IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

Performance Analysis of CNN and Transformers Models with Weighted Ensemble for Agricultural Pest Image Classification

Narsaiah Battu[1] and Arathi Chitla[2]

Research Scholar Department of Computer Science and Engineering at Telangana University, Dichpally, Nizamabad, Telangana, India[1]

Professor of Computer Science and Engineering at Telangana University, Dichpally, Nizamabad, Telangana, India[2]

Abstract: Insect pests cause large and recurring crop losses, and manual identification in the field is slow and needs entomological expertise. This paper compares a convolutional network (ResNet50) with two vision transformers (ViT-B/16 and Swin-B) for fine-grained pest recognition on IP102, a public benchmark of 75,222 images in 102 classes with a strongly long-tailed class distribution. All three models are fine-tuned from ImageNet-pretrained weights under one shared, short (six-epoch) training protocol and evaluated once on the official 22,619-image test split. We then combine them by weighted soft voting, with the weights tuned only on the validation split. Swin-B is the best single model, with 76.29% top 1 accuracy and 0.684 macro F1, narrowly ahead of ViT-B/16 (75.90%, McNemar $p = 0.046$) and well ahead of ResNet50 (70.67%). The ensemble reaches 77.09% top 1 accuracy (95% bootstrap CI 76.57–77.65%), 94.77% top 5 accuracy and 0.694 macro F1, a significant gain over Swin-B (McNemar $p < 10^{-12}$). Validation-based weight tuning assigned ResNet50 a weight of zero, so the final ensemble combines the two transformers. All three models fail on the same 18.0% of test images, per-class F1 rises with the number of training images (Spearman $\rho \approx 0.30$–0.33), and the most frequent errors are between closely related species and between life stages of the same pest. Code, configurations and all predictions are released for reproducibility.

Keywords: Insect pest recognition, IP102, vision transformer, Swin Transformer, ResNet50, ensemble learning, fine-grained classification, precision agriculture.

I. INTRODUCTION

Pests and pathogens remove a large share of the world's harvest. An expert survey covering 67 countries estimated global yield losses to pathogens and pests of 21.5% for wheat, 30.0% for rice and 22.5% for maize, with the heaviest losses in food-deficit regions [1]. Early and correct identification of the pest species is the first step of integrated pest management, because the choice of control measure depends on it. Manual identification needs trained entomologists, is slow, and does not scale to the volume of images that field cameras, traps and smartphones now produce [2], [3].

Deep learning has made automatic pest recognition practical, and the IP102 benchmark [4] has become the standard test for it. IP102 contains 75,222 images of 102 pest species found on eight crops, with a strongly long-tailed class distribution, large changes in appearance across life stages (egg, larva, pupa, adult), and many images taken in cluttered field conditions. These properties make it much harder than earlier pest datasets: the original paper reported 49.4% top 1 accuracy for a fine-tuned ResNet-50 [4], and later CNN-based methods and ensembles raised this to the 60–75% range [5]– [8].

Two families of image models are now competing for this task. Convolutional neural networks (CNNs) such as ResNet50 [9] build in a strong priority for local patterns and translation invariance. Vision transformers drop most of that prior: ViT [10] lets every image patch attend to every other patch from the first layer, while Swin Transformer [11] restores a hierarchical, CNN-like feature pyramid using attention inside local, shifted windows. Recent pest-recognition work has moved toward transformers and CNN–transformer hybrids [12]– [16], but three gaps remain:

1. *Confounded comparisons.* Published IP102 results use different pretraining data, input resolutions, augmentation, training lengths and model-selection rules, so a gap between two papers mixes architecture with recipe. Few studies compare CNN, a plain ViT and a hierarchical transformer under one identical protocol.
2. *Accuracy-only reporting.* On a long-tailed dataset, top 1 accuracy is dominated by the frequent classes. Macro-averaged metrics, per-class behaviour, probability calibration, confidence intervals and significance tests are rarely reported together, which makes small gains hard to judge.

 This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IJARCCE.2026.151180

3. *Ensembles without a clear account of why they work.* Ensembles of CNNs and transformers are common on IP102 [6], [8], [13], [14], but the weights are not always selected on held-out data, and the error overlap that makes an ensemble useful is seldom measured.

This paper addresses these gaps with a controlled comparison. Its contributions are:

1. A comparison of ResNet50, ViT-B/16 and Swin-B on the official IP102 split under one shared fine-tuning protocol (same augmentation, optimizer, schedule, input size and validation-based model selection), so that differences can be attributed to the architecture and its pretraining.
2. A weighted soft-voting ensemble whose weights are selected by grid search on the validation split only, with an ablation over equal, tuned and fixed weights and over every pair of models.
3. An evaluation beyond accuracy: macro and weighted precision, recall and F1, ROC-AUC, MCC, Cohen's kappa, calibration error, 95% bootstrap confidence intervals and McNemar significance tests, plus per-class analysis against training-set frequency.
4. An analysis of where and why the models fails: error overlap between architectures, the most-confused class pairs, and Grad-CAM attention maps on correctly and incorrectly classified images.
5. A fully reproducible pipeline (code, configurations, seeds, logs and all predictions) that runs on free cloud GPUs.

The rest of the paper is organized as follows. Section II reviews related work. Section III describes the dataset and Section IV the models and ensemble. Section V gives the experimental setup, Section VI the results and Section VII the discussion. Section VIII covers limitations and Section IX concludes.

II. Related Work

A. CNN-Based Pest Recognition

Early deep-learning work on pests used small datasets of 10–40 classes collected in controlled settings, where fine-tuned CNNs quickly passed 95% accuracy [17], [18]. On IP102 the task is much harder. Nanni *et al.* fused saliency-based preprocessing with several CNNs and reached 61.93% [5]. Ayan *et al.* combined Inception-V3, Xception and MobileNet with weights found by a genetic algorithm and reached 67.13% [6]. Residual variants designed for pests, such as feature-reuse residual networks [19] and DFF-ResNet [20], improved the backbone itself. Ung *et al.* combined attention, feature-pyramid and fine-grained CNN models to reach 74.13% [7], and Nanni *et al.* reached 73.46% with an ensemble of CNNs trained with different augmentations and Adam variants [8]. A clear lesson from this line of work is that combining diverse models helps on IP102, but the members are almost always CNNs, which tend to share failure modes.

B. Transformer and Hybrid Models

The transformer [21] was adapted to images by ViT [10], which splits an image into 16×16 patches and processes them with global self-attention. ViT needs large-scale pretraining (ImageNet-21k or larger) to beat CNNs [10], [22], [23]. Swin Transformer [11] limits attention to local windows that shift between layers and merges patches to build a feature pyramid, which makes it more data-efficient and well suited to fine-grained tasks. Analyses of what these models learn show that ViTs mix global information even in early layers while CNNs stay local [24].

In pest recognition, Peng and Wang used a transformer as the classification head of a CNN and reported 74.90% on IP102 [12]. Xia *et al.* combined a multi-scale CNN with a DenseNet–vision-transformer hybrid by soft voting [13]. Wang *et al.* fused a CNN stream and a transformer stream with decoupled training aimed at the long tail and reached 76.1% [14]. Kim and Son added region-of-interest maps to a dual-branch cross-attention ViT and reported 81.81% [15], and Wang *et al.* reported 78.77% with an attention-augmented Swin variant [16]. An *et al.* used ResNet, ViT and Swin together with Grad-CAM to locate discriminative regions and fused their features [25]. These studies propose new architectures; they do not isolate how much of the gain comes from the backbone family itself.

C. Ensembles, Calibration and Evaluation

Ensembles reduce variance and improve both accuracy and probability estimate when their members make different errors [26], [27]. For long-tailed data, accuracy alone hides poor performance on rare classes, so macro-average metrics and per-class analysis are needed. Modern deep networks are also often over-confident [28], which matters when a prediction feeds a spraying decision. Paired significance tests such as McNemar's test are the recommended way to compare two classifiers on the same test set [29].

Table I lists representative published results on IP102. They are not directly comparable with each other or with ours, because pretraining, input resolution and training recipes differ; Section VI-H places our results in this context.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

TABLE I

PUBLISHED TOP 1 ACCURACY ON THE IP102 TEST SET (AS REPORTED BY THE AUTHORS)

Study	Year	Approach	Top-1 (%)
Wu *et al.* [4]	2019	ResNet-50 baseline (original benchmark)	49.4
Nanni *et al.* [5]	2020	Saliency preprocessing + CNN ensemble	61.93
Ayan *et al.* [6]	2020	GA-weighted ensemble of 3 CNNs	67.13
Nanni *et al.* [8]	2022	CNN ensemble, augmentation + Adam variants	73.46
Ung *et al.* [7]	2021	Attention + feature-pyramid + fine-grained CNNs	74.13
Peng and Wang [12]	2022	CNN backbone + transformer head	74.90
Wang *et al.* [14]	2023	Two-stream CNN + transformer fusion	76.1
Wang *et al.* [16]	2025	Swin-AARNet (attention-augmented Swin)	78.77
Kim and Son [15]	2025	ROI-aware multi-scale cross-attention ViT	81.81
This work	2026	ViT-B/16 + Swin-B, validation-weighted soft voting (6 epochs, 224 px)	**77.09**

III.DATASET

We use IP102 [4], which contains 75,222 images of 102 insect pest species. The classes follow a two-level hierarchy: each species belongs to the crop it mainly damages, and the eight crops are grouped into field crops and economic crops (Table II). Wu *et al.* also annotated about 19,000 images with bounding boxes for detection; we use only the image-level class labels.

TABLE II

IP102 CLASS HIERARCHY

Top-level category	Crop super-classes
Field crops (FC)	Rice, corn, wheat, beet, alfalfa
Economic crops (EC)	Vitis (grape), citrus, mango

We keep the official split without change: 45,095 training, 7,508 validation and 22,619 test images, a ratio of roughly 6 : 1 : 3. The validation set is used only for checkpoint selection and ensemble weights; the test set is used once per model, after training. Three properties make IP102 hard:

- *Long-tailed distribution.* The rarest class has 71 images and the most frequent 5,740; in the training split the largest class has 82 times as many images as the smallest (Fig. 1). A classifier can score well on accuracy while failing on many rare classes.
- *Large intra-class variation.* A class can contain eggs, larvae, pupae and adults of the same species, which look completely different [4], [5].
- *Small inter-class differences and clutter.* Related species can look almost identical, and many images show small insects on leaves, stems or soil with similar colour and texture [15].

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

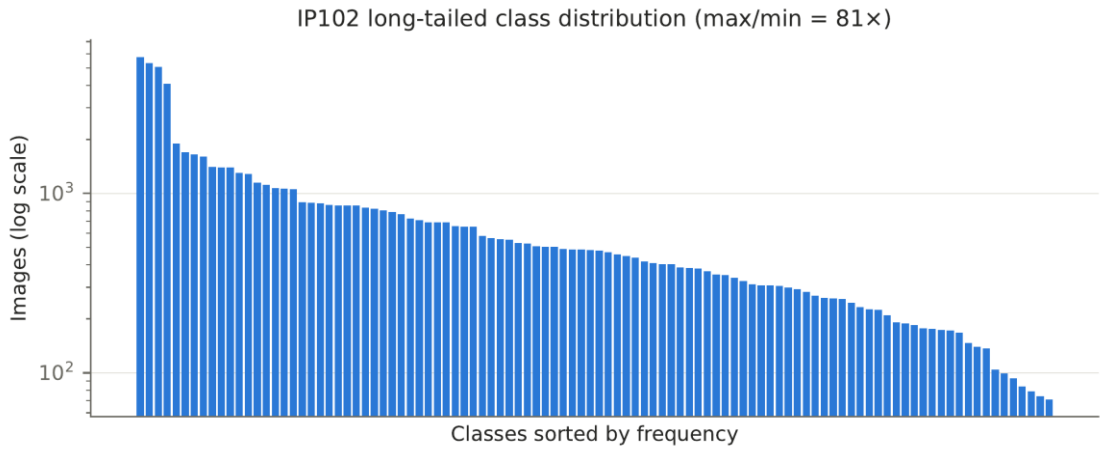


Fig. 1. Number of images per class (train + validation + test), sorted by frequency, on a log scale. The most frequent class has 5,740 images and the rarest 71; in the training split alone the counts range from 3,444 to 42 images per class.

Preprocessing: To reduce disk input/output, every image is resized once so that its shorter side is 256 px, keeping the aspect ratio. All crops and augmentations described in Section V are applied to these images at load time.

IV. METHODOLOGY

The method has two stages (Fig. 2). First, three ImageNet-pretrained [30] backbones are fine-tuned independently on the IP102 training split, and each produces a softmax probability vector over the 102 classes. Second, the three vectors are combined by weighted soft voting, with the weights chosen on the validation split.

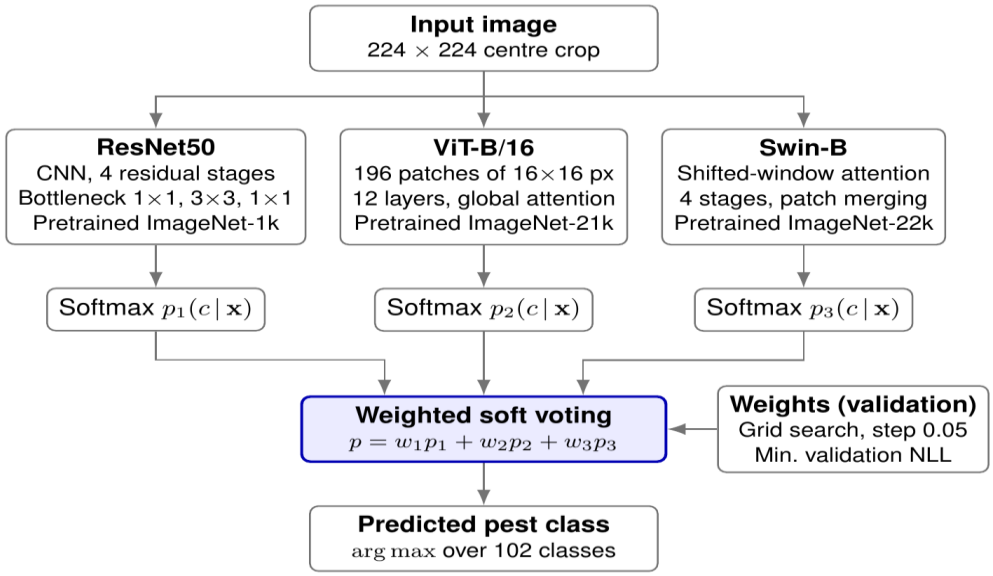


Fig. 2. Overview of the method. The three backbones are trained independently on the 45,095 IP102 training images; their class-probability vectors are combined with weights *w* selected on the validation split, never on the test split.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

TABLE III
BACKBONES USED IN THIS STUDY (FIGURES FROM THE ORIGINAL PAPERS; MEASURED VALUES FOR OUR RUNS ARE IN TABLE VII)

Model	Family	Pretraining	Params (M)	GFLOPs (224 px)	Feature map
ResNet50 [9]	CNN	ImageNet-1k	25.6	3.8	Hierarchical, 4 stages
ViT-B/16 [10], [23]	Transformer, global attention	ImageNet-21k → ImageNet-1k	86	17.6	Single scale, 14 × 14 tokens
Swin-B [11]	Transformer, shifted-window attention	ImageNet-22k → ImageNet-1k	88	15.4	Hierarchical, 4 stages

A. ResNet50

ResNet [9] eases the training of deep networks with identity shortcut connections. Instead of asking a stack of layers to learn a target mapping $H(\mathbf{x})$ directly, each block learns the residual $\mathcal{F}(\mathbf{x}) = H(\mathbf{x}) - \mathbf{x}$ and adds the input back:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}. \qquad (1)$$

ResNet50 uses bottleneck blocks of three convolutions (1×1 to reduce channels, 3×3 to extract spatial features, 1×1 to restore channels), arranged in four stages of 3, 4, 6 and 3 blocks after a 7×7 stem. Global average pooling and a fully connected layer give the class logits; we replace the 1000-way ImageNet head with a 102-way head. Convolution gives the network two built-in priors, locality and translation equivariance, which help when training data are limited.

B. Vision Transformer (ViT-B/16)

ViT [10] applies a standard transformer encoder [21] to a sequence of image patches. A 224×224 image is cut into $N = 196$ non-overlapping patches of $P \times P = 16 \times 16$ pixels. Each patch is flattened and linearly projected to dimension $D = 768$; a learnable class token is prepended and learnable position embeddings are added:

$$\mathbf{z}_0 = \left[\mathbf{x}_{\text{cls}};\ \mathbf{x}_p^1\mathbf{E};\ \ldots;\ \mathbf{x}_p^N\mathbf{E}\right] + \mathbf{E}_{\text{pos}}, \quad \mathbf{E} \in \mathbb{R}^{(P^2 \cdot 3) \times D}. \qquad (2)$$

Each of the $L = 12$ encoder layers applies multi-head self-attention (MSA, 12 heads) and an MLP, each with pre-layer normalisation and a residual connection:

$$\mathbf{z}'_\ell = \text{MSA}\big(\text{LN}(\mathbf{z}_{\ell-1})\big) + \mathbf{z}_{\ell-1}, \qquad (3)$$

$$\mathbf{z}_\ell = \text{MLP}\big(\text{LN}(\mathbf{z}'_\ell)\big) + \mathbf{z}'_\ell, \qquad (4)$$

with scaled dot-product attention in each head:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}. \qquad (5)$$

The normalised final class token, $\text{LN}(\mathbf{z}_L^0)$, feeds the 102-way classifier. Because every token attends to every other token, the receptive field is global from the first layer, but the cost grows quadratically with the number of tokens, $O(N^2)$. ViT has weak image-specific priors and relies on large-scale pretraining [10], [23]; we therefore use weights pretrained on ImageNet-21k and fine-tuned on ImageNet-1k.

C. Swin Transformer (Swin-B)

Swin Transformer [11] brings back two properties of CNNs, locality and a multi-scale hierarchy, while keeping attention. The image is split into 4×4 patches and processed in four stages. Inside each stage, self-attention is computed only within non-overlapping windows of $M \times M = 7 \times 7$ tokens (W-MSA). For a feature map of $h \times w$ tokens with C channels, this changes the cost from quadratic to linear in the number of tokens:

$$\Omega(\text{MSA}) = 4hwC^2 + 2(hw)^2C, \qquad (6)$$

$$\Omega(\text{W-MSA}) = 4hwC^2 + 2M^2hwC. \qquad (7)$$

To let information cross window borders, consecutive blocks alternate between regular windows and windows shifted by $\lfloor M/2 \rfloor$ tokens (SW-MSA):

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.151180

$$\hat{\mathbf{z}}_{\ell} = \text{W-MSA}\big(\text{LN}(\mathbf{z}_{\ell-1})\big) + \mathbf{z}_{\ell-1}, \quad (8)$$

$$\mathbf{z}_{\ell} = \text{MLP}\big(\text{LN}(\hat{\mathbf{z}}_{\ell})\big) + \hat{\mathbf{z}}_{\ell}, \quad (9)$$

$$\hat{\mathbf{z}}_{\ell+1} = \text{SW-MSA}\big(\text{LN}(\mathbf{z}_{\ell})\big) + \mathbf{z}_{\ell}, \quad (10)$$

$$\mathbf{z}_{\ell+1} = \text{MLP}\big(\text{LN}(\hat{\mathbf{z}}_{\ell+1})\big) + \hat{\mathbf{z}}_{\ell+1}. \quad (11)$$

Attention inside a window adds a learned relative position bias $\mathbf{B}$:

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\top}}{\sqrt{d}} + \mathbf{B}\right)\mathbf{V}. \quad (12)$$

Between stages, a patch-merging layer concatenates each 2×2 group of neighbouring tokens ($4C$ channels) and projects them to $2C$, halving the resolution and doubling the width, much like pooling in a CNN. Swin-B uses $C = 128$, stage depths of 2, 2, 18 and 2 blocks, and 4, 8, 16 and 32 heads, giving feature maps at 1/4, 1/8, 1/16 and 1/32 of the input resolution. We use weights pretrained on ImageNet-22k and fine-tuned on ImageNet-1k.

D. Weighted Soft-Voting Ensemble

Let $p_m(c \mid \mathbf{x})$ be the softmax probability that model $m \in \{\text{ResNet50,ViT-B/16,Swin-B}\}$ assigns to class c for image $\mathbf{x}$. The ensemble averages the probability vectors with non-negative weights that sum to one, and predicts the most probable class:

$$p_{\text{ens}}(c \mid \mathbf{x}) = \sum_{m=1}^{3} w_m \; p_m(c \mid \mathbf{x}), \quad w_m \geq 0, \quad \sum_{m=1}^{3} w_m = 1, \quad (13)$$

$$\hat{y} = \underset{c}{\text{argmax}}\, p_{\text{ens}}(c \mid \mathbf{x}). \quad (14)$$

The weights minimize the negative log-likelihood (NLL) on the validation set $\mathcal{V}$:

$$\mathbf{w}^{*} = \underset{\mathbf{w}\in\Delta^2}{\arg\min} \; -\frac{1}{|\mathcal{V}|} \sum_{(\mathbf{x}_i, y_i)\in\mathcal{V}} \log p_{\text{ens}}(y_i \mid \mathbf{x}_i). \quad (15)$$

We solve this by exhaustive search over the probability simplex Δ^2 with a step of 0.05 (231 candidate weight vectors). NLL is used instead of accuracy because it is smooth and uses the full probability vector, which makes the choice less sensitive to a few borderline images; selection by validation accuracy is available as an alternative criterion. The test set plays no part in choosing the weights.

V. Experimental Setup

A. Training Protocol

All three models are fine-tuned end to end with the same short recipe; only the learning rate and stochastic-depth rate differ, following common practice for each family (Table IV). The schedule was kept short (six epochs) so that the whole study runs on free cloud GPUs in a few hours. The models come from the timm library [31] and are trained in PyTorch [32].

TABLE IV
Fine-Tuning Hyper-Parameters

Setting	ResNet50	ViT-B/16	Swin-B
timm checkpoint	resnet50.tv2_in1k	vit_base_patch16_224.augreg2_in21k_ft_in1k	swin_base_patch4_window7_224.ms_in22k_ft_in1k
Epochs	6	6	6
Peak learning rate	1×10^{-3}	1×10^{-4}	1×10^{-4}
Batch size	64	64	64
Iterations per epoch	704	704	704
Stochastic depth (drop path)	0.05	0.1	0.2
Optimiser	AdamW [33], weight decay 0.05 (none on biases and normalisation layers)		
Schedule	1-epoch linear warm-up, cosine decay to 1% of peak [34]		
Loss	Cross-entropy with label smoothing 0.1 [35]		
Gradient clipping	1.0 (global norm)		

 This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IJARCCE.2026.151180

Precision	Mixed (FP16) [36]

Augmentation (training): Random resized crop to 224×224 (scale 0.35–1.0, bicubic), horizontal flip ($p = 0.5$), vertical flip ($p = 0.1$) and random erasing ($p = 0.25$) [37]. No automatic augmentation policy (such as TrivialAugment [38]) was used, to keep the CPU-bound data pipeline fast. Images are normalised with each checkpoint's own pretraining mean and standard deviation.

Evaluation: Resize the shorter side to 256 px and take a 224×224 centre crop, with no test-time augmentation.
Model selection: After each epoch the model is evaluated on the validation split; the checkpoint with the highest validation top-1 accuracy is kept. The test split is evaluated once, with that checkpoint.

Reproducibility: Python, NumPy and PyTorch random seeds are fixed (seed 42), and data-loader workers are seeded. Training ran on Kaggle with two NVIDIA Tesla T4 GPUs (16 GB each): Swin-B and ViT-B/16 were trained at the same time, one per GPU, followed by ResNet50. The software was Python 3.13, PyTorch 2.11.0, torchvision 0.26.0, timm 1.0.30 and CUDA 13.0. Code, configuration files, training logs and all validation and test predictions are available at [repository URL to be added by the author]

B. Evaluation Metrics

With N test images, true label y_i and predicted label $\hat{y}_i$, we report the following.

- *Top-1 and top-5 accuracy:* the share of images whose true class is the top prediction, or among the five most probable classes.
- *Precision, recall and F1:* per class, then averaged two ways. The macro average treats all 102 classes equally and therefore exposes rare-class performance; the weighted average weights each class by its test support:

$$\mathrm{F1}_c = \frac{2P_cR_c}{P_c + R_c}, \qquad \text{Macro-F1} = \frac{1}{K}\sum_{c=1}^{K} \mathrm{F1}_c\,, \quad K = 102. \qquad (16)$$

- *ROC-AUC and average precision:* one-vs-rest per class, then macro- and support-weighted averages.
- *Matthews correlation coefficient (MCC)* in its multi-class form [39] and *Cohen's kappa*, which corrects agreement for chance,

$$\kappa = \frac{p_o - p_e}{1 - p_e}, \qquad (17)$$

where p_o is the observed accuracy and p_e the agreement expected by chance from the class marginals.

- *Calibration:* negative log-likelihood and the expected calibration error over 15 equal-width confidence bins B_b [28]:

$$\mathrm{ECE} = \sum_{b=1}^{15} \frac{|B_b|}{N} |\,\mathrm{acc}(B_b) - \mathrm{conf}(B_b)\,|. \qquad (18)$$

- *Uncertainty and significance:* 95% confidence intervals for top-1 accuracy and macro F1 from 1,000 bootstrap resamples of the test set [40], and the exact McNemar test [29] on the images where two classifiers disagree in correctness.
- *Cost:* parameters, operations at 224×224, single-image latency (batch 1, FP32) and throughput (batch 64, FP16) on the training GPU.

C. Analysis and Visualization

For error analysis we compute the row-normalised confusion matrix, the 15 most frequent confused class pairs, the number of base models (0–3) that classify each test image correctly, and the Spearman correlation between per-class F1 and the number of training images per class. For interpretability we apply Grad-CAM [41], [42] to the last convolutional block of ResNet50 and to the last transformer block of ViT-B/16 and Swin-B, using the same test images for all three models, with one set that all models classify correctly and one set the ensemble misclassifies.

VI. RESULTS

All results are on the official IP102 test split (22,619 images, 102 classes), with seed 42 and the checkpoint selected on the validation split. Every number below was computed from the saved test predictions.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

A. Overall Performance

Swin-B is the strongest single model on every accuracy-type metric, ViT-B/16 is close behind, and ResNet50 trails both by about 5 points of top-1 accuracy (Table V). The validation-tuned ensemble improves on Swin-B by 0.80 points of top-1 accuracy, 0.20 points of top-5 accuracy and 0.010 macro F1.

TABLE V
TEST-SET RESULTS (22,619 IMAGES). TOP-1 ACCURACY IS GIVEN WITH ITS 95% BOOTSTRAP CONFIDENCE INTERVAL (1,000 RESAMPLES)

Model	Top-1 (%) [95% CI]	Top-5 (%)	Macro prec.	Macro rec.	Macro F1	Weighted F1	Macro AUC	MCC	Kappa
ResNet50	70.67 [70.07–71.29]	91.35	0.660	0.601	0.618	0.697	0.980	0.698	0.698
ViT-B/16	75.90 [75.37–76.41]	93.81	0.707	0.671	0.683	0.754	0.985	0.752	0.752
Swin-B	76.29 [75.73–76.85]	94.57	0.708	0.674	0.684	0.757	0.987	0.756	0.756
Weighted ensemble	**77.09 [76.57–77.65]**	**94.77**	**0.721**	**0.683**	**0.694**	**0.765**	**0.988**	**0.765**	**0.764**

Significance: The bootstrap intervals of Swin-B and the ensemble overlap slightly, but the paired McNemar test, which compares the two models image by image, is clear. The ensemble is right where Swin-B is wrong on 400 images, and wrong where Swin-B is right on 218 ($p = 2.2 \times 10^{-13}$). It also beats ViT-B/16 (793 vs. 523 images, $p = 1.0 \times 10^{-13}$). The gap between the two transformers is small and only just significant: Swin-B wins 996 discordant images and ViT-B/16 908 ($p = 0.046$). Both transformers beat ResNet50 by a wide margin ($p < 10^{-100}$ in both cases).

Macro vs. weighted scores: For every model, macro F1 is about 0.07 lower than weighted F1, and macro recall is lower than macro precision. Because the macro average gives rare classes equal weight, this gap is the cost of the long tail (Section VI-C).

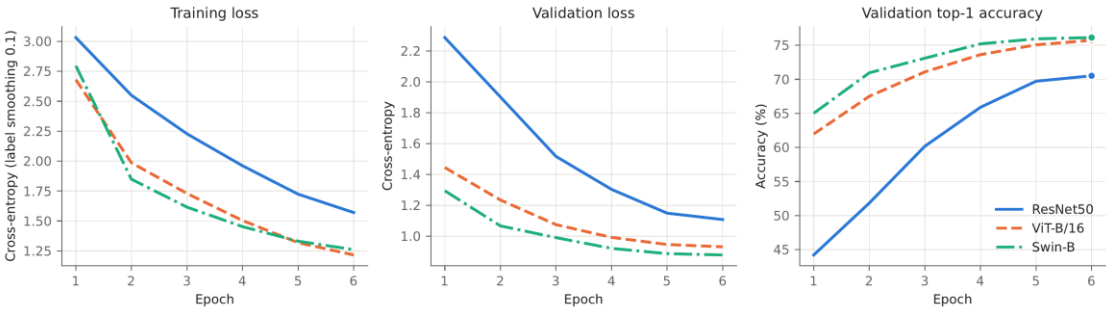


Fig. 3. Training loss, validation loss and validation top-1 accuracy per epoch; the marker shows the selected (best-validation) epoch.

All three models were still improving when training stopped: for each, the best validation accuracy came at the sixth and final epoch (ResNet50 70.51%, ViT-B/16 75.72%, Swin-B 76.12%; Fig. 3). Swin-B starts fastest, with 65.0% validation accuracy after one epoch against 62.0% for ViT-B/16 and 44.2% for ResNet50. At the last epoch, training accuracy under augmentation was 75.6% for ResNet50, 86.7% for ViT-B/16 and 85.0% for Swin-B, so the transformers fit the training set more closely. Validation and test accuracy agree to within 0.2 points for all three models.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

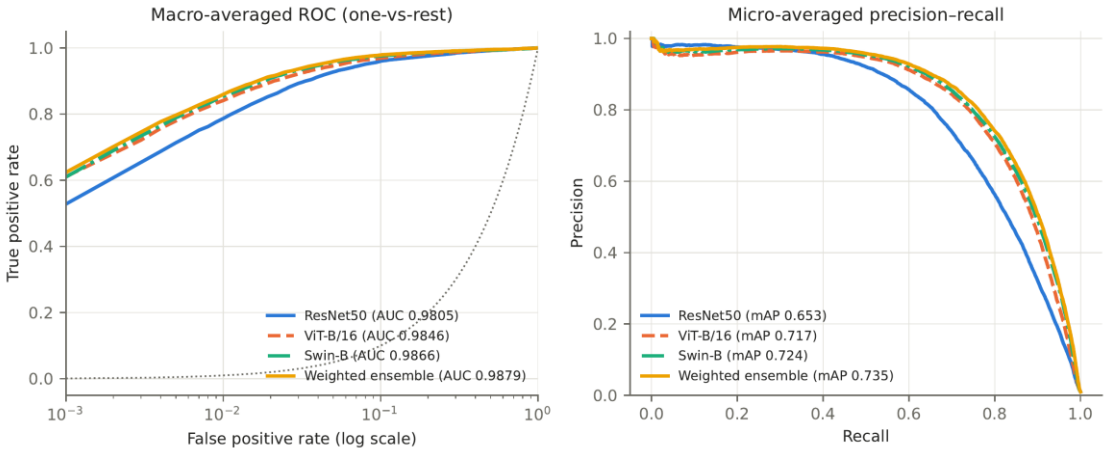


Fig. 4. Left: macro-averaged one-vs-rest ROC curves (log-scaled false-positive rate). Right: micro-averaged precision–recall curves, with macro average precision (mAP) in the legend.

All models rank classes well, with macro ROC-AUC between 0.980 and 0.988 (Fig. 4). The differences are clearer in the low false-positive region and in average precision: macro AP is 0.653 for ResNet50, 0.717 for ViT-B/16, 0.724 for Swin-B and 0.735 for the ensemble.

B. Ensemble Weights and Ablation

Minimizing validation NLL gave weights of 0.00 for ResNet50, 0.35 for ViT-B/16 and 0.65 for Swin-B, so the selected ensemble uses only the two transformers. Table VI compares this choice with other weightings and with every pair of models. Each variant's weights were set without using the test set.

TABLE VI
ENSEMBLE ABLATION. PAIR WEIGHTS MARKED "TUNED" WERE SELECTED ON VALIDATION NLL IN THE SAME WAY AS THE MAIN ENSEMBLE

Variant	Weights (ResNet50 / ViT-B/16 / Swin-B)	Val top-1 (%)	Test top-1 (%)	Test top-5 (%)	Test macro F1
Swin-B alone (reference)	0 / 0 / 1	76.12	76.29	94.57	0.684
All three, equal	0.33 / 0.33 / 0.33	76.53	76.79	94.49	0.691
All three, fixed (preliminary notes)	0.20 / 0.35 / 0.45	76.80	77.07	94.64	0.695
All three, tuned (selected)	**0.00 / 0.35 / 0.65**	**76.57**	**77.09**	**94.77**	**0.694**
ViT-B/16 + Swin-B, equal	0 / 0.50 / 0.50	76.93	77.10	94.76	0.695
ResNet50 + Swin-B, tuned	0.10 / 0 / 0.90	76.11	76.41	94.62	0.685
ResNet50 + Swin-B, equal	0.50 / 0 / 0.50	75.80	75.92	94.12	0.681
ResNet50 + ViT-B/16, tuned	0.20 / 0.80 / 0	75.73	76.11	93.88	0.685
ResNet50 + ViT-B/16, equal	0.50 / 0.50 / 0	75.19	75.57	93.67	0.678

Three points stand out. First, combining the two transformers gives the gain: every variant that contains both reaches 77.07–77.10% top-1, while every pair that includes ResNet50 stays at or below 76.41%. Second, giving ResNet50 an equal share costs about 0.3 points (76.79%), and pairing it with either transformer at equal weight falls below that transformer alone (75.92% vs. 76.29% for Swin-B; 75.57% vs. 75.90% for ViT-B/16). Even with tuned weights, ResNet50 adds only 0.1–0.2 points to either transformer. Third, among the strong variants the exact weights matter very little: the spread of 0.03 points is far smaller than the 95% confidence interval of about ± 0.5 points. The fixed weights from our preliminary notes score the highest validation accuracy, but the NLL criterion prefers the transformer-only weights; on the test set the two are equivalent.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

C. Class Imbalance and Per-Class Performance

Per-class F1 rises with the number of training images for every model (Spearman $\rho = 0.317$ for ResNet50, 0.333 for ViT-B/16, 0.311 for Swin-B and 0.304 for the ensemble; $p \leq 0.002$ in all cases). The correlation is moderate, though: class frequency explains part of the difficulty, not all of it (Fig. 5).

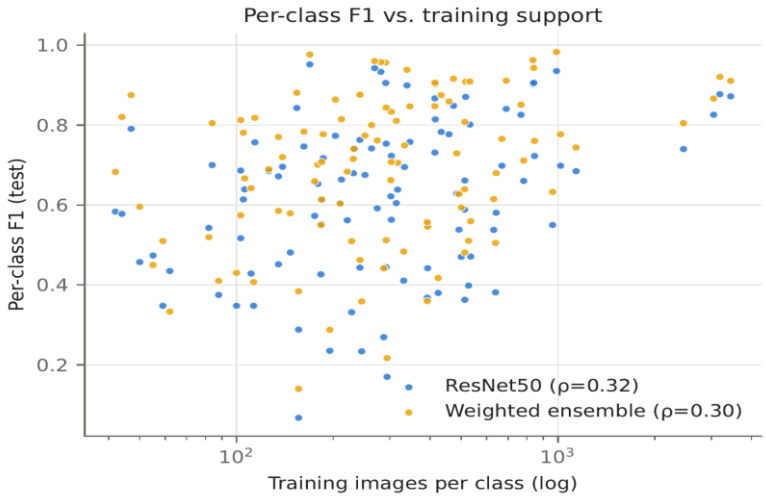


Fig. 5. Per-class test F1 against the number of training images per class (log scale) for ResNet50 and the ensemble.

Splitting the 102 classes into thirds by training-set size shows the same pattern. Mean ensemble F1 is 0.637 for the rarest third (42–186 training images), 0.688 for the middle third (195–414) and 0.760 for the most frequent third (415–3,444). The ensemble's gain over Swin-B is almost the same in every third (+0.009 to +0.011 mean F1), and it improves F1 for 78 classes while lowering it for 18. The transformers also narrow the spread across classes. The number of classes with F1 below 0.5 falls from 29 (ResNet50) to 21 (ViT-B/16), 17 (Swin-B) and 16 (ensemble), while the number with F1 of at least 0.9 rises from 7 to 12, 14 and 16. ResNet50 has the higher F1 for only 6 of the 102 classes.

The ten weakest classes for the ensemble have F1 between 0.14 and 0.42: *Therioaphis maculata* (0.14), large cutworm (0.22), green bug (0.29), beet fly (0.33), rice shell pest (0.36), English grain aphid (0.36), paddy stem maggot (0.38), *Toxoptera citricidus* (0.41), white-margined moth (0.41) and bird cherry-oat aphid (0.42). Five of them are aphids, which are small and look alike. Several are not rare: English grain aphid has 197 test images and bird cherry-oat aphid 213, so visual similarity, and not only sample size, limits performance. The strongest classes reach F1 of 0.94–0.98, among them mole cricket (0.98), *Oides decempunctata* (0.98), Locustoidea (0.96) and *Papilio xuthus* (0.96). These are large insects with distinctive shapes or colour patterns.

D. Calibration and Confidence

All four models are slightly *under-confident*: their average top-1 confidence is lower than their accuracy. The gap is 4.1 points for ResNet50 (66.6% mean confidence vs. 70.7% accuracy), 2.7 for ViT-B/16, 2.1 for Swin-B and 4.2 for the ensemble. The expected calibration errors follow the same order: 4.08%, 2.99%, 2.92% and 4.25%. In the reliability diagram most points lie on or just above the diagonal (Fig. 6), so a stated confidence of, say, 70% corresponds to slightly more than 70% observed accuracy. This is the expected effect of label smoothing, which discourages extreme probabilities. Averaging two models softens the probabilities further, which is why the ensemble has the best negative log-likelihood (0.864, vs. 0.889 for Swin-B, 0.942 for ViT-B/16 and 1.115 for ResNet50) yet a slightly higher ECE than either transformer alone.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

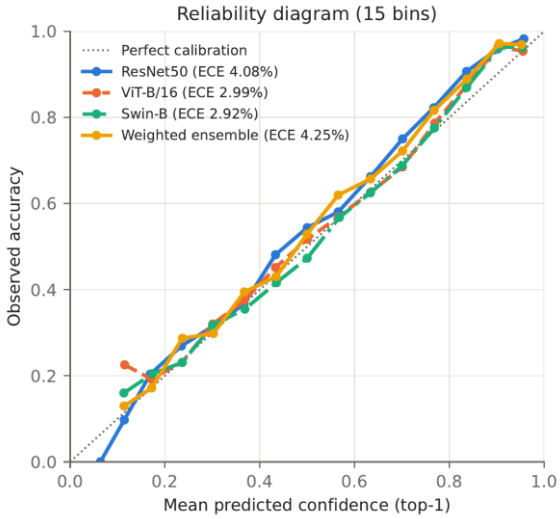


Fig. 6. Reliability diagram: observed accuracy against mean top-1 confidence in 15 equal-width bins. The dotted line is perfect calibration.

Confidence also separates right from wrong predictions well (Fig. 7). For the ensemble, the median confidence is 0.88 on correct predictions and 0.47 on errors. When the ensemble's confidence exceeds 0.9, which happens for 32.2% of test images, it is right 97.7% of the time; only 3.3% of its errors carry such high confidence. In practice, a field tool could therefore accept high-confidence predictions automatically and refer low-confidence images to an expert.

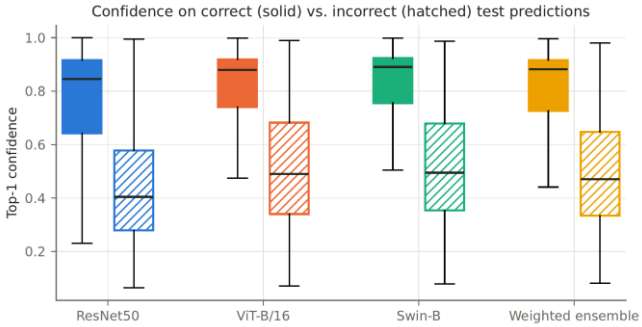


Fig. 7. Distribution of top-1 confidence for correct (solid) and incorrect (hatched) test predictions; boxes show the interquartile range and whiskers 1.5 × IQR.

E. Error Analysis and Model Agreement

The three base models agree on most images. All three are correct on 65.8% of the test set (14,874 images) and all three are wrong on 18.0% (4,074 images). On the remaining 16.2%, only one or two models are correct, and this is the only place an ensemble can help. The ensemble recovers 90.1% of the images where exactly two models are right and 41.4% of those where exactly one is (Fig. 8). It is right on just 13 of the 4,074 images that all three models miss. An oracle that always picked a correct model would reach 82.0%, so the ensemble (77.1%) captures part, but not most, of the available diversity.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

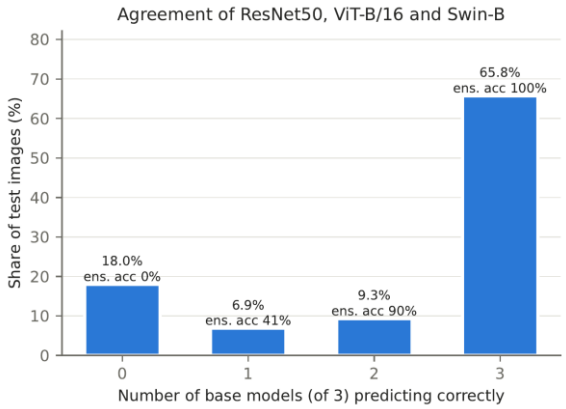


Fig. 8. Share of test images classified correctly by 0, 1, 2 or 3 of the base models, with the ensemble's accuracy inside each group.

The two transformers make the most similar mistakes: 4,456 images are misclassified by both, and the Jaccard overlap of their error sets is 0.70. ResNet50's errors overlap less with either transformer (0.63 for both pairs), so it is the most *different* member. It is also the weakest, with 5.6 points lower accuracy and a markedly higher NLL, and on balance the validation criterion found that its distinct errors did not pay for its extra mistakes (Section VI-B).

The most frequent ensemble errors fall into two groups (Fig. 9):

1. *Overlapping labels.* Some IP102 classes are a family or a generic group that contains other classes in the dataset. Tarnished plant bug and alfalfa plant bug are predicted as *Miridae* (the plant-bug family) in 29% of cases each. English grain aphid and bird cherry-oat aphid are predicted as the generic class *aphids* in 33% and 26% of cases. Rice leafhopper is predicted as *Cicadellidae* (the leafhopper family) in 36%. *Lytta polita* and legume blister beetle are confused with the generic *blister beetle* class. For these images the prediction is taxonomically correct but does not match the label.
2. *Look-alike sibling species and stages.* Large cutworm is predicted as black cutworm for 49% of its test images, and rice shell pest as rice leaf roller for 46%. Brown and small brown planthoppers are confused with white-backed planthopper (25% and 28%), and cabbage armyworm with beet armyworm (23%). Most of these pairs are close relatives (the cutworms and armyworms are noctuid moths, and the three planthoppers belong to one family), and their classes include larvae, which carry few distinguishing features.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

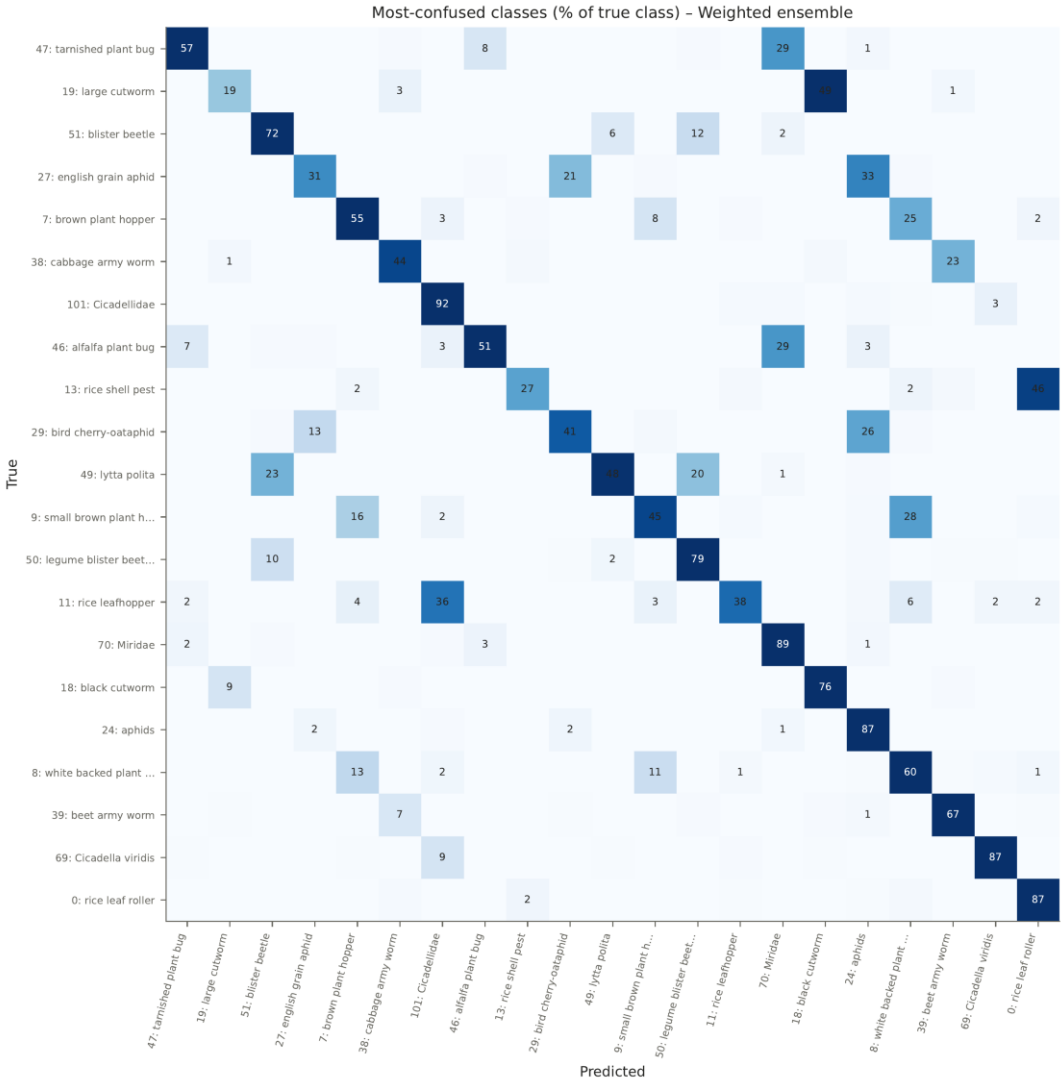


Fig. 9. Row-normalised confusion sub-matrix (% of each true class) over the classes involved in the ensemble's 15 most frequent confusions.

F. Grad-CAM Visualization

Figs. 10 and 11 show Grad-CAM maps for the same test images across the three models. The images were drawn at random (one per class) from those all models classify correctly and from those the ensemble misclassify.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.151180

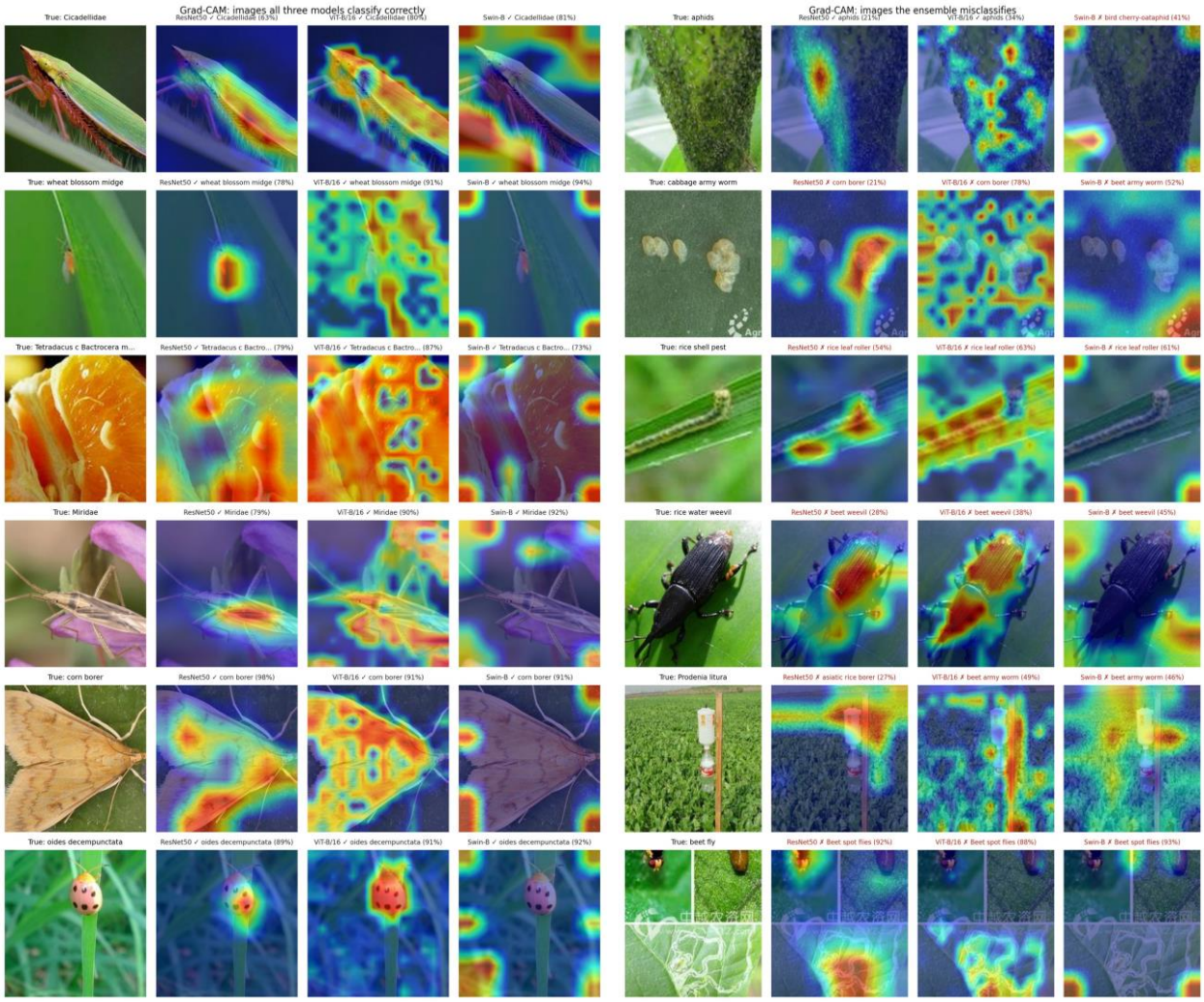


Fig. 10. Grad-CAM maps for six test images that all three models classify correctly. Each title gives the model's prediction and confidence.

Fig. 11. Grad-CAM maps for six test images the ensemble misclassifies; a red title marks a wrong prediction.

The three architectures produce visibly different maps. ResNet50 gives compact maps centred on the insect body, for example on the wheat blossom midge, the plant bug (*Miridae*) and the spotted *Oides* beetle. ViT-B/16 also covers the insect, but its maps are broken into patch-sized blocks and often spread onto the leaf or background. Swin-B's maps, taken from its final 7×7 stage, are coarse and frequently peak at image corners and borders, even when it predicts correctly with more than 90% confidence. We read this as a limitation of last-layer Grad-CAM for shifted-window models rather than as evidence that Swin-B ignores the insect: its accuracy and its agreement with the other models (Section VI-E) would be hard to explain otherwise. In the *Bactrocera minax* example, the image shows damaged citrus fruit rather than an adult fly, and all three models still classify it correctly. This shows that damage symptoms on the crop are also used as evidence.

The misclassified examples show four failure types that match the error analysis.

1. *Generic vs. specific labels.* On a dense aphid colony labelled *aphids*, ResNet50 and ViT-B/16 are right but with low confidence (21% and 34%). Swin-B predicts the specific species *bird cherry-oat aphid* (41%), and the ensemble, which gives Swin-B the larger weight, is also wrong.
2. *Uninformative life stages.* A cluster of cabbage-armyworm eggs is labelled correctly by no model; the predictions are corn borer and beet armyworm. Likewise, a rice-shell-pest larva on a leaf is called *rice leaf roller* by all three, the error made on 46% of that class's test images.
3. *Correct focus, wrong species.* On a rice water weevil, ResNet50 and ViT-B/16 attend to the beetle itself but all three predict *beet weevil*, another weevil.

DOI: 10.17148/IJARCCE.2026.151180

4. *Images without a visible pest.* An image labelled *Prodenia litura* shows a pheromone trap in a field with no visible moth. An image labelled *beet fly* is a watermarked collage of an adult fly and leaf mines, which all three models assign to the related class *beet spot flies* with about 90% confidence.

The last two cases suggest that part of the remaining error is due to dataset properties (overlapping class definitions, context-only images and watermarked web images) and not only to the models.

TABLE VII
MODEL SIZE AND COST, MEASURED ON ONE NVIDIA TESLA T4 AT 224 × 224 INPUT. TRAINING TIME COVERS SIX EPOCHS INCLUDING PER-EPOCH VALIDATION

Model	Parameters (M)	GMACs	Latency, batch 1, FP32 (ms)	Throughput, batch 64, FP16 (img/s)	Training time (h)
ResNet50	23.7	4.09	5.8	863	0.27
ViT-B/16	85.9	16.85	13.6	453	0.77
Swin-B	86.8	15.43	22.0	223	1.03
Ensemble (ViT-B/16 + Swin-B)	172.7	32.28	35.6*	149*	—

*Sum of the two members run one after the other; the selected ensemble does not need ResNet50 because its weight is zero.

The counts in Table VII are multiply-accumulate operations (GMACs), the quantity that most papers, including [9]–[11], report as "FLOPs". ResNet50 is about four times cheaper than either transformer and the fastest model by far. Swin-B has slightly fewer operations than ViT-B/16 but is about 1.6 times slower at batch 1 and half as fast at batch 64 on the T4, because window partitioning and shifting add memory operations that the GMAC count does not capture. The ensemble roughly doubles the cost of Swin-B for a 0.8-point accuracy gain. Training times are indicative only: ViT-B/16 and Swin-B were trained at the same time on separate GPUs that shared four CPU cores for data loading, while ResNet50 was trained alone.

H. Comparison with Published Results

Table I lists published IP102 results. Our ensemble (77.09%) is above the CNN ensembles of Ayan *et al.* (67.13%) [6], Nanni *et al.* (73.46%) [8] and Ung *et al.* (74.13%) [7], the CNN–transformer models of Peng and Wang (74.90%) [12] and Wang *et al.* (76.1%) [14], and the original 49.4% baseline [4]. It is below two recent purpose-built architectures, Swin-AARNet (78.77%) [16] and ROI-ViT (81.81%) [15]. Our single Swin-B (76.29%) already exceeds most of the earlier multi-model systems. These comparisons are only indicative, because training recipes, input resolution and pretraining differ. Our models were trained for only six epochs at 224×224 pixels, without any long-tail-specific loss or test-time augmentation, so the results reflect what standard, publicly available backbones achieve under a short, low-cost protocol.

VII. DISCUSSION

A. Which Backbone Works Best, and Why

Swin-B was the best single model, but its lead over ViT-B/16 is small (0.39 points, $p = 0.046$), while both transformers lead ResNet50 by more than 5 points. Two properties may explain Swin-B's edge. Its local windows and feature pyramid suit small insects on cluttered backgrounds, and it learns fastest: it reached 65.0% validation accuracy after one epoch, against 62.0% for ViT-B/16. That speed matters under a six-epoch budget. With large-scale pretraining, however, ViT-B/16's global attention is almost as effective, so the choice of attention pattern matters less than the jump from CNN to transformer.

That jump cannot be attributed to architecture alone. ResNet50 differs from the transformers in three ways at once: architecture (convolution vs. attention), pretraining data (ImageNet-1k vs. ImageNet-21k/22k) and size (24 M vs. about 86 M parameters). Our design uses each family's standard public checkpoint and therefore does not separate these factors; Section VIII returns to this.

B. Accuracy and Ranking Quality

Our preliminary notes had suggested that ResNet50 ranks classes better (higher macro-AUC) than the transformers despite lower accuracy. The controlled run does not support this: ResNet50 has the lowest macro AUC (0.980, vs. 0.985 for ViT-B/16 and 0.987 for Swin-B) and the lowest macro average precision. Under a shared protocol, ranking quality follows accuracy. The earlier observation was most likely a by-product of the different, undocumented training settings of that preliminary run.

 This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IJARCCE.2026.151180

C. Why the Ensemble Helps, and Why Only a Little

An ensemble can only help where its members disagree, and here they disagree on 16.2% of the test images. The two transformers make strongly overlapping errors (Jaccard 0.70), which limits what averaging them can achieve. The gain is real ($+0.80$ points, $p < 10^{-12}$) but modest, and the ensemble closes only 0.8 of the 5.7 points that separate Swin-B from the oracle (82.0%). ResNet50 makes the most *different* errors, yet the validation criterion gave it zero weight. Its errors are numerous enough that adding it hurts more than its diversity helps. The general lesson is that diversity is useful only between members of comparable accuracy. A stronger CNN, for example one pretrained on ImageNet-21k, could be a more useful ensemble partner than ResNet50. Finally, the exact weights hardly matter (Table VI). A simple average of the two transformers is as good as the tuned weights, which makes the method easy to reproduce.

D. The Long Tail and the Label Structure

Class frequency explains only part of the per-class variation ($\rho \approx 0.3$). Two other factors stand out. First, several weak classes are not rare but visually hard: five of the ten weakest classes are aphid species. Second, part of the measured error comes from how IP102 defines its classes. Family-level classes such as *Miridae* and *Cicadellidae*, and the generic class *aphids*, overlap with species-level classes in the same dataset, and these overlaps account for several of the 15 most frequent confusions. Some images also contain no visible pest or are watermarked collages (Fig. 11). Hierarchical evaluation (crediting a correct family) or a cleaned label set would give a fairer picture of model ability, and long-tail-aware training could lift the rare classes.

E. Practical Implications

For deployment, the models offer clear trade-offs (Table VII):

- *Swin-B alone* gives nearly all of the accuracy (76.3%) at half the ensemble's cost.
- *The ViT-B/16 + Swin-B ensemble* adds 0.8 points for twice the computation.
- *ResNet50* is about four times cheaper and the fastest model, which suits mobile or edge devices, at a cost of 5.6 points.

Because all models are slightly under-confident and confidence separates right from wrong predictions well, a confidence threshold is a practical safeguard. At 0.9, the ensemble labels about a third of images automatically with 97.7% accuracy and can send the rest to an expert. Finally, all three models were still improving at the sixth epoch, so the accuracies reported here are a lower bound for these backbones and not their ceiling.

VIII. Limitations

- *Short training.* Each model was fine-tuned for six epochs, and all three reached their best validation accuracy at the final epoch, so longer training would probably raise every number reported here.
- *One seed.* Results come from a single training run per model. The bootstrap intervals capture test-set sampling, but not run-to-run variation from initialization and data order.
- *Pretraining and size confound.* The checkpoints differ in pretraining data (ImageNet-1k for ResNet50, ImageNet-21k/22k for the transformers) and in size (24 M vs. about 86 M parameters), so the CNN–transformer gap mixes architecture, pretraining scale and capacity.
- *One dataset, one resolution.* All results are on IP102 at 224×224 px. Higher resolutions help small insects [12], [15], and generalisation to other datasets or to new regions was not tested.
- *Simple training recipe.* No automatic augmentation policy, long-tail-specific loss, re-sampling or test-time augmentation was used; each could improve the rare classes [14].
- *Label quality.* IP102 mixes family-level, generic and species-level classes, and contains context-only and watermarked web images (Section VI-F). We used the labels as released, so part of the measured error reflects the dataset rather than the models.
- *Interpretability.* Grad-CAM on transformer blocks, and on Swin-B's coarse final stage in particular, shows where gradients concentrate, not a causal explanation.
- *Cost.* The ensemble needs two transformer forward passes per image, which may not suit edge devices.

IX. Conclusion and Future Work

We compared ResNet50, ViT-B/16 and Swin-B for insect pest recognition on IP102 under one shared, low-cost fine-tuning protocol, and combined them by validation-weighted soft voting. Swin-B was the best single model (76.29% top-1, 0.684 macro F1), closely followed by ViT-B/16 (75.90%) and well ahead of ResNet50 (70.67%). The ensemble reached 77.09% top-1 and 94.77% top-5 accuracy with 0.694 macro F1, a small but statistically significant gain over Swin-B. Validation-based weighting excluded ResNet50 entirely: the transformers' errors overlap strongly, and the more diverse CNN was too weak to help. The analysis also showed that every model is slightly under-confident, which makes confidence thresholds a practical safeguard. Remaining errors concentrate on look-alike species, uninformative life stages

 This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IJARCCE.2026.151180

and IP102's overlapping family-level and species-level labels. For practitioners, Swin-B alone offers the best accuracy–cost balance, the transformer ensemble the highest accuracy, and ResNet50 the cheapest option for edge devices.

The whole study, from data preparation to evaluation, runs in a few hours on free cloud GPUs, and the code, configurations and predictions are released so that others can reproduce and extend it.

In future work we plan to train for longer and over several seeds; to separate architecture from pretraining by comparing backbones with matched pretraining data; to use higher input resolution and region-of-interest cropping for small insects; to apply long-tail-aware losses and hierarchical (family-level) evaluation; to distil the ensemble into a single compact model for on-device use; and to test on independent field images.

REFERENCES

[1] S. Savary, L. Willocquet, S. J. Pethybridge, P. Esker, N. McRoberts, and A. Nelson, "The global burden of pathogens and pests on major food crops," *Nature Ecol. Evol.*, vol. 3, no. 3, pp. 430–439, 2019, doi: 10.1038/s41559-018-0793-y.

[2] W. Li, T. Zheng, Z. Yang, M. Li, C. Sun, and X. Yang, "Classification and detection of insects from field images using deep learning for smart pest management: A systematic review," *Ecol. Informat.*, vol. 66, Art. no. 101460, 2021, doi: 10.1016/j.ecoinf.2021.101460.

[3] D. C. K. Amarathunga, J. Grundy, H. Parry, and A. Dorin, "Methods of insect image capture and classification: A systematic literature review," *Smart Agric. Technol.*, vol. 1, Art. no. 100023, 2021, doi: 10.1016/j.atech.2021.100023.

[4] X. Wu, C. Zhan, Y.-K. Lai, M.-M. Cheng, and J. Yang, "IP102: A large-scale benchmark dataset for insect pest recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 8787–8796, doi: 10.1109/CVPR.2019.00899.

[5] L. Nanni, G. Maguolo, and F. Pancino, "Insect pest image detection and recognition based on bio-inspired methods," *Ecol. Informat.*, vol. 57, Art. no. 101089, 2020, doi: 10.1016/j.ecoinf.2020.101089.

[6] E. Ayan, H. Erbay, and F. Varçın, "Crop pest classification with a genetic algorithm-based weighted ensemble of deep convolutional neural networks," *Comput. Electron. Agric.*, vol. 179, Art. no. 105809, 2020, doi: 10.1016/j.compag.2020.105809.

[7] H. T. Ung, H. Q. Ung, and B. T. Nguyen, "An efficient insect pest classification using multiple convolutional neural network based models," 2021, *arXiv:2107.12189*.

[8] L. Nanni, A. Manfè, G. Maguolo, A. Lumini, and S. Brahnam, "High performing ensemble of convolutional neural networks for insect pest image detection," *Ecol. Informat.*, vol. 67, Art. no. 101515, 2022, doi: 10.1016/j.ecoinf.2021.101515.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[10] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021. [Online]. Available: arXiv:2010.11929

[11] Z. Liu *et al.*, "Swin Transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 10012–10022, doi: 10.1109/ICCV48922.2021.00986.

[12] Y. Peng and Y. Wang, "CNN and transformer framework for insect pest classification," *Ecol. Informat.*, vol. 72, Art. no. 101846, 2022, doi: 10.1016/j.ecoinf.2022.101846.

[13] W. Xia, D. Han, D. Li, Z. Wu, B. Han, and J. Wang, "An ensemble learning integration of multiple CNN with improved vision transformer models for pest classification," *Ann. Appl. Biol.*, vol. 182, no. 2, pp. 144–158, 2023, doi: 10.1111/aab.12804.

[14] C. Wang, J. Zhang, J. He, W. Luo, X. Yuan, and L. Gu, "A two-stream network with complementary feature fusion for pest image classification," *Eng. Appl. Artif. Intell.*, vol. 124, Art. no. 106563, 2023, doi: 10.1016/j.engappai.2023.106563.

[15] G.-E. Kim and C.-H. Son, "ROI-aware multiscale cross-attention vision transformer for pest image identification," *Comput. Electron. Agric.*, vol. 237, Art. no. 110732, 2025, doi: 10.1016/j.compag.2025.110732.

[16] X. Wang, Z. Xiao, and Z. Deng, "Swin attention augmented residual network: A fine-grained pest image recognition method," *Front. Plant Sci.*, vol. 16, Art. no. 1619551, 2025, doi: 10.3389/fpls.2025.1619551.

[17] C. Xie *et al.*, "Multi-level learning features for automatic classification of field crop pests," *Comput. Electron. Agric.*, vol. 152, pp. 233–241, 2018, doi: 10.1016/j.compag.2018.07.014.

[18] K. Thenmozhi and U. S. Reddy, "Crop pest classification based on deep convolutional neural network and transfer learning," *Comput. Electron. Agric.*, vol. 164, Art. no. 104906, 2019, doi: 10.1016/j.compag.2019.104906.

[19] F. Ren, W. Liu, and G. Wu, "Feature reuse residual networks for insect pest recognition," *IEEE Access*, vol. 7, pp. 122758–122768, 2019, doi: 10.1109/ACCESS.2019.2938194.

 This work is licensed under a Creative Commons Attribution 4.0 International License

DOI: 10.17148/IJARCCE.2026.151180

[20] W. Liu *et al.*, "DFF-ResNet: An insect pest recognition model based on residual networks," *Big Data Mining Anal.*, vol. 3, no. 4, pp. 300–310, 2020, doi: 10.26599/BDMA.2020.9020021.

[21] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017.

[22] T. Ridnik, E. Ben-Baruch, A. Noy, and L. Zelnik-Manor, "ImageNet-21K pretraining for the masses," in *Proc. NeurIPS Datasets Benchmarks Track*, 2021. [Online]. Available: arXiv:2104.10972

[23] A. Steiner, A. Kolesnikov, X. Zhai, R. Wightman, J. Uszkoreit, and L. Beyer, "How to train your ViT? Data, augmentation, and regularization in vision transformers," *Trans. Mach. Learn. Res.*, 2022. [Online]. Available: arXiv:2106.10270

[24] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy, "Do vision transformers see like convolutional neural networks?" in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 34, 2021.

[25] J. An, Y. Du, P. Hong, L. Zhang, and X. Weng, "Insect recognition based on complementary features from multiple views," *Sci. Rep.*, vol. 13, 2023, doi: 10.1038/s41598-023-29600-1.

[26] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017.

[27] T. G. Dietterich, "Ensemble methods in machine learning," in *Multiple Classifier Systems* (Lecture Notes in Computer Science, vol. 1857). Berlin, Germany: Springer, 2000, pp. 1–15, doi: 10.1007/3-540-45014-9_1.

[28] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 1321–1330.

[29] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, 1998, doi: 10.1162/089976698300017197.

[30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.

[31] R. Wightman, "PyTorch image models (timm)," GitHub repository, 2019, doi: 10.5281/zenodo.4414861.

[32] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, 2019.

[33] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019. [Online]. Available: arXiv:1711.05101

[34] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017. [Online]. Available: arXiv:1608.03983

[35] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.

[36] P. Micikevicius *et al.*, "Mixed precision training," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018. [Online]. Available: arXiv:1710.03740

[37] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 7, 2020, pp. 13001–13008.

[38] S. G. Müller and F. Hutter, "TrivialAugment: Tuning-free yet state-of-the-art data augmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 774–782.

[39] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, Art. no. 6, 2020, doi: 10.1186/s12864-019-6413-7.

[40] B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap*. New York, NY, USA: Chapman & Hall, 1993.

[41] 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 *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 618–626, doi: 10.1109/ICCV.2017.74.

[42] J. Gildenblat *et al.*, "PyTorch library for CAM methods," GitHub repository, 2021. [Online]. Available: https://github.com/jacobgil/pytorch-grad-cam

 This work is licensed under a Creative Commons Attribution 4.0 International License