given the small test-set size.

4 RESULTS

4.1 Individual backbone performance

Table 4 reports test-set performance with 95% bootstrap confidence intervals for each of the six backbones, computed after the continued-training stage described in Section 3.6, matching the values shown in the corresponding figures. ConvNeXt-Tiny obtains the highest point-estimate accuracy (0.746) and macro-F1 (0.764) among individual backbones; ViT-Small and ResNet-50 follow, while Xception and ConvNeXt V2-Tiny are the weakest of the six backbones on this dataset. Confidence intervals are wide relative to the differences between point estimates, a direct consequence of the small test set of 27 trees (63 correlated images); this point is discussed further in Section 5.

Table 4: Test-set performance with 95% bootstrap confidence intervals (2000 resamples), computed after the continued-training stage

Model	Accuracy	Balanced Accuracy	Macro F1	Macro AUC	AURPC
ResNet-50	0.667 [0.556, 0.778]	0.726 [0.629, 0.822]	0.648 [0.520, 0.764]	0.902 [0.851, 0.950]	0.818 [0.726, 0.910]
EfficientNet-B0	0.651 [0.540, 0.762]	0.721 [0.623, 0.815]	0.631 [0.501, 0.744]	0.927 [0.881, 0.967]	0.844 [0.753, 0.930]
ConvNeXt-Tiny	**0.746** [0.635, 0.841]	0.772 [0.678, 0.862]	**0.764** [0.638, 0.855]	0.909 [0.858, 0.952]	0.776 [0.670, 0.894]
Xception	0.619 [0.508, 0.746]	0.694 [0.591, 0.793]	0.603 [0.476, 0.725]	0.916 [0.874, 0.955]	0.810 [0.703, 0.901]
ViT-Small	0.714 [0.603, 0.825]	0.747 [0.647, 0.843]	0.689 [0.554, 0.808]	0.888 [0.835, 0.937]	0.750 [0.643, 0.863]
ConvNeXt V2-Tiny	0.587 [0.476, 0.698]	0.678 [0.582, 0.772]	0.580 [0.462, 0.695]	0.897 [0.846, 0.943]	0.695 [0.601, 0.853]
Soft-vote ensemble	0.762 [0.667, 0.857]	0.791 [0.698, 0.882]	0.749 [0.614, 0.853]	**0.931** [0.891, 0.968]	**0.868** [0.795, 0.940]
F1-weighted ensemble	0.762 [0.667, 0.857]	0.791 [0.698, 0.882]	0.749 [0.614, 0.853]	0.930 [0.889, 0.968]	0.868 [0.796, 0.940]
Calibration-weighted ensemble	**0.762** [0.651, 0.857]	**0.794** [0.698, 0.883]	0.756 [0.621, 0.858]	0.928 [0.888, 0.966]	0.861 [0.787, 0.934]
Stacking (cross-validated meta-learner)	0.730 [0.619, 0.841]	0.669 [0.511, 0.811]	0.704 [0.511, 0.824]	0.906 [0.850, 0.955]	0.818 [0.706, 0.916]

Figure 4 shows training and validation loss, accuracy, and macro-F1 across epochs for every backbone, addressing the requirement to monitor for overfitting or underfitting. All six backbones show training accuracy continuing to rise after validation accuracy has plateaued, the expected signature of mild overfitting on a dataset of this size; ConvNeXt-Tiny and Xception show the largest train-validation accuracy gap by the end of training, with training accuracy approaching 0.90 to 0.93 against a validation accuracy plateau near 0.75 to 0.80, while EfficientNet-B0 and ResNet-50 show a comparatively smaller gap. This pattern is the basis for using early stopping on validation macro-F1 with a patience of 25 epochs (**Table 3**) rather than training to a fixed epoch budget.

Per-class confusion matrices (**Figure 5**) and receiver operating characteristic curves (**Figure 6**) are reported for every backbone. Across backbones, the dead and badly damaged classes are generally recognized with the highest per-class area under the curve, frequently above 0.93, while the infected class, which represents the earliest and most subtle stage of visible degradation, is the class most frequently confused with non-infected; this pattern is consistent across backbones and is the primary practical limitation of the classifier for genuinely early-stage detection, a point returned to in Section 5.

4.2 Effect of the continued-training stage

Table 5 reports test accuracy and macro-F1 immediately after the primary training run described in Section 3.5 and after the continued-training stage of Section 3.6, for every backbone. As discussed in Section 3.6, the effect is not uniformly beneficial: ConvNeXt-Tiny and ViT-Small improve, ResNet-50, EfficientNet-B0, and ConvNeXt V2-Tiny lose accuracy and macro-F1, and Xception improves modestly.

4.3 Ablation study

To isolate the contribution of multimodal fusion, three baselines were trained under the identical protocol of Section 3.5 using the ResNet-50 backbone: an RGB-only classifier, a thermal-only classifier, and a naive late-concatenation fusion classifier that omits the FPN neck, the cross-modal attention layer, and the auxiliary contrastive and spectral losses of the

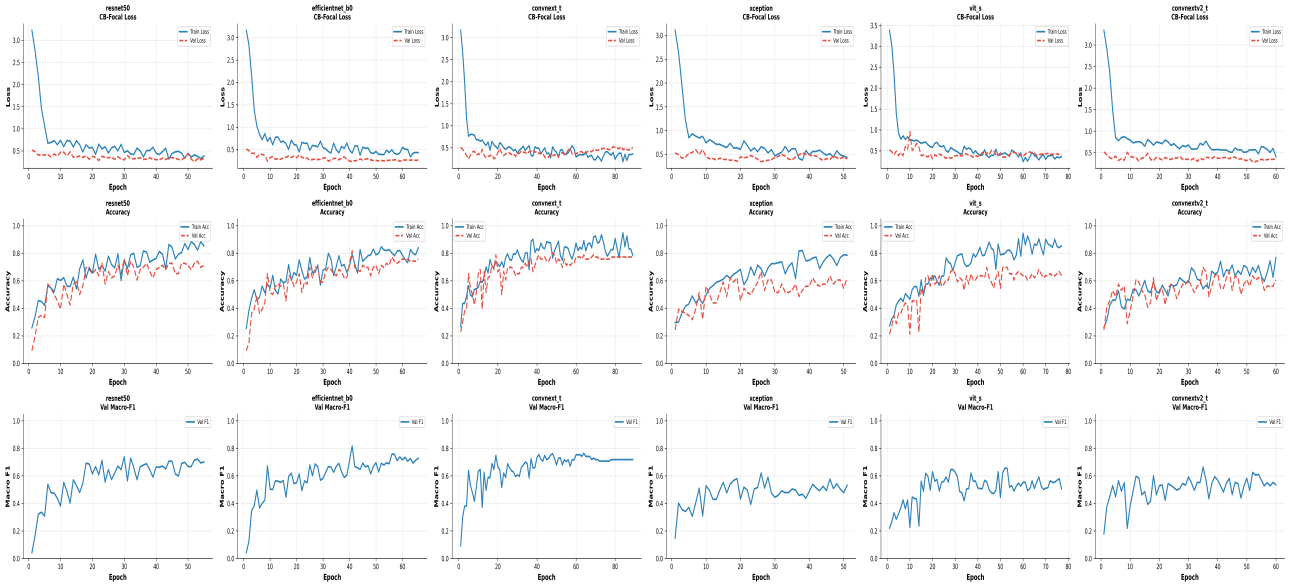
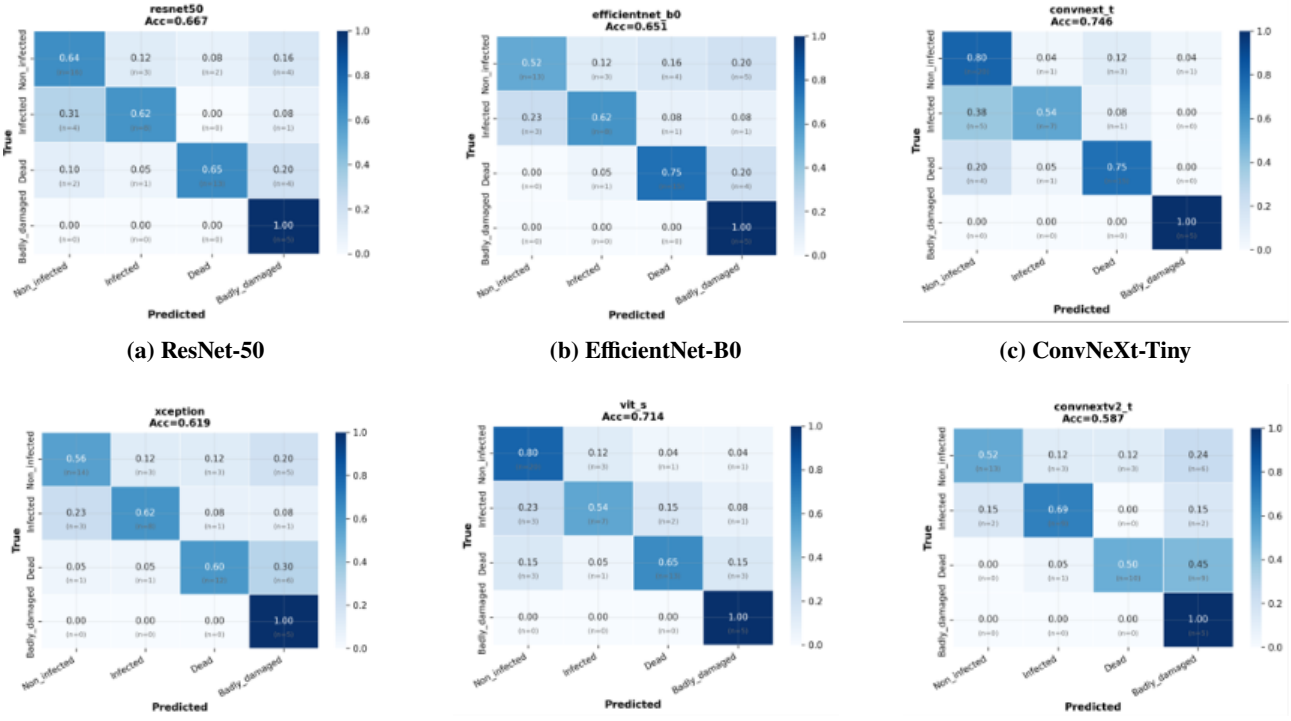


Figure 4: Training and validation class-balanced focal loss, accuracy, and validation macro-F1 across epochs for each of the six backbones, prior to the continued-training stage



(a) ResNet-50 **(b) EfficientNet-B0** **(c) ConvNeXt-Tiny**

(d) Xception **(e) ViT-Small** **(f) ConvNeXt V2-Tiny**

Figure 5: Row-normalized confusion matrices on the 63-image, 27-tree test set for each backbone, computed after the continued-training stage

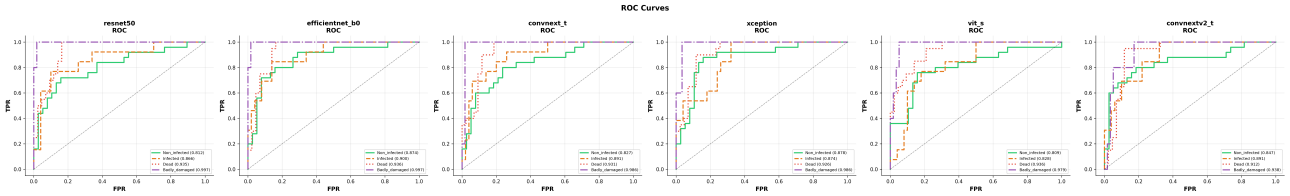


Figure 6: Per-class one-versus-rest receiver operating characteristic curves for each backbone, computed after the continued-training stage

Table 5: Effect of the continued-training stage of Section 3.6 on test accuracy and macro-F1, before and after

Backbone	Accuracy (before)	Accuracy (after)	Macro F1 (before)	Macro F1 (after)
ResNet-50	0.730	0.667	0.707	0.648
EfficientNet-B0	0.714	0.651	0.661	0.631
ConvNeXt-Tiny	0.667	0.746	0.581	0.764
Xception	0.571	0.619	0.492	0.603
ViT-Small	0.587	0.714	0.523	0.689
ConvNeXt V2-Tiny	0.556	0.587	0.541	0.580

proposed model. **Table 6** reports the result. The RGB-only baseline reaches the highest accuracy, 0.762, of any single model reported in this study, including every proposed fusion backbone, though the calibration-weighted ensemble of all six fusion backbones (**Table 4**) matches it. The thermal-only model is markedly weaker than either the RGB-only model or the full fusion model on every metric, while the naive-concatenation fusion model improves over thermal-only but does not match RGB-only on accuracy or macro-F1, despite having access to both modalities. This pattern indicates that, on a dataset of this size, simply concatenating modality features does not reliably extract the complementary information that the more structured fusion mechanisms of Section 3.4 are designed to capture, and that the gains attributable specifically to multimodal fusion, as opposed to model capacity or augmentation strength, are modest and architecture-dependent rather than uniformly positive.

Table 6: Ablation comparison of single-modality and naive-fusion baselines (ResNet-50 backbone, identical training protocol) on the test set

Variant	Accuracy	Balanced Accuracy	Macro F1	Macro AUC	AURPC
RGB only	**0.762**	**0.784**	**0.736**	0.897	0.787
Thermal only	0.619	0.668	0.603	0.842	0.630
Naive concatenation	0.714	0.700	0.690	**0.916**	**0.855**

4.4 Ensembling

Four ensemble strategies were evaluated by combining the post-continued-training softmax probabilities of all six backbones: unweighted soft voting, validation-macro-F1-weighted voting, inverse-expected-calibration-error-weighted voting, and a logistic-regression stacking meta-learner trained with five-fold stratified cross-validation on validation-set probabilities to avoid leaking validation labels directly into the meta-learner fit. **Table 4** reports the result for each strategy. The three voting-based ensembles, soft-vote, F1-weighted, and calibration-weighted, perform similarly to one another and improve test accuracy to 0.762, exceeding every individual backbone except for the macro-F1 of ConvNeXt-Tiny alone. The calibration-weighted ensemble obtains the highest balanced accuracy, 0.794, among all reported models. The stacking ensemble underperforms the voting ensembles on this dataset, most likely a consequence of fitting a meta-learner with limited validation data, 27 trees and 66 images. **Figure 7** reports the per-backbone influence on the stacking meta-learner, in which ConvNeXt-Tiny and EfficientNet-B0 receive the largest absolute coefficient weight, consistent with their relatively strong individual performance in **Table 4**.

Pairwise McNemar testing across all ten models in **Table 4**, 45 comparisons in total, found six statistically significant pairs at $\alpha = 0.05$: ConvNeXt V2-Tiny against each of the three voting ensembles ($p = 0.0074$ in each case) and Xception against the calibration-weighted ensemble ($p = 0.0225$) and against the soft-vote and F1-weighted ensembles ($p = 0.0352$ in each case). No pairwise comparison among ResNet-50, EfficientNet-B0, ConvNeXt-Tiny, ViT-Small, and the ensembles reached significance, nor did any comparison among the four ensemble strategies themselves. This pattern indicates that, at the present sample size, ensembling provides a statistically detectable improvement specifically over the two weakest individual backbones, while the apparent numerical advantage of ensembling over the stronger individual backbones, ResNet-50, EfficientNet-B0, ConvNeXt-Tiny, and ViT-Small, is not statistically distinguishable from chance variation given 27 test trees.

Pairwise per-class DeLong testing of area-under-the-curve differences, 288 comparisons across all model pairs and four classes, produced a similarly conservative result: of all comparisons, only ViT-Small against each of the three voting ensembles on the infected class reached significance (soft-vote, $p = 0.0054$; F1-weighted, $p = 0.0090$; calibration-weighted, $p = 0.0111$), indicating that ensembling significantly improves the ability to discriminate the infected class specifically relative to the weaker-performing ViT-Small backbone, the only class-level comparison for which this holds. No other pairwise area-under-the-curve comparison, including comparisons between any two of the stronger backbones or between

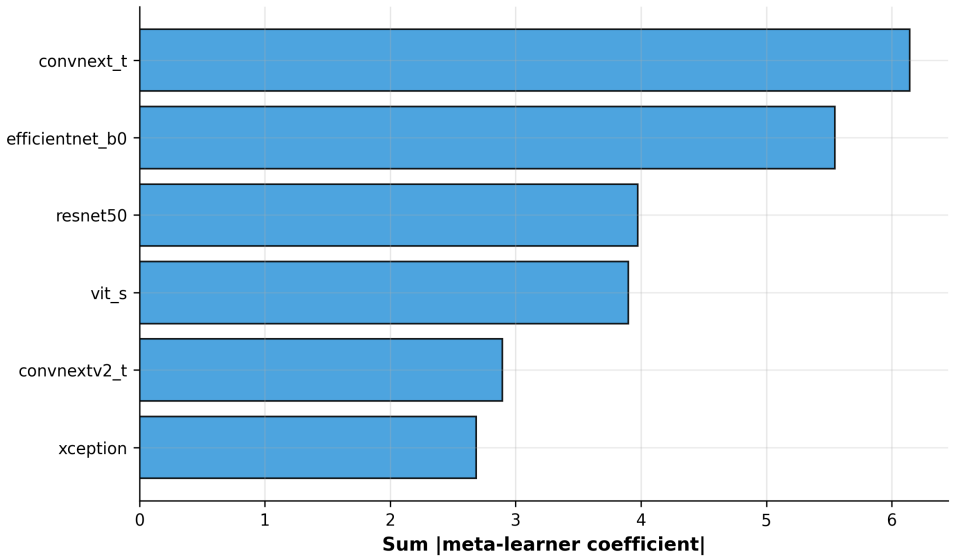


Figure 7: Sum of absolute stacking meta-learner coefficients attributed to each backbone, summed across the four output classes

the ensembles and ResNet-50, EfficientNet-B0, or ConvNeXt-Tiny, reached significance.

4.5 Calibration analysis

Figure 8 reports reliability diagrams and expected calibration error for each backbone. ConvNeXt-Tiny shows the lowest expected calibration error, 0.080, among the six backbones, consistent with its leading point-estimate accuracy and macro-F1 in **Table 4**, while ViT-Small (0.198) and ConvNeXt V2-Tiny (0.173) are the least well calibrated. All backbones show some degree of underconfidence at low predicted-confidence bins and overconfidence at high predicted-confidence bins, the characteristic reliability-diagram pattern for models trained with strong regularization, including label smoothing, mixup, and CutMix, on a small dataset.

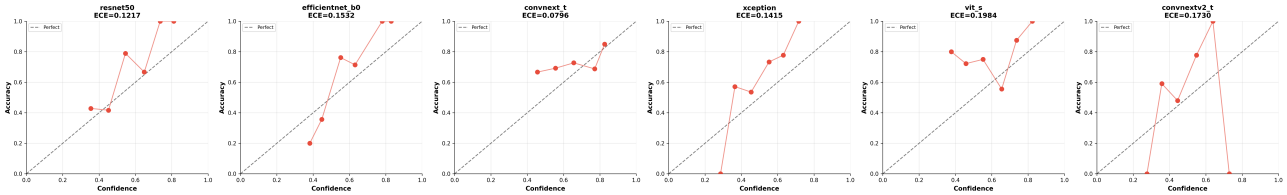


Figure 8: Reliability diagrams with expected calibration error for each backbone, computed on the test set after the continued-training stage

4.6 Multi-seed stability analysis

A multi-seed stability analysis (three seeds, 40 epochs per run, identical tree-level split) was completed for the ResNet-50 backbone within the available computation budget for this study; the configured analysis additionally specified Xception and ConvNeXt V2-Tiny, but the corresponding runs were not captured in the available execution record and are reported as outstanding future verification rather than omitted intentionally. For ResNet-50, test accuracy across the three seeds was 0.651, 0.619, and 0.635 (mean 0.635 ± 0.013), and test macro-F1 was 0.601, 0.577, and 0.571 (mean 0.583 ± 0.013). The standard deviation of approximately 1.3 percentage points across random seeds, on a fixed data split, indicates that initialization and data-ordering randomness alone account for a meaningful share of the run-to-run variability observed for individual backbones in **Table 4**.

4.7 Embedding visualization

Figure 9 shows a t-distributed stochastic neighbor embedding (t-SNE) projection of the fused feature representation produced by the ConvNeXt-Tiny model on the test set. The dead class forms a visually distinct, well-separated cluster,

consistent with its high per-class area under the curve in **Figure 6**, while the infected class shows partial overlap with the non-infected class, consistent with the confusion pattern noted above and with the practical difficulty of distinguishing early-stage infestation from a healthy tree using external visual and thermal cues alone.

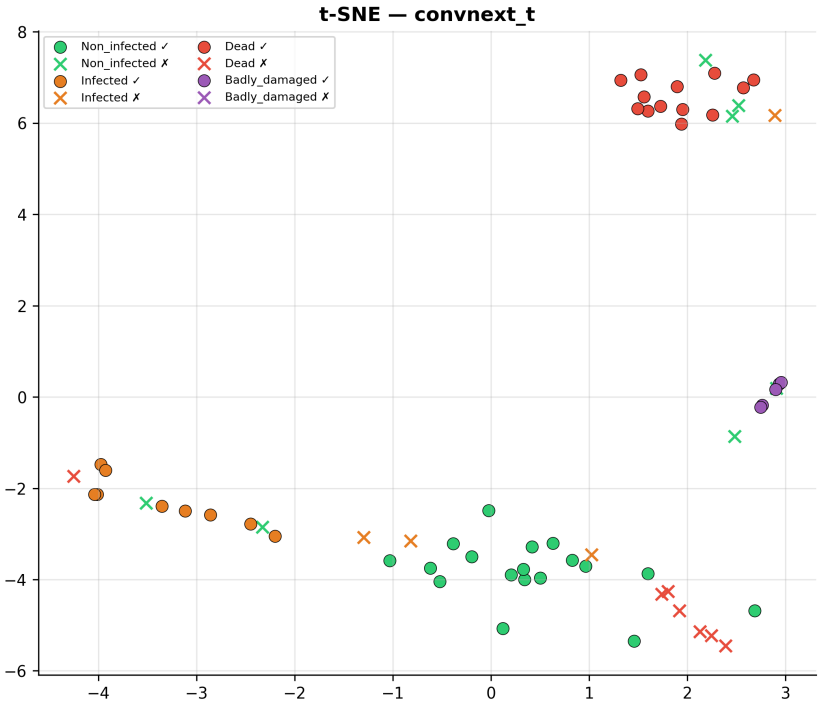


Figure 9: t-SNE projection of the fused test-set feature representation for the ConvNeXt-Tiny model

5 DISCUSSION

5.1 Summary of findings

Across the classification task, fusion of RGB and thermal imagery and ensembling of multiple backbones provide measurable improvements over the weakest individual configurations, but the improvement over the strongest individual backbone, and over a simple RGB-only baseline, is small and, in most pairwise comparisons, not statistically distinguishable from chance variation at the available sample size. ConvNeXt-Tiny is the strongest single backbone on macro-F1 and the best-calibrated model overall (Section 4), while the calibration-weighted ensemble of all six backbones gives the best balanced accuracy and the most consistent gain over the weaker backbones. The continued-training stage intended as pseudo-label self-training (Section 3.6) accepted zero pseudo-labels at the configured 0.95 confidence threshold and instead functioned as additional training on an expanded train-plus-validation set, with an inconsistent effect across backbones; this stage should be interpreted cautiously and is not the basis for any claim of fusion superiority made in this manuscript.

5.2 Comparison with recently published RPW detection studies

Several recently published studies report substantially higher accuracy figures for RPW-related classification than those obtained here, generally in the 93% to 100% range [8, 9, 10], as summarized in **Table 1**. These figures are not directly comparable to the results of this study for several reasons made explicit here rather than left to the reader to infer. First, several of these studies address a binary detection task, infested versus healthy or insect-presence versus absence, rather than the four-class health-severity task addressed here, and binary classification is intrinsically less demanding than four-way discrimination among classes that lie on a disease-progression continuum. Second, several use image-level rather than tree-level splitting, which, as discussed in Section 3.1, permits correlated views of the same specimen to appear in both training and test data and can substantially inflate reported accuracy relative to a leakage-free protocol. Third, dataset sizes and acquisition protocols differ considerably across studies, and none of the cited comparison studies reports bootstrap confidence intervals or tree-level statistical independence of the test set, which makes a literal numerical ranking across studies an unreliable basis for claiming superiority in either direction. The results reported in **Table 4** and **Table 6** are the figures actually obtained under an explicitly leakage-free, tree-level, multi-backbone, bootstrap-validated protocol and are intended to be read alongside this caveat rather than as a claim of outperforming the cited prior work.

5.3 Limitations

Several limitations constrain the generality of the conclusions drawn from this study.

Dataset size and class support. The test set comprises 27 trees, or 63 correlated images, with only two trees in the badly damaged class. Per-class metrics for this class, and any metric sensitive to it, including macro-averaged statistics and the geometric mean of per-class recall, should be interpreted as having very wide true uncertainty even where the reported point estimate appears favorable; the bootstrap confidence intervals in **Table 4** partially, but not fully, communicate this, since bootstrap resampling of a two-tree class cannot recover information that the underlying sample does not contain.

Correlated images within trees. Up to four images per tree per modality are used; although splitting is performed at the tree level to prevent leakage across splits, the 63 test images are not 63 independent observations, since multiple images of the same tree share label noise, lighting conditions, and acquisition-day artifacts. All reported test statistics should be understood as referring to 27 independent specimens, not 63 independent images.

Architectural inconsistency in the FPN fusion neck. As noted in Section 3.4, the hierarchical multi-scale fusion neck is active for three of the six backbones (ResNet-50, EfficientNet-B0, ConvNeXt-Tiny) and inactive for the remaining three (Xception, ViT-Small, ConvNeXt V2-Tiny), because intermediate feature hooks were registered only for the first group in the implementation actually executed. The six backbones are therefore not compared under a perfectly identical fusion mechanism, and part of the performance spread reported in **Table 4** may be attributable to this architectural inconsistency rather than to backbone capacity alone.

Continued-training stage. As discussed in Section 3.6, the intended pseudo-label self-training stage accepted no pseudo-labels and instead trained further on an expanded set that included the validation split, compromising the strict separation between model selection and the data used to train the reported checkpoint for that stage. Results in **Table 5** are reported transparently for this reason.

Stability analysis coverage. The multi-seed stability analysis (Section 4.6) was completed for one of the three originally configured backbones within the available computation budget; extending it to Xception and ConvNeXt V2-Tiny, and to the proposed ensembles, is identified as near-term future work rather than presented here.

Generalization beyond the source region. The classification dataset [11] originates from a single growing region; cross-site, cross-cultivar, and cross-season generalization have not been evaluated, and performance under different palm varieties, illumination conditions, sensor models, or geographic regions remains untested.

Maturity for operational deployment. Given the points above, in particular the small and class-imbalanced test set and the architectural inconsistency across backbones, the results of this study support a promising and methodologically transparent preliminary framework rather than a system ready for unsupervised operational deployment in pest-management decision-making. Treatment or removal decisions based on the output of this classifier should retain expert verification until a substantially larger, multi-site, externally validated dataset is available.

6 CONCLUSION

This study presents a dual-branch RGB-thermal fusion framework, evaluated across six backbone architectures under a leakage-free, tree-level split with bootstrap confidence intervals and paired statistical testing, for four-class date palm health classification. ConvNeXt-Tiny is the strongest individual backbone, reaching 74.6% accuracy and a macro-F1 of 0.764, and a calibration-weighted ensemble of all six backbones gives the best balanced accuracy, 79.4%, and an accuracy of 76.2%, although the improvement of ensembling over the strongest individual backbones is not statistically significant at the available sample size, while the improvement over the two weakest backbones is. An ablation against single-modality baselines shows that the gains attributable specifically to multimodal fusion, as distinct from model capacity or augmentation strength, are modest and architecture-dependent: an RGB-only classifier matches the best fusion ensemble on accuracy, while the full fusion model is required to surpass it on macro-F1 and balanced accuracy. The limitations enumerated in Section 5.3, most notably dataset size, class imbalance, and the architectural asymmetry of the fusion neck across backbones, indicate that substantial further data collection and external validation are required before this classifier can be considered ready for unsupervised operational use. Priority directions for future work are extending the multi-seed stability analysis to all evaluated backbones and to the ensemble strategies, registering intermediate feature hooks consistently across all six backbones so that the multi-scale fusion neck is evaluated under a uniform architecture, and acquiring a larger, multi-site RGB-thermal dataset with improved representation of the badly damaged class to narrow the wide confidence intervals reported in this study.

AUTHOR CONTRIBUTION STATEMENT

ll authors contributed equally to the study conception and design. Material preparation, data collection, and analysis were performed by the authors. The first draft of the manuscript was written by the authors, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

ETHICS APPROVAL AND CONSENT TO PARTICIPATE

This study did not involve human participants or animals. Ethical approval and consent to participate are therefore not applicable.

CONSENT FOR PUBLICATION

Not applicable.

DATA AVAILABILITY

The datasets generated during the current study are available at `https://figshare.com/articles/dataset/A_Dataset_of_Date_Palm_Trees_Thermal_and_RGB_Images_for_Pest_Management_/25974295?file=53580344`.

ACKNOWLEDGMENTS

The authors thank the Reviewers, Associate Editor, and Editor-in-Chief for their constructive comments and suggestions, which substantially improved the manuscript. The authors also acknowledge the use of DeepSeek for assistance in improving the clarity of the English language.

FUNDING

This research received no external funding.

DISCLOSURE STATEMENT

The authors declare that they have no competing interests.

REFERENCES

[1] W. Wakil, J. R. Faleiro, and T. A. Miller, “Sustainable pest management in date palm: current status and emerging challenges,” tech. rep., Springer, 2015.

[2] H. Naveed, V. Andoh, W. Islam, L. Chen, and K. Chen, “Sustainable pest management in date palm ecosystems: Unveiling the ecological dynamics of red palm weevil (coleoptera: Curculionidae) infestations,” *Insects*, vol. 14, no. 11, p. 859, 2023.

[3] W. Boulila, A. Alzahem, A. Koubaa, B. Benjdira, and A. Ammar, “Early detection of red palm weevil infestations using deep learning classification of acoustic signals,” *Computers and Electronics in Agriculture*, vol. 212, p. 108154, 2023.

[4] I. Ashry, B. Wang, Y. Mao, M. Sait, Y. Guo, Y. Al-Fehaid, A. Al-Shawaf, T. K. Ng, and B. S. Ooi, “Cnn–aided optical fiber distributed acoustic sensing for early detection of red palm weevil: A field experiment,” *Sensors*, vol. 22, no. 17, p. 6491, 2022.

[5] S. Delalieux, T. Hardy, M. Ferry, S. Gomez, L. Kooistra, M. Culman, and L. Tits, "Red palm weevil detection in date palm using temporal uav imagery," *Remote Sensing*, vol. 15, no. 5, p. 1380, 2023.

[6] D. Kagan, G. F. Alpert, and M. Fire, "Automatic large scale detection of red palm weevil infestation using street view images," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 182, pp. 122–133, 2021.

[7] M. Ashraf, M. Z. Aslam, N. Saeed, and S. J. Hussain, "Palmnext: a convnext-based deep learning model for pest detection in date palm leaves," *Frontiers in Plant Science*, vol. 16, p. 1738129, 2026.

[8] G. I. Sayed, S. Ibrahim, and A. E. Hassanien, "Early detection of red palm weevil in agricultural environment using deep learning," *Optical Memory and Neural Networks*, vol. 34, no. 1, pp. 63–76, 2025.

[9] M. A. Arasi, L. Almuqren, I. Issaoui, N. S. Almalki, A. Mahmud, and M. Assiri, "Enhancing red palm weevil detection using bird swarm algorithm with deep learning model," *IEEE Access*, vol. 12, pp. 1542–1551, 2023.

[10] B. Martin, S. Saranya, and P. J. Chowdary, "Smart iot thermal imaging approach for early identification of red palm weevil (rpw) infestation on palms," *Scientific Reports*, vol. 16, no. 1, p. 5392, 2026.

[11] A. Nadeem, M. Ashraf, A. Mehmood, K. Rizwan, and M. S. Siddiqui, "Dataset of date palm tree (phoenix dactylifera l.) thermal images and their classification based on red palm weevil (rhynchophorus ferrugineus) infestation," *Frontiers in Agronomy*, vol. 7, p. 1604188, 2025.

[12] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9268–9277, 2019.

[13] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

[14] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[15] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

[16] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Eca-net: Efficient channel attention for deep convolutional neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11534–11542, 2020.

[17] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson, "Averaging weights leads to wider optima and better generalization," *arXiv preprint arXiv:1803.05407*, 2018.

[18] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," in *Artificial intelligence and machine learning for multi-domain operations applications*, vol. 11006, pp. 369–386, SPIE, 2019.

[19] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[20] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.

[21] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," *Advances in neural information processing systems*, vol. 33, pp. 18661–18673, 2020.

[22] A. K. Menon, S. Jayasumana, A. S. Rawat, H. Jain, A. Veit, and S. Kumar, "Long-tail learning via logit adjustment," *arXiv preprint arXiv:2007.07314*, 2020.

[23] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "Augmix: A simple data processing method to improve robustness and uncertainty," *arXiv preprint arXiv:1912.02781*, 2019.

[24] 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*, pp. 618–626, 2017.

[25] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks," in *2018 IEEE winter conference on applications of computer vision (WACV)*, pp. 839–847, IEEE, 2018.

[26] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International conference on machine learning*, pp. 3319–3328, PMLR, 2017.

[27] E. R. DeLong, D. M. DeLong, and D. L. Clarke-Pearson, "Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach," *Biometrics*, pp. 837–845, 1988.

[28] Q. McNemar, "Note on the sampling error of the difference between correlated proportions or percentages," *Psychometrika*, vol. 12, no. 2, pp. 153–157, 1947.

[29] 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*, pp. 770–778, 2016.

[30] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, pp. 6105–6114, PMLR, 2019.

[31] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

[32] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

[33] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[34] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I. S. Kweon, and S. Xie, "Convnext v2: Co-designing and scaling convnets with masked autoencoders," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16133–16142, 2023.