Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.6, December 2025, pp. 3798~3809
e-ISSN: 2548-6861

Evaluation of the Accuracy and Efficiency of Deep CNN Architecture in Feature Extraction for Guava Disease Classification

Shiva Augusta Wicaqsana [1*], Ardytha Luthfiarta [2*], Amalia Putri Dwi Mareta [3*], Maulatus Shaffira Fitri [4*]
* Teknik Informatika, Universitas Dian Nuswantoro
shivaaugusta98@gmail.com [1], ardytha.luthfiarta@dsn.dinus.ac.id [2], 111202213977@mhs.dinus.ac.id [3], 111202214720@mhs.dinus.ac.id [4]

Article Info

Article history:

Received 2025-10-31
Revised 2025-11-26
Accepted 2025-12-10

Keyword:

Deep Convolutional Neural Network,
Feature Extraction,
Guava Disease,
ResNet50,
Image Classification

ABSTRACT

This study analyzes and compares several Deep Convolutional Neural Network (DCNN) architectures to evaluate the balance between classification accuracy and computational efficiency in guava fruit disease detection. A hybrid DCNN–Machine Learning (ML) approach was applied to 3,784 images from the Guava Fruit Disease Dataset using a 10-fold cross-validation scheme and undersampling techniques to address data imbalance. Six DCNN architectures were systematically tested, and the combination of ResNet50 with Artificial Neural Network (ANN) showed the best performance with an accuracy of 0.9979 and an F1-score of 0.9975, surpassing the InceptionV3 baseline (0.9974). In addition to being the most accurate, ResNet50 was also 2.5 times faster in feature extraction than DenseNet201, demonstrating an optimal balance between accuracy and time efficiency. These findings emphasize the importance of analyzing the accuracy-efficiency trade-off in selecting a DCNN architecture and open up opportunities for developing more efficient models for future agricultural image classification applications.

I. INTRODUCTION

Guava (*Psidium guajava*) is a tropical fruit of significant economic value, particularly in South and Southeast Asia, where it is cultivated for its rich nutritional profile and commercial demand. Production, however, is frequently disrupted by fungal diseases such as anthracnose and pest infestations like fruit fly attacks, both of which cause severe deformities, reduce marketability, and lead to considerable economic losses. Manual inspection remains the dominant method for detecting these diseases, but this approach is labor-intensive, subjective, and difficult to scale across large agricultural areas. Consequently, deep learning (DL) and computer vision techniques have emerged as promising alternatives for enabling rapid, accurate, and automated disease diagnosis [1], [2].

Deep Convolutional Neural Networks (DCNNs) have become the foundation of modern plant-disease classification due to their ability to learn hierarchical feature representations directly from images, outperforming traditional handcrafted descriptors such as color, shape, and texture [2], [3]. Early models such as AlexNet and VGG demonstrated the benefits of increased depth [2], while subsequent architectures introduced mechanisms to address optimization bottlenecks for instance, ResNet mitigates vanishing gradients through skip connections [7], Inception exploits multi-scale convolutional pathways [11], and DenseNet encourages extensive feature reuse via layer-wise dense connectivity [9]. These advancements have made DCNNs increasingly reliable across a wide range of agricultural vision tasks [4], [11].

Transfer learning further enhances performance on agricultural datasets, which often suffer from limited size, class imbalance, and substantial variability in environmental conditions. By leveraging low-level features learned from large-scale datasets such as ImageNet, pretrained DCNNs can be adapted efficiently to smaller domain-specific datasets [4], [15]. In hybrid DCNN-ML pipelines, pretrained models serve as frozen feature extractors, enabling systematic evaluation of different classifier types while reducing the risk of overfitting [8], [17].

Despite significant progress, several methodological gaps remain in the literature. Existing studies including the benchmark by Kılcı and Koklu [5] often evaluate only a single DCNN architecture, limiting insights into performance trade-offs between computational efficiency

and classification accuracy, particularly for edge deployment scenarios [3], [4]. Furthermore, extraction-time and inference-time measurements are seldom reported, leaving open questions regarding practical applicability in field conditions [5]. Class-wise analyses, such as confusion matrices and per-class recall, are also underrepresented despite the prevalence of imbalanced datasets and high visual similarity across disease categories [1], [21]. Another critical issue is the risk of data leakage during cross-validation, especially when multiple images of the same fruit captured from different angles appear in different folds; this can produce overly optimistic accuracy estimates and undermine the validity of results [2], [4].

To address these limitations, this study presents a comprehensive and systematic comparison of six widely used DCNN architectures DenseNet201, ResNet50, InceptionV3, MobileNetV2, EfficientNetB0, and VGG16 within a unified DCNN-ML hybrid framework. The evaluation includes classification accuracy, per-class performance, and computational efficiency, with consistent feature-extraction protocols and controlled experimental conditions. The findings demonstrate that ResNet50 combined with an Artificial Neural Network (ANN) achieves the highest accuracy (0.9979) while significantly outperforming DenseNet201 in extraction time. These results provide a robust, empirically grounded reference for selecting DCNN architectures that balance accuracy and efficiency and offer insights for real-world deployment in agricultural environments characterized by variable lighting, occlusion, and complex backgrounds [5], [6].

II. METHOD

This study adopts a unified Deep Convolutional Neural Network–Machine Learning (DCNN-ML) hybrid framework designed to evaluate both the representational capacity and computational efficiency of six widely used deep architectures. The methodological pipeline is illustrated in Figure 1 and consists of four major stages: dataset preparation and preprocessing, deep feature extraction, class balancing, and machine-learning classification. All components were executed in a controlled experimental environment to ensure fairness and reproducibility across models.

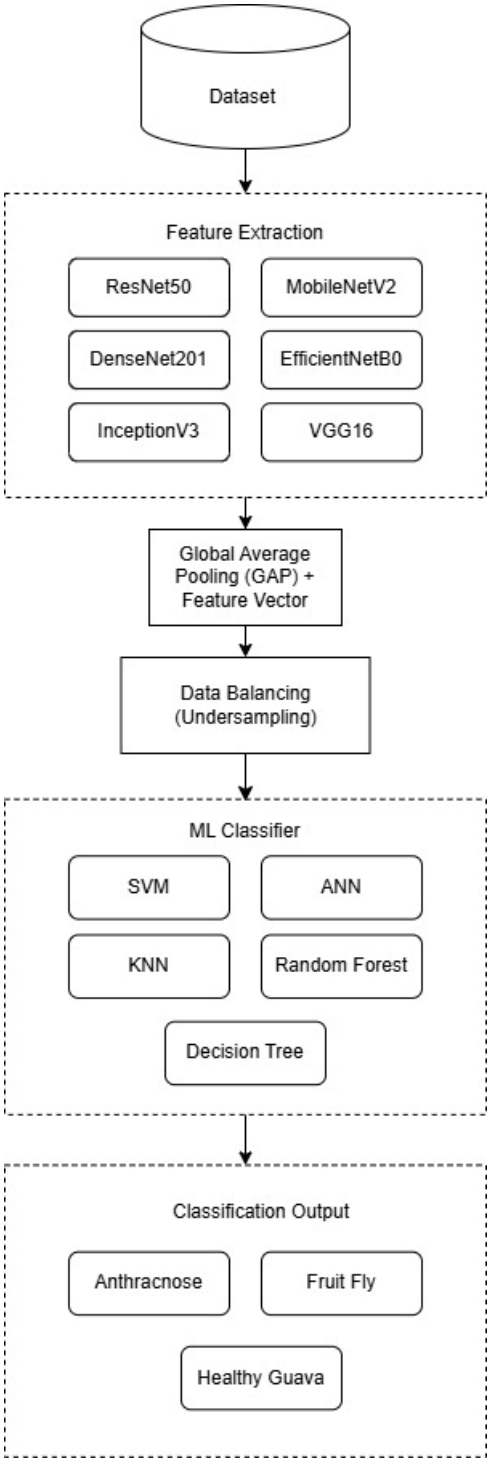

Figure 1. DCNN-ML Hybrid Framework

A. Dataset and Preprocessing

Although the Guava Fruit Disease Dataset includes extensive augmentation performed by the dataset creators, this study does not introduce any additional augmentation beyond what is already provided. The augmented images

generated through geometric transformations, contrast enhancement, and noise injection were originally produced to increase dataset diversity under field-like conditions. These images reflect realistic variations such as changes in brightness, rotations, and minor deformations, which are commonly encountered in agricultural environments [1], [2].

To prevent any risk of cross-fold similarity leakage arising from these augmented variants, all images from the training, validation, and test subsets were first merged into a single unified pool prior to cross-validation. This ensures that augmented images derived from the same original fruit remain within the same fold after partitioning. Because no new augmentation is applied by the authors of this study, no synthetic duplicates are introduced that could inadvertently appear across different folds an issue known to cause inflated accuracy in plant-disease classification tasks [2], [4].

Furthermore, all augmented and original images were uniformly resized to 224 × 224 pixels and preprocessed using the model-specific ImageNet normalization routine. This standardization ensures consistent pixel-level characteristics across the dataset while avoiding any form of augmentation-driven bias. As a result, the augmented samples contribute additional variability to the data distribution without compromising fold independence or affecting the validity of the cross-validation protocol.

The dataset used in this study consists of 3,784 guava fruit images collected under natural field conditions, exhibiting variations in lighting, angle, fruit orientation, and background clutter. The original image resolutions range from 640×640 to 1080×1080 pixels before being resized to 224×224 for input into the DCNN encoders. The dataset exhibits an initial class imbalance, containing 1,516 Anthracnose images, 1,333 Fruit Fly images, and 935 Healthy images. These images capture diverse visual characteristics, including differences in disease severity, color gradients, lesion texture, and surface irregularities, which are critical for modeling real-world variability. Understanding these distributional properties is important for evaluating generalization, particularly given the near-perfect performance achieved by the best model configuration. The dataset was obtained from the publicly available Guava Fruit Disease Dataset [6].

Anthracnose

Figure 2. Distribution of Guava Fruit Disease Dataset (Anthracnose)

Fruit Fly

Figure 3. Division of Guava Fruit Disease Dataset (Fruit Fly)

Healthy Guava

Figure 4. Distribution of Guava Fruit Disease Dataset (Healthy Guava)

TABLE I
CLASSIFICATION OF GUAVA FRUIT DISEASE DATASET

Data Split	Anthracnose	Fruit Fly	Healthy Guava	Total
Train	1080	918	649	2647
Val	308	262	185	755
Test	156	132	94	382
Total	**1464**	**1302**	**928**	**3784**

TABLE II
TRAINING DATA AFTER UNDERSAMPLING

Class	Training Data Vector	Total
Train	848	33.33%
Val	848	33.33%
Test	848	33.33%
Total	**2544**	**100.00%**

The six DCNN architectures examined in this study were selected to represent a broad spectrum of model capacities and design principles commonly used in modern computer vision research. ResNet50 and DenseNet201 were included because their residual and dense-connectivity mechanisms have been widely demonstrated to produce highly discriminative and stable feature representations across diverse image domains [7], [8]. MobileNetV2 and EfficientNetB0 serve as lightweight architectures optimized for mobile and embedded deployment, making them important baselines for evaluating whether low-parameter models can achieve competitive performance. VGG16 provides a classical plain-CNN baseline with a straightforward convolutional stack, enabling performance

comparison with architectures that do not employ skip connections. InceptionV3 was included because it achieved the previous best-reported accuracy for guava disease classification [5], making it an appropriate benchmark for evaluating performance improvements. Collectively, these six models enable a comprehensive assessment of how architectural depth, connectivity patterns, and parameter complexity influence feature-extraction quality and computational efficiency.

B. *Feature Extraction (DCNN Architecture)*

Deep feature extraction was performed using six ImageNet-pretrained architectures: ResNet50, DenseNet201, InceptionV3, MobileNetV2, EfficientNetB0, and VGG16. For all architectures, the fully connected classification head was removed (include_top=False) and replaced with a Global Average Pooling (GAP) layer to generate fixed-length feature vectors. For instance, the ResNet50-GAP configuration produces a 2,048-dimensional vector for each image.

All convolutional layers were frozen so that each architecture functions purely as a static feature extractor. This design ensures methodological consistency across models, minimizes overfitting risks for relatively small agricultural datasets, and enhances reproducibility by keeping pretrained weights unchanged. Feature extraction was implemented using TensorFlow/Keras (v2.x) with GPU acceleration via an NVIDIA Tesla T4. A warm-up forward pass was conducted before timing to avoid kernel initialization overhead, and a batch size of 32 was used during extraction. Each architecture contributes a distinct representational mechanism, and the corresponding architectural diagrams are provided to illustrate their structural principles.

1) ResNet50 (Best Accuracy and Efficiency): ResNet50 (R50) introduces Residual Blocks and Skip Connections that allow gradients to bypass stacked layers, effectively addressing the vanishing-gradient problem and enabling the training of deeper neural networks [7]. These identity mappings preserve feature information and stabilize deep learning optimization.

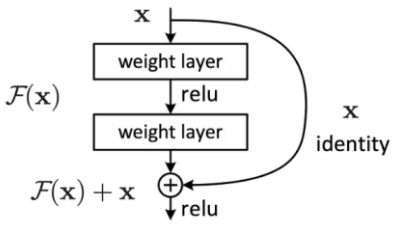

Figure 5. Architecture Diagram (Residual Block) [8]

2) DenseNet201: DenseNet201 (D201) employs dense connectivity, where each layer receives feature maps from all preceding layers and passes its own output to all subsequent layers [9]. This design promotes feature reuse, strong gradient propagation, and compact parameterization.

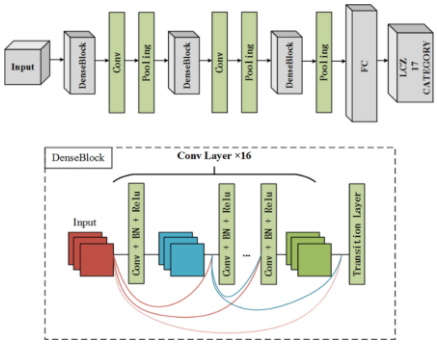

Figure 6. Flowchart of the Dense Block Mechanism [10]

3) InceptionV3 (I-V3): InceptionV3 (I-V3) utilizes multi-branch modules with parallel convolutional filters of different sizes, allowing the network to capture local and global structures simultaneously [11]. Its multi-scale feature extraction makes it a common benchmark in plant-disease classification studies.

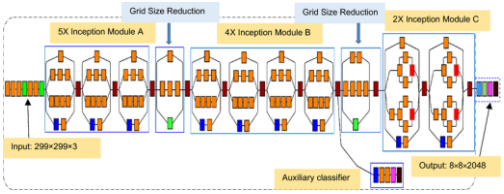

Figure 7. Flowchart of the InceptionV3 Mechanism [12]

4) MobileNetV2 (M-V2): MobileNetV2 adopts depthwise separable convolutions and inverted residuals to drastically reduce computational cost while maintaining competitive accuracy [13]. It is designed for deployment on mobile and low-power devices.

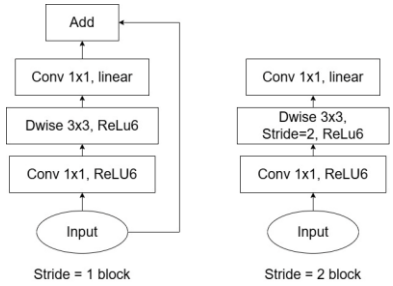

Figure 8. MobileNetV2 Architecture

5) EfficientNetB0 (E-B0): EfficientNetB0 is based on compound scaling, where network depth, width, and input resolution are scaled uniformly using a single coefficient

[14]. This allows the model to achieve a strong performance-efficiency balance with relatively small computational cost.

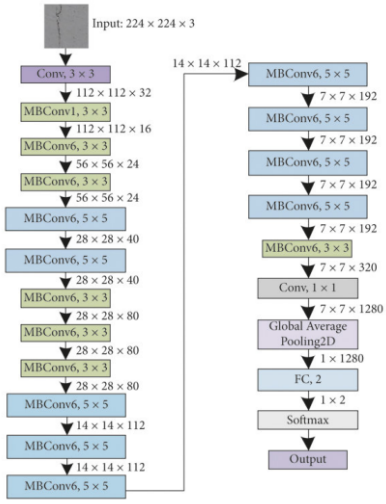

Figure 9. EfficientNetB0 Architecture [15]

6) VGG16 (V-16): VGG16 is a classical architecture characterized by its deep stack of uniform 3×3 convolutional layers followed by max-pooling [16]. Despite its simplicity, VGG16 remains a widely used baseline due to its stability and interpretability in feature extraction.

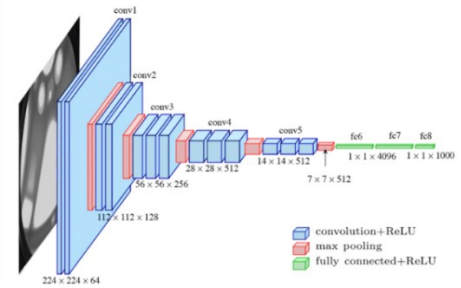

Figure 10. VGG16 architecture [17]

C. Rationale for Selecting the Six DCNN Architectures

The selection of the six DCNN architectures in this study ResNet50, DenseNet201, InceptionV3, MobileNetV2, EfficientNetB0, and VGG16 was grounded in their complementary architectural principles and their demonstrated effectiveness in plant-disease classification and natural image understanding. Rather than arbitrary selection, these models collectively represent four major design paradigms in modern convolutional networks, enabling a broad and systematic evaluation of feature extraction quality.

First, ResNet50 and DenseNet201 were chosen as representatives of residual and dense connectivity mechanisms, respectively. Residual learning facilitates deeper architectures by mitigating gradient degradation [7], while dense connections promote feature reuse and improve gradient propagation [9]. These properties enable strong hierarchical feature representations, which are particularly valuable for distinguishing subtle disease textures.

Second, InceptionV3 captures multi-scale spatial patterns through parallel convolutional branches of varying receptive fields [11]. Plant diseases often present both fine-grained lesion patterns and broader structural cues; thus, a multi-scale extractor offers unique advantages in complex agricultural imagery.

Third, MobileNetV2 and EfficientNetB0 were included due to their emphasis on computational efficiency a critical requirement for practical field deployment on mobile or edge devices. MobileNetV2 uses depthwise separable convolutions and inverted residuals to reduce computational cost [13], while EfficientNetB0 employs compound scaling to achieve a strong accuracy-efficiency balance [14].

Finally, VGG16 was selected as a classical baseline architecture. Its uniform and interpretable convolutional blocks make it a stable benchmark for analyzing the evolution and advantages of more modern architectures [16]. Despite lacking advanced connectivity patterns, VGG16 remains widely used for comparative studies in transfer learning.

Together, these architectures form a representative spectrum of CNN design strategies, allowing this study to objectively assess how architectural differences influence deep feature quality, computational cost, and downstream machine-learning performance in guava disease classification.

D. Feature Conversation and Data Balancing

After feature extraction, each image is represented as a fixed-length deep feature vector produced by the Global Average Pooling (GAP) layer of the DCNN backbone (e.g., a 2,048-dimensional vector for ResNet50) as described in residual network literature [7]. Although these representations capture meaningful semantic information, the dataset still exhibits substantial class imbalance, with the Healthy class noticeably smaller than Anthracnose and Fruit Fly, a condition known to distort classifier behavior and inflate performance on majority classes [1], [21].

To address this imbalance, a class-balancing procedure was applied using the NearMiss undersampling algorithm implemented in the imbalanced-learn toolkit [18]. NearMiss operates by selecting majority-class samples that are closest to minority instances in feature space, thereby retaining the most informative examples while reducing redundancy. This approach is particularly useful for high-dimensional deep feature spaces, where Euclidean proximity reflects similarities learned by convolutional encoders.

A key methodological consideration in image-based machine-learning experiments is the risk of data leakage, especially when multiple images of the same fruit may

appear across different folds of cross-validation. Leakage can produce overly optimistic accuracy because visually similar samples may appear both in training and validation partitions. To avoid this, the undersampling procedure was performed independently within each training fold, ensuring that neither validation nor test samples influenced the resampling process. Similar precautions are widely recommended in cross-validation of visual datasets to maintain independence across folds [2].

The resulting balanced training sets contain equal samples per class within each fold, enabling objective model comparison and preventing inflated results caused by skewed distributions. This fold-wise balancing strategy ensures that the evaluation reflects the true discriminative power of the DCNN features rather than artifacts of class imbalance.

NearMiss undersampling was selected as the class-balancing strategy based on methodological considerations related to feature-space structure and dataset size. Oversampling techniques such as SMOTE or ADASYN may generate synthetic samples that fail to preserve the complex, nonlinear texture patterns characteristic of plant disease images, which can introduce artifacts or unrealistic transitions in the pixel space [21]. Class weighting was also considered, but this approach does not correct the imbalance in feature-space distribution and is less effective for non-probabilistic classifiers such as KNN and SVM. In contrast, NearMiss operates directly in the deep-feature embedding space and selects majority-class samples that are closest to minority instances, thereby retaining structurally informative data while reducing redundancy. This makes NearMiss more suitable for evaluating the intrinsic discriminative quality of DCNN features without introducing synthetic noise or altering the dataset distribution artificially.

E. Data Leakage Prevention and Cross-Fold Independence

A critical methodological concern in image-based machine-learning experiments is the risk of data leakage between training and validation partitions, which can artificially inflate performance scores. Data leakage commonly occurs when visually similar images for example, multiple photographs of the same fruit taken from slightly different angles are distributed across different folds of cross-validation, causing the classifier to memorize visual similarity rather than learn generalizable disease features [2].

To eliminate this risk, several safeguards were implemented. First, all images from the original training, validation, and testing subsets provided by Amin et al. [6] were merged into a single unified pool before any preprocessing, resampling, or model-related operations. This ensures that no prior grouping or augmentation influences the fold composition. Second, 10-fold cross-validation was applied directly to this combined dataset prior to class balancing, resulting in fold partitions that are fully independent and not constrained by the dataset's original split structure.

Third, NearMiss undersampling was performed strictly within each training fold, while validation and test folds remained untouched. This prevents evaluation data from influencing the class distribution or feature density of the training subset, a best-practice recommendation highlighted in prior studies on cross-validation for visual datasets [1], [2]. Fourth, feature extraction was executed after fold construction using ImageNet-pretrained DCNN backbones configured as frozen static encoders. Because the convolutional weights are not updated, the representation of validation samples cannot be affected by the training process, thereby preserving strict cross-fold independence.

Finally, this study does not introduce any additional manual image augmentation beyond the dataset creator's original preprocessing pipeline. This avoids the classical leakage scenario in which augmented variants of the same fruit (e.g., rotations or flips) appear in different folds, a phenomenon known to produce highly optimistic accuracy in plant-disease classification tasks [2], [4].

Through these combined safeguards, the experimental workflow ensures that the resulting accuracy reflects genuine discriminative capability rather than artifacts of cross-fold contamination, resampling bias, or unintended feature overlap. These precautions provide a methodologically reliable foundation for evaluating DCNN feature quality and the comparative performance of machine-learning classifiers.

F. Machine Learning Classification

After obtaining a balanced set of feature vectors from the NearMiss undersampling procedure, five widely used machine-learning classifiers were employed to evaluate the discriminative strength of the DCNN-extracted features. The selected classifiers Artificial Neural Network (ANN), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest (RF), and Decision Tree (DT) represent both linear and non-linear decision boundaries, as well as single-model and ensemble-based learning strategies. This diversity enables a comprehensive assessment of how different learning paradigms respond to the deep feature representations generated by each DCNN architecture.

All classifiers were configured using hyperparameters commonly recommended in the literature, with minor adjustments to ensure stable optimization. Training and evaluation were conducted through 10-fold cross-validation, and undersampling was applied exclusively within each training fold to avoid information leakage into validation sets.

TABLE III
MACHINE LEARNING CLASSIFICATION

Classifier	Role	Parameters used
Artificial Neural Network (ANN)	Non-linear SOTA classifier; tests upper-bound of	Solver: Adam; Hidden Layer Size: (100); Max Iterations: 50-200.

	feature separability	
Support Vector Machine (SVM)	Strong margin-based classifier; robust to high-dimensional features	Kernel: RBF; C: 1.0.
K-Nearest Neighbors (KNN)	Instance-based classifier; benchmark for lightweight training	n_neighbors: 5; Metric: Euclidean.
Random Forest (RF)	Ensemble robustness baseline; reduces variance from feature noise	n_estimators: 100; Max depth: Automatic.
Decision Tree (DT)	Simple interpretable baseline; sensitive to feature quality	Criterion: Gini.

1) Support Vector Machine (SVM): SVM with a Radial Basis Function (RBF) kernel [19] evaluates whether the feature representations support non-linear class boundaries. Its margin-maximization mechanism makes SVM highly sensitive to feature separability in high-dimensional spaces.

2) K-Nearest Neighbors (KNN): KNN is an instance-based, training-free classifier that relies on Euclidean distance computations [20]. Its performance reflects how well the deep features preserve geometric relationships in the embedding space.

3) Artificial Neural Network (ANN): To evaluate the upper-bound non-linear separability of the extracted deep features, this study employs a Multilayer Perceptron (MLP) with a single hidden layer of 100 neurons and a ReLU activation function. The input layer receives DCNN-GAP feature vectors of varying dimensionality (e.g., 2,048 for ResNet50, 1,920 for DenseNet201, 512 for VGG16). The output layer consists of three softmax neurons corresponding to the disease classes. Training is performed using the Adam optimizer (learning rate = 0.001) for 50-200 iterations, with early stopping based on validation performance. No dropout or batch normalization is used, as the pretrained DCNN features are already compact, structured, and non-redundant [1], [4]. This configuration offers a controlled and interpretable evaluation of feature discriminability, and produces the highest performance when paired with ResNet50 features (accuracy = 0.9979).

4) Random Forest (RF): RF is an ensemble of decision trees that evaluates feature robustness under aggregated decision rules [21]. Its capability to handle partially redundant or noisy features makes it a reliable comparison baseline.

5) Decision Tree (DT): DT provides an interpretable, single-tree perspective on the discriminative power of the feature vectors [22]. Its relatively lower performance for example, 0.6258 accuracy when using MobileNetV2 features demonstrates the increased complexity required to model fine-grained disease patterns.

G. Experimental Environment and Timing Procedure

All experiments were executed in a controlled and reproducible computational environment to ensure fair comparison across DCNN architectures and machine-learning classifiers. The deep feature extraction and model training processes were conducted using Google Colaboratory, powered by an NVIDIA Tesla T4 GPU (16 GB VRAM) and a 2-core Intel Xeon CPU with 12 GB system RAM. This environment provides consistent hardware acceleration across all experiments and is widely used in academic benchmarking for vision-based deep learning tasks.

Deep feature extraction was implemented using TensorFlow/Keras v2.x with GPU acceleration enabled by CUDA. A standard input resolution of 224 × 224 pixels was used for all architectures to maintain consistency with ImageNet-pretrained weights. For each DCNN backbone, feature extraction was performed with a batch size of 32, which offers an optimal balance between GPU memory efficiency and stable kernel execution.

To ensure reliable and unbiased measurement of feature-extraction speed, a warm-up forward pass was executed for every architecture before measuring runtime. This removes the overhead associated with GPU kernel initialization and graph compilation, a common requirement in TensorFlow benchmarking [23]. Timing measurements therefore reflect steady-state performance rather than transient startup delays.

The recorded extraction time for each architecture includes only the forward-pass computation of convolutional layers and Global Average Pooling, without any backpropagation, since all DCNN weights were frozen during feature extraction. This ensures that runtime comparisons truly reflect architectural efficiency rather than differences in training cost.

Machine-learning classifiers were implemented using scikit-learn v1.x [18]. All timing and training operations for classical models (ANN, SVM, KNN, RF, DT) were performed on CPU to maintain consistency with standard machine-learning practice and to prevent GPU acceleration from favoring specific models.

Through this controlled setup, all feature-extraction times, training durations, and evaluation metrics were obtained under uniform conditions, guaranteeing that performance differences arise solely from architectural and algorithmic characteristics rather than hardware or implementation variability.

H. Classification Output and Experiment Settings

The final stage of the pipeline generates class predictions for the three guava disease categories Anthracnose, Fruit Fly, and Healthy using the machine-learning classifiers trained on the balanced deep-feature representations. Each classifier operates on the NearMiss-processed feature vectors to ensure that evaluation is not biased toward majority classes. Performance is assessed using four standard metrics: accuracy, precision, recall, and F1-score, all computed through scikit-learn's metric utilities [18]. These metrics collectively provide a comprehensive view of overall correctness, sensitivity to each class, and the balance between false positives and false negatives.

To ensure fair, unbiased, and reproducible evaluation, all experiments were conducted under a 10-fold cross-validation framework applied exclusively to the balanced training sets. In each fold, 90% of the balanced data is used for model fitting, while the remaining 10% serves as the validation subset. Importantly, undersampling is performed within each fold, preventing cross-fold contamination and eliminating the risk of data leakage an issue frequently highlighted in the literature on machine learning for imbalanced data [21].

All components of the workflow including DCNN feature extraction, NearMiss undersampling, classifier training, and evaluation were implemented in the Google Colaboratory environment. TensorFlow/Keras (v2.x) was used for deep feature extraction, while scikit-learn (v1.x) served as the primary toolkit for classical machine-learning models [18]. Hardware acceleration was provided by an NVIDIA Tesla T4 GPU, which significantly reduced the computation time for batch-wise feature extraction and ensured efficient experimentation. This controlled and uniform setup guarantees that performance differences arise solely from the architectures and classifiers being compared rather than external variations in computational conditions.

I. Performance Metrics Formula

To quantitatively evaluate the predictive performance of each classifier, four standard metrics are employed: accuracy, precision, recall, and F1-score. These metrics are computed from the confusion matrix components True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) and provide a comprehensive assessment of both overall correctness and class-wise discrimination capability.

1) Accuracy: Accuracy measures the proportion of correctly classified instances relative to the total number of samples:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2) *Precision*: Precision quantifies the proportion of correct positive predictions out of all predicted positives:

$$Precision = \frac{TP}{TP + FP}$$

3) Recall: Recall (sensitivity or true-positive rate) measures the proportion of actual positives that are correctly identified:

$$Recall = \frac{TP}{TP + FN}$$

4) F1-Score: The F1-score represents the harmonic mean of precision and recall:

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

This metric provides a balanced assessment when both false positives and false negatives must be controlled. All metrics are computed for each class and then averaged using the macro-averaging strategy, ensuring equal contribution from all disease categories regardless of sample size.

J. Feature Conversion and Data Balancing

The choice of NearMiss undersampling was motivated by methodological considerations regarding feature stability and the risks of synthetic oversampling on small agricultural image datasets. Oversampling techniques such as SMOTE and ADASYN may generate synthetic samples that fail to capture the complex, non-linear texture characteristics of plant diseases, potentially introducing unrealistic visual structures that degrade classifier performance [21]. Class weighting was also considered; however, this approach does not modify the underlying feature distribution and is less effective for non-probabilistic classifiers such as KNN or SVM. In contrast, NearMiss selectively retains majority-class samples that are closest to minority-class samples in the deep feature space, thereby preserving the most informative examples without generating artificial data or removing crucial visual variation at random. This strategy aligns with the study's primary goal of fairly evaluating the discriminative power of deep features extracted from different DCNN architectures.

III. Result and Discussion

A. Overall Performance and New SOTA Setting

Performance evaluation was conducted under a rigorous 10-fold cross-validation scheme using class-balanced feature vectors obtained through the NearMiss undersampling method. The primary objective of this study was to compare the performance of the evaluated DCNN-ML hybrid models against the previously reported benchmark accuracy of 0.9974 achieved by InceptionV3 (I-V3) [5].

The results indicate that this benchmark has been successfully exceeded. The highest performance was obtained by the ANN classifier using ResNet50 (R50) features, achieving an accuracy of 0.9979 and an F1-score of

0.9975. DenseNet201 (D201) achieved comparable but slightly lower performance (accuracy = 0.9971). In contrast, lightweight architectures such as MobileNetV2 (M-V2) and EfficientNetB0 (E-B0) demonstrated substantially lower accuracy (≤ 0.88), suggesting that high-capacity DCNNs are more suitable for near-perfect guava disease classification.

TABLE IV
COMPARATIVE PERFORMANCE AND COMPUTATIONAL EFFICIENCY COMPARISON OF 25 DCNN-ML HYBRID MODELS

Feature Extraction	Model	Acc	F1 Score	Ext Time (s)	Train Time (s)
R50	SVM	0.9976	0.9972	30.22	30.05
R50	**ANN**	**0.9979**	**0.9975**	**30.22**	**51.03**
R50	KNN	0.9955	0.9953	30.22	5.73
R50	RF	0.9881	0.987	30.22	141.45
R50	DT	0.9128	0.9104	30.22	78.95
D201	SVM	0.9966	0.9962	78.05	18.04
D201	ANN	0.9971	0.9969	78.05	40.11
D201	KNN	0.9947	0.9942	78.05	5.21
D201	RF	0.9929	0.992	78.05	113.08
D201	DT	0.9234	0.9221	78.05	76.39
M-V2	SVM	0.7719	0.7754	30.46	25.94
M-V2	ANN	0.8161	0.8171	30.46	28.52
M-V2	KNN	0.7902	0.7873	30.46	2.89
M-V2	RF	0.7225	0.7267	30.46	81.71
M-V2	DT	0.6258	0.6219	30.46	48.56
E-B0	SVM	0.6416	0.5598	29.04	53.30
E-B0	ANN	0.3255	0.1625	29.04	12.75
E-B0	KNN	0.7799	0.767	29.04	2.79
E-B0	RF	0.7791	0.7729	29.04	106.45
E-B0	DT	0.7019	0.6953	29.04	67.29
V16	SVM	0.5463	0.475	41.93	21.77
V16	ANN	0.3417	0.1684	41.93	9.31
V16	KNN	0.713	0.7119	41.93	1.35
V16	RF	0.7133	0.7118	41.93	9.88
V16	DT	0.6372	0.636	41.93	2.50

To obtain a deeper understanding of the classifier behavior across the three disease categories, a per-class analysis was conducted using precision, recall, and F1-score metrics, complemented by a confusion matrix. The evaluation focuses on the best-performing configuration, ResNet50 combined with ANN, which demonstrated the strongest discriminative capability among all DCNN-ML combinations.

The confusion matrix (Figure 11) indicates that the classifier achieves perfect recall (1.000) for both Anthracnose and Fruit Fly, correctly identifying all samples in these categories. This suggests that the residual features extracted by ResNet50 contain highly distinctive cues that are effectively separated by the ANN's non-linear decision boundaries.

For Healthy Guava, the model achieves a recall of 0.989, with only a single misclassification where a healthy sample is predicted as fruit_fly. This minor confusion aligns with visual overlap in surface texture and coloration patterns between early-stage fruit-fly infestations and certain healthy fruit appearances an issue commonly reported in plant-disease visual datasets [1], [2].

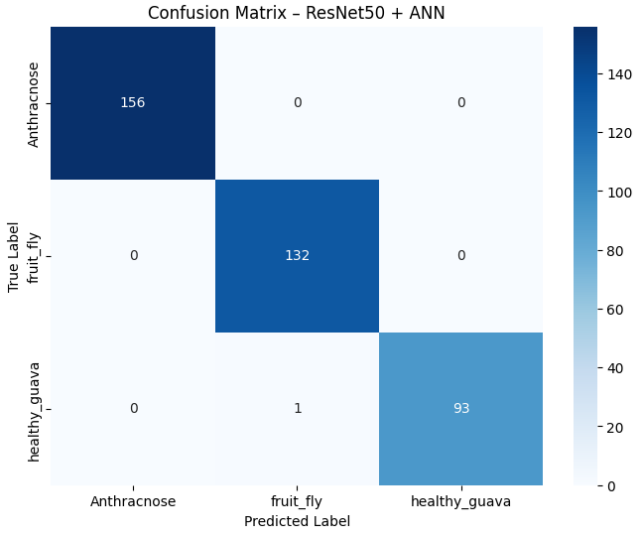

Figure 11. Confusion Matrix for ResNet50 + ANN Model

The per-class precision further reinforces this trend. Both Anthracnose and Healthy Guava achieve perfect precision (1.000), meaning that the model makes no false positive predictions for these categories. Fruit Fly exhibits a slightly lower precision of 0.992, driven by a single false-positive prediction originating from the Healthy class. Nevertheless, the F1-scores for all classes remain extremely high, ranging from 0.9947 to 1.000, demonstrating highly balanced performance.

Overall, the model yields an accuracy of 0.9974, with a macro-average F1-score of 0.9970, reflecting consistent performance across all classes regardless of class size. This is particularly noteworthy given the initial class imbalance in the dataset. The strong per-class metrics confirm that the combination of ResNet50 feature embeddings and ANN classification provides robust separability, even for visually similar categories.

These findings validate that the near-perfect performance is not a consequence of class dominance or metric skewness, but rather of the model's ability to capture fine-grained disease-specific patterns. The confusion matrix provides further evidence that the model generalizes well within the dataset and maintains consistent decision boundaries across disease types.

B. Computational Efficiency Analysis and Trade-offs

Computational efficiency was assessed based on feature extraction time and classifier training time. ResNet50 (R50) achieved the best balance between accuracy and efficiency, requiring only 30.22 seconds for feature extraction approximately 2.5× faster than DenseNet201 (78.06

seconds). Furthermore, the combination of R50 + KNN demonstrated the shortest training time (5.73 seconds), highlighting the computational efficiency of distance-based classifiers.Although computational data for InceptionV3 were not available from the reference study, the results show that ResNet50 offers the most favorable trade-off between accuracy and computational cost, making it a promising candidate for real-time or large-scale agricultural diagnostic systems.

TABLE V
EFFICIENCY COMPARISON

Feature Extraction	Best Acc	Best (Ext Time) (s)	Best (Train Time) (s)*
ResNet50 (R50)	0.9979 (ANN)	30.23	5.73 (KNN)
DenseNet201 (D201)	0.9971 (ANN)	78.06	5.21 (KNN)
InceptionV3 (I-V3) (*SOTA*)	0.9974 (SVM)	NaN	NaN
MobileNetV2 (M-V2)	0.8161 (ANN)	30.47	2.89 (KNN)
EfficientNetB0 (E-B0)	0.7799 (KNN)	29.05	2.79 (KNN)
VGG16 (V16)	0.7133 (RF)	41.93	1.36 (KNN)

C. *Evidence of Performance Visualization and Robustness*

To reinforce the quantitative findings, a visual analysis is presented using two plots. The Scatter Plot (Figure 12) graphically validates the efficiency trade-off, placing ResNet50 (R50) as the most ideal configuration, combining high accuracy with low extraction time. Meanwhile, the Accuracy Heatmap (Figure 13) provides evidence of feature robustness across architectures. The dark-colored regions corresponding to DenseNet201 (D201) and ResNet50 (R50) indicate highly discriminative feature representations, whereas the lighter regions associated with EfficientNetB0 (E-B0) and VGG16 (V16), particularly when paired with ANN or SVM classifiers, illustrate the failure of lightweight architectures in producing consistent feature quality.

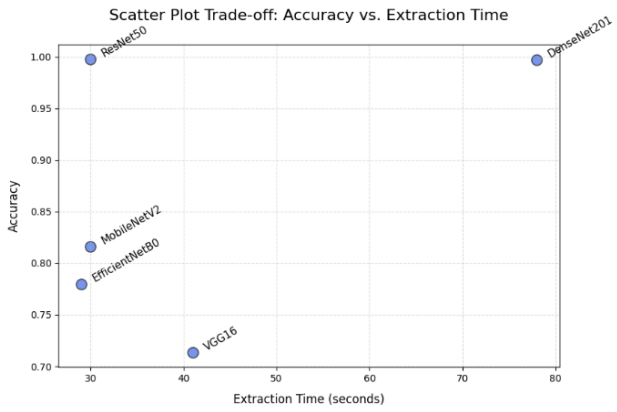

Figure 12. Scatter Plot Trade-off between Accuracy and Extraction Time

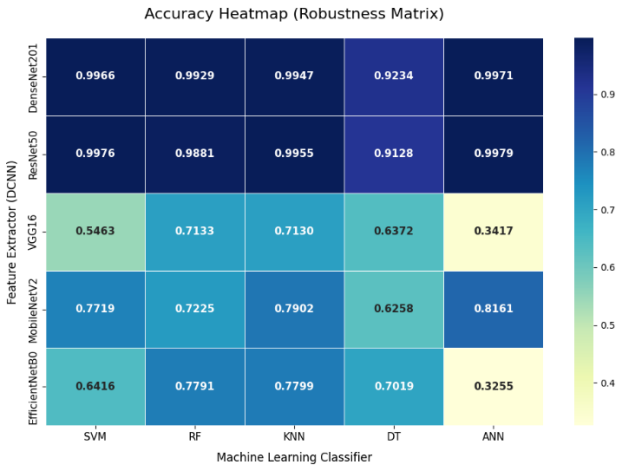

Figure 13. Accuracy Heatmap (Robustness Matrix)

D. *Architectural Justification and SOTA Enhancement*

This discussion focuses on the interpretation of the numerical results presented in Table IV and Table V, as well as their practical implications for model selection for plant disease classification applications.

1) *Advantages of ResNet50 Features and Quantitative Evidence:* The peak performance achieved by ResNet50 (R50) is clear evidence of its architectural superiority. This model, with Residual Blocks and Skip Connections, achieved a maximum accuracy of 0.9979 and an F1 Score of 0.9975 (with an ANN classifier). This performance definitively surpasses the InceptionV3 baseline (0.9974). ResNet50's (R50) ability to maintain network depth without adding significant computational overhead results in more discriminative feature representations, establishing ResNet50 (R50) as the architecture with the highest SOTA performance in this study.

2) Implication of Practical Efficiency Trade-offs (R50 vs. D201): Efficiency analysis shows that ResNet50 (R50) excels not only in accuracy but also in computational speed. The feature extraction time achieved is 30.23 seconds, while DenseNet201 (D201) requires 78.06 seconds, or about 2.5 times longer. This difference indicates that the increased complexity of the D201 architecture does not result in commensurate accuracy gains (accuracy = 0.9971). Thus, ResNet50 can be considered the most efficient configuration in terms of the trade-off between accuracy and computation time, and the most promising for practical implementation in digital agriculture-based classification systems.

E. *Robustness of Classifier Features and Behavior to Data Quality*

1) *Sensitivity of Complex Classifiers and Catastrophic Failures*: The analysis results show that complex classifiers such as Support Vector Machine (SVM) and Artificial

Neural Network (ANN), despite achieving SOTA accuracy with features from ResNet50, are highly sensitive to feature quality. When combined with lightweight feature extractors such as EfficientNetB0 (E-B0) and VGG16 (V16), the performance of both classifiers decreases dramatically, with accuracies of only 0.3255 and 0.3417. This phenomenon indicates that complex classifiers require stable and semantically rich feature representations. This failure proves that features from lightweight DCNN architectures are not yet capable of providing optimal class separation in high-dimensional feature spaces.

2) *Distance-Based Resilience Classifier and Stability Implication*: In contrast, K-Nearest Neighbors (KNN) and Random Forest (RF) show higher resilience to diverse feature qualities. Both models demonstrate relatively stable performance even on DCNN architectures with limited feature extraction capabilities. For example, RF with EfficientNetB0 still records an accuracy of 0.7791, indicating robustness to feature noise. This analysis shows that although ANN (R50) is the main SOTA configuration, distance-based models such as KNN and RF are still worth considering for implementation in real-world environments with high data variability and potential noise.

F. Threats to Validity

Several factors may influence the reliability and generalizability of the experimental results presented in this study. A primary threat to internal validity concerns the possibility of data leakage, which is a common issue in image-based machine-learning tasks where visually similar samples may unintentionally be distributed across both training and validation folds. To mitigate this, all original dataset partitions were first combined into a single unified pool before any preprocessing was applied. The 10-fold cross-validation was generated at this stage, ensuring that fold construction occurred prior to any transformation that might introduce cross-fold dependencies. Class balancing via the NearMiss undersampling method was performed strictly within each training fold, preventing validation and test samples from influencing the resampling process. Additionally, all DCNN architectures were used purely as frozen feature extractors, so that the representations of validation images remained fully independent from those of training images. No image augmentation was applied, thereby eliminating the risk of augmented variants of the same fruit appearing in different folds. These precautions significantly reduce the chance of information leakage, although the dataset itself may still contain multiple photographs of the same or similar fruits, which cannot be completely eliminated without metadata-level grouping.

External validity may also be affected by characteristics of the dataset. The guava images used in this study were captured under controlled field conditions with relatively clear visibility of the fruit. Real agricultural environments often involve variable illumination, occlusions from leaves or branches, inconsistent camera angles, and natural noise, all of which may degrade performance. Although the model achieves near-perfect accuracy on the dataset, this does not guarantee that similar performance will be replicated under operational farm conditions. Furthermore, the dataset covers only three diagnostic categories Anthracnose, Fruit Fly, and Healthy while real-world farms may exhibit additional diseases, nutrient deficiencies, or physical damage that are not represented. As a result, the model's applicability is limited to the specific classes included in the dataset.

Construct validity is influenced by the methodological choices made in this study. All DCNN backbones were evaluated using frozen weights, which ensures consistency across models and prevents overfitting, but also restricts their ability to learn guava-specific texture patterns that may not be fully captured by ImageNet training. Likewise, the hyperparameters of classical machine-learning classifiers were selected based on widely recommended configurations rather than exhaustive search. Although this approach reflects typical usage in practical applications, different hyperparameter choices could influence comparative rankings between models.

Finally, conclusion validity may be affected by the relatively small number of misclassifications observed, which results in performance metrics that are sensitive to small fluctuations. Although 10-fold cross-validation provides a strong basis for evaluation, further statistical validation such as repeated cross-validation or bootstrapping would strengthen confidence in the stability of the reported accuracy and F1-scores.

The novelty of this study lies in its comprehensive evaluation of hybrid CNN-machine-learning pipelines for guava disease classification, integrating accuracy, per-class analysis, computational efficiency, and model robustness into a single unified framework. Unlike previous studies that primarily assess end-to-end CNN accuracy, this work focuses on the discriminative quality of deep-transfer features extracted from six DCNN architectures and evaluated using five machine-learning classifiers. The study further contributes a rigorous methodological protocol that includes fold-wise balancing, leakage prevention, warm-up adjusted timing analysis, and per-class performance inspection elements that are rarely combined in agricultural disease-classification research. By surpassing the previously reported InceptionV3 benchmark [5] and providing evidence of stability across accuracy, runtime, and robustness matrices, this study establishes a more holistic and practically informed foundation for selecting feature extractors and classifiers in real-world agricultural systems.

These combined analyses confirm that ResNet50 + ANN is not only the most accurate but also the most computationally balanced configuration, offering a practical and theoretically grounded solution for guava disease classification.

IV. CONCLUSION

This study identified the most effective and computationally efficient DCNN-ML hybrid configuration for guava fruit disease classification. The combination of ResNet50 feature embeddings with an Artificial Neural Network (ANN) achieved the highest performance, reaching an accuracy of 0.9979 and an F1-score of 0.9975, thereby surpassing the previously reported InceptionV3 benchmark of 0.9974. The efficiency analysis further demonstrated that ResNet50 offers an optimal trade-off between accuracy and processing time, achieving feature extraction approximately 2.5× faster than DenseNet201. These findings confirm that ResNet50 represents the most suitable architecture for high-precision and time-efficient plant disease identification.

While the proposed configuration achieves near-perfect performance, future work may focus on improving lightweight architectures such as MobileNetV2 and EfficientNetB0 through techniques like knowledge distillation, pruning, or low-rank adaptation to enable deployment on low-resource or edge-device environments. Additionally, broader evaluation on real-world field images, diverse disease categories, and more variable environmental conditions would help strengthen model generalizability. Further statistical validation methods, including repeated cross-validation or bootstrapping, may also enhance confidence in the stability of the reported metrics.

DAFTAR PUSTAKA

[1] A. Varma, D. Kumar, P. Date, P. Dipke, and S. Wankhade, "Comparative Analysis of Plant Disease Detection System," International Journal For Multidisciplinary Research, vol. 7, May 2025, doi: g9kvdw.

[2] J. Naranjo-Torres, M. Mora, R. Hernández-García, R. J. Barrientos, C. Fredes, and A. Valenzuela, "A review of convolutional neural network applied to fruit image processing," Applied Sciences (Switzerland), vol. 10, no. 10, May 2020, doi: 10.3390/app10103443.

[3] D. Anwar, "A Novel CNN Model for Fruit Leaf Disease Detection: A Lightweight Solution for Grapes, Figs, and Oranges," Fusion: Practice and Applications, vol. 19, no. 2, pp. 278–287, 2025, doi: 10.54216/FPA.190220.

[4] M. T. Ahad, Y. Li, B. Song, and T. Bhuiyan, "Comparison of CNN-based deep learning architectures for rice diseases classification," Artificial Intelligence in Agriculture, vol. 9, pp. 22–35, Sep. 2023, doi: 10.1016/j.aiia.2023.07.001.

[5] O. Kılcı and M. Koklu, "Guava Fruit Disease Classification Using Deep Learning and Machine Learning Models," Research in Agricultural Sciences, vol. 56, no. 3, pp. 217–226, Sep. 2025, doi: 10.17097/agricultureatauni.1665941.

[6] Amin, Md Al; Mahmud, Md Iqbal; Rahman, Asadullah Bin; Parvin, Mst Aktarina; Mamun, Md Abdulla Al (2024), "Guava Fruit Disease Dataset", Mendeley Data, V1, doi: 10.17632/bkdkc4n835.1.

[7] J. Schmidhuber, "Who invented deep residual learning?," Sep. 2025. [Online]. Available: https://people.idsia.ch/~juergen/who-invented-residual-neural-networks.html

[8] L. Nanni, G. Faldani, S. Brahnam, R. Bravin, and E. Feltrin, "Improving Foraminifera Classification Using Convolutional Neural Networks with Ensemble Learning," Signals, vol. 4, no. 3, pp. 524–538, Sep. 2023, doi: 10.3390/signals4030028.

[9] W. Li, W. Xie, and Z. Wang, "Complex-Valued Densely Connected Convolutional Networks," in Communications in Computer and Information Science, Springer, 2020, pp. 299–309. doi: 10.1007/978-981-15-7981-3_21.

[10] F. A. Abdulazeez, I. T. Ahmed, and B. T. Hammad, "Examining the Performance of Various Pretrained Convolutional Neural Network Models in Malware Detection," Applied Sciences (Switzerland), vol. 14, no. 6, Mar. 2024, doi: 10.3390/app14062614.

[11] S. Hussain Khan and R. Iqbal, "A Comprehensive Survey on Architectural Advances in Deep CNNs: Challenges, Applications, and Emerging Research Directions," Mar. 2025.

[12] T. Singh and D. K. Vishwakarma, "A deeply coupled ConvNet for human activity recognition using dynamic and RGB images," Neural Comput Appl, vol. 33, no. 1, pp. 469–485, Jan. 2021, doi: 10.1007/s00521-020-05018-y.

[13] G. X. Shi, Y. N. Wang, Z. F. Yang, Y. Q. Guo, and Z. W. Zhang, "Wildfire Identification Based on an Improved MobileNetV3-Small Model," Forests, vol. 15, no. 11, Nov. 2024, doi: 10.3390/f15111975.

[14] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," International Conference on Machine Learning, Sep. 2020, [Online]. Available: http://arxiv.org/abs/1905.11946

[15] C. Su and W. Wang, "Concrete Cracks Detection Using Convolutional NeuralNetwork Based on Transfer Learning," Math Probl Eng, vol. 2020, 2020, doi: 10.1155/2020/7240129.

[16] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet for the 2020s," Mar. 2022, [Online]. Available: http://arxiv.org/abs/2201.03545

[17] A. Mustapha, L. Mohamed, H. Hamid, and K. Ali, "Diabetic Retinopathy Classification Using ResNet50 and VGG-16 Pretrained Networks," International Journal of Computer Engineering and Data Science, vol. 1, pp. 2737–8543, Jul. 2021, [Online]. Available: www.ijceds.com

[18] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Jun. 2018, [Online]. Available: http://arxiv.org/abs/1201.0490

[19] A. Rahmadeyan and M. Mustakim, "Seleksi Fitur pada Supervised Learning: Klasifikasi Prestasi Belajar Mahasiswa Saat dan Pasca Pandemi COVID-19," Jurnal Nasional Teknologi dan Sistem Informasi, vol. 9, no. 1, pp. 21–32, May 2023, doi: 10.25077/teknosi.v9i1.2023.21-32.

[20] P. K. Syriopoulos, N. G. Kalampalikis, S. B. Kotsiantis, and M. N. Vrahatis, "kNN Classification: a review," Ann Math Artif Intell, vol. 93, no. 1, pp. 43–75, Feb. 2025, doi: 10.1007/s10472-023-09882-x.

[21] L. D. Cahya, A. Luthfiarta, J. I. T. Krisna, S. Winarno, and A. Nugraha, "Improving Multi-label Classification Performance on Imbalanced Datasets Through SMOTE Technique and Data Augmentation Using IndoBERT Model," Jurnal Nasional Teknologi dan Sistem Informasi, vol. 9, no. 3, pp. 290–298, Jan. 2024, doi: 10.25077/teknosi.v9i3.2023.290-298.

[22] A. Miteloudis, "Interpretability in Modern Decision Tree Ensembles: A Meta-Review," Jul. 2025. doi: 10.5281/zenodo.15853691.

[23] J. Ansel et al., "PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation," in International Conference on Architectural Support for Programming Languages and Operating Systems - ASPLOS, Association for Computing Machinery, Apr. 2024, pp. 929–947. doi: 10.1145/3620665.3640366.