Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.XXXXXXX

MSCPNet: A Multi-Scale Convolutional Pooling Network for Maize Disease Classification

MEHDHAR S. A. M. AL-GAASHANI[1], REEM ALKANHEL[2], MUTHANA ALI SALEM ALI[3], MOHAMMED SALEH ALI MUTHANNA [4], AHMED AZIZ[5,6], AND AMMAR MUTHANNA.[7,8], (Member, IEEE)

[1]School of Resources and Environment, University of Electronic Science and Technology of China,4 1st Ring Rd East 2 Section, Chengdu, 610056, Sichuan, China (e-mail: mr.mehdhar@uestc.edu.cn and mr.mehdhar@gmail.com) (Orcid:https://orcid.org/0000-0003-2612-0978)
[2]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia (e-mail: rialkanhal@pnu.edu.sa)
[3]Department of Computer-Aided Design and Engineering Design, National University of Science and Technology (MISiS) (e-mail: m2112648@edu.misis.ru)
[4]Department of international business Management, Tashkent state university of Economics, Tashkent, Uzbekistan (e-mail: a.muthanna@tsue.uz)
[5]Department of computer science, Faculty of computer and Artificial intelligence, Benha university, Egypt
[6]Engineering school, Central Asian University, Tashkent, Uzbekistan (e-mail: ahmed.aziz@fci.bu.edu.eg)
[7]Department of Applied Probability and Informatics, Peoples'Friendship University of Russia (RUDN University), 6 Miklukho-Maklaya St, Moscow, Russia 117198(e-mail: muthanna.asa@spbgut.ru)
[8]Department of Telecommunication Networks and DataTransmission, St. Petersburg State University of Telecommunication, St. Petersburg, Russia (e-mail: muthanna.asa@sut.ru)

Corresponding author: Reem Alkanhel (e-mail: rialkanhal@pnu.edu.sa)

This paper is supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R323), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia, in part the studies at St. Petersburg State University of Telecommunications. prof. M.A. Bonch-Bruevich were supported by the Ministry of Science and High Education of the Russian Federation by the grant 075-15-2022-1137

ABSTRACT Maize (*Zea mays*) is a critical crop for global food security and economic stability. However, it is highly vulnerable to various diseases such as northern leaf blight, common rust, and maize lethal necrosis, which can lead to significant crop losses if not detected early. Traditional CNN-based models, while effective in extracting spatial features, often fail to capture subtle multi-scale variations necessary for distinguishing between disease symptoms. These models also suffer from high computational complexity when deeper layers are introduced to handle fine-grained details. Transformer-based models, on the other hand, provide long-range dependencies but come with significant computational overhead, limiting their use in real-time agricultural applications. To overcome these challenges, we propose MSCPNet, a novel architecture that combines a truncated MobileNetV2 backbone with a Multi-Scale Convolutional PoolFormer block. The truncated backbone ensures that only essential layers for general feature extraction are retained, enhancing the model's adaptability across domains. The Multi-Scale Convolutional PoolFormer block captures both local and global dependencies through parallel convolutional branches of varying kernel sizes, while the PoolFormer module efficiently handles feature aggregation without the heavy computational cost associated with traditional attention mechanisms. This design allows the model to balance computational efficiency and high accuracy, making it highly suitable for real-time maize disease detection. Extensive evaluations on the maize leaf disease classification task yielded outstanding results, with the proposed MSCPNet achieving an accuracy of 97.44%, precision of 96.76%, recall of 97.37%, F1-score of 97.04%, and an MCC of 0.9653, with a model size of 998,084 parameters and 315,258,752 FLOPs. Furthermore, the model was evaluated on the PlantVillage dataset for tomato leaf disease classification, where it achieved an accuracy of 99.32%, precision of 99.32%, recall of 99.33%, F1-score of 99.32%, and an MCC of 0.9925. These results demonstrate the effectiveness and efficiency of MSCPNet in disease classification across different domains.

INDEX TERMS Maize Disease, Deep Learning, Feature Pooling, Image Classification, Multi-Scale Feature Aggregation

I. INTRODUCTION

Maize (*Zea mays*) is one of the most essential crops worldwide, providing a major source of food, livestock feed, and industrial raw materials. It is widely cultivated across diverse geographic regions, playing a crucial role in ensuring food security, particularly in areas heavily dependent on agriculture. The significance of maize production cannot be overstated, as it feeds millions globally and serves as a key commodity in various industries. However, despite its critical importance, maize is highly susceptible to a range of diseases, including northern leaf blight, common rust, and maize lethal necrosis, all of which can severely impact crop yield and quality. These diseases, if not detected early, can lead to catastrophic losses, with reductions in annual maize production ranging from 10% to 30% in severely affected regions [1]–[3]. The late identification of such diseases exacerbates the problem by allowing pathogens to spread rapidly, increasing the difficulty and cost of disease management. Thus, there is a pressing need for efficient, early detection methods that can minimize crop loss and ensure the sustainability of maize production systems [4].

In recent years, image processing techniques, combined with traditional machine learning algorithms, have been widely employed to address the problem of early plant disease detection [5]. These techniques leverage visual symptoms observed in leaf images to detect diseases at early stages, which helps reduce crop loss and supports food security efforts. Various machine learning algorithms, including Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Artificial Neural Networks (ANN), have shown considerable promise in early-stage disease detection by extracting important features such as color, texture, and shape from leaf images [6], [7]. However, these approaches have notable limitations. They often require manual feature extraction, which can be time-consuming and less effective when applied to complex, real-world scenarios. Additionally, the accuracy of traditional models tends to degrade in the presence of environmental variability, such as inconsistent lighting or occlusion from overlapping leaves. As a result, more advanced, automated solutions are needed to address these challenges and provide reliable disease detection across diverse conditions [8], [9].

Vision Transformers (ViTs) have transformed the field of computer vision by overcoming long-range dependencies through the use of self-attention mechanisms. These models capture global context in images, enabling the understanding of complex relationships across different regions of the image [10]. However, one of the main drawbacks of ViTs is their computational expense. The attention operations scale quadratically with the number of tokens, leading to significant memory and processing requirements as the input size grows. This limitation makes ViTs computationally expensive and difficult to deploy in resource-constrained environments [11]. Furthermore, the parametric nature of self-attention mechanisms exacerbates this issue, as each token's query, key, and value vectors must be independently computed and processed. Despite these challenges, recent advancements such as additive attention and separable self-attention mechanisms have been explored to reduce the computational burden while maintaining the ability to model global dependencies effectively. These innovations aim to make ViTs more practical for real-time applications without compromising performance [12], [13].

In this study, we propose a novel architecture designed to enhance the model's ability to capture both local and global dependencies by introducing the Multi-Scale convolutional PoolFormer block. This block process features at multiple scales simultaneously by incorporating parallel convolutional branches, each with a distinct kernel size. Following the convolutional branches, the PoolFormer block from [14] is applied, facilitating non-parametric token mixing, and improving the model's ability to capture important details across different spatial resolutions. Following the convolutional branches, the PoolFormer block from [14] is applied, facilitating non-parametric token mixing, and improving the model's ability to capture important details across different spatial resolutions. Pooling, in traditional architectures, is typically used to downsample feature maps and reduce dimensionality, often at the cost of losing fine-grained spatial information. In contrast, the PoolFormer module retains and aggregates multi-scale features effectively, which is crucial for accurately detecting variations in maize disease symptoms. Unlike conventional pooling methods, which may overlook subtle spatial details, PoolFormer ensures that both local and global dependencies are captured, thereby improving model robustness in agricultural image analysis. This makes it particularly beneficial in the context of maize disease classification, where subtle differences in symptoms can be the key to accurate diagnosis. This approach is particularly beneficial for handling the diverse presentation of maize diseases, such as variations in lesion textures and spot sizes, making the model more robust to subtle visual changes [15]. Moreover, many existing models rely on pretrained backbones without considering that the deeper layers of these networks tend to capture domain-specific features [16] [17], which may not generalize well to different tasks, such as maize disease detection. To address this, we truncate the pretrained backbone, removing the layers closest to the output while retaining the initial layers responsible for extracting generic features. This adjustment ensures the network remains flexible and capable of generalizing across different domains, improving its performance in real-world applications [18].

A. MOTIVATION AND CONTRIBUTIONS

The classification of maize diseases based on visual leaf symptoms presents unique challenges that we address in this study. Variability in leaf images due to factors such as lighting conditions, disease severity, and plant growth stages complicates the generalization of deep learning models. Moreover, traditional models often rely on pre-trained backbones without accounting for their limitations in domain-specific applications. For instance, deeper layers tend to capture highly specialized features that may not transfer well to tasks like disease classification, which requires both local and global context understanding. To this end, we propose a novel architecture that effectively handles these complexities by introducing a novel approach to disease classification.

Our work aims to provide a solution to the common challenges in maize disease detection, particularly focusing on balancing computational efficiency, feature extraction across multiple scales, and ensuring interpretability. The Multi-Scale Convolutional PoolFormer block is designed to process multi-scale information, capturing both local and global dependencies, which is crucial for addressing the variations in disease presentation, such as differences in lesion texture and spot size. Furthermore, by truncating the MobileNetV2 backbone and retaining only the initial layers responsible for capturing generic features, we improve the model's ability to generalize across domains, focusing on low-level representations that are more universally applicable to diverse tasks.

The main contributions of this work are summarized as follows:

1) We propose a novel architecture that integrates a truncated MobileNetV2 backbone with a Multi-Scale convolutional PoolFormer block. This custom block captures multi-scale information using parallel convolutional layers with varying kernel sizes, significantly improving the model's ability to differentiate between subtle variations in disease presentation.
2) Our model incorporates a non-parametric token mixing mechanism using PoolFormer module, which enhances global feature extraction without the computational overhead of traditional attention mechanisms. This allows the model to remain lightweight while achieving high performance, making it suitable for real-time agricultural applications.
3) We address the issue of class imbalance by applying data augmentation strategies specifically designed for under-represented classes, such as gray leaf spot. These strategies ensure that the model generalizes well across all disease categories, resulting in more balanced and accurate predictions.
4) We conduct extensive evaluations on publicly available maize disease dataset, demonstrating that our model outperforms existing approaches in terms of both accuracy and computational efficiency. The model's design enables real-time diagnostic capabilities in agricultural systems.
5) We employ Grad-CAM visualizations to highlight the regions of the input images that most influence the model's predictions. This enhances interpretability and trust, providing valuable insights for agricultural professionals when diagnosing diseases based on leaf symptoms.

II. RELATED WORK

Automated plant disease recognition has been a significant topic, and many AI-based methods have been developed for this task. With the advent of machine learning, researchers have extensively applied it to automate plant disease identification [19], [20]. Traditional machine learning approaches have also been used to automate this task. For instance, the authors of [21] used pattern recognition algorithms based on image-processing technology to identify alfalfa leaf diseases. Their study integrated clustering algorithms, such as K-means and fuzzy C-means, with supervised classification algorithms like SVM, achieving high accuracy in disease recognition. Similarly, in [22], the authors proposed a novel approach for cucumber disease recognition using Global-Local Singular Value Decomposition (GL-SVD) combined with SVM, which showed improved performance in recognizing cucumber leaf diseases. Additionally, [23] applied SVM in detecting sugarcane borer diseases by using image processing techniques and grid search methods to optimize parameters, achieving a 96% accuracy rate in detecting diseased and healthy sugarcanes. Moreover, Hamdani et al. [24] proposed a color histogram-based approach to detect oil palm leaf diseases. The authors used features from multiple color spaces (RGB, L*a*b, HSI, and HSV) and applied Principal Component Analysis (PCA) for feature reduction before classification with an artificial neural network (ANN), achieving an accuracy of 99.67%. Singh and Misra [25] utilized image segmentation and soft computing techniques, including genetic algorithms, to detect plant leaf diseases, demonstrating the effectiveness of automated approaches for early disease identification. Islam et al. [26] presented a potato disease detection system using image segmentation and a multiclass SVM, which achieved 95% classification accuracy on the PlantVillage dataset. This work showcases the strength of SVM-based approaches in handling multiclass classification problems in plant disease detection. Finally, Omrani et al. [27] employed radial basis function-based support vector regression (SVR) for detecting apple leaf diseases. Their method utilized K-means clustering for segmentation and outperformed ANNs in disease classification.

While traditional machine learning techniques have shown success in plant disease detection, the advancement of deep convolutional neural networks (CNNs) has transformed the field significantly [28]. CNNs can automatically learn hierarchical features from raw images, reducing the need for manual feature extraction. This capability has proven particularly useful in complex image recognition tasks, such as plant disease classification, where subtle visual patterns differentiate healthy and diseased plants. Several studies have extensively explored CNN-based methods for this purpose. For example, Joseph et al. [29] reviewed various CNN models for intelligent plant disease diagnosis and highlighted their potential in enhancing crop health monitoring. Similarly, Dhaka et al. [30] conducted a survey of CNN models applied to plant leaf disease prediction, identifying key architectures and techniques that have achieved remarkable results. Furthermore, Boulent et al. [31] provided an in-depth analysis of CNN applications for automatic crop disease identification, underscoring the role of deep learning in precision agriculture. These models have been shown to excel at automatically extracting meaningful features from raw images, thereby outperforming traditional machine learning methods. For instance, Chen et al. [32] demonstrated the effectiveness of transfer learning by applying pre-trained VGGNet models to plant disease identification. Their approach achieved notable accuracy, even when applied to complex datasets under real-world conditions. Moreover, Tariq et al. [33] employed the VGG16 model enhanced with explainable AI techniques to diagnose corn leaf diseases, achieving high accuracy while also offering interpretable results via Layer-wise Relevance Propagation (LRP). This combination of high performance and transparency is crucial in agricultural applications where trust in AI models is essential. Another compelling example is the work by Theerthagiri et al. [34], who utilized the SqueezeNet architecture for maize leaf disease detection. Their model, optimized for fewer parameters while maintaining high precision, achieved an accuracy of 97% across various maize disease classes, including blight, rust, and grey leaf spot. This work highlights the potential of lightweight models like SqueezeNet in resource-constrained environments where computational efficiency is a priority. In addition to purely deep learning approaches, several hybrid methods have successfully combined CNNs for feature extraction with traditional machine learning classifiers to improve plant disease recognition. For example, Al-Gaashani et al. [35] leveraged transfer learning and feature extraction from CNN models like MobileNetV2 and ResNet50V2, followed by a Gravitational Search Algorithm (GSA) to optimize these features, which were then passed to a Multinomial Logistic Regression (MLR) classifier for final disease classification, achieving a precision of 99.2%. Similarly, Al-Gaashani et al. [36] employed feature extraction from MobileNetV2 and NASNetMobile, reduced dimensionality using kernel PCA, and classified tomato leaf diseases using traditional classifiers like Random Forest and SVM, achieving high accuracy. Furthermore, Dash et al. [37] utilized deep features extracted by DenseNet201 and applied an optimized SVM for maize disease classification, achieving a classification accuracy of 94.6%.

While CNNs have demonstrated impressive success in plant disease detection, they face certain limitations when applied to real-world scenarios. Specifically, traditional CNNs often struggle to differentiate between critical and irrelevant features within complex images. Standard CNNs apply convolutional operations uniformly across the entire image, which may lead to equal attention being given to both significant and insignificant regions. This can result in noise or irrelevant details affecting the classification outcome, reducing the model's accuracy and generalization ability [38], [39].

To mitigate these shortcomings, attention mechanisms have been introduced in CNN architectures, enabling the models to focus on the most important regions of an image while suppressing irrelevant areas [40]. This is particularly vital in plant disease detection, where distinguishing between healthy and diseased sections of a leaf is often subtle but crucial. For instance, Albahli et al. [39] implemented a spatial-channel attention mechanism in their Efficient Attention Network (EANet), which significantly improved accuracy by reducing the impact of background noise in maize crop images. Similarly, Karthik et al. [41] embedded an attention mechanism within a residual CNN to enhance the detection of disease-specific regions in tomato leaves, leading to better classification performance. Recent developments have further integrated attention mechanisms with CNNs to address the complexities of plant disease classification. For example, Zhao et al. [45] proposed the RIC-Net model, which combines Inception and residual blocks with an embedded attention module to accurately classify diseases in corn, potatoes, and tomatoes. This fusion of attention and multi-scale feature extraction contributed to high classification accuracy. Additionally, Chen et al. [43] leveraged lightweight attention networks in rice disease detection, achieving robust performance even under challenging backdrop conditions. Furthermore, Lei Chen et al. [44] introduced an improved domain adaptation approach, incorporating a novel attention mechanism to handle discrepancies between source and target domains in rice disease image recognition, achieving significant improvements in accuracy under small sample scenarios. These advancements demonstrate the critical role of attention mechanisms in enhancing the effectiveness of CNNs for plant disease detection, helping models focus on key areas and improving both classification accuracy and robustness across various conditions [42].

While CNNs equipped with attention modules have enhanced the ability to focus on the most relevant features in plant disease detection, they still face limitations in capturing long-range dependencies and global context, especially in complex image scenarios. Vision Transformers (ViTs) have emerged as a more effective solution, offering the ability to model both local and global features through self-attention mechanisms. Unlike traditional CNNs that rely on convolutions and attention modules to extract local features, ViTs segment images into patches and apply self-attention to model the relationships between patches, making them better suited for capturing long-range dependencies.

Karthik et al. [46] demonstrated the potential of this approach in their dual track deep fusion network, which integrates the Swin Transformer with a depthwise feature pyramid for citrus disease classification, achieving an accuracy of 99.2%. Similarly, Guo et al. [47] proposed the Convolutional Swin Transformer (CST) for classifying plant diseases based on type and severity, which reached 98.5% accuracy, further proving the effectiveness of Transformers in processing spatial information. Pacal [48] explored a Vision Transformer model for maize leaf disease detection, leveraging a large dataset and achieving an impressive 99.5% accuracy, while Li et al. [49] introduced LMBRNet for tomato leaf disease classification, incorporating Vision Transformers and residual connections to achieve 99.7% accuracy. Jin et al. [50] utilized a multiple attention transformer method for super-resolution in grape disease recognition, which significantly enhanced model performance in detecting subtle disease markers, achieving high accuracy. Zhou et al. [51] presented a residual-distilled transformer for rice leaf disease identification, demonstrating how self-attention improves performance, achieving an accuracy of 98.3%. Li et al. [52] applied a spatial convolutional self-attention transformer module for strawberry disease classification in complex backgrounds, reaching 98.7% accuracy. Finally, Gole et al. [53] developed TrIncNet, a lightweight Vision Transformer network for identifying plant diseases, achieving competitive accuracy with fewer parameters and making it efficient for deployment in resource-constrained environments. Furthermore, Thakur, et al. [54] proposed a hybrid model that combines CNNs and vision transformers for plant disease detection, achieving an accuracy of 98.86% on the PlantVillage dataset by leveraging the feature extraction strengths of both models. Similarly, Thai et al. [55] introduced FormerLeaf, an efficient ViT-based model that achieved superior performance in cassava leaf disease detection by optimizing attention heads in each transformer layer, thereby reducing model complexity while maintaining high accuracy. These approaches demonstrate the growing trend of incorporating transformer-based architectures to further enhance the accuracy and robustness of plant disease detection systems.

While the ViT architecture and its variants have significantly advanced plant disease classification by leveraging attention mechanisms for token mixing, these methods are often computationally expensive, especially when processing large datasets. Recent works, such as [46]–[49], have demonstrated the power of hybrid approaches that combine convolutional and transformer-based models to enhance feature extraction and classification accuracy. However, these approaches still rely on computationally intensive attention-based token mixers, which limit their efficiency. To address these limitations, the PoolFormer block was introduced in [55], offering a simplified yet highly effective feature map mixing strategy that replaces attention mechanisms with non-parametric pooling operations. This model, built upon the MetaFormer architecture, demonstrated competitive performance across various vision tasks with fewer parameters and reduced computational complexity compared to traditional attention-based transformers. By utilizing the PoolFormer block, our proposed model incorporates the advantages of MetaFormer's general architecture while ensuring efficient and scalable feature extraction for plant disease classification.

III. MATERIALS AND METHODOLOGY

This section delineates the comprehensive architecture of the proposed model, together with the dataset utilized for training and testing, encompassing the preprocessing and augmentation techniques implemented. This study concentrates on the categorization of maize leaf diseases. We detail the integration of a truncated MobileNetV2 backbone with the novel Multi-Scale convolutional PoolFormer block, which leverages PoolFormer modules for efficient Feature map mixing. The objective is to achieve a compromise between collecting multi-scale features and guaranteeing computing efficiency, making the model suitable for real-time agricultural applications. Furthermore, proper dataset preparation, especially in the context of handling class imbalance, is crucial to ensuring balanced and robust model performance.

A. DATASET DESCRIPTION

The original dataset utilized for this study is the Maize Leaf Disease Dataset, obtained from Kaggle [56]. This dataset contains four distinct classes: blight, common rust, gray leaf spot, and healthy maize leaves. The dataset distribution for both the training and testing sets is provided in Table 1. The images were curated from two major datasets, PlantVillage and PlantDoc, ensuring that irrelevant and low-quality images were excluded [57], [58]. To address the class imbalance issue, particularly with gray leaf spot being underrepresented, different augmentation strategies were applied to balance the dataset.

TABLE 1: Details of the original dataset used for maize disease detection

Class	Training Images	Testing Images
Blight	975	171
Common Rust	1111	195
Gray Leaf Spot	488	86
Healthy	988	174
Total	**3562**	**626**

B. PREPROCESSING AND RESIZING

The original images from the dataset were resized to 224×224 pixels, aligning with the input size required by most pre-trained deep learning models. This resizing ensures compatibility with architectures such as MobileNetV2, which has been extensively used for plant disease classification tasks. Normalization was also applied to the images, using mean and standard deviation values of $\{0.485, 0.456, 0.406\}$ and $\{0.229, 0.224, 0.225\}$, respectively. These values are standard for ImageNet-pretrained models and help speed up convergence during training while improving generalization. The dataset was split into training and test sets, with augmentation applied exclusively to the training set, ensuring that the test set remained untouched for unbiased evaluation.

C. DATA AUGMENTATION AND CLASS BALANCING

To address the class imbalance, we applied data augmentation to the training set. For all classes except the gray leaf spot, five augmented images were generated for every original image. For the gray leaf spot, which had the fewest samples, 10 augmented images per original image were generated to ensure class balance. The augmentation process employed a variety of techniques to simulate real-world variations in image conditions. These techniques included random rotations, flips, brightness and contrast adjustments, motion blur, and grid distortion. The use of these augmentations increased the diversity of the training data and enhanced the model's ability to generalize to unseen conditions. The three different scenarios for the dataset are shown in Table 2. These include (1) the original dataset without any augmentation, (2) the dataset with uniform augmentation across all classes, and (3) the dataset with augmentation applied with additional handling for the less-represented class.

TABLE 2: Dataset distribution of training part under different scenarios

Scenario	Blight	Common Rust	Gray Leaf Spot	Healthy
Without Augmentation	975	1111	488	988
With Uniform Augmentation	5850	6666	2928	5928
With Augmentation and Class Balancing	5850	6666	5368	5928

D. AUGMENTATION STRATEGY

Data augmentation was performed using the albumentations library [59], which offers a wide range of transformation techniques suitable for image classification tasks. Augmentation was selectively applied to the training data, while the test data remained unaltered. Algorithm 1 outlines the augmentation pipeline used during the training process.

Algorithm 1: Augmentation and Class Balancing for Maize Disease Dataset

```
Input: $\mathcal{D}_{\text{train}} = \{(I_j, y_j)\}_{j=1}^{N}$: Training dataset, $\mathcal{A}$:
       Set of augmentation techniques, $K$: Number
       of augmented images per original image
Output: $\mathcal{D}_{\text{aug}}$: Augmented dataset
1  Initialize $\mathcal{D}_{\text{aug}} \leftarrow \mathcal{D}_{\text{train}}$;
2  foreach $(I_j, y_j) \in \mathcal{D}_{train}$ do
3      if $y_j = Gray\ Leaf\ Spot$ then
4          $K = 10$;
5      else
6          $K = 5$;
7      for $k = 1$ to $K$ do
8          Randomly apply augmentation from $\mathcal{A}$;
9          $I_{j,k} \leftarrow \mathcal{A}(I_j)$;
10         $\mathcal{D}_{\text{aug}} \leftarrow \mathcal{D}_{\text{aug}} \cup \{(I_{j,k}, y_j)\}$;
11 return $\mathcal{D}_{\text{aug}}$;
```

Through the thoughtful application of preprocessing and augmentation techniques, we significantly enhanced the robustness and generalizability of our deep learning model. By addressing the inherent class imbalance, particularly for the under-represented categories, we ensured a more equitable representation of the data, which in turn enabled the model to achieve strong performance across all categories. This approach not only improved the model's accuracy but also its ability to generalize to real-world data, reinforcing its applicability in practical settings. The augmentation techniques applied in our study are visually depicted in Figure 1. This figure illustrates the original image at the center, surrounded by different transformations applied to it.

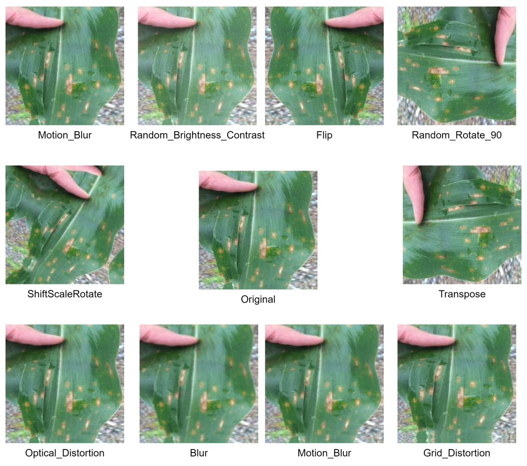


FIGURE 1: Visualization of augmentation methods applied to the dataset. The original image is at the center, with various augmentations surrounding it.

E. PROPOSED MODEL ARCHITECTURE

The architecture of the proposed MSCPNet model is composed of three essential components: a truncated MobileNetV2 utilized for initial feature extraction, a Multi-Scale Convolutional PoolFormer block designed for comprehensive multi-scale feature aggregation, and PoolFormer modules, which efficiently manage non-parametric token mixing. These components collectively enable the model to achieve a balance between accuracy and computational efficiency, making it particularly suitable for deployment in resource-constrained environments. The overall structure of the proposed model is illustrated in Figure 3. The overall process of our approach encompasses essential steps, including dataset preparation—comprising augmentation, normalization, and resizing—subsequently followed by model training and evaluation. The MSCPNet model is trained in conjunction with pre-trained models, and their performance is evaluated across different metrics. The whole methodology of our suggested approach is illustrated in Figure 2. Moreover, the Grad-CAM visualization is further employed to analyze the interpretability of the model predictions, as depicted in Figure 4.

1) Multi-Scale Convolutional PoolFormer Block

The Multi-Scale Convolutional PoolFormer Block is designed to capture features at multiple spatial scales by applying convolutional filters with different kernel sizes. The block includes four parallel branches, where three branches apply convolutions with kernel sizes 2×2, 3×3, and 5×5 respectively, followed by PoolFormer modules, and one branch applies a 1×1 convolution without any PoolFormer module.

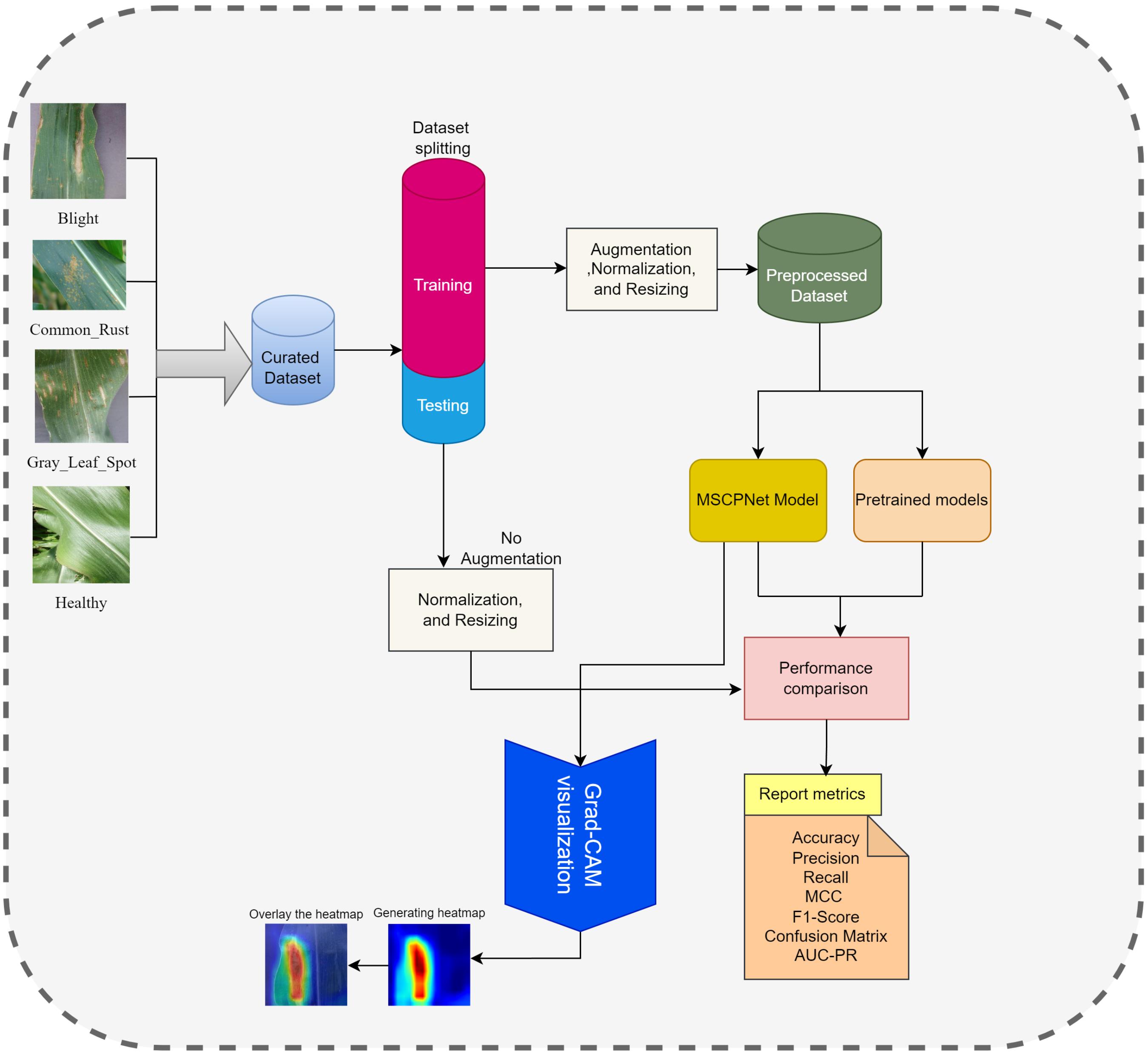


FIGURE 2: Flowchart of the complete methodology

These operations are represented as follows:

$$\mathbf{F}_k = \mathcal{P}_k(\text{Conv}_{k\times k}(\mathbf{F}_r)), \quad k \in \{2,3,5\},$$

where $\mathbf{F}_r$ is the reduced feature map from the truncated MobileNetV2 backbone, $\text{Conv}_{k\times k}$ represents the convolutional operation with a kernel size of k, and $\mathcal{P}_k$ denotes the PoolFormer operation applied to the resulting feature map.

The 1×1 convolution branch is represented as:

$$\mathbf{F}_{1x1} = \text{Conv}_{1\times 1}(\mathbf{F}_r)$$

After applying these operations in parallel, the resulting feature maps $\mathbf{F}_2$, $\mathbf{F}_3$, $\mathbf{F}_5$, and $\mathbf{F}_{1x1}$ are concatenated along the channel dimension:

$$\mathbf{F}_{\text{concat}} = \text{Concat}(\mathbf{F}_{1x1}, \mathbf{F}_2, \mathbf{F}_3, \mathbf{F}_5)$$

This concatenation operation aggregates the multi-scale information from all branches.

After concatenation, the merged feature map is passed through a final 1×1 convolution, represented as:

$$\mathbf{F}_{\text{merged}} = \text{Conv}_{1\times 1}(\mathbf{F}_{\text{concat}})$$

This 1×1 convolution reduces the dimensionality of the concatenated feature map, projecting the high-dimensional representation into a more compact form, which is more suitable for downstream tasks like classification.

Finally, the output is normalized using a Layer Normalization operation:

$$\mathbf{F}_{\text{output}} = \text{LayerNorm}(\mathbf{F}_{\text{merged}})$$

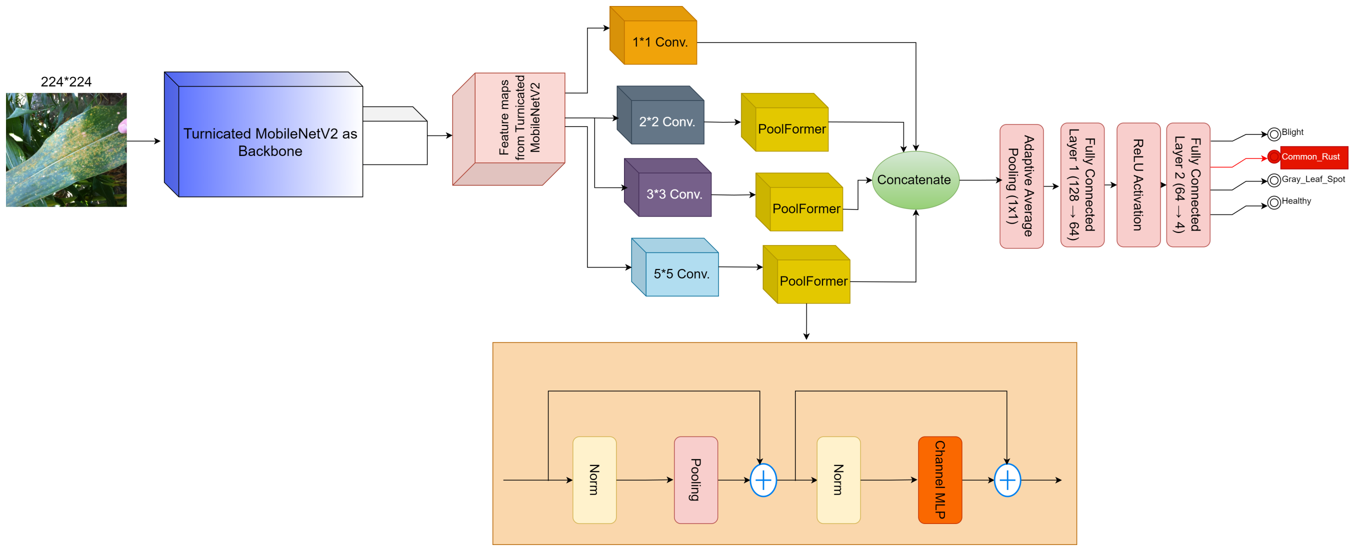

224*224
Turnicated MobileNetV2 as Backbone
Feature maps from Turnicated MobileNetV2
1*1 Conv.
2*2 Conv.
3*3 Conv.
5*5 Conv.
PoolFormer
PoolFormer
PoolFormer
Concatenate
Adaptive Average Pooling (1x1)
Fully Connected Layer 1 (128 → 64)
ReLU Activation
Fully Connected Layer 2 (64 → 4)
Blight
Common_Rust
Gray_Leaf_Spot
Healthy
Norm
Pooling
Norm
Channel MLP

FIGURE 3: MSCPNet model for maize disease classification

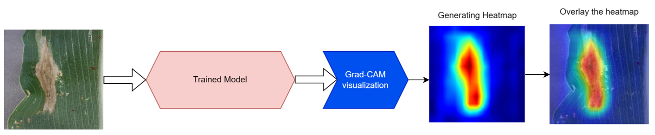


FIGURE 4: Grad-CAM visualization of MSCPNet model for maize disease classification

This normalization step ensures that the feature map is well-conditioned and stabilizes the training process. The resulting feature map $\mathbf{F}_{\text{output}}$ is then passed to the final classification layers.

2) Classification Head

The classification head consists of adaptive average pooling and two fully connected layers. The pooled features are flattened and passed through the fully connected layers:

$$\mathbf{F}_{\text{pool}} = \text{GAP}(\mathbf{F}_{\text{final}})$$

$$\mathbf{z}_1 = \text{ReLU}(\mathbf{W}_1\mathbf{F}_{\text{pool}})$$

$$\hat{y} = \mathbf{W}_2\mathbf{z}_1$$

Where $\hat{y}$ represents the predicted class probabilities for maize disease categories, including Blight, Common Rust, Gray Leaf Spot, and Healthy.

The integration of the Multi-Scale Convolutional PoolFormer block with the truncated MobileNetV2 backbone provides a balanced approach to feature extraction and computational efficiency. Truncating MobileNetV2 reduces the complexity of the model without sacrificing the critical low-level features required for accurate maize disease classification. At the same time, the Multi-Scale ConvPoolFormer block addresses the limitations of traditional CNNs in capturing global dependencies by leveraging multi-scale convolutions and PoolFormer blocks, which combine multi-scale feature aggregation with efficient feature maps mixing. The use of non-parametric operations in the PoolFormer module ensures that token mixing is performed without the high computational cost associated with attention mechanisms, preserving the model's ability to capture contextual dependencies in a computationally efficient manner. By capturing information across different spatial scales, this architecture enhances the model's ability to generalize to various maize disease patterns. Furthermore, the PoolFormer module ensures that the model remains computationally lightweight, making it ideal for real-world deployment in resource-constrained environments, such as small farms or mobile diagnostic applications.

IV. RESULT AND DISCUSSION

In this section, we present the experimental setup, the outcomes of our proposed model, and an ablation study to assess the impact of different components. Additionally, we provide a comparative evaluation of our model's performance with other existing methodologies.

A. EXPERIMENTAL CONFIGURATION

All experiments were conducted on a Windows 10 machine with a 64-bit architecture powered by an Intel64 processor (Family 6, Model 183, Stepping 1). The system also featured an NVIDIA GeForce RTX 3090 GPU, enabling GPU-accelerated training. The software environment consisted of Python 3.10.14, integrated with key libraries such as NumPy (v1.24.3), OpenCV (v4.10.0), PyTorch (v2.3.1), Torchvision (v0.18.1), Matplotlib (v3.9.0), Seaborn (v0.13.2), and Scikit-learn (v1.5.0). CUDA version 11.8 and cuDNN version 8700 were utilized to optimize GPU-based computations, ensuring that the deep learning workflows were efficient and reproducible.

Algorithm 2: MSCPNet for Maize Disease Classification

```
Input: $\mathbf{X} \in \mathbb{R}^{H\times W\times 3}$: Input maize leaf image
Output: $\hat{y} \in \mathbb{R}^{\text{num\_classes}}$: Predicted class probabilities
1 Step 1: Feature Extraction
    • Load pre-trained MobileNetV2 and truncate after a certain layer.
    • Extract base feature map:
          $\mathbf{F}_b \leftarrow \text{MobileNetV2}(\mathbf{X})$
  Step 2: Channel Reduction
    • Apply $1 \times 1$ convolution to reduce feature map dimensions:
          $\mathbf{F}_r \leftarrow \text{Conv}_{1\times 1}(\mathbf{F}_b)$
  Step 3: Multi-Scale Convolution and PoolFormer Application
    • Apply multi-scale convolutions and PoolFormer blocks:
          $\mathbf{F}_k \leftarrow \mathcal{P}_k(\text{Conv}_{k\times k}(\mathbf{F}_r)), \quad k \in \{2,3,5\}$
    • Apply $1 \times 1$ convolution branch in parallel:
          $\mathbf{F}_{1x1} \leftarrow \text{Conv}_{1\times 1}(\mathbf{F}_r)$
  Step 4: Feature Concatenation
    • Concatenate the outputs from all convolution branches:
          $\mathbf{F}_{\text{concat}} \leftarrow \text{Concat}(\mathbf{F}_2, \mathbf{F}_3, \mathbf{F}_5, \mathbf{F}_{1x1})$
  Step 5: Apply 1 × 1 convolution and Layer Normalization
    • Apply $1 \times 1$ convolution to reduce dimensions of concatenated features:
          $\mathbf{F}_{\text{final}} \leftarrow \text{Conv}_{1\times 1}(\mathbf{F}_{\text{concat}})$
    • Apply layer normalization:
          $\mathbf{F}_{\text{enh}} \leftarrow \text{LayerNorm}(\mathbf{F}_{\text{final}})$
  Step 6: Global Average Pooling and Classification
    • Perform global average pooling on the normalized features:
          $\mathbf{F}_{\text{pool}} \leftarrow \text{GAP}(\mathbf{F}_{\text{enh}})$
    • Flatten the pooled features and pass through fully connected layers:
          $\mathbf{z}_1 \leftarrow \text{ReLU}(\mathbf{W}_1\mathbf{F}_{\text{pool}}), \quad \hat{y} \leftarrow \mathbf{W}_2\mathbf{z}_1$
  Output: Return predicted class probabilities $\hat{y}$.
```

B. PERFORMANCE METRICS

To assess the performance of the model, we utilized a range of evaluation metrics, including accuracy, precision, recall, F1-score, and the Matthews Correlation Coefficient (MCC). These metrics provide a comprehensive evaluation of the classification tasks and are defined mathematically as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

$$F_1\text{-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

$$\text{MCC} = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{5}$$

Where TP refers to True Positives, TN refers to True Negatives, FP refers to False Positives, and FN refers to False Negatives.

In addition to the above metrics, we employed the Precision-Recall curve to illustrate the balance between precision and recall at different thresholds, which is especially helpful for imbalanced datasets. The area under this curve, known as AUC-PR, succinctly captures the model's performance in such settings. Furthermore, we used a confusion matrix, which provides detailed insight into the model's classification results by highlighting true positives, true negatives, false positives, and false negatives for each class. By employing these diverse metrics, our evaluation not only captures the overall performance of the model but also highlights specific strengths and potential areas for improvement, enabling a deeper understanding of how the model performs in various scenarios.

C. HYPERPARAMETER TUNING

In this study, we explored various hyperparameter settings to improve the model's performance. The hyperparameters and their configurations are outlined in Table 4. After extensive experimentation, the optimal settings were identified: the Adam optimizer, a batch size of 32, a learning rate of 0.001, and an input size of 224. These hyperparameters were found to provide the best balance between accuracy and generalization, while also maintaining computational efficiency.

D. TRAIN PROPOSED MODEL WITH AND WITHOUT AUGMENTATION

The results of the study are summarized in two figures, 5 and 6. These figures illustrate the performance of the model under different training conditions, including variations in data augmentation and class balancing strategies. Figure 5 shows a comparison of the overall model performance across three scenarios: training without augmentation, training with augmentation, and training with augmentation combined with class balancing for the underrepresented class. The accuracy, precision, recall, F1-score, MCC, and inference time metrics are displayed. It is evident that data augmentation significantly improves the model's performance. Specifically, augmenting the training data and handling class imbalance led to an increase in all metrics, with the highest accuracy of 97.44% achieved in the third scenario. This demonstrates that a balanced and diverse dataset, enriched by augmentation, is crucial for achieving better generalization in plant disease classification tasks. Additionally, precision and recall exhibit a noticeable increase when the augmentation is applied, with the recall rising from 94.64% in the first scenario (without augmentation) to 97.37% in the third scenario (augmentation with class balancing).

The improved recall indicates that the model is better at identifying true positive cases, and minimizing false negatives. The augmentation techniques have thus enhanced the model's sensitivity to identifying diseased plant leaves across different conditions. Figure 6 provides a class-wise breakdown of precision, recall, and F1-score for each of the four disease categories. It is clear from the figure that the augmentation strategies have yielded consistent improvements across all disease categories. For instance, the precision for gray leaf spot increased significantly from 0.83 without augmentation to 0.92 after augmentation and class balancing. This suggests that the model struggles less with underrepresented classes after augmentation, effectively capturing more nuanced features of the gray leaf spot category. For other categories like blight and common rust, the improvements are also noticeable, particularly in terms of recall. Blight's recall improved from 0.92 to 0.97, indicating that the model is now more adept at detecting diseased blight leaves. The F1 score for each class reflects similar trends, with the highest F1 scores observed in the augmented and class-balanced scenario, further reinforcing the positive impact of a well-augmented and balanced dataset. These results highlight the importance of a comprehensive data augmentation strategy to boost model performance across various metrics. Both figures demonstrate that augmenting the training data not only increases the model's overall accuracy but also ensures better class-wise performance, particularly for underrepresented classes like gray leaf spot.

E. PERFORMANCE COMPARISON BETWEEN MSCPNET, TRUNCATED MOBILENETV2, AND PRETRAINED MODELS

In this section, we assess the performance of the proposed MSCPNet model, featuring a truncated MobileNetV2 backbone paired with a Multi-Scale Convolutional PoolFormer block and benchmark it against several well-established pre-trained models. The comparison includes DenseNet121, ResNet50, ShuffleNetV2, SqueezeNet, and MobileNetV2. Each model was evaluated on the same dataset, and we report the key evaluation metrics: accuracy, precision, recall, F1-score, MCC, and inference time. The results, detailed in Table 5, indicate that DenseNet121 achieved an accuracy of 95.53% with an F1-score of 94.69%. While the model performs exceptionally well in categorizing the healthy class with perfect precision, recall, and F1-score, it faces difficulties with certain disease categories, particularly with lower F1-scores for some. DenseNet121 also has the highest computational demand, requiring 2.86 billion FLOPs and an inference time of 0.0289 seconds. On the other hand, ResNet50 provides slightly lower accuracy at 95.21%, though its inference time is faster at 0.0120 seconds. The model's computational cost is higher at 4.10 billion FLOPs, and while its performance across most classes is satisfactory, the F1-scores are slightly lower for more challenging disease categories. Optimized for efficiency, ShuffleNetV2 demonstrates a much lower inference time of 0.0107 seconds with only 147.8 million FLOPs, making it one of the most efficient models. However, its accuracy of 94.09% is somewhat lower, with an F1-score of 92.82%, especially struggling in the more challenging disease categories. SqueezeNet, another lightweight model, performs similarly with an accuracy of 93.61%, an F1-score of 92.12%, and an inference time of just 0.0091 seconds. Despite being the fastest model, its performance is lower in terms of precision and recall for certain classes. MobileNetV2, with its balance between speed and accuracy, achieves a commendable 96.65% accuracy and an F1-score of 96.04%. The model's inference time of 0.0110 seconds and 312.9 million FLOPs make it highly competitive, especially in classifying the more difficult disease categories. Our

TABLE 4: Hyperparameter configurations

Hyperparameter	Configurations
Input Size	224
Optimizers	Adagrad AdamW SGD Adam RMSprop
Batch Size	8, 16, 32
Learning Rate	0.01, 0.001, 0.0001
Early Stopping	Patience of 10 epochs
Loss Function	Categorical Cross Entropy
Number of Epochs	200

proposed MSCPNet model outperforms all the aforementioned models, achieving the highest accuracy at 97.44%, the best F1-score of 97.04%, and an MCC of 0.9653. MSCPNet demonstrates remarkable performance across all categories, maintaining precision and recall near perfect for the healthy class and showing improvements in challenging disease categories. Additionally, the computational efficiency is on par with MobileNetV2, with 315.2 million FLOPs and an inference time of 0.0111 seconds. The truncated MobileNetV2 achieves an accuracy of 95.85% with an F1-score of 95.09%. It is noteworthy for its balance between performance and efficiency, with a significantly lower inference time of 0.0083 seconds and a total FLOP count of 132.47 million, making it the most computationally efficient model in the comparison. Although it performs slightly lower in some metrics compared to MobileNetV2 without truncation. These results are summarized in Table 5, and confusion matrices for each model are illustrated in Figure 7. The confusion matrices provide insight into how well each model performs across all categories, highlighting classification errors and success rates. Furthermore, precision-recall curves are presented in Figure 8, showcasing the trade-offs between precision and recall for each model across all classes.

F. ABLATION STUDY OF MULTI-SCALE CONVOLUTIONAL POOLFORMER BLOCK WITH DIFFERENT BACKBONES

In this section, we analyze the effectiveness of the proposed Multi-Scale Convolutional PoolFormer block by integrating it with different pre-trained backbones, including DenseNet121, ResNet50, ShuffleNetV2, SqueezeNet, and MobileNetV2. The purpose of this ablation study is to highlight the performance improvements across various metrics such as accuracy, precision, recall, F1-score, MCC, and inference time when the proposed block is added to these backbones. As shown in Table 6, the inclusion of the MSCP-Net block led to notable improvements in accuracy for all the backbones, although in some cases there was a slight increase in inference time. For instance, DenseNet121, as a standalone model without the proposed block (discussed in the previous section), achieved an accuracy of 95.53% and an inference time of 0.0289 seconds. When the proposed block was incorporated, the accuracy improved to 95.85%, although the inference time slightly increased to 0.0302 seconds. Similarly, ResNet50 saw an accuracy increase from 95.21% to 95.53%, with a corresponding increase in inference time from 0.0120 seconds to 0.0180 seconds. The lightweight architectures, such as ShuffleNetV2 and SqueezeNet, also benefited from the inclusion of the proposed block. ShuffleNetV2, for example, improved its accuracy from 94.09% to 95.53%, while still maintaining a fast inference time of 0.0150 seconds. Likewise, SqueezeNet exhibited an increase in accuracy from 93.61% to 94.41%, with an inference time of 0.0136 seconds. The most significant performance enhancement was observed in MobileNetV2. Without the proposed block, MobileNetV2 had an accuracy of 96.65%, and after incorporating the Multi-Scale Convolutional PoolFormer block, it achieved the best performance across all metrics, with an accuracy of 97.28%, an F1-score of 97.04%, and an MCC of 0.9653. The inference time was 0.0111 seconds, comparable to its standalone performance. The proposed MSCPNet model, utilizing a truncated MobileNetV2 backbone, achieved the highest accuracy among all models, reaching 97.44 percent. It also demonstrated strong performance in terms of precision, recall, and F1-score, with values of 96.76 percent, 97.37 percent, and 97.04 percent, respectively. MCC further confirmed the model's robustness, reaching a value of 0.9653. Additionally, the model maintained high computational efficiency, with an inference time of 0.0111 seconds and a total of 315.2 million FLOPs, positioning it as both fast and highly accurate. The confusion matrices for each model, shown in Figure 9, further illustrate the impact of the proposed block in reducing misclassification errors. For example, DenseNet121 and ResNet50 both

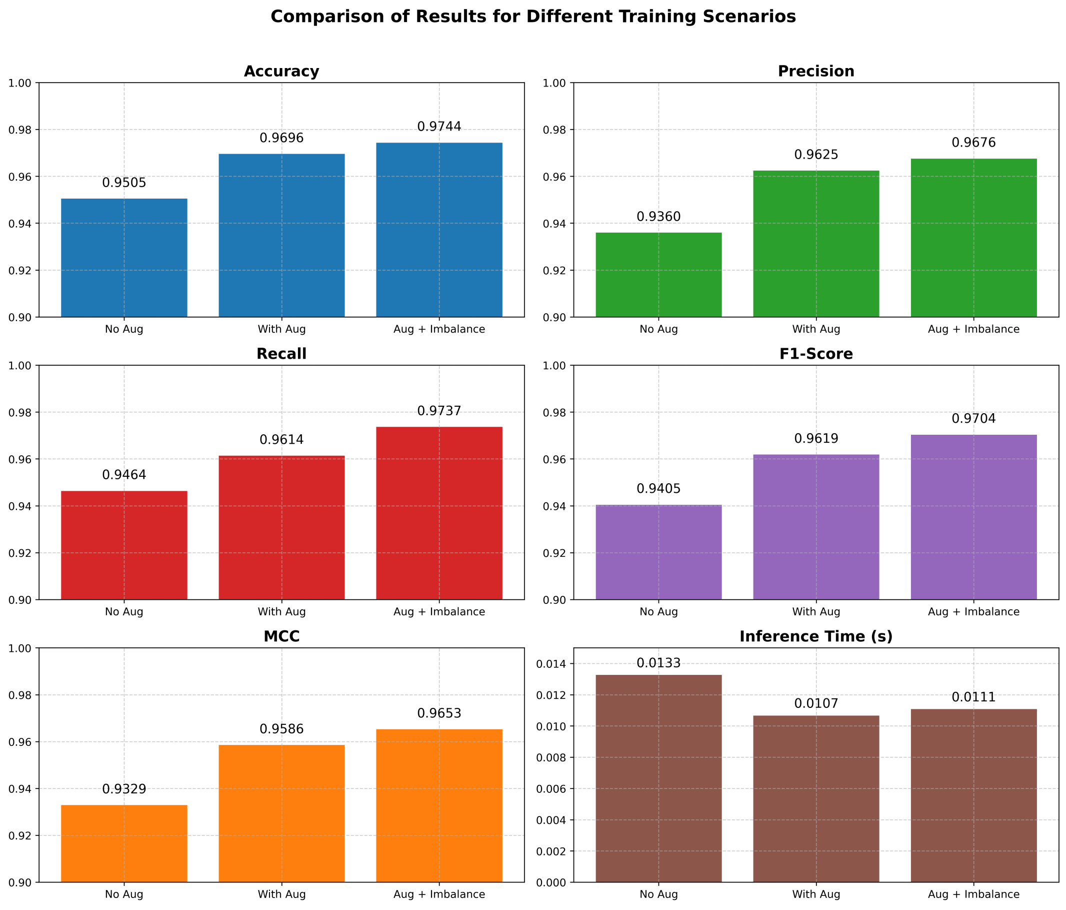


FIGURE 5: Comparison of performance metrics across different training scenarios: no augmentation, with augmentation, and augmentation with class balancing.

TABLE 5: Performance comparison of different pre-trained models, truncated MobileNetV2, and the proposed MSCPNet model.

Model	Accuracy	Precision	Recall	F1-Score	MCC	Inference Time (s)
DenseNet121	95.53%	94.37%	95.15%	94.69%	0.9396	0.0289
ResNet50	95.21%	93.71%	95.41%	94.37%	0.9356	0.0120
ShuffleNetV2	94.09%	92.20%	93.98%	92.82%	0.9209	0.0107
SqueezeNet	93.61%	92.13%	92.25%	92.12%	0.9135	0.0091
MobileNetV2	96.65%	96.26%	95.85%	96.04%	0.9542	0.0110
Truncated MobileNetV2	95.85%	95.02%	95.19%	95.09%	0.9434	0.0083
Proposed MSCPNet	**97.44%**	**96.76%**	**97.37%**	**97.04%**	**0.9653**	**0.0111**

showed improvements in the correct classification of difficult classes such as gray leaf spot. Moreover, the precision-recall curves displayed in Figure 10 confirm that the addition of the proposed block improved the AUC for each backbone, thereby enhancing the overall performance across various classes.

In summary, this ablation study demonstrates that the proposed MSCPNet block consistently enhances the performance of various backbones across all metrics, while maintaining efficient inference times. The ability of the block to reduce misclassification errors, as seen in the confusion matrices, further solidifies its value in real-world applications.

G. ANALYSIS OF FLOPS AND PARAMETER EFFICIENCY

In this section, we analyze the computational efficiency and parameter usage of various backbone models, both in their pre-trained configurations and after integrating the Multi-Scale Convolutional PoolFormer block, which is a key component of the proposed MSCPNet model. The evaluation focuses on two main metrics: FLOPs and the number of parameters, as these directly impact the model's suitability for real-time applications and resource-constrained environments. Figures 11 and 12 provide a detailed comparison of the total FLOPs and the number of parameters for various backbone models, both in their pre-trained configurations

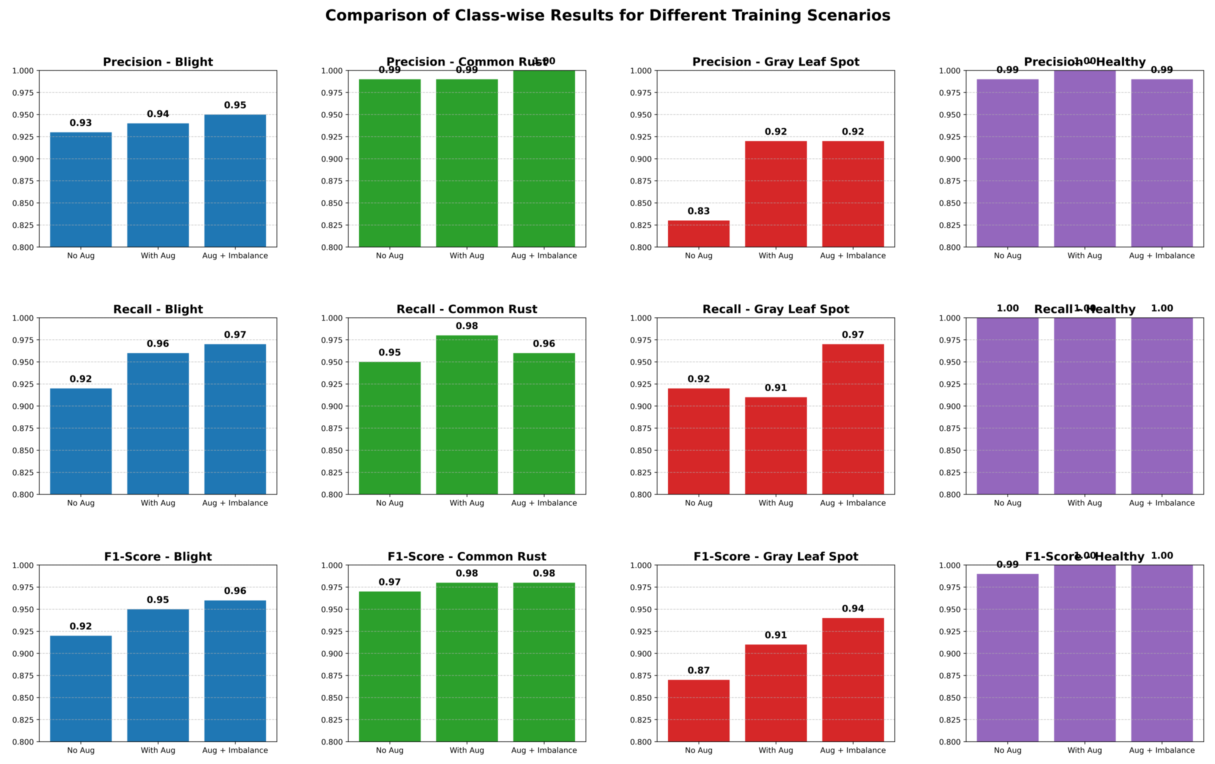
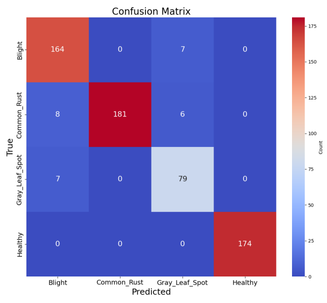


FIGURE 6: Class-wise comparison of precision, recall, and F1-score for blight, common rust, gray leaf spot, and healthy across the three training scenarios.

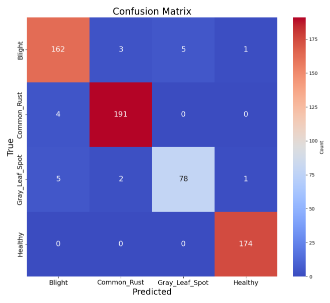
TABLE 6: Performance Comparison of different backbone models with the Proposed MSCPNet block.

Backbone Model	Accuracy	Precision	Recall	F1-Score	MCC	Inference Time (s)	Total FLOPs
DenseNet121	95.85%	94.82%	95.76%	95.24%	0.9436	0.0302	3,066,045,440
ResNet50	95.53%	94.16%	94.42%	94.29%	0.9390	0.0180	4,915,031,552
ShuffleNetV2	95.53%	93.96%	95.31%	94.51%	0.9396	0.0150	318,950,600
SqueezeNet	94.41%	92.84%	93.78%	93.25%	0.9241	0.0136	1,385,824,800
MobileNetV2	97.28%	96.37%	97.03%	96.67%	0.9631	0.0181	517,634,624
Proposed MSCPNet	**97.44%**	**96.76%**	**97.37%**	**97.04%**	**0.9653**	**0.0111**	**315,258,752**

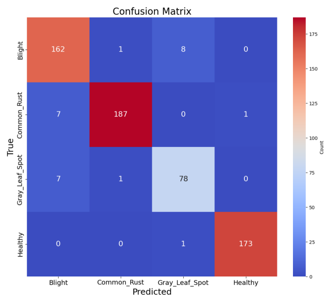
and after the inclusion of the proposed Multi-Scale Convolutional PoolFormer block. The FLOPs comparison, shown in Figure 11, highlights that lightweight models such as ShuffleNetV2 and MobileNetV2 maintain the lowest FLOP counts, with 147.8 million and 312.9 million FLOPs, respectively, in their pre-trained forms. After adding the Multi-Scale Convolutional PoolFormer block, these values increase to 318.9 million for ShuffleNetV2 and 517.6 million for MobileNetV2. Despite this increase, these models remain suitable for real-time applications or environments with limited computational resources. More complex models, such as DenseNet121 and ResNet50, exhibit significantly higher FLOP counts. For instance, DenseNet121's FLOPs increase from 2.86 billion in its pre-trained state to 3.06 billion after incorporating the Multi-Scale Convolutional PoolFormer block, while ResNet50's FLOPs increase from 4.10 billion to 4.91 billion. Although these models demand more computational power, they offer superior performance in tasks that require deep feature extraction. Interestingly, the proposed MSCPNet model achieves a competitive FLOP count, with 315.3 million FLOPs, surpassing even the modified MobileNetV2 and ShuffleNetV2 models in computational efficiency. This demonstrates that despite integrating multi-scale convolutional pooling block, which typically increases representational capacity, the MSCPNet architecture maintains low FLOPs while delivering notable performance improvements. In terms of parameter efficiency, as shown in Figure 12, ShuffleNetV2 and MobileNetV2 have relatively few parameters, with 1.26 million and 2.23 million parameters, respectively, in their pre-trained forms. After adding the Multi-Scale Convolutional PoolFormer block, the parameter count rises to 4.54 million for ShuffleNetV2 and 6.20 million for MobileNetV2. On the other hand, DenseNet121 and ResNet50 show much larger parameter counts, with DenseNet121 increasing from 6.96 million to 10.86 million parameters and ResNet50 increasing from 23.52 million to



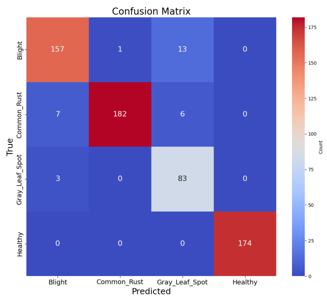
(a) DenseNet121



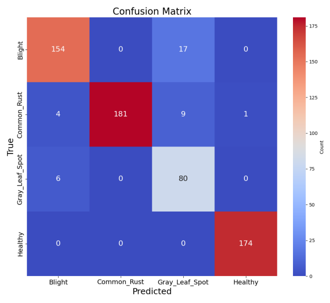
(b) MobileNetV2



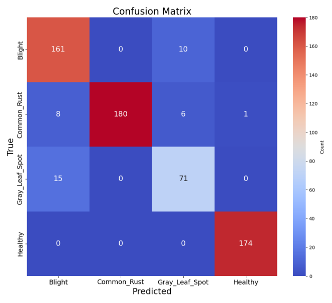
(c) Truncated MobileNetV2



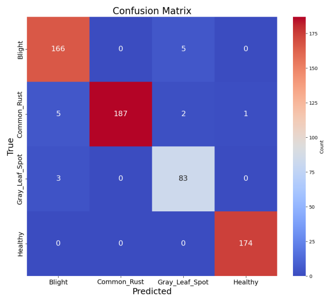
(d) ResNet50



(e) ShuffleNetV2



(f) SqueezeNet



(g) MSCPNet

FIGURE 7: Confusion matrices for different models.

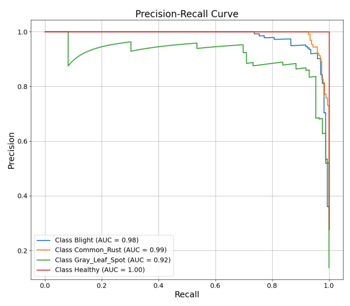
39.06 million. The proposed MSCPNet model maintains a low parameter count of 998,084, making it highly suitable for environments that require both high performance and parameter efficiency. These comparisons emphasize the significant trade-offs between computational efficiency, representational power, and parameter efficiency, reinforcing MSCPNet as an attractive solution for tasks that demand both high accuracy and resource-conscious designs.

H. EXPERIMENTATION OF MSCPNET MODEL USING DIFFERENT HYPERPARAMETERS

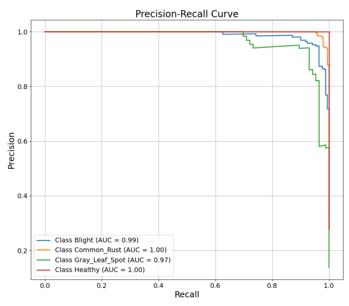
In this section, we present the experimental results obtained by evaluating our proposed MSCPNet model with different optimizers, learning rates, and batch sizes. These experiments aim to further explore the effect of various hyperparameters on the performance of our model and justify the final choices for the hyperparameters used in our best-performing configuration.

1) Experiments with Different Optimizers

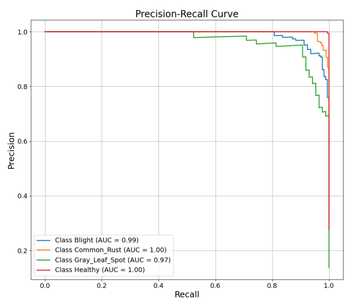
The choice of optimizer significantly influences both convergence speed and model performance. We experimented with several popular optimizers: Adagrad, Adam, AdamW, RMSprop, and SGD. As shown in Table 7, the results reveal that Adam achieved the best overall performance with an accuracy of 97.44%, an F1-score of 97.04%, and an MCC of 0.9653. The SGD optimizer also performed well, providing competitive results with an accuracy of 97.12% and an F1-score of 96.45%. These findings suggest that Adam and SGD are robust optimizers for this task, whereas RMSprop and AdamW yielded slightly lower performance. Adagrad performed the worst in terms of accuracy and F1-score, emphasizing the importance of carefully selecting optimizers for achieving optimal results.



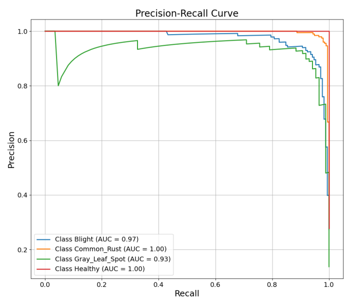
(a) DenseNet121



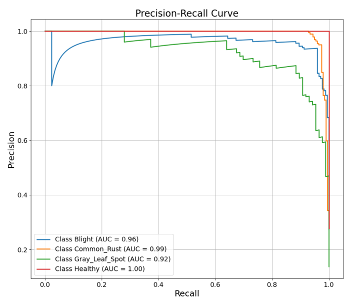
(b) MobileNetV2



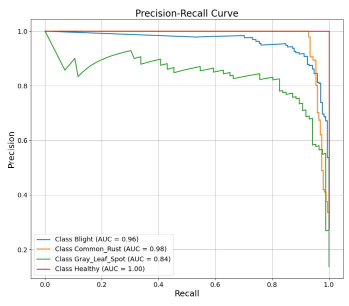
(c) Truncated MobileNetV2



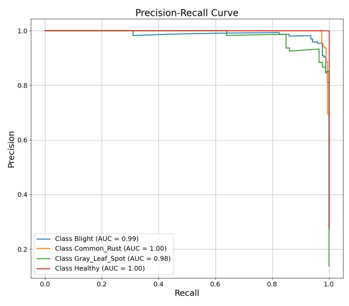
(d) ResNet50



(e) ShuffleNetV2



(f) SqueezeNet



(g) MSCPNet

FIGURE 8: Precision-Recall curves for different models.

TABLE 7: Performance comparison with different optimizers.

Optimizer	Accuracy	Precision	Recall	F1-Score	MCC
Adagrad	96.33%	95.62%	94.96%	95.26%	0.9498
Adam	**97.44%**	**96.76%**	**97.37%**	**97.04%**	**0.9653**
AdamW	96.33%	95.44%	95.75%	95.58%	0.9500
RMSprop	96.49%	95.84%	95.90%	95.87%	0.9521
SGD	97.12%	96.83%	96.16%	96.45%	0.9609

2) Experiments with Different Learning Rates

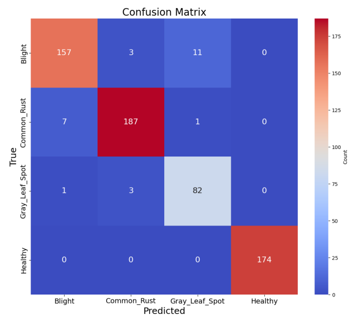
We explored the impact of different learning rates on the performance of MSCPNet, testing three values: 0.0001, 0.001, and 0.01. As summarized in Table 8, the optimal learning rate was found to be 0.001, achieving the highest accuracy of 97.44%, an F1-score of 97.04%, and an MCC of 0.9653. Lower and higher learning rates yielded lower performance, confirming that 0.001 strikes a balance between fast convergence and stable training, ensuring generalization without overfitting.

TABLE 8: Performance comparison with different learning rates.

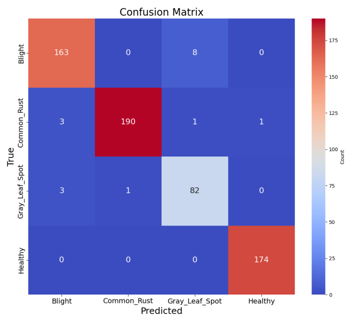
Learning Rate	Accuracy	Precision	Recall	F1-Score	MCC
0.0001	96.81%	96.16%	96.10%	96.12%	0.9564
0.001	**97.44%**	**96.76%**	**97.37%**	**97.04%**	**0.9653**
0.01	96.81%	95.84%	96.62%	96.20%	0.9566

3) Experiments with Different Batch Sizes

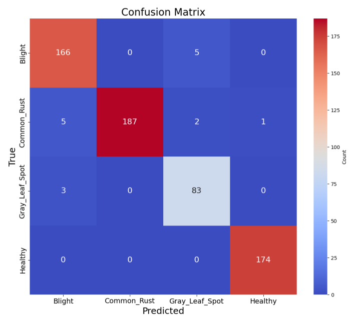
The batch size is another key hyperparameter that can influence model performance and convergence speed. We experimented with batch sizes of 8, 16, and 32, as displayed in Table 9. The results indicate that a batch size of 32 yielded the



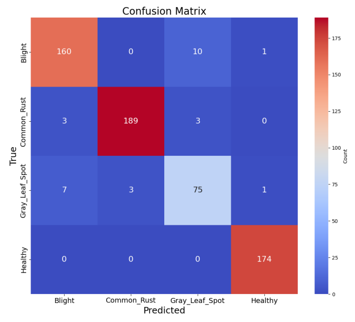
(a) DenseNet121 Confusion Matrix



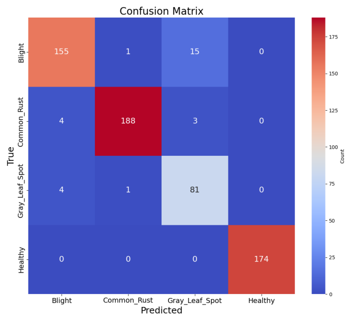
(b) MobileNetV2 Confusion Matrix



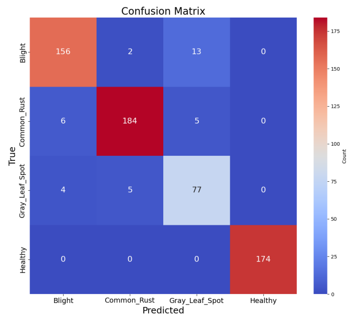
(c) MSCPNet Confusion Matrix



(d) ResNet50 Confusion Matrix

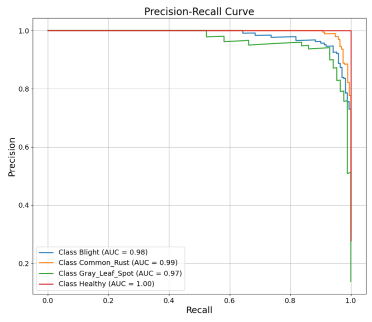


(e) ShuffleNetV2 Confusion Matrix

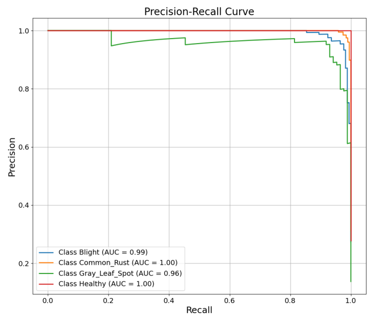


(f) SqueezeNet Confusion Matrix

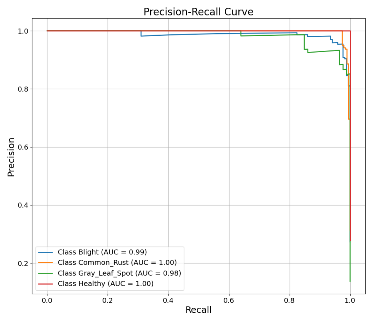
FIGURE 9: Confusion matrices for different model backbones, including truncated MobileNetV2 and other pre-trained models.



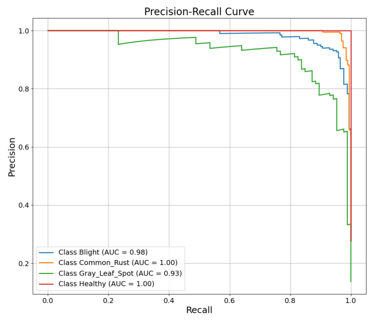
(a) DenseNet121 Precision-Recall Curve



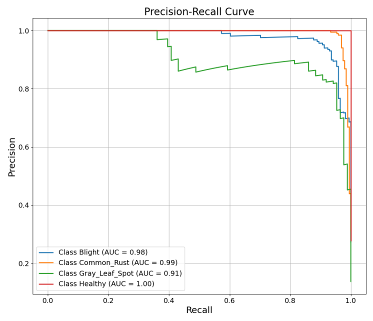
(b) MobileNetV2 Precision-Recall Curve



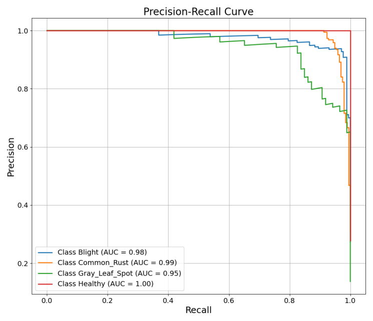
(c) MSCPNet Precision-Recall Curve



(d) ResNet50 Precision-Recall Curve



(e) ShuffleNetV2 Precision-Recall Curve



(f) SqueezeNet Precision-Recall Curve

FIGURE 10: Precision-Recall curves for different model backbones, including truncated MobileNetV2 and other pre-trained models.

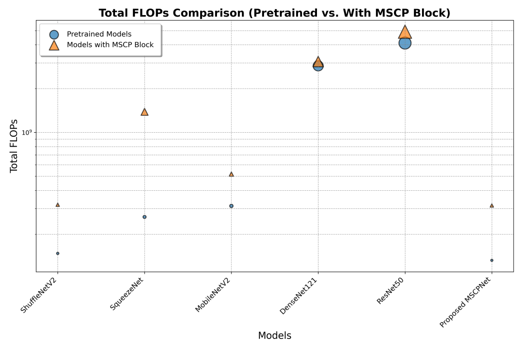


FIGURE 11: Total FLOPs comparison of pre-trained models and models with the multi-scale convolutional poolformer block. Circle sizes are proportional to flops, and the logarithmic scale illustrates the wide range of values. Blue circles represent the flops of the pre-trained models, while red circles show the flops after incorporating the multi-scale convolutional poolformer block.

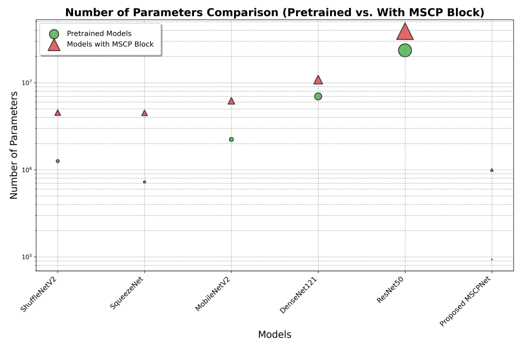


FIGURE 12: Comparison of the number of parameters for pre-trained models and models with the multi-scale convolutional poolformer block. Circle sizes are proportional to the number of parameters, with a logarithmic scale to accommodate the large range of values. Green circles represent the pre-trained models, while red triangles represent the models with the multi-scale convolutional PoolFormer block..

best performance, with an accuracy of 97.44%, an F1-score of 97.04%, and an MCC of 0.9653. This suggests that larger batch sizes allow the model to better approximate gradients, leading to more accurate updates. While a batch size of 8 provided comparable results, it slightly underperformed in terms of recall and F1-score.

TABLE 9: Performance comparison with different batch sizes.

Batch Size	Accuracy	Precision	Recall	F1-Score	MCC
8	97.28%	96.93%	95.96%	96.39%	0.9630
16	96.49%	95.78%	96.17%	95.96%	0.9521
32	**97.44%**	**96.76%**	**97.37%**	**97.04%**	**0.9653**

I. ABLATION STUDY ON TRUNCATION OF THE PRE-TRAINED MOBILENETV2 MODEL

An ablation study was conducted to explore the impact of utilizing a truncated version of MobileNetV2. The results, presented in Table 10, indicate that truncating the MobileNetV2 backbone and incorporating our Multi-Scale Convolutional PoolFormer block led to notable improvements. The proposed MSCPNet model, which is based on the truncated MobileNetV2, achieved an accuracy of 97.44%, an F1-score of 97.04%, and an MCC of 0.9653. These metrics surpass the performance of the complete MobileNetV2 model, which recorded an accuracy of 97.28% and an MCC of 0.9631. The truncation of MobileNetV2 reduced computational complexity, resulting in fewer FLOPs and an improved inference time of 0.0111 seconds, enhancing both speed and efficiency. This ablation study reveals that truncating the pre-trained model while introducing the proposed block can lead to performance gains in terms of both accuracy and computational efficiency. This finding supports the hypothesis that a truncated backbone, when combined with effective attention and pooling mechanisms, is advantageous. To further assess the impact of truncating the MobileNetV2 backbone, a detailed analysis was performed by comparing the confusion matrices and precision-recall curves for various models, including the proposed MSCPNet and other pre-trained backbones. The confusion matrices, depicted in Figures 9, illustrate the model's capacity to accurately classify different disease categories. Furthermore, the precision-recall curves, shown in Figures 10, highlight each model's performance in handling imbalanced datasets, particularly in rare cases. The proposed MSCPNet model, with its truncated MobileNetV2 backbone, demonstrates improvements in precision, recall, and overall classification accuracy, as evidenced by both confusion matrix and precision-recall curve analyses. These results, along with the findings in Table 10, confirm the enhanced performance and generalizability of the proposed model.

TABLE 10: Ablation study on truncation of the pre-trained MobileNetV2 model.

Model	Accuracy	Precision	Recall	F1-Score	MCC	Inference Time (s)
Proposed MSCPNet (MobileNetV2 backbone)	97.28%	96.39%	96.96%	96.39%	0.9631	0.0181
Proposed MSCPNet (Truncated MobileNetV2 backbone)	**97.44%**	**96.76%**	**97.37%**	**97.04%**	**0.9653**	**0.0111**

J. GRAD-CAM VISUALIZATIONS FOR DISEASE CLASSIFICATION

In order to gain insights into the feature learning process of the model and its focus during classification, we employed Gradient-weighted Class Activation Mapping (Grad-CAM) [60] to visualize the regions of interest in the input images. Grad-CAM generates a heatmap that highlights the areas in the image that most contributed to the model's decision. This visualization allows us to interpret the attention of the model and ensure that the network is focusing on disease-relevant regions, such as leaf lesions and discoloration. In Figure 13, we present a series of Grad-CAM visualizations for the disease classes in the maize dataset. Each disease category is represented with both the heatmap and the overlaid heatmap on the original image to clearly depict the regions the model focused on. Figure 13(a) shows the results for the blight category. The heatmap highlights a significant portion of the diseased area, with the overlaid version clearly marking the affected region on the leaf. Figure 13(b) presents the common rust category, where the heatmaps show focused attention on the rust spots spread across the leaf surface, with the overlay accurately mapping these areas. Figure 13(c) displays the results for gray leaf spot, illustrating the model's attention to the spots spread across the leaf, showing that the model correctly identifies the disease features. These Grad-CAM visualizations provide transparency into the decision-making process of the model and validate that the attention is placed on the correct regions, improving the model's interpretability. This step is crucial in applications like plant disease diagnosis, where understanding the model's predictions can provide confidence in real-world deployment.

K. ERROR ANALYSIS OF MISCLASSIFIED SAMPLES

In this study, several misclassified samples were observed during the evaluation of the model, underscoring the challenges it faced in accurately distinguishing between visually similar plant diseases. As shown in Figure 14, the misclassified samples include instances where the true and predicted labels do not align, offering insights into areas where the model struggled. A prominent source of confusion arose between blight and common rust, as both conditions exhibit similar visual features, such as the appearance of brown spots and lesions on the leaves, making it difficult for the model to differentiate between them. Additionally, some blight cases were incorrectly predicted as Healthy, likely due to the model's inability to capture subtle visual cues present in early-stage blight, where symptoms may not be as prominent or distinct from healthy leaves. These misclassifications suggest that while the model generally performed well, there are certain instances where improvements could be made. Enhancing feature extraction capabilities, particularly for diseases with overlapping visual characteristics, may help in reducing errors. Moreover, incorporating additional training data could further aid the model in better identifying the subtle differences between plant diseases and improving its overall robustness.

L. COMPARISON WITH OTHER STATE-OF-THE-ART METHODS

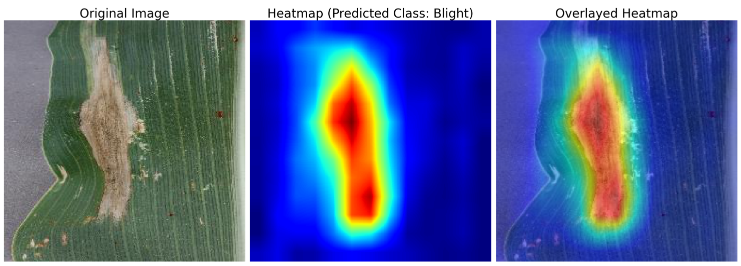
In this subsection, we compare the performance of our proposed MSCPNet with other state-of-the-art methods for plant disease classification using four classes from the Kaggle repository [56]. Table 11 presents the techniques, accuracies, and parameter counts for the selected models. As shown in Table 11, traditional CNNs empowered by specific mechanisms like VGG16 with Layer-wise Relevance Propagation (LRP) [33] achieve a respectable accuracy of 94.67%, but at the cost of higher complexity. Similarly, methods utilizing DenseNet201 coupled with SVM [37] showed a slightly lower accuracy of 94.6% and do not report the parameter count, indicating a focus on hybrid techniques for feature extraction and classification. Our proposed MSCPNet surpasses the performance of these methods, achieving an accuracy of 97.44% with 998,084 parameters, making it highly efficient compared to alternative models. Notably, MSCPNet outperforms the SqueezeNet-based method [34], which achieves 97% accuracy, but with a slightly higher parameter count. Additionally, the use of attention mechanisms, as demonstrated by Albahli et al. [39], shows that the integration of attention modules into CNNs can achieve superior results, with an impressive accuracy of 98.94% but at the expense of significantly larger model complexity of 8.23M parameters. Similarly, ResNet50 [61], ShuffleNetV2 [62], and MobileNetV2 [65] exhibit solid performance in the range of 94.09% to 96.65%, but their parameter counts range from 1.26M to 23.5M, highlighting the trade-offs between accuracy and model size. DenseNet121 [64], achieves 95.53% accuracy with 6.96M parameters, balancing performance and complexity effectively. However, despite its effectiveness, it still falls short of the efficiency provided by our proposed MSCPNet. These results demonstrate that while various deep learning models achieve competitive accuracies, our proposed MSCPNet not only delivers higher accuracy than most of these models but also maintains a significantly lower parameter count, making it suitable for real-world applications that require both high performance and efficiency.

Study	Dataset	#Classes	Technique	Accuracy	#parameters
[33]	Maize Dataset [56]	4	VGG16 empowered by LRP	94.67%	–
[37]	Maize Dataset [56]	4	DenseNet201 + SVM	94.6%	–
[34]	Maize Dataset [56]	4	SqueezeNet based model	97%	–
[39]	Maize Dataset [56]	4	CNN architecture with Attention module	98.94%	8.23M
[61]	Maize Dataset [56]	4	ResNet50	95.21%	23.5M
[62]	Maize Dataset [56]	4	ShuffleNetV2	94.09%	1.26M
[63]	Maize Dataset [56]	4	SqueezeNet	93.61%	724,548
[65]	Maize Dataset [56]	4	MobileNetV2	96.65%	2.23M
[64]	Maize Dataset [56]	4	DenseNet121	95.53%	6.96M
Proposed MSCPNet	Maize Dataset [56]	4	Multi-Scale Convolutional Pooling Network	97.44%	998,084

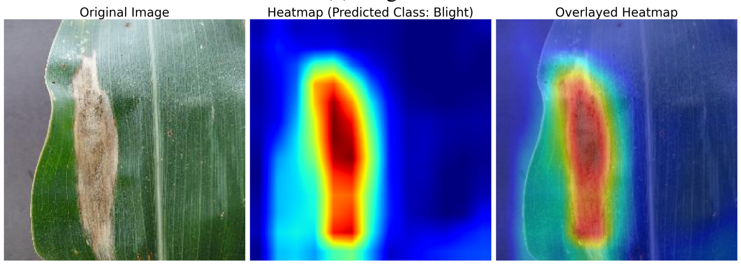
TABLE 11: Comparison of our proposed MSCPNet with other state-of-the-art methods

M. EVALUATION OF MSCPNET ON OTHER PLANT DISEASES

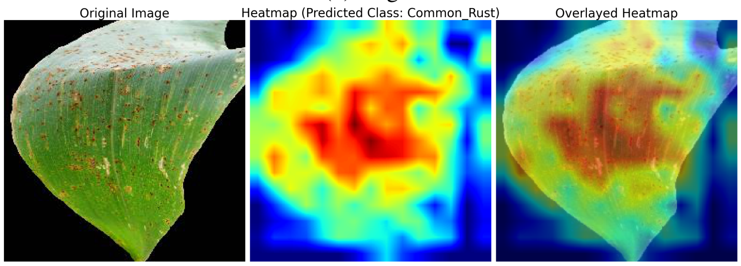
In this section, we evaluate the performance of the proposed MSCPNet model on tomato leaf disease classification from plantvillage dataset [66]. The dataset contains a total of 18,494 training images and 4,885 testing images, distributed across ten classes, as outlined in Table 12. The MSCPNet model was trained using the hyperparameters detailed in Table 13. The input size was fixed at 224×224 pixels, with a batch size of 32, using the Adam optimizer with a learning rate of 0.001. Early stopping was employed with a patience of 10 epochs to avoid overfitting, and the model was trained for a maximum of 200 epochs. The loss function used for optimization was categorical cross-entropy, which is well-suited for multi-class classification tasks. The classification performance metrics for the model, including precision, recall, and



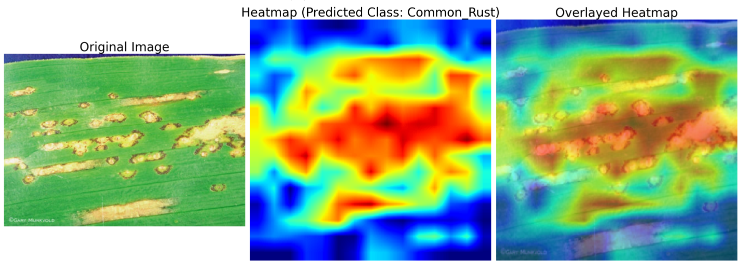
(a) Blight



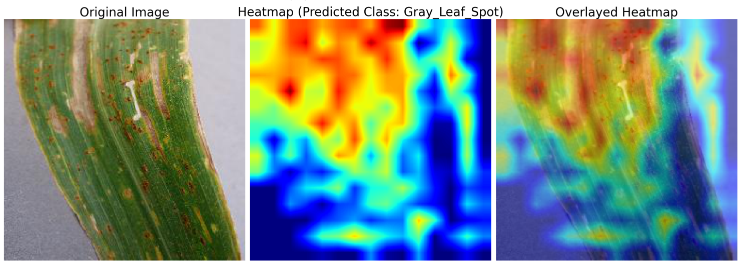
(b) Blight



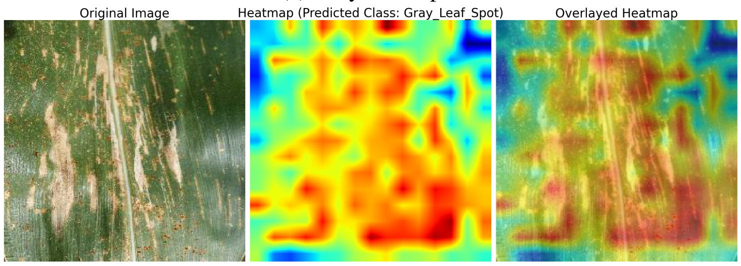
(c) Common Rust



(d) Common Rust



(e) Gray Leaf Spot



(f) Gray Leaf Spot

FIGURE 13: Grad-CAM visualizations for different disease classes. (a) Blight: Heatmap and overlaid heatmap. (b) Common Rust: Heatmap and overlaid heatmap. (c) Gray Leaf Spot: Heatmap and overlaid heatmap.

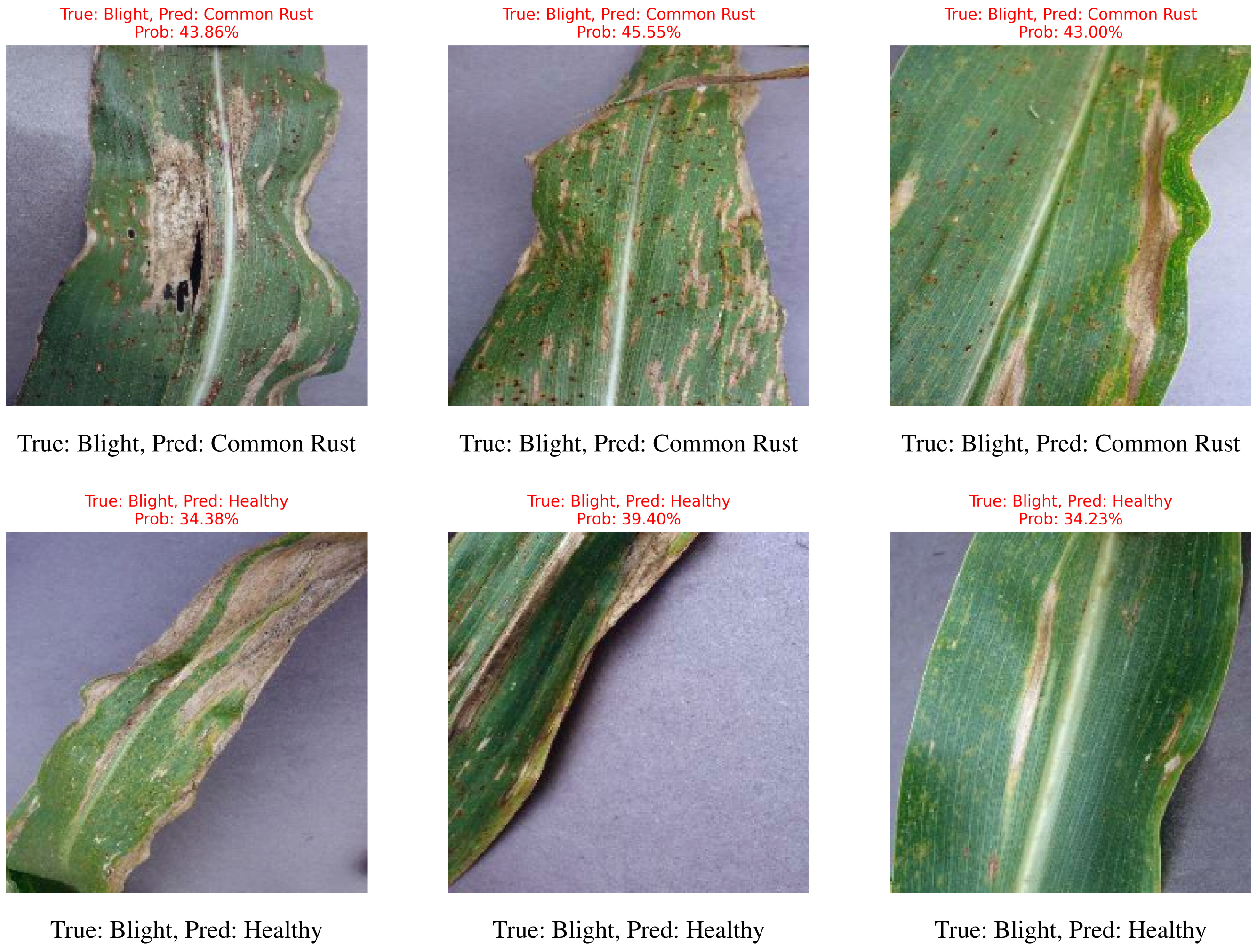

True: Blight, Pred: Common Rust
Prob: 43.86%
True: Blight, Pred: Common Rust
Prob: 45.55%
True: Blight, Pred: Common Rust
Prob: 43.00%
True: Blight, Pred: Common Rust
True: Blight, Pred: Common Rust
True: Blight, Pred: Common Rust
True: Blight, Pred: Healthy
Prob: 34.38%
True: Blight, Pred: Healthy
Prob: 39.40%
True: Blight, Pred: Healthy
Prob: 34.23%
True: Blight, Pred: Healthy
True: Blight, Pred: Healthy
True: Blight, Pred: Healthy

FIGURE 14: Misclassified samples with their true and predicted labels.

F1-score, are summarized in Table 14. The model achieved an overall accuracy of 99.32%, with high scores across all metrics, demonstrating its effectiveness in accurately classifying various tomato plant diseases. Figure 15 displays the precision-recall curve for each class, illustrating the strong performance of the model in distinguishing between tomato diseases. Additionally, the confusion matrix in Figure 16 provides insight into the model's classification accuracy and the degree of confusion between classes. Furthermore, Table 15 presents a comparison of MSCPNet with other state-of-the-art models from the literature.

TABLE 12: Dataset sizes for tomato disease detection

Class	Training Images	Testing Images
Tomato Bacterial Spot (Bact_Spot)	1702	425
Tomato Early Blight (Early_Blight)	1920	480
Tomato Healthy (Healthy)	1926	481
Tomato Late Blight (Late_Blight)	1851	463
Tomato Leaf Mold (Leaf_Mold)	1882	470
Tomato Septoria Leaf Spot (Septoria)	1745	436
Tomato Spider Mites (Spider_Mites)	1741	435
Tomato Target Spot (Target_Spot)	1827	457
Tomato Mosaic Virus (Mosaic_Virus)	1790	448
Tomato Yellow Leaf Curl Virus (Yellow_Leaf_Curl)	1961	490
Total	**18,494**	**4,885**

TABLE 13: Hyperparameter configurations

Hyperparameter	Configurations
Input Size	224
Optimizer	Adam
Batch Size	32
Learning Rate	0.001
Early Stopping	Patience of 10 epochs
Loss Function	Categorical Cross Entropy
Number of Epochs	200

As seen in Table 15, the proposed MSCPNet model outperforms several prior models in terms of accuracy, achieving 99.32% on the tomato leaves disease from the PlantVillage dataset [66]. This is a notable improvement compared to the standard CNN models [67], which achieved an accuracy of 98.49% with a significantly larger number of parameters of 1.42 million. Compared to the DenseNet model [68], which achieved 94.94% accuracy, MSCPNet demonstrates a considerable increase in performance with fewer parameters, highlighting its efficiency. Additionally, when DenseNet was trained with synthetic images, its accuracy improved to 97.11%, yet MSCPNet still surpasses it. Another interesting

TABLE 14: Classification performance metrics for tomato disease detection model

Class	Precision	Recall	F1-score	Support
Bact_Spot	0.99	1.00	0.99	425
Early_Blight	0.99	0.99	0.99	480
Late_Blight	0.99	0.99	0.99	463
Leaf_Mold	1.00	0.99	0.99	470
Septoria	0.99	0.99	0.99	436
Spider_Mites	1.00	1.00	1.00	435
Target_Spot	1.00	0.99	0.99	457
Mosaic_Virus	1.00	1.00	1.00	490
Yellow_Leaf_Curl	1.00	1.00	1.00	448
Healthy	1.00	1.00	1.00	481
Macro avg	0.99	0.99	0.99	4585
Weighted avg	0.99	0.99	0.99	4585
Accuracy	0.9932			
Precision	0.9932			
Recall	0.9933			
F1-score	0.9932			
MCC	0.9925			
Total FLOPs	315,259,136			
Avg Inference Time	9.489 milliseconds			

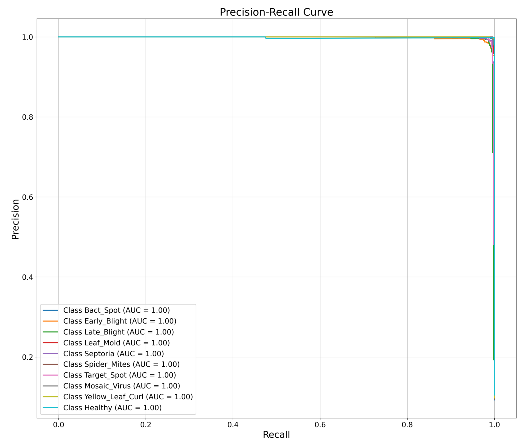


FIGURE 15: Precision-Recall Curve for Tomato Disease Detection Model using MSCPNet.

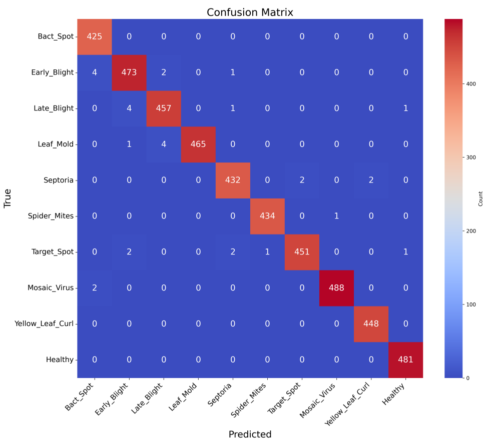


FIGURE 16: Confusion matrix for tomato disease detection model using MSCPNet.

comparison is with the CNN model proposed by Alnamoly et al. [69], which used depthwise separable convolutions and soft attention mechanisms. While this model achieved an accuracy of 99.04%, it required fewer parameters than MSCPNet. However, MSCPNet still achieved a higher accuracy with moderate complexity in terms of parameter count, making it a more balanced solution between accuracy and model complexity. The LMBRNet model [49] achieved the highest accuracy of 99.70% but at the cost of a very high number of parameters of 4.1 million. In contrast, MSCPNet offers a competitive accuracy while maintaining a more efficient model with just under 1 million parameters. A recent study by Kumar et al. [70] used a modified MobileNetV2 with transfer learning for tomato disease classification, achieving 99.28% accuracy with 2 classes and 3.28 million parameters. While this model achieved comparable accuracy, MSCPNet still provides better performance with fewer parameters. The MSCPNet model demonstrates a superior balance between accuracy and computational efficiency compared to other state-of-the-art approaches, making it a promising solution for the real-time detection and classification of plant diseases in practical agricultural settings.

N. LIMITATIONS AND FUTURE WORK

While the proposed MSCPNet model has demonstrated promising performance in maize disease detection, several limitations remain that must be addressed to further enhance its applicability and robustness. First, the evaluation of the model has been conducted on a specific maize dataset, which limits its generalizability to other crops and disease types. To address this, future work should expand the dataset to include a broader range of crops, diseases, and environmental conditions, such as variations in lighting, occlusion, and weather patterns. Such expansion would improve the model's robustness and its ability to perform reliably in diverse agricultural settings. Additionally, while data augmentation techniques were employed during training to increase variability, there is potential to explore more advanced augmentation methods. For example, employing generative adversarial networks (GANs) or domain-specific transformations could further enhance the model's robustness against real-world variations in disease appearance, particularly when dealing with complex or noisy data. Moreover, while the PoolFormer block has shown efficiency in token mixing, there is room for improvement in capturing long-range dependencies within the feature representations. Future research could investigate incorporating more sophisticated attention mechanisms, or exploring hybrid models that combine CNNs with ViTs or other attention-based architectures. These innovations could enhance the model's ability to capture fine-grained, long-range interactions and push the boundaries of accuracy and efficiency. Finally, deploying MSCPNet in real-time agricultural applications, especially in resource-constrained environments like small-scale farms, presents additional challenges. Future work will focus on optimizing the model for real-time deployment, including reducing memory usage and

TABLE 15: Performance comparison of MSCPNet with State-of-the-Art models

Study	Dataset	#Classes	Technique	Accuracy	#parameters
[67]	PlantVillage	10	CNN	98.49%	1,422,542
[68]	PlantVillage	10	DenseNet	94.94%	771,452
[68]	PlantVillage with Synthetic Images	10	DenseNet	97.11%	771,452
[69]	PlantVillage	10	CNN with Depthwise Separable Operation and Soft Attention	99.04%	221,594
[49]	PlantVillage	9	Multiple Branches Residual Net (LMBRNet)	99.70%	4.1 M
[70]	Kaggle	2	Modified MobileNetV2 with Transfer Learning	99.28%	3,284,450
Proposed MSCPNet Model	PlantVillage	10	Multi-Scale Convolutional Pooling Network	99.32%	998,474

improving inference speed. These optimizations are crucial for ensuring that the model can be applied effectively on edge devices and mobile platforms, making it feasible for practical, on-the-ground use.

V. CONCLUSION

In this study, we have proposed MSCPNet, a novel deep-learning architecture for the early and accurate detection of maize diseases, addressing the critical challenge of crop yield loss due to delayed diagnosis. Our approach, integrating a truncated MobileNetV2 backbone with a Multi-Scale Convolutional PoolFormer block, effectively captures both local and global features at multiple scales, enhancing the model's ability to handle the diverse and subtle visual manifestations of maize diseases. The backbone truncation strategy further reduces computational complexity while maintaining the essential layers required for general feature extraction, allowing the model to adapt across various domains. The inclusion of non-parametric feature maps mixing through the PoolFormer module has been shown to significantly improve feature aggregation, offering an efficient alternative to the traditional self-attention mechanism. This not only reduces the computational overhead but also enables real-time performance, a crucial requirement for practical deployment in agricultural environments. The proposed MSCPNet model achieved notable results on maize disease datasets, with an accuracy of 97.44%, precision of 96.76%, recall of 97.37%, F1-score of 97.04%, and a MCC of 0.9653. These results surpass the performance of existing methods while maintaining computational efficiency, with only 998,084 parameters and 315 million FLOPs. Additionally, MSCPNet was further evaluated on a tomato leaf disease classification task, where it achieved an accuracy of 99.32%, precision of 99.32%, recall of 99.33%, and F1-score of 99.32%, demonstrating its generalizability to other plant disease datasets. The proposed architecture demonstrated superior performance compared to existing methods, both in terms of accuracy and computational cost, as validated on publicly available maize disease datasets. Moreover, the interpretability of the model was enhanced through Grad-CAM visualizations, providing clear insights into the regions of the image that most influence the model's predictions.

ACKNOWLEDGEMENTS

This paper is supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R323), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia, in part the studies at St. Petersburg State University of Telecommunications. prof. M.A. Bonch-Bruevich were supported by the Ministry of Science and High Education of the Russian Federation by the grant 075-15-2022-1137

DATA AVAILABILITY

The datasets utilized in this study are thoroughly described and properly cited within the article.

DECLARATION OF INTERESTS

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

REFERENCES

[1] Phytopathology Research, Advances in research on maize lethal necrosis, available at: https://phytopatholres.biomedcentral.com/
[2] Maize Lethal Necrosis disease: review of molecular and genetic resistance mechanisms, socio-economic impacts, and mitigation strategies in sub-Saharan Africa. BMC Plant Biology, available at: https://bmcplantbiol.biomedcentral.com/
[3] Efficient attention-based CNN network (EANet) for multi-class maize crop disease classification. Frontiers in Plant Science, available at: https://www.frontiersin.org/
[4] Early Monitoring of Maize Northern Leaf Blight Using Vegetation Indices and Plant Traits from Multiangle Hyperspectral Data. Agriculture, available at: https://www.mdpi.com/
[5] Singh, V., Sharma, N., & Singh, S. "A review of imaging techniques for plant disease detection." Artificial Intelligence in Agriculture, 4 (2020): 229-242.
[6] Javidan, S. M., Banakar, A., Rahnama, K., Vakilian, K. A., & Ampatzidis, Y. "Feature engineering to identify plant diseases using image processing and artificial intelligence: a comprehensive review." Smart Agricultural Technology (2024): 100480.
[7] Plant disease detection using computational intelligence and image processing. Journal of Plant Diseases and Protection, available at: https://link.springer.com/
[8] Qadri, S. A. A., Huang, N.-F., Wani, T. M., & Bhat, S. A. "Advances and Challenges in Computer Vision for Image-Based Plant Disease Detection: A Comprehensive Survey of Machine and Deep Learning Approaches." IEEE Transactions on Automation Science and Engineering (2024).
[9] Development of plant disease detection for smart agriculture. Multimedia Tools and Applications, available at: https://link.springer.com/
[10] Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., & Shah, M. "Transformers in vision: A survey." ACM Computing Surveys (CSUR) 54, no. 10s (2022): 1-41.
[11] A survey of the vision transformers and their CNN-transformer based variants. Artificial Intelligence Review, available at: https://link.springer.com/
[12] SwiftFormer: Efficient Additive Attention for Transformer-based Real-time Mobile Vision Applications. arXiv, available at: https://arxiv.org/
[13] Mehta, S., & Rastegari, M. "MobileViT: Light-weight, general-purpose, and mobile-friendly vision transformer." arXiv preprint arXiv:2110.02178 (2021).
[14] Yu, Weihao, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. "MetaFormer is actually what you need for vision." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10819-10829. 2022.

[15] EmbedFormer: Embedded Depth-Wise Convolution Layer for Token Mixing. Sensors, available at: https://www.mdpi.com/

[16] Coulibaly, S., Kamsu-Foguem, B., Kamissoko, D., Traore, D. (2019). Deep neural networks with transfer learning in millet crop images. *Computers in Industry*, 108, 115-120.

[17] Chen, J., Chen, J., Zhang, D., Sun, Y., Nanehkaran, Y. A. (2020). Using deep transfer learning for image-based plant disease identification. *Computers and Electronics in Agriculture*, 173, 105393.

[18] Comparison of Deep Learning Models for Multi-Crop Leaf Disease Detection with Enhanced Vegetative Feature Isolation. Sensors, available at: https://www.mdpi.com/

[19] Iqbal, Zahid, Muhammad Attique Khan, Muhammad Sharif, Jamal Hussain Shah, Muhammad Habib ur Rehman, and Kashif Javed. "An automated detection and classification of citrus plant diseases using image processing techniques: A review." *Computers and electronics in agriculture*, 153 (2018): 12-32.

[20] Kaur, Sukhvir, Shreelekha Pandey, and Shivani Goel. "Plants disease identification and classification through leaf images: A survey." *Archives of Computational Methods in Engineering*, 26 (2019): 507-530.

[21] Qin, Feng, Dongxia Liu, Bingda Sun, Liu Ruan, Zhanhong Ma, and Haiguang Wang. "Identification of alfalfa leaf diseases using image recognition technology." *PloS one*, 11, no. 12 (2016): e0168274.

[22] Zhang, Shanwen, and Zhen Wang. "Cucumber disease recognition based on Global-Local Singular value decomposition." *Neurocomputing*, 205 (2016): 341-348.

[23] Huang, Tisen, Rui Yang, Wenshan Huang, Yiqi Huang, and Xi Qiao. "Detecting sugarcane borer diseases using support vector machine." *Information processing in agriculture*, 5, no. 1 (2018): 74-82.

[24] Hamdani, Hamdani, Anindita Septiarini, Andi Sunyoto, Suyanto Suyanto, and Fitri Utaminingrum. "Detection of oil palm leaf disease based on color histogram and supervised classifier." *Optik*, 245 (2021): 167753.

[25] Singh, Vijai, and Ak K. Misra. "Detection of plant leaf diseases using image segmentation and soft computing techniques." *Information processing in Agriculture*, 4, no. 1 (2017): 41-49.

[26] Islam, Monzurul, Anh Dinh, Khan Wahid, and Pankaj Bhowmik. "Detection of potato diseases using image segmentation and multiclass support vector machine." In *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1-4. IEEE, 2017.

[27] Omrani, Elham, Benyamin Khoshnevisan, Shahaboddin Shamshirband, Hadi Saboohi, Nor Badrul Anuar, and Mohd Hairul Nizam Md Nasir. "Potential of radial basis function-based support vector regression for apple disease detection." *Measurement*, 55 (2014): 512-519.

[28] Abade, André, Paulo Afonso Ferreira, and Flavio de Barros Vidal. "Plant diseases recognition on images using convolutional neural networks: A systematic review." *Computers and Electronics in Agriculture*, 185 (2021): 106125.

[29] Joseph, Diana Susan, Pranav M. Pawar, and Rahul Pramanik. "Intelligent plant disease diagnosis using convolutional neural network: a review." *Multimedia Tools and Applications*, 82, no. 14 (2023): 21415-21481.

[30] Dhaka, Vijaypal Singh, Sangeeta Vaibhav Meena, Geeta Rani, Deepak Sinwar, Muhammad Fazal Ijaz, and Marcin Woźniak. "A survey of deep convolutional neural networks applied for prediction of plant leaf diseases." *Sensors*, 21, no. 14 (2021): 4749.

[31] Boulent, Justine, Samuel Foucher, Jérôme Théau, and Pierre-Luc St-Charles. "Convolutional neural networks for the automatic identification of plant diseases." *Frontiers in Plant Science*, 10 (2019): 941.

[32] Chen, Junde, Jinxiu Chen, Defu Zhang, Yuandong Sun, and Yaser Ahangari Nanehkaran. "Using deep transfer learning for image-based plant disease identification." *Computers and Electronics in Agriculture*, 173 (2020): 105393.

[33] Tariq, Maria, Usman Ali, Sagheer Abbas, Shahzad Hassan, Rizwan Ali Naqvi, Muhammad Adnan Khan, and Daesik Jeong. "Corn leaf disease: insightful diagnosis using VGG16 empowered by explainable AI." *Frontiers in Plant Science*, 15 (2024).

[34] Theerthagiri, Prasannavenkatesan, A. Usha Ruby, J. George Chellin Chandran, Tanvir Habib Sardar, and Ahamed Shafeeq BM. "Deep SqueezeNet learning model for diagnosis and prediction of maize leaf diseases." *Journal of Big Data*, 11, no. 1 (2024): 112.

[35] Al-Gaashani, Mehdhar SAM, Nagwan Abdel Samee, Reem Alkanhel, Ghada Atteia, Hanaa A. Abdallah, Asadulla Ashurov, and Mohammed Saleh Ali Muthanna. "Deep transfer learning with gravitational search algorithm for enhanced plant disease classification." *Heliyon*, 10, no. 7 (2024).

[36] Al-gaashani, Mehdhar SAM, Fengjun Shang, Mohammed SA Muthanna, Mashael Khayyat, and Ahmed A. Abd El-Latif. "Tomato leaf disease classification by exploiting transfer learning and feature concatenation." *IET Image Processing*, 16, no. 3 (2022): 913-925.

[37] Dash, Arabinda, Prabira Kumar Sethy, and Santi Kumari Behera. "Maize disease identification based on optimized support vector machine using deep feature of DenseNet201." *Journal of Agriculture and Food Research*, 14 (2023): 100824.

[38] Zeng, Weihui, Haidong Li, Gensheng Hu, and Dong Liang. "Identification of maize leaf diseases by using the SKPSNet-50 convolutional neural network model." *Sustainable Computing: Informatics and Systems*, 35 (2022): 100695.

[39] Albahli, Saleh, and Momina Masood. "Efficient attention-based CNN network (EANet) for multi-class maize crop disease classification." *Frontiers in Plant Science*, 13 (2022): 1003152.

[40] Chelloug, S. A., Alkanhel, R., Muthanna, M. S. A., Aziz, A., & Muthanna, A. "MULTINET: A Multi-Agent DRL and EfficientNet Assisted Framework for 3D Plant Leaf Disease Identification and Severity Quantification." *IEEE Access*, 2023.

[41] Karthik, R., M. Hariharan, Sundar Anand, Priyanka Mathikshara, Annie Johnson, and R. Menaka. "Attention embedded residual CNN for disease detection in tomato leaves." *Applied Soft Computing*, 86 (2020): 105933.

[42] Al-Gaashani, Mehdhar SAM, Ammar Muthanna, Samia Allaoua Chelloug, and Neeraj Kumar. "EAMultiRes-DSPP: an efficient attention-based multi-residual network with dilated spatial pyramid pooling for identifying plant disease." *Neural Computing and Applications* (2024): 1-21.

[43] Chen, Junde, Defu Zhang, Adnan Zeb, and Yaser A. Nanehkaran. "Identification of rice plant diseases using lightweight attention networks." *Expert Systems with Applications*, 169 (2021): 114514.

[44] Chen, Lei, Jiaxian Zou, Yuan Yuan, and Haiyan He. "Improved domain adaptive rice disease image recognition based on a novel attention mechanism." *Computers and Electronics in Agriculture*, 208 (2023): 107806.

[45] Zhao, Yun, Cheng Sun, Xing Xu, and Jiagui Chen. "RIC-Net: A plant disease classification model based on the fusion of Inception and residual structure and embedded attention mechanism." *Computers and Electronics in Agriculture*, 193 (2022): 106644.

[46] Karthik, R., Sameeha Hussain, Timothy Thomas George, and Rashmi Mishra. "A dual track deep fusion network for citrus disease classification using group shuffle depthwise feature pyramid and swin transformer." *Ecological Informatics*, 78 (2023): 102302.

[47] Guo, Yifan, Yanting Lan, and Xiaodong Chen. "CST: Convolutional Swin Transformer for detecting the degree and types of plant diseases." *Computers and Electronics in Agriculture*, 202 (2022): 107407.

[48] Pacal, Ishak. "Enhancing crop productivity and sustainability through disease identification in maize leaves: Exploiting a large dataset with an advanced vision transformer model." *Expert Systems with Applications*, 238 (2024): 122099.

[49] Li, Mingxuan, Guoxiong Zhou, Aibin Chen, Liujun Li, and Yahui Hu. "Identification of tomato leaf diseases based on LMBRNet." *Engineering Applications of Artificial Intelligence*, 123 (2023): 106195.

[50] Jin, Haibin, Xiaoquan Chu, Jianfang Qi, Jianying Feng, and Weisong Mu. "Learning multiple attention transformer super-resolution method for grape disease recognition." *Expert Systems with Applications*, 241 (2024): 122717.

[51] Zhou, Changjian, Yujie Zhong, Sihan Zhou, Jia Song, and Wensheng Xiang. "Rice leaf disease identification by residual-distilled transformer." *Engineering Applications of Artificial Intelligence*, 121 (2023): 106020.

[52] Li, Gaoqiang, Lin Jiao, Peng Chen, Kang Liu, Rujing Wang, Shifeng Dong, and Chenrui Kang. "Spatial convolutional self-attention-based transformer module for strawberry disease identification under complex background." *Computers and Electronics in Agriculture*, 212 (2023): 108121.

[53] Gole, Pushkar, Punam Bedi, Sudeep Marwaha, Md Ashraful Haque, and Chandan Kumar Deb. "TrIncNet: a lightweight vision transformer network for identification of plant diseases." *Frontiers in Plant Science*, 14 (2023): 1221557.

[54] Thakur, Poornima Singh, Shubhangi Chaturvedi, Pritee Khanna, Tanuja Sheorey, and Aparajita Ojha. "Vision transformer meets convolutional neural network for plant disease classification." *Ecological Informatics*, 77 (2023): 102245.

[55] Thai, Huy-Tan, Kim-Hung Le, and Ngan Luu-Thuy Nguyen. "FormerLeaf: An efficient vision transformer for Cassava Leaf Disease detection." *Computers and Electronics in Agriculture*, 204 (2023): 107518.

[56] Smaranjit Ghose (2020). Corn or Maize Leaf Disease Dataset. https://www.kaggle.com/datasets/smaranjitghose/corn-or-maize-leaf-disease-dataset.
[57] Singh, D., Jain, N., Jain, P., Kayal, P., Kumawat, S., Batra, N. (2020). PlantDoc: A dataset for visual plant disease detection. *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*, 249-253.
[58] J, Arun Pandian, & Gopal, Geetharamani (2019). Data for: Identification of Plant Leaf Diseases Using a 9-layer Deep Convolutional Neural Network. *Mendeley Data*, V1, doi: 10.17632/tywbtsjrjv.1.
[59] Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., & Kalinin, A. A. "Albumentations: fast and flexible image augmentations." *Information*, vol. 11, no. 2, 2020, p. 125.
[60] Selvaraju, Ramprasaath R., Cogswell, Michael, Das, Abhishek, Vedantam, Ramakrishna, Parikh, Devi, & Batra, Dhruv (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 618–626.
[61] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778. 2016.
[62] Ma, Ningning, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. "Shufflenet v2: Practical guidelines for efficient CNN architecture design." In *Proceedings of the European conference on computer vision (ECCV)*, pp. 116-131. 2018.
[63] Iandola, Forrest N., Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size." *arXiv preprint arXiv:1602.07360*, 2016.
[64] Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. "Densely connected convolutional networks." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700-4708. 2017.
[65] Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. "MobileNetV2: Inverted residuals and linear bottlenecks." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510-4520. 2018.
[66] Hughes, David, and Marcel Salathé. "An open access repository of images on plant health to enable the development of mobile disease diagnostics." *arXiv preprint arXiv:1511.08060*, 2015.
[67] Trivedi, Naresh K., Vinay Gautam, Abhineet Anand, Hani Moaiteq Aljahdali, Santos Gracia Villar, Divya Anand, Nitin Goyal, and Seifedine Kadry. "Early detection and classification of tomato leaf disease using high-performance deep neural network." *Sensors*, vol. 21, no. 23, 2021, p. 7987.
[68] Abbas, Amreen, Sweta Jain, Mahesh Gour, and Swetha Vankudothu. "Tomato plant disease detection using transfer learning with C-GAN synthetic images." *Computers and Electronics in Agriculture*, vol. 187, 2021, p. 106279.
[69] Alnamoly, Mahmoud H., Anar A. Hady, Sherine M. Abd El-Kader, and Ibrahim El-Henawy. "FL-ToLeD: An Improved Lightweight Attention Convolutional Neural Network Model for Tomato Leaf Diseases Classification for Low-end Devices." *IEEE Access*, 2024.
[70] Kumar, B. Anil, Bansal, M., & Sharma, R. "Caffe-MobileNetV2 based tomato leaf disease detection." In 2023 3rd International Conference on Artificial Intelligence and Signal Processing (AISP), pp. 1-6. IEEE, 2023.

MEHDHAR S. A. M. AL-GAASHANI received the B.Sc. degree from Belgorod State Technological University (BSTU), Russia, and the M.Sc. degree from Don State Technical University, Russia. He received his Ph.D. degree from Chongqing University of Posts and Telecommunications, Chongqing, China, in 2023. He is currently a Postdoctoral Researcher with the University of Electronic Science and Technology of China (UESTC). His research interests include machine learning, deep learning, computer vision, image processing, and IoT.

PLACE PHOTO HERE

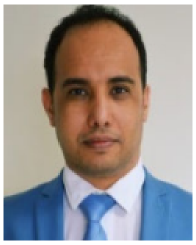
REEM ALKANHEL received the B.S. degree in computer sciences from King Saud University, Riyadh, Saudi Arabia, in 1996, the M.S. degree in information technology (computer networks and information security) from Queensland University of Technology, Brisbane, Australia, in 2007, and the Ph.D. degree in information technology (networks and communication systems) from Plymouth University, Plymouth, U.K., in 2019. She has been with Princess Nourah bint Abdulrahman University, Riyadh, since 1997. She is currently a Teacher Assistant with the College of Computer and Information Sciences. Her current research interests include communication systems, networking, the Internet of Things, information security, information technology, quality of service and experience, software defined networks, and deep reinforcement learning.

PLACE PHOTO HERE

MUTHANA ALI SALEM ALI received his M.S. degree in information technology Department of Computer-Aided Design and Engineering Design, National University of Science and Technology (Misis). Now, he is pursuing his Ph.D degree at the National University of Science and Technology. His research interests include machine learning and image processing.

MOHAMMED SALEH ALI MUTHANNA received the M.S. degree from the Computer Science Department, Saint Petersburg Electrotechnical University "LETI," Russia, in 2016, and the Ph.D. degree from Chongqing University of Posts and Telecommunications, Chongqing, China, in 2021. Currently, he is a Postdoctoral Fellow with the Institute of Computer Technologies and Information Security, Southern Federal University, Russia. His main research interests include mobile edge computing, software-defined networks (SDN), the IoT, industrial wireless, and sensor networks.

AHMED AZIZ (Member, IEEE) received the B.Sc. degree (Hons.) in computer science and the M.S. degree in computer science from the Faculty of Computers and Informatics, Benha University, Benha, Egypt, in June 2007 and October 2014, respectively, and the Ph.D. degree in computer science from the School of Computer and System Science, Jawaharlal Nehru University, New Delhi, India, in 2019. From December 2007 to December 2010, he was a Lecturer Assistant with the Computer Science Department, Faculty of Science, Benha University, where he was an Assistant Professor with the Faculty of Computer and Artificial Intelligence, from 2014 to 2019. From August 2019 to September 2020, he was an Associate Professor with the Department of Computer Science and Engineering, Sharad University, India. Since October 2020, he has been a Professor with the Department of International Business Management, Tashkent State University of Economics (TSUE), Tashkent, Uzbekistan. He is currently the Dean of the CAU Engineering School, Tashkent. He has published more than 18 research articles of SCI with high-impact factors, such as IEEE SENSOR JOURNAL (IF3.7), IEEE INTERNET OF THINGS JOURNAL (IF 9.07), Journal of Network and Computer Applications (IF 5.9), and IEEE ACCESS (IF 4.05); and quarter one Scopus index journals. His research interests include sensor networks, compressive sensing, computing, wireless networks, and the IoT.



AMMAR MUTHANNA (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees from Saint Petersburg State University of Telecommunications, in 2009, 2011, and 2016, respectively. He is currently an Associate Professor with the Department of Telecommunication Networks, the Deputy Head of Science, and the Head of the SDN Laboratory. From 2017 to 2019, he was a Postdoctoral Researcher with RUDN University. In 2012 and 2013, he took part in the Erasmus Student Program with the Faculty of Electrical Engineering, University of Ljubljana, and in 2014, he was a Visitor Researcher with Tampere University, Finland. He has been an active member of the technical program committee on many international conferences and journals. He has been an Expert of the Judges Panel and Challenge Management Board at AI-5G-Challenge, ITU, and a Russian Host Organizer. His research interests include wireless communications, 5G/6G cellular systems, the IoT applications, edge computing, and software-defined networking