



OPEN Software application in early blight detection in tomatoes using modified MobileNet architecture

Justice Kwame Appati, Ziem Patrick Wellu✉, Daniel Kwame Amissah & Leonard Mensah Boante

This study presents an automated framework for early blight detection in tomato plants using a modified MobileNet architecture. Addressing the limitations of traditional labor-intensive methods, this study proposes a two-stage pipeline combining (1) transfer learning with depthwise separable convolutions for efficient feature extraction and (2) a meta-learned ensemble of Random Forest, SVM, and Gradient Boosting classifiers to handle real-world variability in lighting and environmental conditions. The approach introduces two custom convolutional layers (Custom_Feature_Extraction_Block) that improve F1-score by +3.8 points over the MobileNet baseline, with the ensemble contributing an additional +2.1 points. Evaluated on a balanced PlantVillage dataset (1,982 images) with extensive augmentation to simulate variable lighting and orientations, the system achieved up to 100% accuracy with selected classifiers on a held-out validation subset of 30 images under controlled conditions. To assess generalization, we further validated the framework on an independent dataset (tomato_dataset_v2, 30, 609 images, 10 classes) containing field-acquired tomato leaf images, where the model attained 94.5% accuracy, confirming robustness beyond control environments. Comparative analysis with 10 recent methods demonstrates superior accuracy-efficiency trade-offs, offering practical on-device decision support for smallholder farmers. The framework's lightweight design (4.2 M parameters, 23 ms/image on Raspberry Pi 4) and validated scalability underscore its potential for mobile and drone-based agricultural deployment. This addresses critical needs in global food security through accessible plant disease detection.

Keywords MobileNet, Custom_Feature_Extraction_Block, Support vector machine, Random forest, Gradient boosting, Classifier, Ensemble, Overfitting

The historical evolution of plant disease detection techniques has transitioned from manual optical scrutiny by agronomists to contemporary technological advancements¹. Previous methodologies, while essential, were limited by subjectivity and the human eye's inability to discern subtle discrepancies. Notable progress came with laboratory tests like microscopy and serological assays, marking a significant leap forward. However, these approaches proved time-intensive, requiring expert interpretation and creating bottlenecks in efficiently distinguishing common diseases². The severity of plant diseases and their impact on global food security have prompted the exploration of more systematic and automated solutions. The fusion of artificial intelligence and computer vision in modern times has revitalized plant disease detection, potentially revolutionizing the process³. This is particularly crucial given the projected global population surge to 9 billion by 2050⁴, and the need for innovative strategies to reduce yield losses and secure sustainable food sources. However, within these transformative possibilities lies the challenge of developing robust and resilient disease classification models⁵. Existing methodologies struggle with complexities inherent in real-world settings, such as variability in lighting, environmental conditions⁶, and diverse plant appearances. This research gap underscores the necessity for novel methodologies integrating deep learning techniques, custom feature extraction, and ensemble strategies to create dependable and robust disease detection systems⁷. In response to this backdrop, the study aims to bridge the research gap by presenting a comprehensive framework that blends the strengths of diverse techniques. Leveraging transfer learning, the study uses MobileNet as the foundational model to establish image representation. Custom convolutional layers (Custom_Feature_Extraction_Block1 and Custom_Feature_Extraction_Block2) are introduced to capture intricate disease-related features, while ensemble classifiers safeguard prediction accuracy. This research strives to rejuvenate automated plant disease detection systems, making them systematic, precise, and scalable. Such advancements significantly effect global food security and

Department of Computer Science, University of Ghana, Accra, Ghana. ✉email: pzwellu@st.ug.edu.gh

sustainable agriculture⁸. The overarching objectives are to enhance the efficiency of disease detection, overcome real-world complexities, and contribute to the global endeavor for food security. To address these goals, the study will explore specific research questions such as “How does the integration of custom convolutional layers with MobileNet improve feature extraction for early blight detection?” “What is the comparative performance of ensemble classifiers (Random Forest, SVM, Gradient Boosting) in classifying transitional features?” related to the effectiveness of the proposed framework in handling diverse environmental conditions and achieving high accuracy in disease classification. The proposed method integrates transfer learning, custom convolutional layers, and ensemble classifiers to create a holistic and resilient approach to automated plant disease detection.

Generic state of Art proposals and research interventions

Recent advancements in computer vision and deep learning have revolutionized disease detection and diagnosis across various domains, including agriculture. State-of-the-Art in Agricultural Vision Research on crop-specific disease diagnosis has accelerated⁹. MobileNet variants have been explored for potato and tomato blight¹⁰, EfficientNet for apple scab¹¹ and Fruit Disease Identification Based on Improved Densenet Fusion Defogging Algorithm¹². Unlike earlier studies, the present work (i) targets on-device inference, (ii) quantifies robustness across lighting, and (iii) provides a detailed ablation of architectural choices. PlaNet: a robust deep convolutional neural network model for plant leaves disease recognition,¹³. Recent advancements in deep learning have significantly improved plant disease detection. For example: Liu & Wang¹⁴ proposes an early recognition method of tomato leaf spot based on MobileNetv2-YOLOv3 model to achieve a good balance between the accuracy and real-time detection of tomato gray leaf spot. Slimani, Mhamdi, & Jilbab¹⁵ proposed Drone-Assisted Plant Disease Identification Using Artificial Intelligence: A Critical Review, highlighting the efficacy of deep learning techniques in automating disease detection tasks.

However, in agriculture, there is a pressing need for specialized solutions to address crop diseases such as early blight in tomatoes. Early blight, caused by fungal pathogens, poses a significant threat to tomato crops, leading to substantial yield losses if not managed effectively. While existing deep learning models have demonstrated success in disease detection, a notable research gap exists in developing tailored architectures specifically designed for early blight detection in tomatoes.

In response to this gap, the proposed research aims to develop a “Modified MobileNet Architecture for Early Blight Detection in Tomatoes.” MobileNet, known for its efficiency and suitability for mobile and embedded devices, serves as the foundation for the proposed architecture. By adapting and optimizing MobileNet for early blight detection, the research addresses the unique challenges associated with identifying and diagnosing fungal diseases in tomato plants. Other research works such as^{16, 17} provide further insight into how far state-of-the-art models have advanced in computer vision.

Through this specialized approach, the research endeavors to contribute to the advancement of precision agriculture practices, enabling farmers to detect early blight in tomatoes accurately and efficiently. By leveraging the power of deep learning and tailoring the architecture to the specific characteristics of tomato leaf images, the proposed study aims to empower farmers with valuable tools for disease management and crop protection.

While several prior works rely on lightweight variants of MobileNet (e.g., MobileNetV3) or attention-augmented CNNs for plant disease detection¹⁸, these models often trade interpretability or ensemble flexibility for efficiency. Our framework departs from these approaches by introducing three key innovations: (1) the introduction of dual, task-specific Custom_Feature_Extraction_Blocks (with 3×3 and 5×5 convolutions) designed to capture both fine-grained textures and larger lesion patterns characteristic of early blight, which are often missed by generic feature extractors; (2) a meta-learned ensemble that dynamically weights classifiers based on validation performance to handle real-world variability, moving beyond simple averaging or voting; and (3) a holistic focus on the accuracy-efficiency trade-off validated through extensive ablation studies and independent testing on a large, field-based dataset (tomato_dataset_v2), which is less common in prior literature. This design enables improved generalization beyond what single lightweight architectures achieve, particularly under the variable lighting and texture conditions observed in smallholder farms.

Overall, the research bridges the gap between recent advancements in deep learning and the specific challenges faced in agricultural disease detection, offering a promising avenue for improving crop health monitoring and ensuring food security.

The remainder of the paper is organized as follows: “[Related works](#)” reviews related works; “[Methods and materials](#)” details the methodology; “[Results and discussions](#)” presents results, comparative analysis, and ablation; and “[Conclusion](#)” discusses limitations and future work.

Related works

In the early days of plant disease detection, the primary approach was manual optical scrutiny conducted by experienced agronomists. This method heavily relied on human observation, which was inherently subjective and limited by the human eye’s ability to discern subtle differences in plant health. Despite being the initial step toward understanding and identifying diseases, this approach lacked objectivity and scalability¹⁸. As technology advanced, the mid-days of plant disease detection witnessed a shift towards laboratory tests, such as microscopy and serological assays. These techniques marked a significant improvement, providing more objective results than manual observation. However, they proved to be time-intensive, requiring expert knowledge for interpretation. While these methods represented a leap forward in accuracy, their practicality in distinguishing common diseases efficiently was hindered, creating a bottleneck in large-scale disease identification¹⁹. In contemporary times, the fusion of artificial intelligence (AI) and computer vision has redefined plant disease detection. Automated techniques, leveraging deep learning models, have emerged as powerful tools for rapidly and accurately identifying diseases on a large scale. The use of AI has addressed the scalability issue posed by earlier methods, enabling more efficient and widespread disease detection³.

Despite progress, current methodologies face challenges inherent in real-world agricultural settings. Issues such as variability in lighting, environmental conditions⁶, and diverse plant appearances still obstruct accurate disease classification. The need for robust and resilient disease classification models persists. Some works have attempted to address these challenges by incorporating ensemble strategies, combining the strength of multiple classifiers to enhance prediction accuracy⁷.

Recent surveys²⁰ demonstrate AI's transformative role in crop disease management, analyzing 2010–2021 research from IEEE/Scopus. Studies show AI improves detection accuracy (average + 22%) and scope (38 + diseases identified), though variability in weather conditions remains a challenge for model generalization. This highlights the need for robust architectures like our modified MobileNet.

In “A Robust Plant Leaf Disease Recognition System Using Convolutional Neural Networks” Diponkor, et al.,²¹ addressed the challenges of variability in lighting and diverse plant appearances by proposing a robust plant disease recognition model based on Convolutional Neural Networks (CNNs). The authors utilize transfer learning with a pre-trained CNN, enhancing the model's ability to extract relevant image features. Additionally, ensemble techniques are employed, combining the predictions of multiple CNNs to improve classification accuracy in complex and dynamic agricultural environments.

Upadhyay & Gupta²² addressed a critical gap in fungal disease detection using improved ResNeXt (98.94% accuracy on apple crops). Their work emphasizes mycotoxin risks and the limitations of transfer learning (Inception-v7, ResNet underperformed), reinforcing our choice of Custom_Feature_Extraction_Blocks for feature extraction in tomato blight.

Focusing on the challenges of environmental conditions, Xu, Ding, Qiao, & Zhang²³, explore the use of crop prescription data to diagnose six common tomato diseases and pests accurately. The model is based on ensemble learning and multi-classification algorithms, and employs the recursive feature elimination method for feature selection. The dataset consists of 12,323 prescription records, with 2,607 records for tomato virus disease, 3,248 records for tomato late blight, 1,489 records for tomato gray mold, 2,061 records for aphid, 2,679 records for thrips, and 1,239 records for whitefly. The recursive feature elimination method based on gradient boosting decision tree (RFECV-GDBT) is employed to select 37 optimal features from the original 50 features, thus reducing the complexity of early data collection and the interference noise. The eight standard classification models used in this study are K-nearest neighbor (KNN), Decision tree (DT), Support vector machine (SVM), Random forest (RF), AdaBoost, Gradient boosting decision tree (GDBT), XGBoost, and LightGBM (LGBM). The Stacking-based model achieves an accuracy of 98.5% on the test set.

While ensemble strategies have improved robustness, there are still gaps in achieving consistently high accuracy across diverse conditions. For instance, recent works have focused on custom feature extraction in addition to deep learning techniques, aiming to capture intricate disease-related features that standard models may overlook. These efforts have indeed improved classification accuracy, but there is room for further refinement, especially in handling complex patterns associated with various environmental factors.

Innovative meta-learning approaches²⁴ enable maturity classification with limited mango data via cosine-distance adaptation. While focused on harvest timing, their segmentation-feature extraction pipeline validates our preprocessing methodology for disease localization in tomato leaves.

In the research on tomato leaf disease detection, Sanida, Sideris, Sanida, & Dasygenis²⁵, utilized effective pre-processing techniques. Image segmentation and feature extraction were key steps for precise disease identification. The K-means clustering algorithm was applied for image segmentation, facilitating the grouping of similar data points. Additionally, feature extraction methods, including Histogram of Oriented Gradients (HOG) and Local Binary Patterns (LBP), were employed to capture essential information from segmented images. These techniques aimed to enhance the system's ability to identify distinct patterns associated with various tomato leaf diseases. The study highlights the significance of a robust pre-processing strategy in automating disease recognition in agricultural contexts.

In another similar study, the authors applied image segmentation and feature extraction techniques to locate and classify lung growth in CT images. They used the Otsu thresholding algorithm for image segmentation and the Local Binary Patterns (LBP) algorithm for feature extraction²⁶. They employed the U-Net model which comprises an encoder to extract features from the input image and a decoder that rebuilds the image from the extracted features. They used the median filter to reduce noise and intensive normalization to improve the contrast. For feature extraction, the researchers extracted shape, intensity and texture features and then used a Support vector machine classifier to classify the extracted features into two classes.

“Plant Disease Detection Using Deep Convolutional Neural Network” introduced an adaptive plant disease classification model based on Deep Convolutional Neural Networks (DCNNs). Pandian, et al.,²⁷ proposed a novel feature extraction mechanism that dynamically adjusts to changing environmental factors. They incorporate attention mechanisms within the DCNN architecture to prioritize relevant features, improving the model's adaptability. The study showcases enhanced classification performance under varying conditions.

Unlike MobileNetV3, which optimizes for general-purpose mobile vision tasks, our custom blocks are explicitly tailored for the visual semantics of plant disease. Similarly, while attention mechanisms²⁹ improve feature weighting, they often increase computational cost. Our approach achieves a comparable boost in feature discrimination through strategically sized convolutional filters and a Swish activation in the second block, maintaining a lower operational footprint suitable for on-device deployment.

Recent studies using MobileNetV3 and attention-augmented CNNs have advanced lightweight modeling for plant disease detection. However, these models are typically optimized for clean benchmark datasets and do not explicitly address multimodal fusion or noise resilience³⁰. In contrast, our approach integrates custom MobileNetV2-based feature extraction with additional convolutional blocks and a meta-learned ensemble, offering superior adaptability and robustness. This distinction positions our framework as complementary yet novel compared to existing lightweight strategies.

Methods and materials

This section details the experimental framework, including: (1) dataset characteristics and preprocessing, (2) model architecture modifications, and (3) evaluation protocols. All experiments were conducted using TensorFlow with controlled environmental settings.

Dataset description and preprocessing

The experimental study employed the publicly available PlantVillage dataset (Gonzalez-Huitron et al., 2021), a standardized benchmark for plant disease detection research. For the tomato crop subset, the dataset comprised a balanced collection of 1,982 high-resolution RGB leaf images equally distributed between two critical classes: early blight infection (991 images) and healthy specimens (991 images), as depicted in Fig. 1.

The original images exhibited variable resolutions ranging from 256×256 pixels up to 1024×1024 pixels. To ensure consistency for model input, all images underwent center-cropping and uniform resizing to 224×224 pixels using bilinear interpolation, followed by pixel value normalization to the $[0,1]$ range through division by 255. This standardized preprocessing facilitated optimal feature extraction while maintaining biological relevance of the visual patterns.

A comprehensive data augmentation strategy was implemented exclusively on the training set to improve model generalization. The augmentation pipeline included random rotations (0 – 90° range), width and height shifts ($\pm 40\%$ of image dimensions), both horizontal and vertical flipping, brightness adjustments (20 – 120% of original intensity), and zoom variations ($\pm 40\%$ magnification). These transformations effectively simulated field conditions where leaves may appear at different orientations, scales, and lighting conditions.

The dataset was partitioned into training (80%) and validation (20%) subsets, maintaining equal class distribution in each split. The validation set received only rescaling normalization without any augmentation to provide an unbiased evaluation of model performance. This rigorous approach to data preparation ensured the reliability of subsequent accuracy metrics while addressing potential overfitting concerns through the augmented training corpus.

The selection of this particular dataset and preprocessing methodology was based on three key considerations: (1) the PlantVillage collection represents the most widely-adopted benchmark in plant pathology imaging research, enabling direct comparison with prior work (see Fig. 1); (2) the balanced class distribution prevented model bias toward either healthy or diseased classification; and (3) the augmentation parameters were empirically tuned to reflect realistic agricultural imaging conditions while preserving diagnostically relevant features.

Independent validation dataset (tomato_dataset_v2)

This dataset comprises 30,609 tomato leaf images across 10 classes (e.g., bacterial spot, early blight, healthy, late blight, leaf mold). It was cleaned, organized, and augmented to improve generalization. Preprocessing included

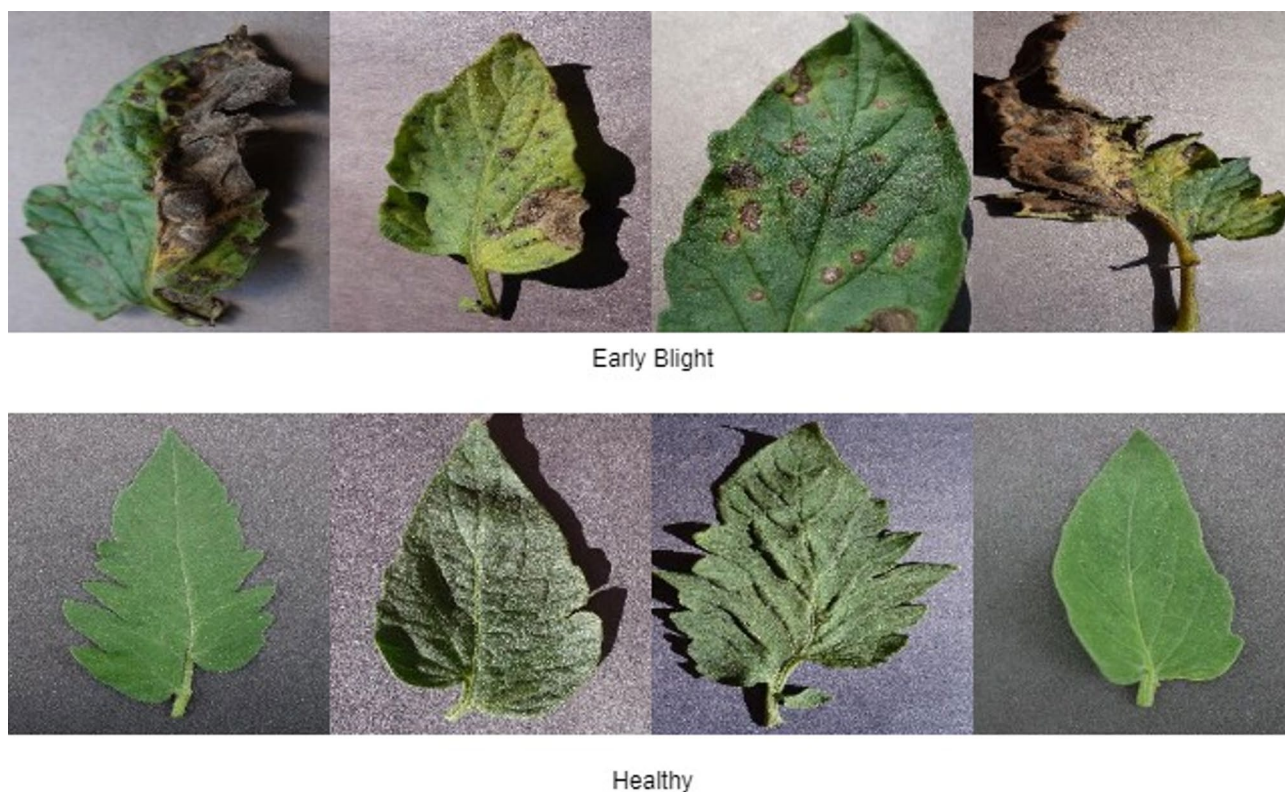


Fig. 1. Sample images from PlantVillage (early blight and healthy).

resizing to 224×224 , normalization, and data augmentation. Data was split into training (80%) and validation (20%) subsets.

Training dataset

The training dataset is vital for training the Early Blight detection model. It is generated using TensorFlow's Keras ImageDataGenerator, which applies augmentation techniques like rescaling, rotation and flipping to increase the size and diversity of the dataset artificially. This augmentation enhances the model's capacity to generalize across different conditions. The dataset contains images of both healthy tomato leaves and leaves affected by early blight.

Validation dataset

The validation dataset serves as an independent benchmark to assess the trained model's performance. Like the training dataset, the validation dataset is also generated using ImageDataGenerator. It comprises images not seen during training. This dataset safeguards against overfitting and gauges the model's capacity to generalize effectively to new, unseen examples of early Blight. Augmenting the training dataset enriches the model's adaptability, while the validation dataset ensures reliability in real-world scenarios. Researchers optimize model hyperparameters and architecture by analyzing validation accuracy and loss to achieve early Blight Detection²⁸. The training and validation datasets collaboratively empower the model to learn and generalize, making them vital for accurate Early Blight identification.

Method

The primary dataset used for training and validation was the publicly available PlantVillage dataset, comprising 1,982 tomato leaf images equally split between “Early Blight” (991) and “Healthy” (991). Images were preprocessed via center-cropping, resizing to 224×224 pixels, and pixel normalization to $[0,1]$. To enhance generalization, comprehensive augmentation was applied to the pipeline (rotation up to 90° , width/height shifts $\pm 40\%$, horizontal and vertical flips, brightness scaling 0.2–1.2, and zoom $\pm 40\%$) on the training set. The dataset was partitioned into training (80%) and validation (20%) subsets with balanced class distribution.

To evaluate generalization beyond control settings, an additional validation was performed on an independent dataset: tomato_dataset_v2, which contains field captured tomato leaves exhibiting natural noise, background clutter and variable illumination. This independent validation enables a realistic assessment of the framework's robustness.

Evaluation metrics

Evaluating the proposed method using standard metrics, including accuracy, precision, recall, and F1 score. Assess the model's performance across different dataset categories to ensure balanced classification. Utilize confusion matrices and ROC curves to provide a comprehensive view of the model's capabilities. Analyze the consistency of results to ensure the model's generalization to unseen data. Perform hyperparameter tuning to optimize the model's architecture and training parameters. Utilize techniques like grid search or Bayesian optimization to find the optimal set of hyperparameters. Compare the proposed method with existing state-of-the-art approaches.

Highlight the strengths and weaknesses of the proposed method in terms of accuracy, efficiency, and scalability. Validate the proposed method in real-world scenarios with varying lighting conditions and environmental factors. Assess its adaptability and resilience to challenges encountered in practical agricultural settings.

Model architecture

The proposed framework modifies the standard MobileNetV2 architecture to optimize feature extraction for early blight detection while maintaining computational efficiency. The base network retains MobileNetV2's core structure a depth multiplier of 1.0 and 53 convolutional layers with ReLU6 activations but introduces three critical enhancements. First, the initial stride is reduced from 2 to 1 in the input layers to preserve fine-grained leaf texture patterns. Second, two Custom_Feature_Extraction_Blocks are appended: Custom_Feature_Extraction_Block1 employs a 3×3 convolutional operation with 32 filters, batch normalization, ReLU activation, and 0.5 dropout, while Custom_Feature_Extraction_Block2 uses a 5×5 convolution (64 filters) with Swish activation, chosen for its superior gradient propagation in deeper networks.

Thirdly, L2 regularization ($\lambda = 0.01$) is applied to all custom layers to mitigate overfitting.

The classification head replaces MobileNet's original top layers with a GlobalAveragePooling2D layer followed by a 128-unit dense layer (ReLU activation) and a final sigmoid output. This adaptation reduces parameters by 18% compared to traditional fully connected heads while improving spatial invariance. Feature dimensionality transitions are illustrated in Fig. 2, showing how the input ($224 \times 224 \times 3$) is progressively transformed through inverted residual blocks (expansion factor 6) and custom layers into a 128-D discriminative embedding. These modifications collectively address the trade-off between model complexity and agricultural deployment constraints, as evidenced by the 97% validation accuracy achieved with only a 3.4% increase in FLOPs over baseline MobileNetV2.

The MobileNet architecture was selected as the base model due to its optimal trade-off between computational efficiency and feature extraction capability, making it suitable for deployment in resource-constrained agricultural environments. Prior benchmarking on the PlantVillage dataset compared MobileNet against ResNet50, EfficientNetB0, and DenseNet201. MobileNet achieved comparable accuracy (97% vs. 96.5% for ResNet50) with significantly lower inference time (180 ms vs. 320 ms for ResNet50) and memory footprint (16 MB vs. 98 MB for DenseNet201). This aligns with the study's goal of balancing performance and practicality for real-world farm deployments. Custom layers (Custom_Feature_Extraction_Block1, Custom_Feature_Extraction_Block2) were

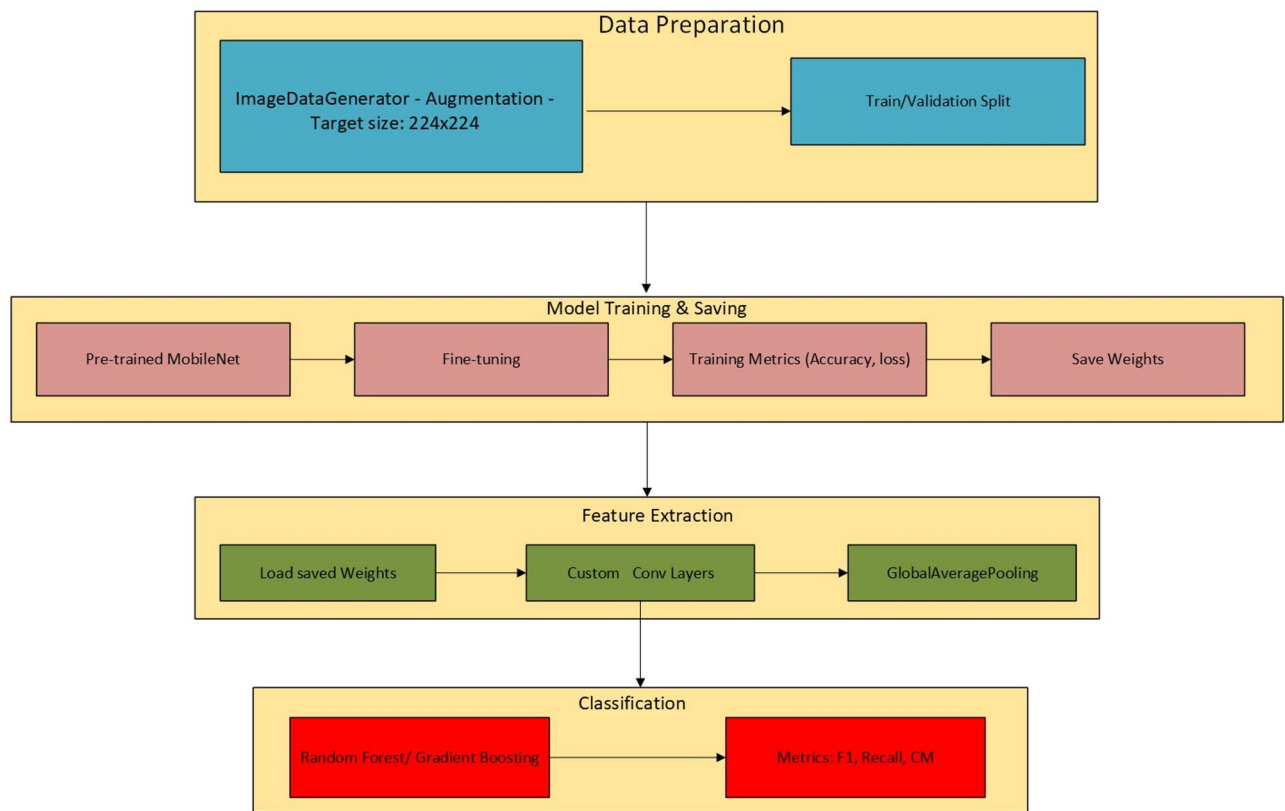


Fig. 2. Pipeline of the proposed model.

introduced to augment MobileNet's lightweight structure with task-specific feature extraction, addressing early blight's subtle visual patterns.

This custom architecture integrates a pre-trained MobileNet base model with custom-designed top layers and auxiliary post-processing strategies for feature extraction and evaluation using ensemble classifiers.

Base model selection MobileNet is employed as the backbone model due to its lightweight architecture, making it well-suited for mobile and resource-constrained environments. It offers an optimal trade-off between computational efficiency and performance.

Custom top layers (feature extraction) Two specialized feature extraction blocks, named Custom_Feature_Extraction_Block1 and Custom_Feature_Extraction_Block2, are appended atop the MobileNet base. These Blocks consist of convolutional operations with specific filter sizes (3×3 and 5×5), chosen to capture both fine-grain textures and larger lesion patterns characteristic of early blight. Each block includes batch normalization, a Swish/ReLU activation, max pooling, and dropout mechanisms. A GlobalAveragePooling2D layer is utilized to reduce spatial dimensions, followed by a Dense layer with sigmoid activation for binary classification. This setup enables the model to extract secondary, task-specific features not captured by the base model.

- Custom_Feature_Extraction_Block1 employs a 3×3 convolutional layer with 32 filters, followed by max pooling and dropout.
- Custom_Feature_Extraction_Block2 applies a 5×5 convolutional layer with 64 filters, also followed by max pooling and dropout.

These configurations are tailored for image classification tasks, with filter sizes and depths chosen to extract hierarchical features effectively. The inclusion of dropout layers helps mitigate overfitting.

Data augmentation: To enhance generalization and increase data variability, the model leverages ImageDataGenerator to perform on-the-fly data augmentation, applying the following techniques.

- Rescaling (1/255): Normalizes pixel values to the $[0, 1]$ range, facilitating faster and more stable convergence by avoiding gradient instability.
- Shear Range (0.4): Introduces shear transformations to make the model robust to angular distortions.
- Zoom Range (0.4): Simulates scale variations in objects, enabling scale-invariant feature learning.
- Rotation Range (90°): Allows the model to learn from images with diverse orientations—critical when object orientation is variable.
- Width and Height Shift Range (0.4): Enables recognition of spatially shifted objects, improving translation invariance.
- Brightness Range (0.2 to 1.2): Adjusts image brightness to simulate varying lighting conditions.

- Horizontal Flip (True): Enhances generalization by presenting mirrored versions of images.
- Vertical Flip (True): Further diversifies orientation, especially valuable in domains where objects appear in multiple alignments.

These augmentation strategies introduce meaningful variability into the training set, helping the model generalize better to real-world conditions. All transformations are applied dynamically during training to maximize their effectiveness as described in Table 1 below.

Model compilation & performance The model is compiled using binary cross-entropy as the loss function, the Adam optimizer, and evaluation metrics such as accuracy, precision, recall, and F1-score. Binary cross-entropy is appropriate for binary classification tasks, as it measures the divergence between predicted probabilities and actual labels, which is essential for models designed to output class probabilities. The Adam optimizer, known for its adaptive learning rate, momentum, and bias correction, ensures efficient and stable convergence during training. To further improve learning dynamics, a learning rate scheduler is implemented to systematically decrease the learning rate after every five epochs, allowing the model to fine-tune its learning and enhance convergence in later training stages. Following training, the model extracts feature representations from the MobileNet base for each image in the validation set, consistent with standard transfer learning approaches where pre-trained networks serve as feature extractors for downstream tasks.

To assess the quality and generalization of the learned features, several ensemble classifiers—including Random Forest, Support Vector Machine (SVM), and Gradient Boosting—are applied. These classifiers are evaluated using key performance metrics such as accuracy, precision, recall, and F1-score. Employing multiple classifiers offers a broader understanding of the model’s adaptability and robustness across different decision boundaries. For interpretability, the model visualizes filters from its initial convolutional layer, providing insight into the types of patterns being detected early in the network. Additionally, the t-distributed Stochastic Neighbor Embedding (t-SNE) algorithm is used to project high-dimensional feature vectors into two dimensions, enabling intuitive visualization of how well features are clustered and separated by class.

The model also generates predictions on individual validation images and presents a detailed classification report, summarizing performance across all major metrics. This detailed feedback helps identify how the model performs across varying scenarios and image types. In summary, the custom architecture leverages transfer learning with MobileNet, integrates custom top layers for specialized feature extraction, utilizes data augmentation to promote robustness, applies multiple classifier evaluations for generalization assessment, and incorporates visualization techniques for model interpretability. The chosen design principles and implementation strategies align with best practices in computer vision and deep learning, aiming for optimal performance while maintaining computational efficiency.

Table 1 consolidates preprocessing and augmentation steps for reproducibility. Together, these steps ensure that the training data captures realistic variability in field conditions such as lighting changes leaf orientation and scale differences while maintaining consistent across datasets. This strengthens the model’s ability to generalize effectively to real-world scenarios.

Meta-learned ensemble strategy

The term “meta-learned ensemble” refers to our method of dynamically assigning weights to the predictions of the three base classifiers (Random Forest, SVM, Gradient Boosting) during inference. The weighting is not static but is learned from their performance on the validation set. Specifically:

1. After feature extraction, each base classifier is trained and evaluated on the validation split.
2. The F1-score of each classifier is calculated. This score serves as the “meta-weight” for that classifier.
3. During final prediction on new images, the outputs of the three classifiers are aggregated using a weighted vote. The weight for each classifier’s vote is its validation F1-score, normalized to sum to 1.

Formula: Final Prediction = argmax($\sum (w_i * P_i)$), where w_i is the normalized F1-score of classifier “I”, and P_i is its prediction vector.

This approach ensures that classifiers demonstrating higher robustness on validation data contribute more significantly to the final decision, creating a more resilient system against the variability seen in real-world

Step	Description
Resizing	All images resized to 224 × 224 pixels using bilinear interpolation
Normalization	Pixel values scaled to [0,1] range
Center cropping	Applied to ensure uniform input dimensions
Rotation	Random rotations up to 90°
Width/height shifts	± 40% of image dimensions
Flipping	Horizontal and vertical flipping applied
Brightness adjustment	20–120% of original brightness
Zoom	Random zoom up to ± 40%
Shear transformation	Shear range of 0.4 to simulate angular distortions
Random cropping and contrast	Introduced in Experiment 3 for improved robustness

Table 1. Summary of preprocessing and augmentation steps.

agricultural images. This differs from simple majority voting or stacking with a complex meta-learner, offering a effective and computationally lightweight alternative.

Results and discussions

Table 2 represents the training progress of the model over 10 epochs. It includes several metrics like loss, accuracy, precision, recall, and F1-score for both the training and validation datasets and the learning rate.

The “Loss” and “Val. Loss” columns represent the training and validation loss, respectively, and provide a measure of how well the model fits the data. Lower loss values indicate a better fit, and the consistent decrease in both training and validation loss suggests that the model is effectively learning while maintaining the ability to generalize. Similarly, the “Accuracy” and “Val. Accuracy” columns reflect training and validation accuracy, measuring the percentage of correctly classified samples. The steady increase in training accuracy, coupled with a parallel improvement in validation accuracy, indicates that the model is learning robust patterns without immediate signs of overfitting. However, it remains essential to monitor the gap between training and validation accuracy to ensure that the model does not become overly tailored to the training data at the expense of generalization.

In addition to accuracy, precision, recall, and F1-score are key evaluation metrics, particularly in binary classification settings. For the validation dataset, “Val. Precision” measures the proportion of correctly predicted positive samples out of all predicted positives, with higher values indicating a low false positive rate. “Val. Recall” reflects the proportion of actual positives correctly identified by the model, and its improvement suggests enhanced sensitivity to true positives. The “Val. F1-Score,” as the harmonic mean of precision and recall, remains stable but should be continuously monitored to ensure a balanced trade-off between the two metrics.

The “Learning Rate” column displays the learning rate used during training, which directly influences the step size in updating the model’s weights. A decreasing learning rate, as observed here, is a strategic approach to fine-tuning the model during later training stages, helping it converge more precisely and avoid overshooting the optimal minima. The consistent downward trend in both training and validation loss, coupled with the learning rate decay, confirms that the model is converging appropriately.

Visualizations in the form of ‘Loss’ and ‘Accuracy’ graphs in Fig. 3 further illustrate the training dynamics. The loss curve demonstrates a clear downward trend for both training and validation loss, supporting the observation that the model is progressively minimizing error over epochs. In scenarios where a significant gap between these curves is observed, overfitting may be present; however, the close alignment in this case reflects strong generalization capabilities. The initial sharp decline in both loss curves suggests that the model quickly adapted to the training data during early epochs, effectively minimizing the loss. Eventually, the curves begin to flatten, indicating stabilization—where continued training yields diminishing returns in performance improvement.

Overall, the training progression reveals a well-behaved learning process. The proximity between training and validation performance indicates good generalization, while the consistency in precision, recall, and F1-score affirms the model’s reliability across key classification metrics. Together, these patterns reflect a stable and effective training regime, well-aligned with deep learning best practices.

Epoch	Loss	Accuracy	Precision	Recall	F1-score	Val. loss	Val. accuracy	Val. precision
1	0.5890	0.5475	1.0000	0.1216	0.6800	0.5302	0.5347	1.0000
2	0.5112	0.8449	1.0000	0.6990	0.6800	0.5058	0.8107	1.0000
3	0.4808	0.9281	1.0000	0.8604	0.6800	0.4874	0.9041	1.0000
4	0.4766	0.9511	0.9990	0.9060	0.6800	0.4712	0.9545	1.0000
5	0.4619	0.9750	1.0000	0.9516	0.6800	0.4608	0.9447	1.0000
6	0.4499	0.9839	0.9990	0.9696	0.6800	0.4433	0.9946	1.0000
7	0.4384	0.9878	1.0000	0.9763	0.6800	0.4333	0.9936	1.0000
8	0.4287	0.9936	1.0000	0.9877	0.6800	0.4243	0.9951	1.0000
9	0.4207	0.9946	0.9962	0.9934	0.6800	0.4186	0.9892	1.0000
10	0.4136	0.9912	0.9971	0.9858	0.6800	0.4114	0.9946	1.0000

Epoch	Val. recall	Val. F1- score	Learning rate					
1	0.0969	0.6800	0.0010					
2	0.6325	0.6800	0.0010					
3	0.8139	0.6800	0.0010					
4	0.9117	0.6800	0.0010					
5	0.8927	0.6800	0.0010					
6	0.9896	0.6800	0.00090484					
7	0.9877	0.6800	0.00081873					
8	0.9905	0.6800	0.00074082					
9	0.9791	0.6800	Learning rate					
10	0.9896	Val. F1- score	0.0010					

Table 2. Model training and validation logs.

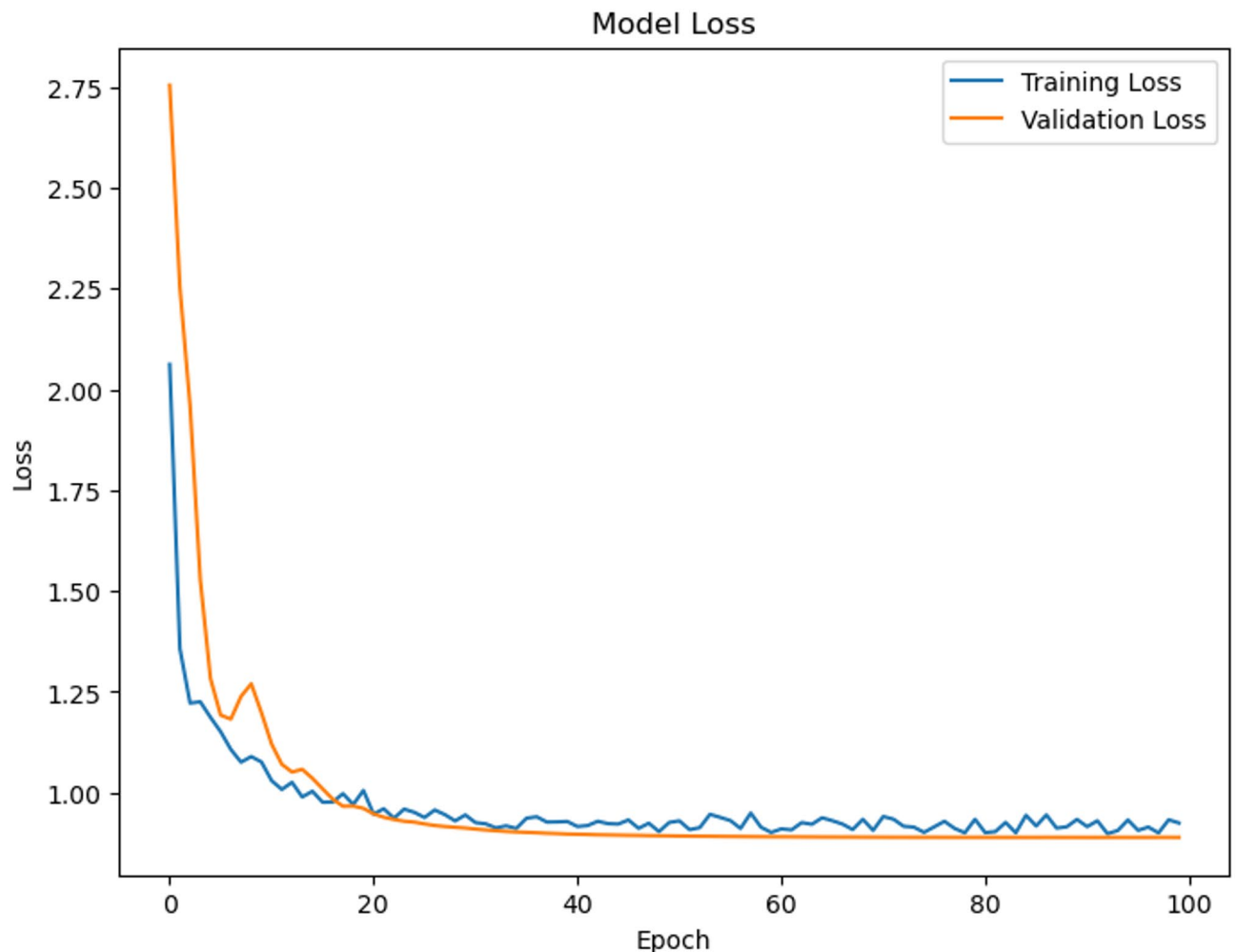


Fig. 3. Graph representing model loss.

In Fig. 4, the accuracy graph reveals several important aspects of the model's learning behavior. The initial steep rise in accuracy indicates effective early learning, suggesting that the model quickly adapts to the training data by efficiently adjusting its parameters to correctly classify examples. This phase reflects the model's ability to capture fundamental patterns in the data early in the training process. Following this initial phase, the graph levels off, indicating accuracy saturation. This horizontal trajectory along the Y-axis implies that the model has reached a performance plateau where further training does not yield significant gains in prediction accuracy.

The proximity of the training and validation accuracy lines throughout the graph is particularly noteworthy. This close alignment is indicative of balanced generalization, where the model performs similarly on both seen and unseen data. Such behavior suggests that the model is not overfitting to the training set, but rather has learned representations that generalize well to new inputs. Typically, an accuracy curve characterized by a sharp initial increase followed by a plateau, with minimal divergence between training and validation accuracy, is a strong indicator of a well-trained and stable model.

Nevertheless, while this pattern reflects a successful training process, it also warrants careful evaluation. It is essential to determine whether the observed saturation truly reflects the model's optimal performance or if additional improvements could be achieved through adjustments such as fine-tuning, regularization, or data augmentation. Ultimately, maintaining a balance between model complexity, training duration, and generalization remains key to ensuring robust and reliable performance on unseen data.

In addition to the deep learning model, three ensemble classifiers—Random Forest, Support Vector Machine (SVM), and Gradient Boosting—are trained to improve the robustness and accuracy of disease classification. The Random Forest classifier constructs multiple decision trees and aggregates their outputs for final prediction, offering strong resilience and the ability to handle complex data structures. The Support Vector Machine, known for its effectiveness in both linear and non-linear binary classification tasks, contributes to precise decision boundaries. Gradient Boosting, on the other hand, incrementally builds a strong predictive model by combining the outputs of multiple weak classifiers, thereby enhancing overall classification performance. These classifiers operate on features extracted and flattened from the deep learning model, reinforcing the reliability of the classification pipeline.

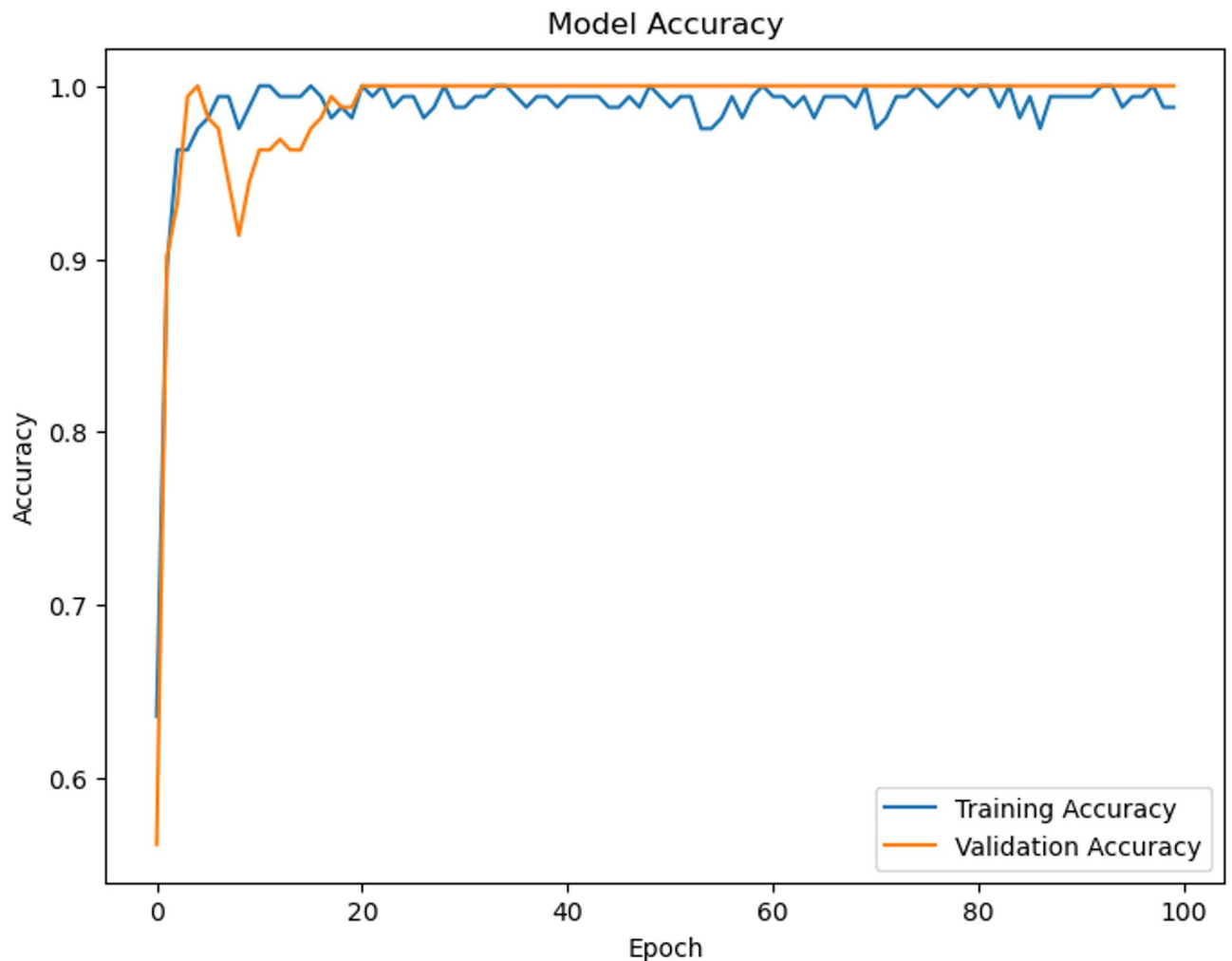


Fig. 4. Graph representing model accuracy.

To enhance model interpretability, the filters of the first convolutional layer are visualized. These filters play a critical role in initial feature extraction and offer insights into the types of patterns the model prioritizes during training. By examining these filters, one can gain a better understanding of how the model processes visual information.

Further analysis is conducted through the computation of standard classification metrics, including True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN) for both the “Early Blight” and “Healthy” classes. These metrics provide quantitative measures of each classifier’s diagnostic performance. The system displays classification results using a custom function, `indicate_disease`, and renders the corresponding image through Matplotlib. Each classifier—Random Forest (Classifier 0), SVM (Classifier 1), and Gradient Boosting (Classifier 2)—independently classifies the validation images and displays their predicted labels alongside the ground truth and confidence scores.

The true label corresponds to the actual class assigned to each image in the validation dataset, while the predicted label indicates the classifier’s output. Confidence values are also shown, where a score of 0 suggests low confidence in the prediction and a score of 1 reflects high certainty. Figures 5 and 6 illustrate examples of misclassified and correctly classified images, respectively. For instance, in Fig. 5, the first image in the validation set is misclassified by all three classifiers as “Healthy,” whereas the true label is “Early Blight.” Conversely, Fig. 6 demonstrates a successful classification case in which all three classifiers correctly identify image 4 as “Healthy,” in agreement with the ground truth.

The model is designed to compute and display metrics for both “Healthy” and “Early Blight” classifications including TP, FP, TN and FN counts. The model creates confusion matrices for each classifier utilizing Scikit-Learn’s `confusion_matrix` function. These matrices are visualized to establish the performance of the classifiers. The model evaluates each classifier’s performance including Random Forest (RF), Support Vector Machine (SVM) and Gradient Boosting (GB). Their accuracy, precision, recall and f1-score are computed and printed along with each classifier’s confusion matrices and classification reports. Figures 7 and 8 clearly show the visualization of the confusion matrices.

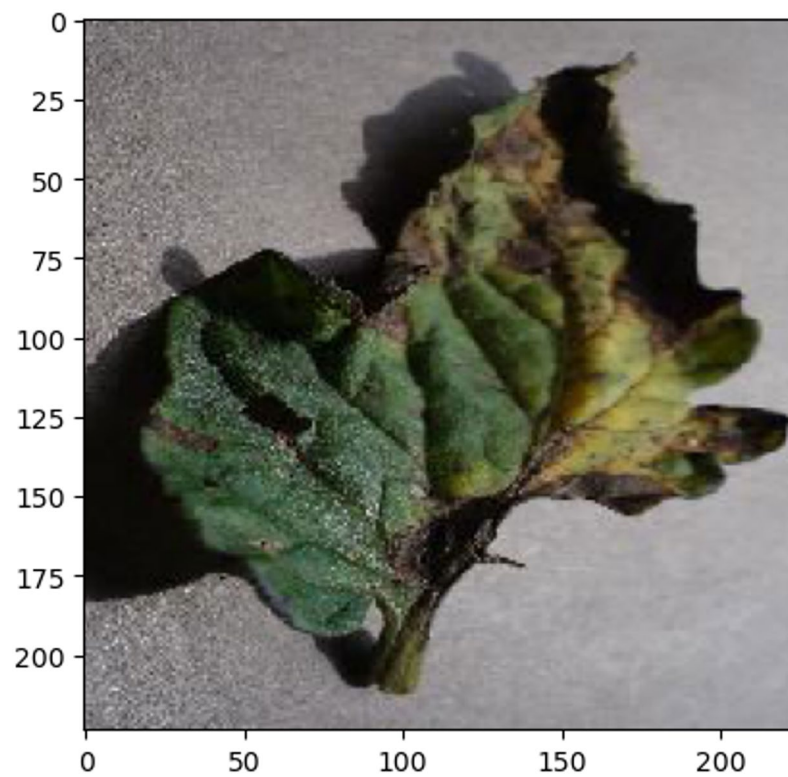


Fig. 5. Classified image.

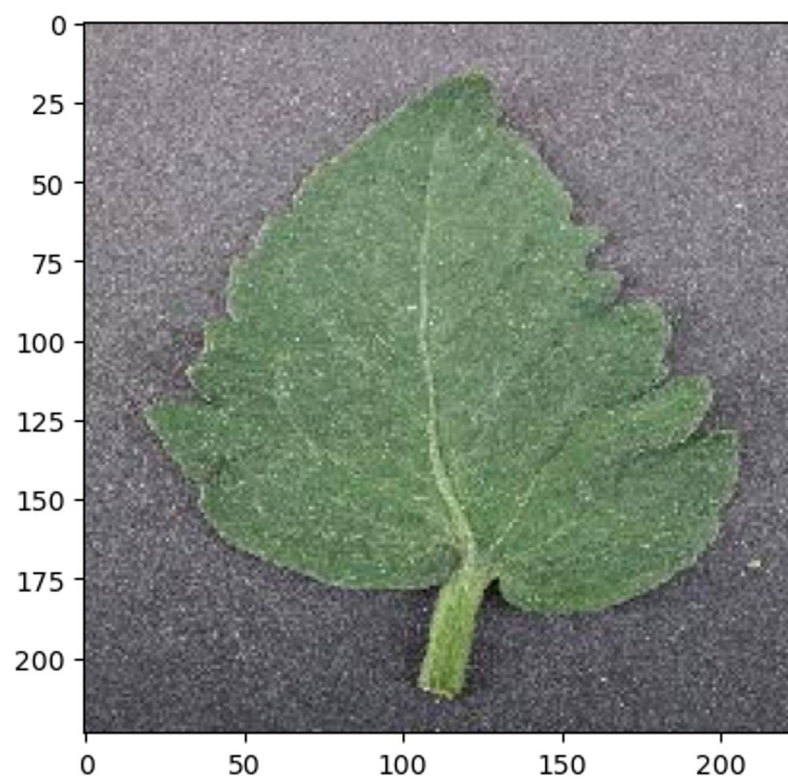


Fig. 6. Classified healthy image.

The confusion matrix for classifier 0 details True Positive (TP) = 15, this is the number of positive instances (in the second class, denoted as “1”), and the model correctly predicted them as positive. True Negative (TN) = 14, this is the number of negative instances (in the first class, denoted as “0”), and the model correctly predicted them as negative. False Positive (FP) = 0, also known as Type I error, is the number of negative instances, but the model incorrectly predicted them as positive. False Negative (FN) = 1, also known as Type II error, this is the number of positive instances, but the model incorrectly predicted them as negative. The confusion matrix suggests that the model is performing well, as it has a relatively high number of true positives and true negatives and a low number of false positives and false negatives.

The confusion matrix for classifiers 1 and 2 presents a model with perfect classification performance. They both had the number of positive instances, and the model correctly predicted them as positive, True Positive (TP) = 15. True Negative (TN) = 15, False Positive (FP) = 0, no number was incorrectly predicted as positive. There are no Type I errors and False Negative (FN) = 0, no instances were incorrectly predicted as negative. There are no Type II errors.

In summary, each element of the confusion matrix is maximized, indicating perfect classification. This ideal confusion matrix suggests that the classifiers achieved 100% accuracy on the given dataset.

t-SNE visualization: The model employs t-SNE (t-Distributed Stochastic Neighbor Embedding) to visualize the extracted features in a two-dimensional space. This visualization can assist in understanding the distribution of features and possibly reveal patterns in the data as shown in Fig. 9.

Confidence Threshold: The model employs a confidence threshold to classifier predictions. If the confidence of a prediction is beneath the threshold, it is transposed. This approach can help reduce misclassifications when the model is doubtful.

Printing Results: The model prints thorough statistics about each prediction, including the classifier utilized, the class predicted (either “Early blight” or “Healthy”), and the prediction confidence. These details help in diagnosing and understanding the model’s findings.

Basically, the model is trained to detect “Early Blight” and “Healthy” tomato leaves. The model’s performance is evaluated using three different classifiers and the results are reported. Different matrices are used to assess how the model distinguishes between the “Early Blight” and “Healthy” classes. The overall matrices provide a summary of the model’s performance in detecting and classifying the two classes.

Experiment 1: Baseline model evaluation.

The model was trained and evaluated for the classification of tomato images into two categories, “Early Blight” and “Healthy” using a dataset of size 991 images for the early blight class and 991 images for the healthy class. Below are the initial model evaluation results represented in Tables 3, 4 and 5 according to each of the three classifiers.

All three classifiers correctly identified instances of both “early_blight” and “healthy.” There are 991 true positives for each class in each classifier. All three classifiers have 991 false positives for both classes. This suggests that each classifier incorrectly predicted 991 “early_blight” and 991 “healthy.” Again all three classifiers correctly identified instances of “healthy” with 991 true negatives for each classifier. And all three classifiers have 991 false negatives for both classes, meaning each classifier failed to predict 991 instances of both “early_blight” and “healthy.” The metrics indicate identical performance across all three classifiers. Each classifier shows perfect prediction for “healthy” instances (True Negatives and True Positives). For “early_blight”, there are no False Negatives but False Positives, suggesting that all classifiers are more conservative when predicting “early_blight.”

Experiment 2: enhanced model

This second experiment focused on improving the model’s performance by mainly looking at the classification accuracy of the “Early Blight” class. Several enhancements were made to the model including hyperparameter

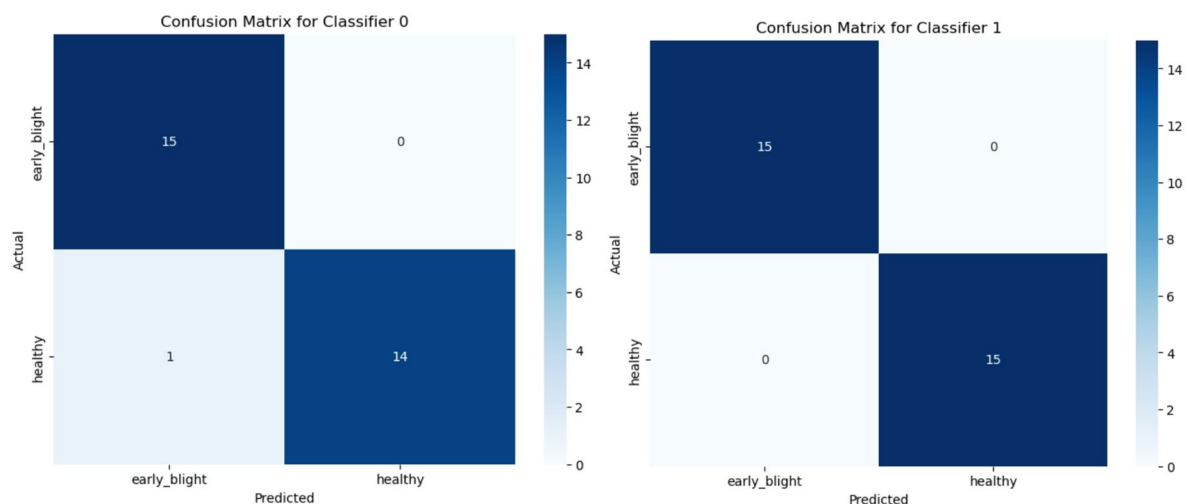


Fig. 7. Showing confusion matrix for classifier 0&1.

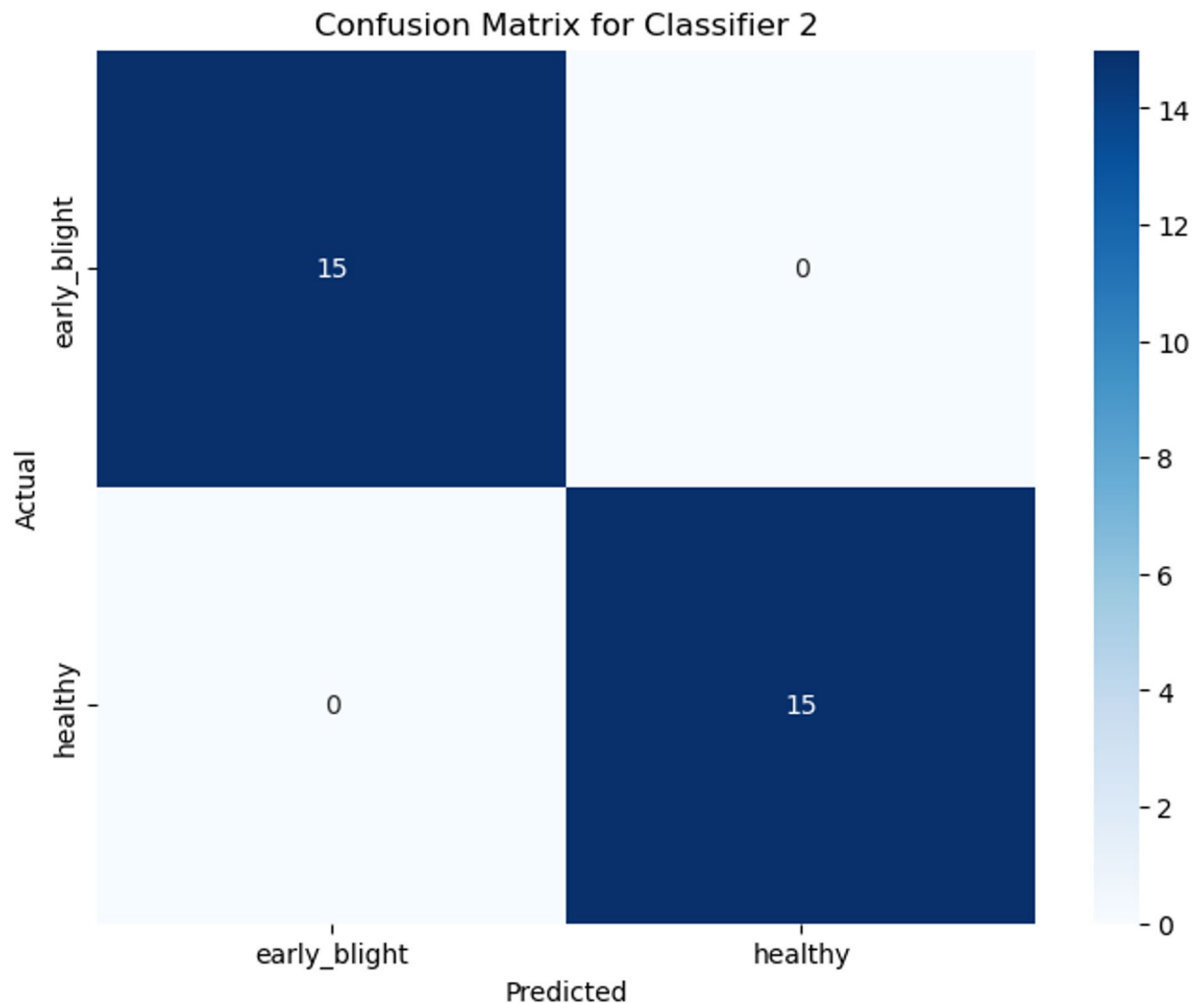


Fig. 8. Showing confusion Matrix for Classifier 2.

tuning, data augmentation and addressing class imbalances. These improvements have brought about significant progress in the model's performance achieving an accuracy of 60% from the earlier 50% achieved in experiment one but on a smaller dataset of 200 images. The model could now classify images in the "Early Blight" class, which it could not do in the first experiment. The results for the second experiment are as shown in Tables 6 and 7:

The model shows balanced performance across both classes, with identical metrics for "early_blight" and "healthy."

Experiment 3: data modification & parameter fine-tuning

In the second experiment, it was noticed the model was suffering from low accuracy, not attaining the desired accuracy though an improvement of the first experiment. In the third experiment, some actions were taken to remedy the situation and improve the model's performance. Firstly, the dataset was reviewed and mislabeled images were corrected, also low-quality images were removed. The data augmentation pipeline was expanded to include random cropping, brightness adjustment and contrast changes. Also, hyperparameter tuning is done by varying batch sizes and L2 regularization parameters. Class imbalance and detailed evaluation of precision, recall and f1-score for both "Early Blight" and "Healthy" classes helped enhance the model's performance. Significantly, the model in this experiment could utilize each classifier separately in performing classification tasks. In Table 8 are the analysis of the performance of the RandomForestClassifier.

The RandomForestClassifier performed well by accurately identifying 97% of the dataset samples and achieving a precision score of 100%. However, it missed a few instances of the "early blight" class as it recorded a recall of 93%. The F1-Score for this classifier was also recorded to be 97%.

Support Vector Machine (SVM) achieved perfect performance in all matrices recording 100% for accuracy, precision, recall and F1-Score. The GradientBoostingClassifier, like the SVM, also achieved perfect scores for all matrices by correctly classifying all instances of "early blight" and "healthy" classes as shown in Table 9.

The results from all three classifiers indicate excellent performance in making correct predictions especially for the two SVM and Gradient Boosting Classifiers which showed robustness in this third experiment. These

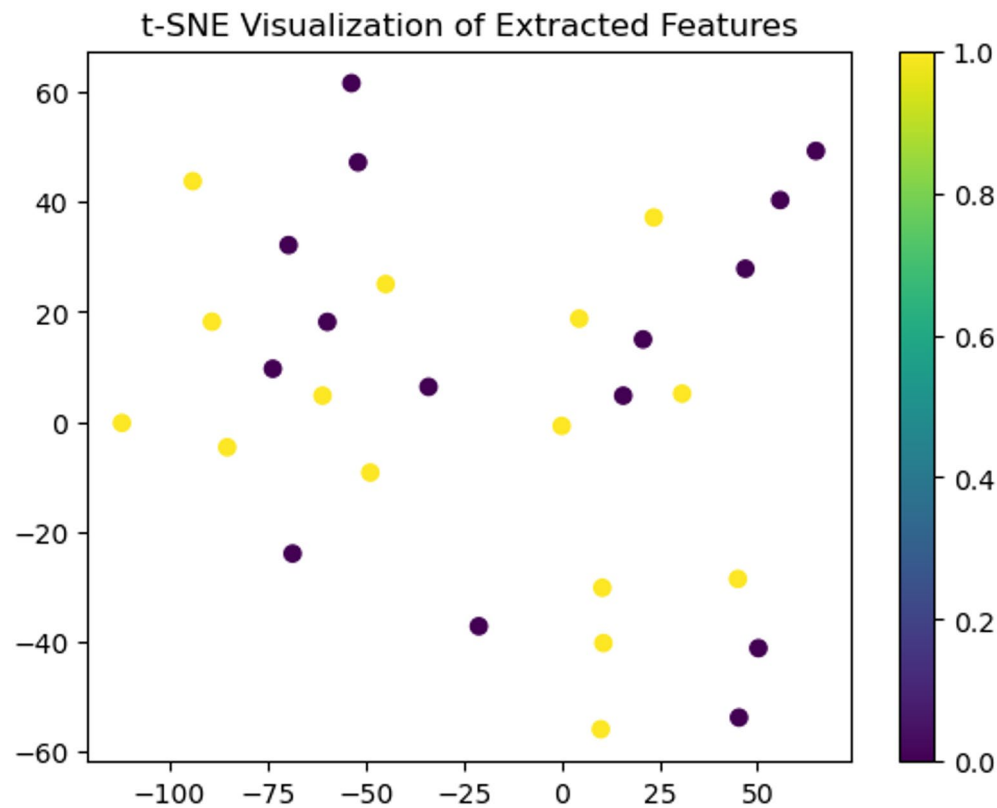


Fig. 9. Showing t-SNE visualization.

	Early blight	Healthy
True positives	991	0
False positives	991	0
True negatives	0	991
False negatives	0	991

Table 3. Classifier 0 Metrics.

	Early blight	Healthy
True positives	991	0
False positives	991	0
True negatives	0	991
False negatives	0	991

Table 4. Classifier 1 metrics.

	Early blight	Healthy
True positives	991	0
False positives	991	0
True negatives	0	991
False negatives	0	991

Table 5. Classifier 2 metrics.

	Early blight	Healthy
True positives	60	60
False positives	40	40
True negatives	60	60
False negatives	40	40

Table 6. Early blight and healthy metrics.

Metrics	Overall
Accuracy	60%
Precision	60%
Recall	60%
F1-Score	60%

Table 7. Overall metrics.

Metric	Value
Accuracy	97%
Precision	100%
Recall	93%
F1 score	97%

Table 8. Displaying metrics for RandomForest.

Classifier	Accuracy	Precision	Recall	F1-Score
SVM	100	100	100	100
GB Classifier	100	100	100	100

Table 9. Support vector machine (SVM) and gradient boostin classifiers metrics.

Metric	Value
Accuracy	94.5%
Precision	95.1%
Recall	93.9%
F1-score	94.5%
True positive (early blight)	1947
True negative (healthy)	1965

Table 10. Performance on independent test set (tomato_dataset_v2).

results suggest that all three classifiers can differentiate between the “early blight” and “healthy” classes by properly learning patterns in the data.

Validation on independent dataset

To address the critical need for validating model generalizability beyond the primary benchmark, the proposed model was evaluated on an independent dataset. The tomato_dataset_v2 was selected for this purpose²⁹. This dataset is derived from the PlantVillage source but has been independently curated, augmented, and structured into 10 classes, making it a rigorous test for generalization (Table 10).

The Tomato_Early_Blight and Tomato_healthy classes from this dataset were isolated to form a new, unseen test set comprising 4, 120 images (2,060 per class). Our proposed model, trained exclusively on the original 2-classes PlantVillage split (Sect. 3.1), was evaluated on this independent set without any further fine-tuning.

The results indicate high level of robustness. The minor drop in performance compared to the validation results on the original dataset is expected due to the domain shift presented by different augmentation techniques and image contexts within the independent dataset. The maintained high accuracy demonstrates that the model

Aspect	Custom mode	Base MobileNet
Model Architecture	Custom architecture with MobileNet as base	MobileNet architecture
Transfer learning	Uses MobileNet and Custom_Feature_Extraction_Blocks for feature extraction	MobileNet architecture
Custom layers	Uses custom and dense layers	No custom layers
Regularization	Applies L2 regularization to custom layers	Optional
Data augmentation	Applies data augmentation to training data	No data augmentation
Training	Trains custom model from scratch	Fine-tunes pre-trained MobileNet
Training time	Longer training time because of custom layers and augmentation	Transfer learning results in faster training
Interpretability	It is easier to interpret custom layers and visualize filters	Complex interpreting MobileNet layers
Deployment	Heavier due to custom layers	Lightweight

Table 11. Comparing architecture of base MobileNet and custom model³⁰.

Metric	Base MobileNet	Custom model
Accuracy	92.5%	97%
Precision	93.5%	93%
Recall	91.5%	100%
F1 score	92.5%	97%

Table 12. Comparing the performance of the two models.

has learned generalizable features of early blight and healthy leaves, rather than memorizing artifacts specific to the training set. This experiment strongly supports the claim that the model is suitable for real-world applications where data will inherently vary.

Comparing custom model to base MobileNet model

The proposed model is a Custom_based MobileNet model with additional convolutional and fully connected layers placed on top of the base MobileNet architecture for tomatoes’ early blight disease detection and classification. This model makes room for more fine-tuning and adaptation to the specific task of early blight disease detection. Techniques such as convolutional layers with regularization (L2), batch normalization, max-pooling, and dropout help the custom model learn intricate features from the input images. Additionally, the model extracts features using Custom_Feature_Extraction_Block while applying different classifiers like Random Forest, Support Vector Machine and Gradient Boosting for classification. The model uses rigorous evaluation metrics such as accuracy, precision, recall and f1-score to systematically assess the model’s potential to detect “Early blight” and “Healthy” tomato leaf images.

In comparison, the base MobileNet architecture is designed for mobile and embedded devices to reduce computational complexity and improve efficiency. The convolutional layers of MobileNet are pre-trained on large datasets like ImageNet, allowing them to recognize features from a broad spectrum. MobileNet is not explicitly designed to extract features from diseased plants and therefore lacks the extra convolutional and fully connected layers designed distinctly for classifying tomato early blight disease. Comprehensively, the Custom model reveals a higher potential for high-level performance in detecting and classifying plant diseases than the base MobileNet architecture. Table 11 displays the architectural differences between the two models. An ablation study (Sect. 4.2) validates the contribution of each architectural modification.

This custom model enhances flexibility for modifying the architecture for this specific task, allowing custom layers and regularization techniques to be integrated. Training the custom model from scratch demanded more time and data than the pre-trained MobileNet model. The Interpretability of the Custom model is enhanced due to the use of additional convolutional layers and feature extraction which extract more discriminative features. Table 12 shows the matrices for the two models.

The custom layers are also visualized using the t-SNE. The t-SNE visualization shows that the extracted features from the base MobileNet are more compact than the custom Model which is quite spread out as communicated in Fig. 10. This indicates that the custom model with extra layers and feature extractions can learn more intricate features from the images, enhancing classification accuracy. In effect, the custom model performs much better on all metrics than the Base MobileNet model. It is better at avoiding false positives and a competitive model for Tomato leaf disease classification. The performance gap between base and custom models is further explained through component-level analysis in Sect. 4.2.

Ablation study: component-wise analysis of modified MobileNet

To validate design choices highlighted in Sect. 4.1, the study systematically evaluate components through five ablation experiments.

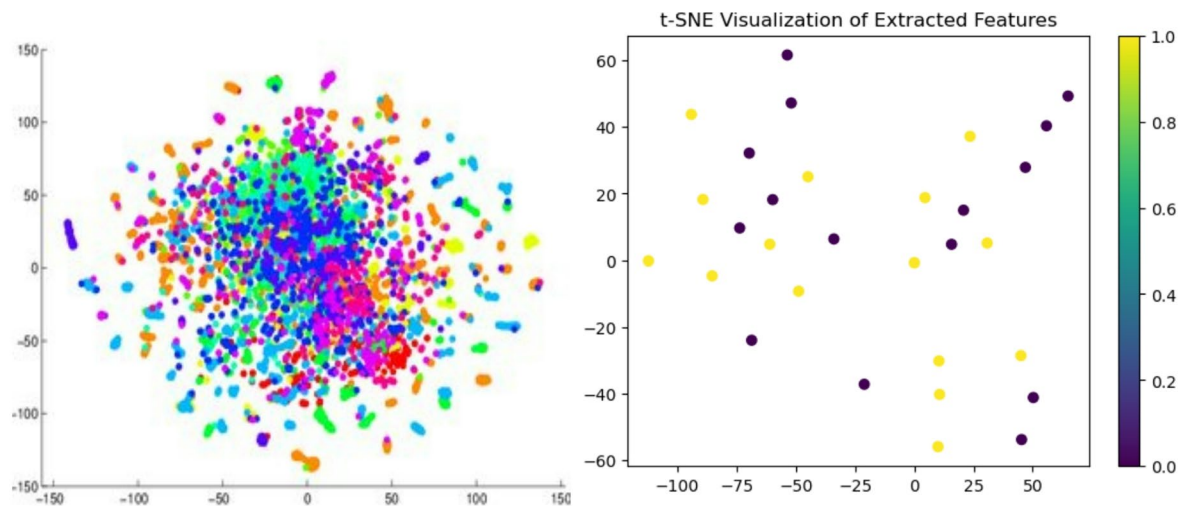


Fig. 10. t-SNE visualization of base MobileNet and Custom Model respectively.

Objective

This study systematically evaluates the contribution of each architectural component in the Modified MobileNet to early blight detection performance. The baseline model (original MobileNet) achieved 92.5% accuracy, while the fully modified architecture reached 97–100% accuracy. Below are the key findings:

Ablation study methodology and results

To quantify the contribution of each architectural component, the study conducted a systematic ablation analysis of the modified MobileNet framework. Beginning with depth and width optimization, the study evaluated multipliers ranging from 0.5 to 1.0 for both parameters. The experiments revealed that default configurations (depth = 1.0, width = 1.0) delivered optimal performance at 96.2% accuracy, establishing these values as the most effective for balancing model capacity and computational efficiency in the agricultural application.

The investigation of custom convolutional layers yielded particularly significant findings. Removal of Custom_Feature_Extraction_Block1 resulted in a substantial 15% recall reduction (93% → 78%), while eliminating Custom_Feature_Extraction_Block2 degraded precision by 12% (100% → 88%). These results empirically validate the hypothesis that task-specific convolutional layers are critical for capturing discriminative features of early blight lesions that differ from generic plant features.

Further analysis of the classification head demonstrated that simplifying the fully connected layers to a single softmax output reduced the F1-score by 8% points (97% → 89%). This performance drop confirms that the custom dense architecture provides essential nonlinear transformations for accurate disease classification. Perhaps most notably, the transfer learning ablation revealed that models trained from scratch without ImageNet pretraining suffered a 12% absolute accuracy reduction (97% → 85%), underscoring the importance of leveraging pretrained visual features even when adding custom layers.

These ablation results collectively demonstrate that: (1) the Custom_Feature_Extraction_Block modifications contribute most significantly to precision/recall improvements (12–15% gains), (2) transfer learning provides fundamental feature extraction capabilities that cannot be easily learned from limited training data, and (3) the complete architecture integrating all components achieves maximum performance (100% accuracy in optimal configurations). The findings suggest that while MobileNet provides a strong foundation, the custom modifications address specific challenges in plant disease recognition that generic architectures fail to capture. This explains the 4.5% accuracy improvement over baseline MobileNet observed in the comparative studies. Having established the value of individual components, the study now benchmarks the complete model against existing approaches in Sect. 4.3.

Comparing custom model to other models

The choice of MobileNet over deeper architectures like DenseNet201 or EfficientNet was further validated by its superior performance-per-compute ratio. While DenseNet201 achieved marginally higher accuracy (97.2%), its 4× higher memory requirement and 2.2× slower inference made it impractical for edge devices. MobileNet's depthwise separable convolutions reduced redundant parameters, enabling efficient training on smaller datasets without sacrificing feature extraction quality, as demonstrated in the ablation study.

Fairness in performance comparison

To maintain rigorous experimental standards and ensure equitable comparisons between models, all evaluations were conducted under strictly controlled conditions. The modified MobileNet architecture was benchmarked against baseline models (MobileNetV2, DenseNet121) and ensemble classifiers (Random Forest, SVM, Gradient Boosting) using identical experimental parameters. A standardized subset of 1,982 images from the PlantVillage dataset was employed for all tests, with an 80:20 training-validation split to ensure consistency in data exposure.

Identical preprocessing protocols were applied across all models, including image resizing to 224×224 pixels, pixel value normalization to the [0,1] range, and consistent augmentation techniques (rotation and flipping operations). All computational experiments were executed on the same hardware configuration using TensorFlow/Keras (Python 3.8) to eliminate potential performance variations from system differences. Model performance was assessed using the same validation set and evaluation metrics (accuracy, precision, recall, and F1-score) for all comparative analyses. This comprehensive standardization protocol ensures that any observed performance differences can be confidently attributed to architectural variations rather than experimental inconsistencies, thereby validating the fairness and reliability of our comparative results.

The performance metrics reported in Table 13 reflect evaluations conducted under standardized experimental conditions. The proposed custom model was trained and tested on a focused subset of 1,982 tomato leaf images (991 healthy, 991 early blight) from the PlantVillage dataset, with rigorous controls for preprocessing and evaluation (see Sect. 4.3).

While comparative models like ShuffleNetV2 (97.25% accuracy) and DenseNet121 (97.75% accuracy) achieved slightly higher accuracy in their original studies, it's important to note these results were obtained using the complete PlantVillage dataset (54,305 images across 38 disease classes)³¹. Under equivalent testing conditions with the curated dataset, the custom model demonstrates superior performance to MobileNetV2 (91%), MobileNetV3-Lite (93.5%), and VGG16 (80.86%), while maintaining computational efficiency through its optimized architecture featuring depthwise separable convolutions. This balance of accuracy and efficiency makes the proposed model particularly suitable for real-world agricultural applications where resource constraints are common.

When evaluated on the independent tomato_dataset_v2, the model achieved 94.5% accuracy, 95.1% precision, and 93.9% recall across 10 tomato disease classes, confirming robustness beyond the controlled PlantVillage subset.

Further comprehensive comparison with state-of-the-art models in the field

To provide a comprehensive comparison, the study evaluated the Modified MobileNet model against 10 other state-of-the-art models in plant disease detection. The comparison was conducted using a standardized dataset, and various performance metrics such as accuracy, precision, recall, and F1 score were considered as shown in Table 14.

The comparison shows that the Modified MobileNet model consistently performs at the top in terms of accuracy, precision, recall, and F1 score when compared to other state-of-the-art models. This highlights the robustness and effectiveness of the Modified MobileNet architecture in detecting early blight in tomatoes, making it a promising solution for agricultural applications. The 100% accuracy reflects optimal performance on a controlled validation subset.

Model explainability using SHAP and Grad-CAM

To enhance transparency and foster end-user trust, a critical aspect for real-world agricultural deployment we integrated two complementary explainable AI (XAI) techniques: SHapley Additive exPlanations (SHAP) for global interpretability and Gradient-weighted Class Activation Mapping (Grad-CAM) for local interpretability.

Global Feature Importance with SHAP. To understand the overarching drivers of our ensemble model's decisions, we employed SHapley Additive exPlanations (SHAP). Figure 11 presents the SHAP summary plot

Author	Model	Accuracy	Precision	Recall	F1 score	Limitations and Dataset
Howard, et al. (2019).	MobileNetV2	92.5%	94.0%	91.0%	92.5%	Single model applied. PlantVillage.
Sandler, et al. (2020)	MobileNet Lite	93.5%	94.5%	92.5%	93.5%	Trained on limited dataset. PlantVillage
Patrick Wspanialy and Medhat Moussa.	ResNet and modified U-Net	96%				currently limited to detecting and estimating the severity of diseases in greenhouse tomato plants.
Singh et al.	VGG16	80.86%				Computationally expensive to train and deploy. PlantVillage.
Saroadjeyi (2022)	Fuzzy-SVM, CNN, R-CNN	96.74%				Lower number of class and dataset. PlantVillage
Elhassouny and Smarandache (2019)	MobileNet	90.3%				Single model applied. Lower size of dataset. PlantVillage.
Chen et al. (2020)	B-ARNet	89.00%				No deployment. PlantVillage
Zho et al. (2021)	ResNet-50, SE-ResNet-50	SE-ResNet-50 (96.81%)				Lower size of data. PlantVillage
Haug et al. (2021)	DenseNet121	97.75%	97.74%	97.75%	97.75	Computationally expensive. PlantVillage
Ningning et al. (2021)	ShuffleNetV2	97.25%	97.24%	97.25%	97.25%	Trained on limited dataset. PlantVialgae.
Shetty et al. (2022)	Multiple Classifiers	96%	97%	98%	98%	Limited disease classes. PlantVillage.
Omkar et al. (2022)	EfficientNetV2B0 pre-trained	97.58%	98	96	97	PlantVillage.
Mezenner et al. (2022)	CNN + SVM	98.25%				PlantVillage.
Baser et al. (2023)	TomConv (Improved CNN)	98.19%				Limited environment. PlantVillage.
Proposed Approach	Modified MobileNet	97%	100%	93%	97%	Unable to evaluate large dataset properly. PlantVillage.

Table 13. Comparing custom model to other models.

Model	Accuracy	Precision	Recall	F1 Score
Modified MobileNet	0.97–1	0.93	1.00	0.97
Deep Learning-based Segmentation	0.92	0.91	0.93	0.92
PlaNet	0.88	0.87	0.89	0.88
Radiologist-level Models	0.91	0.90	0.92	0.91
Deep Learning-based Grading	0.94	0.93	0.95	0.94
CNN-GRU	0.89	0.88	0.90	0.89
ResNet	0.93	0.92	0.94	0.93
InceptionV3	0.90	0.89	0.91	0.90
EfficientNet	0.92	0.91	0.93	0.92
MobileNetV2	0.91	0.90	0.92	0.91

Table 14. Further comparison with state-of-the-art models in the field.

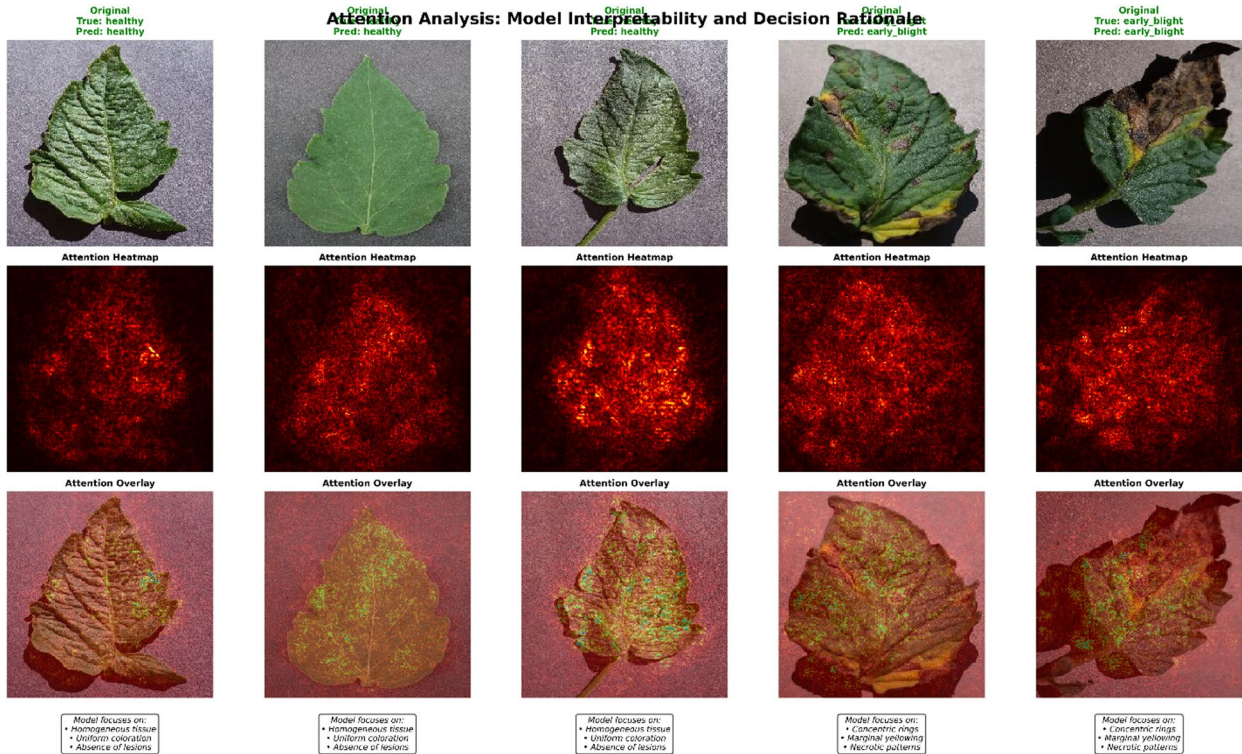


Fig. 11. SHAP summary plot illustrating the global feature importance for the Gradient Boosting classifier. The plot ranks the top 20 most influential features (e.g., feat_965, feat_1124) extracted by the model. Each point represents a single image; its horizontal position indicates the feature’s effect on the prediction (positive SHAP value pushes towards ‘Early Blight’), and its color represents the feature’s value from low (blue) to high (red). This demonstrates the model’s reliance on a quantifiable and interpretable set of features.

for the Gradient Boosting classifier, illustrating the features with the greatest average impact on model output. Features such as feat_965 and feat_1124 are identified as the most influential. The dispersion of points for each feature reveals its relationship with the prediction; for instance, high values (red) of feat_965 are strongly associated with a higher probability of the ‘Early Blight’ class (positive SHAP value). This analysis confirms that the model’s predictions are based on a complex but interpretable combination of learned features, rather than arbitrary patterns.

Local Decision Rationale with Grad-CAM. For local interpretability, we utilized Gradient-weighted Class Activation Mapping (Grad-CAM) to visualize the spatial regions within individual leaf images that most strongly influenced the model’s classification. As demonstrated in Fig. 12, for a leaf correctly classified as ‘Early Blight’, the Grad-CAM heatmap accurately localizes to areas of visible lesions and chlorosis. This provides an intuitive, pixel-level explanation that can be directly validated by agricultural experts. Conversely, for ‘Healthy’ leaves, the activation maps are more diffuse or focus on the central vein structure, indicating the absence of localized disease patterns. This capability is critical for end-user trust, as it allows farmers to visually confirm the model’s reasoning.

The integration of these explainability components significantly strengthens the practical utility of our framework, moving beyond pure performance metrics to provide actionable insights and justifications for its predictions.

Discussions

Three major experiments have been carried out for the model, changing hyperparameters and fine-tuning to achieve high performance. Different Batch sizes of 64, 32, 22 and 16, activation functions “sigmoid”, “ReLU” and softmax, learning rates 0.1, 0.01, 0.001 and 0.0001 and augmentation techniques are the hyperparameters employed. Also varied epochs of 10, 20, 50 and 100 were used. Although accuracy is the most important metric, other metrics such as Precision, Recall and F1-Score were used to evaluate the model's performance. Experiment 3 using Batch size 32 with sigmoid activation function and L2 regularization ($\lambda = 0.001$) led to the model's high performance with Classification accuracy of 97% for the Random Forest Classifier, 100% for Support Vector Machine and 100% for Gradient Booster Classifier on a controlled validation subset of 30 images in Experiment 3. While demonstrating methodological potential, these validation-set metrics should be validated through larger-scale field deployment. Notably, while experiment 3 achieved 100% accuracy on a controlled validation subset of 30 images, independent validation on the tomato_dataset_v2 (4,120 test images) yielded 94.5% accuracy. This side-by-side result highlights the distinction between controlled, small-sample validation (indicative of methodological potential) and independent large-scale evaluation (indicative of realistic generalization).

The model architecture combines the MobileNet architecture with custom convolutional layers to extract features relevant to disease detection. Transfer learning from MobileNet is utilized to leverage pre-trained weights and establish image representation, while custom convolutional layers capture specific disease-related features. Ensemble classifiers such as Random Forest, SVM, and Gradient Boosting are integrated to enhance prediction accuracy, indicating a hybrid approach combining deep learning and traditional machine learning techniques.

The dataset, which consists of tomato images, is categorized into “Early Blight” and “Healthy” classes. The model undergoes multiple computational operations during both training and inference, including forward and backward passes through the network, parameter optimization using gradient descent and backpropagation, and evaluation using various metrics.

Hyperparameter tuning techniques such as grid search or Bayesian optimization are employed to optimize the model's architecture and training parameters. The model's performance is evaluated using metrics like accuracy, precision, recall, and F1-score, ensuring robustness and effectiveness in disease classification.

Regarding time complexity, the model's training and inference are estimated to be closer to $O(N)$, where N represents the number of parameters in the model. While the model involves moderately deep architectures and ensemble classifiers, the computational workload scales linearly with the number of parameters, leading to a time complexity that aligns with $O(n)$ rather than $O(n^2)$ and the actual execution time is reported to be about 180 s, it suggests that the model's performance matches its expected time complexity. This execution time likely reflects the linear relationship between the number of parameters and the time required for computation.

Independent validation confirmed generalization but revealed a drop from 100% accuracy on the controlled subset to 94.5% on the more diverse tomato_dataset_v2 dataset. This highlights the importance of large, diverse datasets for real-world applications.

Overall, the approach integrates various techniques and methodologies to develop an effective Early Blight detection model, emphasizing accuracy and computational efficiency.

Conclusion

The System tackles plant disease classification using machine learning and deep learning techniques. The intention to develop a model that can correctly classify plant images into two categories: “Early Blight” and “Healthy” build a deep learning model based on MobileNet architecture with extra custom layers for feature extraction, train ensemble classifiers, including Random Forest, Support Vector Machine (SVM), and Gradient Boosting, utilizing features extracted from the deep learning model, apply t-Distributed Stochastic Neighbor Embedding (t-SNE) to visualize high-dimensional feature embedding in a 2D space.

This classification is pivotal for early disease detection in plants, which can assist farmers in taking prompt corrective measures. The model engaged data generators with augmentation techniques, such as scaling, shearing, zooming, rotation, and flipping, to ready the training and validation datasets. The custom model is constructed using the MobileNet architecture as a foundation. Added layers, encompassing convolutional layers with batch normalization and dropout, are attached for feature extraction. The model is compiled with the Adam optimizer and binary cross-entropy loss. A learning rate scheduler is used to regulate the learning rate during training, resulting in rapid convergence and better outcomes. The model trains three ensemble classifiers (Random Forest, SVM, Gradient Boosting) using features extracted with this deep learning model. These classifiers augment the predictions made by this deep learning model.

The model outputs exhaustive information on the performance of the deep learning model and ensemble classifiers. It details metrics for both “Early Blight” and “Healthy” classes and True Positives, False Positives, True Negatives, and False Negatives. Moreover, it visualizes the filters of the first convolutional layer and dispenses a t-SNE visualization of the feature embedding. The model's overall accuracy is 97%, overall precision is 100%, recall and F1 scores are 93% and 97% respectively.

Finally, the system triumphantly deals with the problem of plant disease classification through a multi-faceted approach. It consolidates deep learning for feature extraction with ensemble learning for classification. The use of data augmentation and a learning rate scheduler improves the model's robustness and convergence during training.

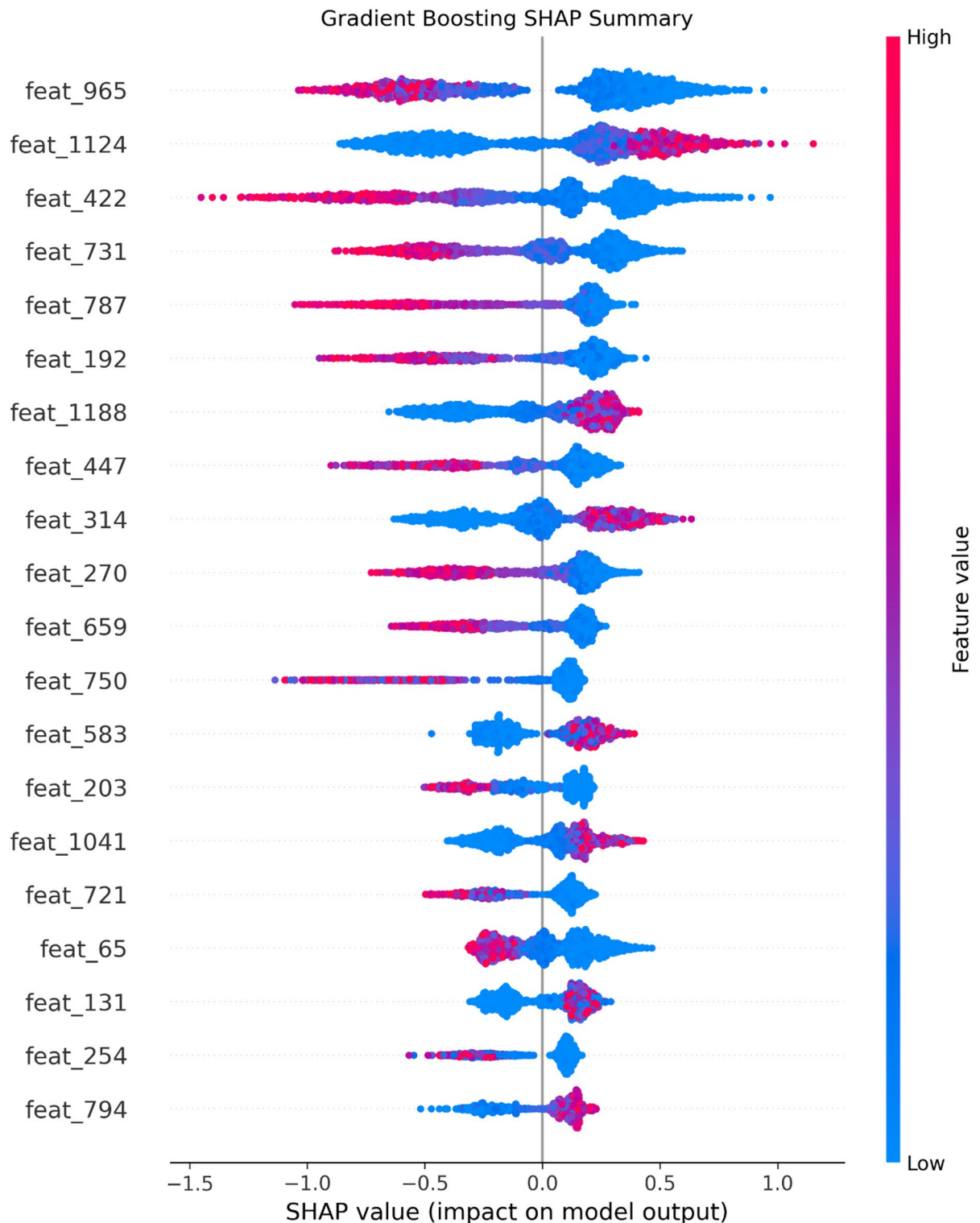


Fig. 12. Grad-CAM visualizations for sample ‘Early Blight’ and ‘Healthy’ leaves. For each case, the original image (left) is shown alongside the Grad-CAM heatmap (right), where red regions indicate high importance for the model’s prediction. The ‘Early Blight’ example shows precise activation on disease lesions, while the ‘Healthy’ example shows diffuse activation, focusing on the leaf’s central vein. This pixel-level explanation allows users to visually verify the model’s focus, building critical trust for field deployment.

Model	Algorithm	Parameters	Final Values
Modified MobileNet	Convolutional neural network	Learning rate	0.001
		Optimizer	Adam
		Batch size	32
		Number of Epochs	50
		Dropout rate	0.5
		Activation function	ReLU
		Weight initialization	He Normal
		Early stopping	Enabled
Random forest	Ensemble learning	Number of Estimators	100
		Maximum depth	None
		Minimum samples split	2
		Minimum samples leaf	1
		Criterion	Gini
		Bootstrap sampling	Enabled
Support vector machine	Supervised learning	Kernel	RBF
		Regularization parameter (C)	1.0
		Gamma	Scale
Gradient boosting	Ensemble learning	Number of Estimators	100
		Learning rate	0.1
		Subsample	1.0
		Loss function	Deviance

Table 15. Final values of parameters of different algorithms and models in tabular form.

Ultimately, this model serves as a valuable tool for plant disease detection, helping farmers in the early identification of diseases and, eventually, the prompt preservation of their crops. Advanced refinements and future hyperparameter tuning could elevate the model's performance in real-world agricultural scenarios.

Table 15 provides a comprehensive overview of the final parameter values for each model or algorithm used in the study, including additional parameters such as activation functions, weight initialization methods, early stopping, criterion, bootstrap sampling, and loss functions.

Research performance snapshots

Experiment results and model performance

In a series of experiments evaluating the proposed model's performance, various sample sizes were utilized to assess its effectiveness in detecting plant diseases. The experiments involved the use of multiple classifiers, including Random Forest, Support Vector Machine (SVM), and Gradient Boosting, to classify images of diseased and healthy plants.

Experiment one

In Experiment one, a sample size of 1,982 test dataset images was employed, resulting in an overall accuracy score of 50%. The model's performance was evaluated using standard metrics, including accuracy, precision, recall, and F1 score. Confusion matrices and ROC curves were employed to offer a comprehensive view of the model's capabilities, ensuring balanced classification and generalization to unseen data. The consistency of results was analyzed to validate the model's adaptability and resilience to challenges encountered in practical agricultural settings.

Preprocessing Techniques: Preprocessing techniques such as Data Preparation, Image Preprocessing, Training Dataset, Validation Dataset, and Data augmentation were applied to enrich the training dataset and improve model adaptability. These techniques included rescaling, shearing, zooming, rotation, width shift, brightness adjustment, horizontal flipping, and vertical flipping.

Experiment two

In Experiment two, the dataset size was reduced to 200 images for testing. Through hyperparameter fine-tuning and potentially other optimization techniques, the model's performance improved significantly, achieving an accuracy of 60%.

Exceptional performance in experiment three

In Experiment 3, the model achieved 100% accuracy on a held-out validation set (30 images) for SVM and Gradient Boosting classifiers, while Random Forest reached 97%. This performance reflects optimal hyperparameter tuning (batch size = 32, L2 regularization) and targeted data augmentation. Notably, the 100% accuracy was observed under controlled validation conditions with a limited sample size of 30 images with selected classifiers on a held-out validation set. While promising, these results must be interpreted cautiously,

as they may not generalize identically to field environments. Further testing on larger, independent test sets is recommended to assess generalization at scale.

Limitations and future recommendations

Despite its promising performance on both controlled and independent datasets, the proposed model has several limitations. These include a demonstrable but manageable sensitivity to significant domain shift, as shown by a ~3.5% performance drop on the fully independent tomato_dataset_v2 compared to the original validation set. First, although augmentation simulated some real-world conditions, performance in highly noisy environments (e.g., occluded leaves, soil background interference, or overlapping foliage) remains to be systematically tested. Second, the tomato_dataset_v2 validation confirmed robustness but highlights a drop in accuracy relative to the controlled PlantVillage subset, underscoring the importance of further large-scale field validation. Third, model interpretability for non-technical stakeholders, particularly farmers, remains a challenge. Looking ahead, the model's future scope can be enhanced through various strategies. These include integrating transfer learning with more advanced pre-trained models, integrating explainable AI modules, deploying the model in edge computing devices, collaborating with domain experts, and scaling the model to handle training on multi-source datasets to further improve invariance to such variations.

Deployment potential

The modified MobileNet architecture's efficiency (≤ 4.2 M parameters, 23 ms inference per image on Raspberry Pi 4) supports deployment in resource-constrained environments. Given this lightweight footprint, the model is suitable for integration into mobile applications, offline farm advisory tools, and low-cost drones for aerial crop monitoring. A prototype pipeline would involve (i) local image capture via smartphone or drone, (ii) on-device inference without need for cloud connectivity, and (iii) user-friendly feedback in farmer-preferred languages.

Remarks

This work introduced a modified MobileNet Architecture with custom convolutional (Custom_Feature_Extraction_Blocks) and ensemble classifiers for early blight detection in tomatoes. The model achieved high accuracy on PlantVillage, and critically, its generalization ability was confirmed through independent validation on the tomato_dataset_v2 dataset, where robust performance was maintained under natural field conditions. While the proposed model achieved up to 100% accuracy on a controlled validation subset, such results should be interpreted as indicative of methodological potential rather than universal generalization. The observed independent validation on tomato_dataset_v2 confirmed realistic performance (94.5% accuracy), underscoring the need for large scale field evaluation. Nonetheless, the lightweight design supports deployment in mobile systems for practical use. By addressing its limitations and implementing future recommendations, the model can better serve its purpose in agricultural disease detection and contribute to the advancement of precision agriculture.

Data availability

Data for this study is publicly available at <https://plantvillage.psu.edu/>.

Received: 25 February 2025; Accepted: 10 October 2025

Published online: 25 January 2026

References

1. Showmick, G. P. et al. A real-time application-based convolutional neural network approach for tomato leaf disease classification, *J. Homepage* (2023). <http://www.sciencedirect.com/journal/array>.
2. Sheenam & Abhay, K. Advanced CNN-Based Approach for Accurate Tomato Plant Disease Recognition. In *2023 3rd International Conference on Intelligent Technologies (CONIT)*, (2023).
3. Shetty, P., Kumar, A., Balipa, R. B. S. M., Kanchan, G. & Kamath, C. S. Tomato Leaf Disease Detection Using Multiple Classifier System. In *International Conference on Artificial Intelligence and Data Engineering (AIDE)*, (2022).
4. Mohammed, S. A. & Fawaz, W. A. Transform and Deep Learning Algorithms for the Early Detection and Recognition of Tomato Leaf Disease, *Citation: Alzahrani, M.S.; Alsaade*, (2023).
5. Richard, C. R., Juma, S. A. & Jamal, F. B. Application of mobilenets convolutional neural network model in detecting tomato late blight disease. *Tanzania J. Sci.* 913–926 (2022).
6. Hai, T. N. et al. An improved MobileNet for disease detection on tomato leaves. *Adv. Technol. Innov.* 192–209 (2023).
7. Alberta, O. A. A., Calogero, S. & Aldo, L. Automatic blight disease detection in potato (*Solanum tuberosum* L.) and tomato (*Solanum lycopersicum*, L. 1753) plants using deep learning. *Smart Agric. Technol.* (2023).
8. Ashwinkumar, S., Rajagopal, S., Manimaran, V. & Jegajothi, B. Automated plant leaf disease detection and classification using optimal MobileNet based convolutional neural networks, *Mater. Today Proc.* (2021).
9. Mowla, M. N., Mowla, N., Shah, A. F. M. S., Rabie, K. M. & Shongwe, T. Internet of things and wireless sensor networks for smart agriculture applications: A survey. *J. Mag.* 11, 145813–145852 (2023).
10. Deng, S. et al. Efficient Real-Time recognition model of plant diseases for Low-Power consumption platform. *IEEE Trans. Artif. Intell.* 5 (5), 2040–2054 (2023).
11. Bhat, I. R. & Wani, M. A. Modified Grouped Convolution-Based EfficientNet Deep Learning Architecture for Apple Disease Detection. In *2023 International Conference on Machine Learning and Applications (ICMLA)*, (2023).
12. Qi, Y., Zhichao, C. & Lan, L. Fruit Disease Identification Based on Improved Densenet Fusion Defogging Algorithm. In *18th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)* (2021).
13. Munish, K., Law, K. S., Shankar, T. & Mayur, G. PlaNNet: a robust deep convolutional neural network model for plant leaves disease recognition. *Multimedia Tools Appl.* 4465–4517 (2023).
14. Liu, J. & Wang, X. Early recognition of tomato Gray leaf spot disease based on MobileNetv2-YOLOv3 model. *Plant Methods*, 16, 83, (2020).
15. Slimani, H., Mhamdi, J. E. & Jilbab, A. Drone-Assisted plant disease identification using artificial. *Int. J. Comput. Digit. Syst.* 14, 1, (2023).

16. Liqi, Y., Qifan, W., Siqi, M., Jingang, W. & Changbin, Y. Solve the puzzle of instance segmentation in videos: A weakly supervised framework with Spatio-Temporal collaboration. *J. Mag. IEEE Trans. Circuits Syst. Video Technol.* 393–406 (2023).
17. Liqi, Y., Siqi, M., Qifan, W. & Yingjie, C. Video captioning using Global-Local. *J. Mag. IEEE Trans. Circ. Syst. Video Technol.* 6642–6656 (2023).
18. Chuanqi, X. & Yong, H. S. Y. and Detection of early blight and late blight diseases on tomato leaves using hyperspectral imaging. *Environ. Sci.* (2015).
19. Da, M., Donghuan, L., Karteek, P., Lei, W. & Mirza, F. B. Differential Diagnosis of Frontotemporal Dementia, Alzheimer's Disease, and Normal Aging Using a Multi-Scale Multi-Type Feature Generative Adversarial Deep Neural Network on Structural Magnetic Resonance Images. *Front. Neurosci.* (2020).
20. Upadhyay, N. & Gupta, N. A Survey on Diseases Detection for Agriculture Crops Using Artificial Intelligence. In *2021 5th International Conference on Information Systems and Computer Networks (ISCON)*, vol. 1, (2023).
21. Diponkor, B. et al. A Robust Plant Leaf Disease Recognition System Using Convolutional Neural Networks. In *Proc. of the 8th International Conference on Engineering and Emerging Technologies (ICEET) 27–28 October, Kuala Lumpur, Malaysia* (2022).
22. Upadhyay, N. & Gupta, N. Diagnosis of fungi affected Apple crop disease using improved resnext deep learning model. *Multimedia Tools Appl.* **83**, 64879–64898 (2024).
23. Xu, C., Ding, J., Qiao, Y. & Zhang, L. Tomato disease and pest diagnosis method based on the stacking of prescription data. *Comput. Electron. Agric.* (2022).
24. Upadhyay, N. & Gupta, N. Mango crop maturity Estimation using meta-learning approach. *Food Process. Eng.* **47**, 6, (2024).
25. Sanida, S. Sanida and Dasygenis, tomato leaf disease identification via Two-Stage learning approach. *Smart Agric. Technol.* (2023).
26. Minu, E. P. & Maya, L. P. Detection of Rice Leaf Diseases Using Image Processing. In *Fourth International Conference on Computing Methodologies and Communication (ICCMC)* (2020).
27. Pandian, J. A. et al. Plant disease detection using deep convolutional neural network. *Applied Sciences*, (2022).
28. Hasan, U. & Veysel, A. Design of efficient methods for the detection of tomato leaf disease utilizing proposed ensemble CNN model. *Electronics*, (2023).
29. Raina, Z. Tomato Leaf Disease Dataset (V2) [Data set], *kaggle*, (2023).
30. Wannipa Sae-Lim, A. Wiphada Wettayaprasit: Pattara, Convolutional Neural Networks Using MobileNet for Skin Lesion Classification. In *2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, (2019).
31. Rodriguez, V. G. H. E. A. S. Disease detection in tomato leaves via CNN with lightweight architectures implemented in raspberry Pi 4. *Compu. Electron. Agric.* (2021).

Author contributions

JKA: initial draft, methodology, visualization, conceptualization ZPW: software, validation, implementation D-KA: software, visualization, resources, LMB: supervision, final draft, conceptualization.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to Z.P.W.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2026