Deep Learning-Based Automated Detection of Tomato Leaf Diseases Using CNNs

Daniel Musafiri Balungu, Maksim Aleksandrovich Malykh, Dmitry Evgenievich Burdin, Nikita Sergeevich Predein
Ural Federal University, Department of Big Data Analytics and Video Analysis methods, Yekaterinburg, Russia
E-mail: danielbal03.db@gmail.com, maksimmalyh99@gmail.com, Burdin.Dmitry@urfu.me, predein697@gmail.com

Keywords: Image processing, computer vision, plant leaf diseases, agricultural technology, crop protection, AI in agriculture, deep learning

Received: June 12, 2025

With the global prevalence of tomato diseases causing 20 to 40% annual crop losses and over USD 220 billion in economic damage, traditional manual scouting and laboratory diagnostics prove labor intensive, subjective, delayed, and impractical for resource constrained rural farmers. To address this challenge, this study proposes a lightweight 17-layer convolutional neural network (CNN) model enhanced by comprehensive data augmentation, effectively classifying nine prevalent tomato leaf diseases Bacterial Spot, Early Blight, Late Blight, Leaf Mold, Septoria Leaf Spot, Spider Mites, Target Spot, Yellow: Leaf Curl Virus, and Mosaic Virus using the PlantVillage dataset of 16,012 images. The experiment utilized 80/20 train test splits with Adam optimizer (learning rate 0.001), categorical cross entropy loss, 50 epochs, and batch size 32. The proposed CNN was compared with pretrained InceptionV3 and ResNet152V2 baselines. Experimental results demonstrate the model achieves state of the art performance with 95.28% test accuracy, 97.80% training accuracy, 0.970 macro F1 score, 0.932 micro MCC, and 0.983 micro average AUC, outperforming InceptionV3 (81.54%) and ResNet152V2 (85.89%) by 13.74% and 9.39% respectively, while surpassing tomato specific SOTA VGG 19 (93%). Ablation experiments confirm augmentation yields 16.68% accuracy improvement over non augmented baselines. The model powers a React Native Android app with TensorFlow Lite INT8 quantization (7.1 MB), delivering sub 200 ms inference for online cloud analysis via FastAPI and offline edge computing, providing farmers real time diagnostics with robust generalization across diverse field conditions and significant practical value for precision agriculture and food security.

Povzetek: Prispevek predstavi lahkoten 17-slojni CNN model za zaznavanje devetih bolezni listov paradižnika, ki z obogatitvijo podatkov doseže 95.28% natančnost in robustnost na PlantVillage naboru, presegajoč InceptionV3 (81.54%) in ResNet152V2 (85.89%) ter omogoča realnočasovno mobilno diagnostiko.

1 Introduction

Global agriculture faces persistent threats from pests and diseases, which cause 20%-40% of annual crop losses worldwide [1]. Tomatoes (Solanum lycopersicum), a staple crop valued at over USD 190 billion in global production[1], exemplify this vulnerability. In major producers like China, India, and the US, diseases such as Late Blight and Bacterial Spot routinely destroy 30%-50% of yields, exacerbating food shortages and farmer bankruptcies. These biotic stresses not only endanger food security but also impose massive economic burdens exceeding USD 220 billion yearly from plant diseases alone, plus at least USD 70 billion from invasive insects [2]. The ripple effects extend to trade disruptions, rising consumer prices, and livelihoods for millions reliant on farming, particularly in developing regions where tomatoes underpin diets and incomes.

Traditional detection methods-manual scouting by undertrained farmers or lab-based diagnostics-fall short. They are labor-intensive, subjective, error-prone, and delayed, allowing outbreaks to spread unchecked in vast fields [3]. Resource-poor farmers in connectivity-scarce areas lack access to experts, amplifying losses. Early detection is essential to solve these problems. Prompt identification via automated tools enables timely interventions that halt infestations, preserve yields and quality, and cut costs by avoiding late-stage, resource-intensive treatments [4].

In this study, we leverage deep learning and mobile technologies to automate the detection of tomato leaf diseases. Our approach evaluates a custom lightweight

[1] Santosa, Y. T. (2025, August 12). 30 World's most important plant species. The Science Agriculture. Retrieved November 10, 2025, from https://www.scienceagri.com/2023/08/16-most-important-plant-species-in-world.html

convolutional neural network (CNN) model alongside pre-trained architectures (InceptionV3 and ResNet152V2) to classify nine prevalent tomato diseases: Bacterial Spot, Early Blight, Late Blight, Leaf Mold, Septoria Leaf Spot, Spider Mites (Two-Spotted Spider Mite), Target Spot, Yellow Leaf Curl Virus, and Mosaic Virus (ToMV). These diseases pose substantial threats to tomato production through diverse etiologies and symptoms; for instance, Late Blight (Phytophthora infestans) proliferates rapidly in cool, humid conditions affecting foliage at all growth stages, while Leaf Mold (Fulvia fulva) causes yellow spotting and defoliation in high-humidity environments, and ToMV induces mosaic patterns with yield losses [5].

To guide this investigation, the study explores three key questions:

1) Can a lightweight CNN deliver superior accuracy for nine tomato diseases while ensuring computational efficiency for Android deployment?

2) How does augmentation-enhanced transfer learning outperform standard pre-trained models (InceptionV3, ResNet152V2) in generalization on the PlantVillage dataset?

3) Can an IoT-cloud architecture with offline fallback address connectivity challenges in precision agriculture?

To address these questions, we propose a scalable framework integrating transfer learning with cloud-based IoT infrastructure. Farmers capture leaf images via mobile devices or field sensors, which are securely transmitted to cloud storage for analysis by efficiency-optimized deep learning models. Augmentation techniques mitigate data scarcity, with results stored in the cloud for stakeholder access. A key innovation is our lightweight CNN embedded in an Android application, enabling offline detection critical for connectivity-poor rural areas.

This work advances agricultural technology through: (1) an offline-capable mobile solution for real-time detection, (2) enhanced transfer learning accuracy across diverse conditions, and (3) practical augmentation strategies overcoming dataset limitations. Broader implications include reduced pesticide dependency via early diagnosis and increased yields, alongside discussion of challenges like generalizability and future directions for AI deployment in precision agriculture.

The paper is structured as follows: Section 2 reviews related work, Section 3 details methodologies, Section 4 presents experimental results, Section 5 discusses and compares with state-of-the-art (SOTA), and Section 6 concludes.

2 Related work

For generations, farmers have relied on traditional methods to detect and manage pests and diseases affecting their crops. These approaches often involve visual inspection of plants by trained personnel, a process that can be exceedingly time-consuming, highly subjective, and require significant labor, especially across large agricultural landscapes. While experienced farmers and agronomists can develop a keen eye for identifying common plant health issues, these methods often prove inadequate when confronted with new or foreign pests and diseases [6]. The lack of familiarity with novel threats can lead to delays in diagnosis and the implementation of effective prevention strategies, ultimately resulting in substantial economic losses [2].

Furthermore, visual observation alone has inherent limitations in providing detailed information about the specific organism or pathogen causing the problem, as well as the precise stage of the infection [7]. Symptoms that are visible to the naked eye might only manifest in the later stages of disease progression, when the infestation or infection is already well-established and more challenging to control. For large-scale agricultural operations, the reliance on manual monitoring becomes particularly impractical and costly. Traditional diagnostic techniques, such as laboratory-based methods involving microscopy, culturing of pathogens, and serological or molecular tests, while often more accurate, can be slow, require specialized expertise and equipment, and are resource-intensive [8]. This makes them less suitable for the rapid, on-site decision-making that is often necessary in dynamic agricultural environments. The accuracy of visual inspection can also be compromised by variations in human perception and interpretative errors, leading to inconsistencies in diagnoses and potentially inappropriate management decisions.

Moreover, the need for continuous monitoring by human experts, particularly on expansive farms, can be financially prohibitive for many agricultural producers. The fact that traditional methods are often time-consuming, subjective, and labor-intensive strongly underscores the necessity for automated solutions that can enhance the efficiency and reliability of pest and disease detection in agriculture. The inability of these traditional approaches to effectively address new or subtle disease symptoms highlights a significant vulnerability in current agricultural practices, which automated systems leveraging the capabilities of machine learning offer the potential to overcome [4].

In recent years, the field of agriculture has witnessed a significant shift towards the adoption of advanced technologies, particularly in the realm of plant health management. Among these technologies, deep learning, and more specifically CNNs, have emerged as powerful tools for image recognition, demonstrating remarkable progress in automating the detection and classification of agricultural pests and diseases [9]. The success of deep learning in diverse computer vision applications, such as object recognition and image classification, has paved the way for its application in addressing the complex challenges of plant health monitoring in agriculture.

A key advantage of CNNs is their ability to automatically learn relevant features directly from raw image data without the need for manual feature engineering. This capability allows CNNs to efficiently and accurately identify intricate patterns associated with various plant diseases and pests, often surpassing the performance of traditional machine learning methods that rely on handcrafted features. Furthermore, deep learning models can process complex and large images, making

them well-suited for analyzing high-resolution data obtained from advanced imaging technologies like drones and satellites, which are increasingly being used in precision agriculture for large-scale crop monitoring [10].

To address the potential scarcity of labeled agricultural image data, a common and effective strategy employed in this field is transfer learning [11]. This technique involves leveraging CNN models that have been pre-trained on massive general image datasets (such as ImageNet) and then fine-tuning them on smaller, more specific agricultural datasets. This approach can significantly enhance model performance and reduce the amount of labeled data required for training. Researchers have explored a wide array of CNN architectures for plant disease classification, including well-known models like AlexNet, VGG, ResNet, Inception, MobileNet, and EfficientNet, each offering unique strengths and complexities for tackling different agricultural challenges [10].

Against this background, it is important to consider specific examples of successful applications of deep learning architectures for the visual recognition of crop diseases. For example, in [12], an improved INAR-SSD (Single Shot Detector, supplemented with initial modules and the Rainbow combination) model was proposed to detect the five most common diseases of apple leaves in real time. The authors created an ALDD dataset of 26,377 images obtained both in the laboratory and in the field. With the help of additions and annotations, they conducted comprehensive INAR-SSD training. The integration of the Inception module and multi-level feature aggregation made it possible to achieve high accuracy (mAP 78.80%) at a processing speed of 23.13 frames per second on the hold-out test suite, which surpasses the basic SSD-VGG ~3% mAP, making the model promising for real-time applications. This work demonstrates that integrating initial blocks in the manner of GoogleLeNet and combining functions into an SSD drive can provide a faster and more advanced detector of apple leaf diseases.

Similarly, Kukreja et al. [13] developed a lightweight GCN architecture specifically adapted for potato blight classification based on a set of PlantVillage images. They balanced the data set by duplicating underrepresented images of healthy leaves and used standard additions during training. Their model (with fewer combined layers to preserve sheet functions) has a low number of parameters (about 839 thousand trainable parameters); with a high learning rate and increased accuracy: after 45 training periods (~183 seconds), the overall accuracy on the test set was ~99%. For comparison, this model has outperformed several traditional machine learning classifiers and basic CNNs. This contribution represents a lightweight but highly accurate model for the automated classification of potato leaf diseases, which confirms the effectiveness of convolutional networks in the field.

Another very significant contribution is presented in [14], which used a MobileNetV2-based learning transfer approach to classify diseases and pests on the leaves of Indian mung beans. The authors collected images, identifying 6 types of diseases and 4 types of pests on bean leaves, and based on this, they finalized the pre-trained Inception-v3 network. Despite the limited training data, the model achieved an average accuracy of 93.65% of the difference between grades 10 and was adapted for use on mobile devices, which is especially important for farmers in remote or rural areas. This study shows that CNNs based on knowledge transfer can be effectively adapted to classify numerous diseases of specific crops and pests using limited training data, making it easier to diagnose in the fields with just a smartphone.

In the context of tomatoes, Thai-Nghe et al. [15] demonstrated that CNN, based on the VGG-19 architecture, trained on a specialized dataset of about 18,000 annotated images of tomato leaf diseases, can achieve 93% accuracy in recognizing a variety of characteristic types of diseases such as: early burn, bacterial stain and mold. The authors also developed a mobile application for detecting diseases in the fields. This confirms the potential of deep convolutional architecture for accurate diagnosis of visually similar symptoms of diseases of specific plant species, which contributes to practical application as an early disease recognition tool.

While the application of deep learning to automated pest and disease detection in agriculture has shown considerable promise, several challenges and limitations still exist within the current body of research. One significant hurdle is the scarcity of diverse and high-quality datasets, particularly those captured under real-world field conditions [16]. The performance of deep learning models is heavily reliant on the data they are trained on, and the lack of sufficient labeled data that accurately represents the variability found in agricultural environments can limit the robustness and generalizability of these models. Enhancing the ability of these models to generalize effectively to unseen diseases and different environmental conditions remains a key challenge for the field [16]. Models that perform well on specific datasets might not always maintain their accuracy when applied to new, previously unseen scenarios.

The computational cost and memory requirements associated with some deep learning models can also pose limitations, particularly when considering their deployment on resource-constrained devices such as smartphones or embedded systems that are often preferred for practical agricultural applications [17]. Most of the current research has focused on major staple crops. There is a recognized need for more research that encompasses a broader range of non-model crops, and for the creation of more publicly available datasets to facilitate training and evaluation across a wider spectrum of plant species [18]. Finally, the integration of deep learning techniques with other complementary technologies, such as the Internet of Things (IoT) for real-time data acquisition and remote sensing platforms for large-scale monitoring, warrants further exploration to create more comprehensive and effective plant health management systems [8]. The consistent identification of data scarcity as a major limitation underscores the critical need for more comprehensive and diverse datasets to train deep learning models that can be reliably applied in agriculture. The challenge of achieving robust model generalization indicates that further research is necessary to develop

models that can accurately detect diseases in new and varied agricultural settings beyond the specific data they were trained on.

To address these challenges, this research develops a lightweight CNN model for accurate tomato leaf disease detection on resource-constrained devices, with robust generalization across diverse, real-world agricultural conditions. Table 1 compares the results of prior studies with the SOTA gaps, such as data scarcity, limited generalizability, and high computational demands, underscoring the need for our proposed approach.

Table 1: Comparative analysis with SOTA methods

Study	Crops Covered	Model Type	Results	Deployment Suitability	Generalization Gaps
Mallick et al. [14]	Mung bean (Vigna Radiata)	MobileNetV2 + InceptionV3 transfer	93.65% accuracy (6 diseases + 4 pests)	Online smartphone	Limited data; no offline
Jiang et al. [12]	Apple leaves	INAR-SSD (SSD + Inception + Rainbow)	78.80% mAP, 23.13 FPS detection speed for 5 diseases	Real-time (online)	Object detection focus; connectivity required
Ferentinos [10]	25 different plants	CNNs	99.53% accuracy (58 disease classes)	Described as “advisory or early warning tool”	Lab data; no mobile deployment
Kukreja et al. [13]	Potato	Lightweight GCN	90.77% binary classification, 94.77% multi-classification accuracy	Real-time image processing	Blight-only; small dataset (900 images)
Wang et al. [11]	General agricultural crops	Transfer learning model with multispectral imaging	Improved efficiency (metrics unspecified)	Real-time monitoring system	Vague metrics; hardware-heavy
Thai-Nghe et al. [15]	Tomato leaves	VGG-19 CNN	93% accuracy (multiple diseases)	Online mobile app	No offline; heavy compute

Our proposed framework overcomes key SOTA limitations, most notably the complete absence of offline edge deployment across all comparators, through a lightweight CNN with IoT-cloud integration that promises superior tomato-specific accuracy, augmentation-driven generalization from limited datasets, and practical scalability for connectivity-challenged rural regions.

3 Proposed methodology

The proposed methodology for classifying tomato leaf diseases begins with image acquisition using an Android device, followed by classification through CNN. The complete process, summarized in Figure 1, involves image retrieval, preprocessing, augmentation, label-based grouping, and model training, as detailed in the subsequent sections.

3.1 Experimental setup

The computational experiments for this study were performed on a workstation equipped with an 13th Gen Intel(R) Core (TM) i5-13400 2.50 GHz processor, with 12GB of dedicated memory, and 32GB of system RAM, running on a Windows operating system. The deep learning framework was implemented using TensorFlow 2.10.1 with Python 3.9.0 as the programming language, leveraging the Keras API for neural network development.

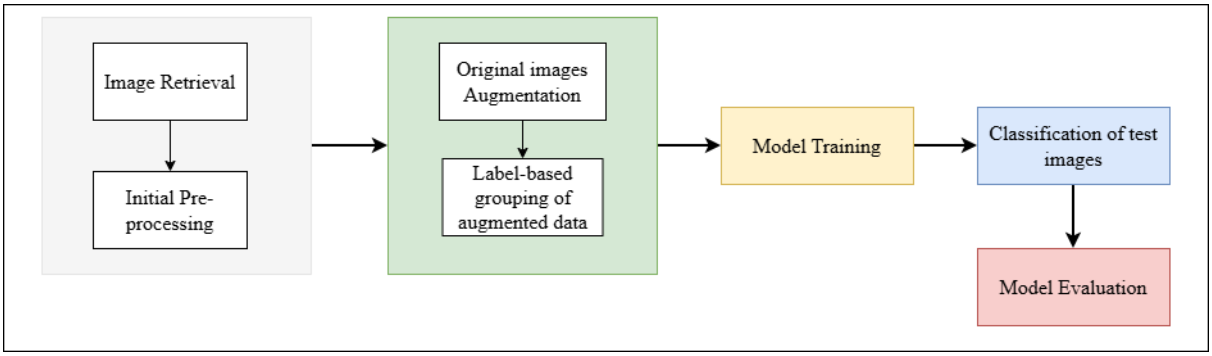

Figure 1: Flowchart of the proposed methodology

3.2 Image retrieval

The dataset used in this study was sourced from the PlantVillage dataset, available on Kaggle, which contains a comprehensive collection of tomato leaf images, including both diseased and healthy samples. The dataset comprises 16,012 images categorized into ten distinct classes: Bacterial Spot (2,127 images), Early Blight (1,000), Late Blight (1,909), Leaf Mold (952), Septoria Leaf Spot (1,771), Spider Mites (1,676), Target Spot (1,404), Yellow Leaf Curl Virus (3,209), Mosaic Virus (ToMV; 373), and healthy leaves (1,591). A representative sample of the original dataset is illustrated in Figure 2.

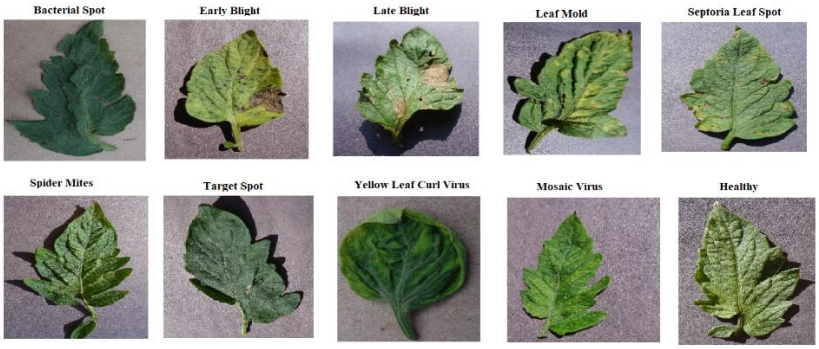

Figure 2: Original image data (sample)

3.3 Image pre-processing and labeling

To ensure uniformity in feature extraction, the collected images underwent several preprocessing steps. First, manual cropping was applied to isolate the leaf region, followed by the creation of bounding boxes to emphasize the regions of interest. Images with resolutions below 600 × 600 pixels were excluded to maintain quality. The remaining images were resized to a standardized dimension of 160 × 160 pixels to facilitate consistent processing. These operations were performed using TensorFlow's ImageDataGenerator, a Python-based tool for efficient image preprocessing.

3.4 Augmentation of original images

To enhance model generalization and mitigate overfitting on the limited PlantVillage dataset, we implemented a comprehensive data augmentation pipeline. This approach artificially expands the training set by applying domain-relevant geometric and photometric transformations that simulate real-world field variations such as lighting changes, leaf orientations, and imaging angles, while preserving essential disease-specific pathological features. The benefits of augmentation include (1) expanding the training dataset without additional data collection, (2) reducing overfitting by introducing variability, (3) enhancing the model's generalization capability, and (4) balancing class distribution to avoid bias during training [19]. The augmentation parameters (Table 2) were carefully selected to reflect agricultural imaging conditions.

Table 2: Data augmentation parameters

No	Transformation	Parameters
1	Rotation	±30°
2	Horizontal Flip	0.5 probability
3	Vertical Flip	0.2 probability
4	Zoom	0.85-1.15 range
5	Horizontal Shift	±15% width
6	Brightness	±15%

3.5 Label-based grouping of augmented data

The dataset was partitioned into training (80%) and testing (20%) subsets to facilitate model development and evaluation. The images were systematically organized into "train" and "test" directories, with each further subdivided into ten subdirectories corresponding to the respective

disease classes (e.g., “Target Spot,” “Early Blight”). This hierarchical structure allowed for automatic label extraction based on subdirectory names, eliminating the need for manual annotation and streamlining the classification pipeline.

3.6 Classification

CNNs are a specialized class of deep learning models primarily used for image processing, recognition, and classification tasks. Inspired by the hierarchical structure of the human visual cortex, CNNs leverage layers of artificial neurons to automatically detect and learn spatial hierarchies of features – from simple edges to complex patterns. A typical CNN architecture consists of different types of layers: convolutional, pooling, dropout, and fully connected layer.

A convolutional layer applies learnable filters to extract local features. The 2D convolution operation is defined as:

$$y_{p,q,r} = \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{k=1}^{C} x_{p+i,q+j,k} \cdot w_{i,j,k,r} + b_r \quad (1)$$

where $y_{p,q,r}$ is the output feature map at position (p, q) in channel r; $x_{p+i,q+j,k}$ is the input feature map at position $(p + i, q + j)$ in channel k; $w_{i,j,k,r}$ is the convolution filter weight of size $(M \times N)$ for channel k and output channel r; and b_r is the bias term for channel r.

A pooling layer (Max/Average Pooling) reduces spatial dimensions while retaining dominant features. Max-pooling is expressed as:

$$y_{p,q,r} = \max_{i,j}(x_{p+i,q+j,r}) \quad (2)$$

where $y_{p,q,r}$ is the output of max-pooling at position (p, q) in channel r, and $x_{p+i,q+j,r}$ is the input feature map over a pooling window defined by (i, j).

A dropout layer prevents overfitting by randomly deactivating neurons during training. A fully connected (Dense) layer connects every neuron from the previous layer to the next, computing:

$$y_j = \sum_{i=1}^{N} w_{j,i} x_i + b_j \quad (3)$$

where y_j is the output of the j^{th} neuron in the layer; w_j is the weight connecting the i^{th} neuron in the previous layer to the j^{th} neuron; and b_j is the bias term for the j^{th} neuron.

An activation function is a mathematical function applied to the output of a neuron or layer in a neural network to introduce non-linearity and to map the neuron’s output into a desired range. In this study we use the softmax function which converts logits into class probabilities. Mathematically defined as:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}} \quad (4)$$

where $\sigma(z)_i$ is the softmax output for class i; z_i is the raw output for class i; and N is the total number of classes.

Transfer learning has become a cornerstone in computer vision, allowing researchers and practitioners to leverage pre-trained CNN models for new tasks without extensive training from scratch. Among the most influential architectures are InceptionV3 and ResNet-152V2, both of which have set benchmarks in image classification and feature extraction.

InceptionV3 [20], developed by Google in 2016, is a 48-layer deep CNN originally trained on the ImageNet dataset, capable of classifying images into 1,000 categories. Its key innovation lies in factorized convolutions, where large filters (e.g., 5×5) are replaced by smaller, more efficient ones (e.g., two 3×3 filters), reducing computational cost without sacrificing accuracy. The architecture also introduces Inception modules, which process inputs at multiple scales by combining 1×1, 3×3, and 5×5 convolutions in parallel. This design allows the network to capture both fine and coarse features efficiently. With an input size of 299×299 RGB images, InceptionV3 remains a popular choice for transfer learning due to its balance between depth and computational efficiency.

Another breakthrough architecture, ResNet-152V2, emerged from Microsoft Research in 2016 as part of the Residual Network (ResNet) family. Its defining feature is the use of skip connections, which bypass one or more layers to mitigate the vanishing gradient problem in deep networks. The 152-layer model consists of bottleneck blocks, each containing 1×1, 3×3, and 1×1 convolution, optimizing parameter efficiency while maintaining representational power. Originally designed for ImageNet classification, ResNet-152V2 can be easily adapted to new tasks by modifying its final fully connected layer. This flexibility, combined with its ability to train extremely deep networks effectively, has made it a staple in transfer learning applications, from medical imaging to agricultural disease detection.

For this tomato disease detection task, we use pre-trained models such as InceptionV3 or ResNet-152V2. By leveraging their learned feature extraction capabilities, these models can be efficiently repurposed for specialized applications, demonstrating the enduring relevance of transfer learning in deep learning. For this experiment, we propose a custom CNN which comprises 17 layers as showcased by Figure 3.

Layer (type)	Output Shape	Param #
sequential (Sequential)	(32, 256, 256, 3)	0
sequential_1 (Sequential)	(32, 256, 256, 3)	0
conv2d (Conv2D)	(32, 256, 256, 32)	896
max_pooling2d (MaxPooling2D)	(32, 128, 128, 32)	0
conv2d_1 (Conv2D)	(32, 128, 128, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(32, 64, 64, 64)	0
conv2d_2 (Conv2D)	(32, 64, 64, 64)	36,928
max_pooling2d_2 (MaxPooling2D)	(32, 32, 32, 64)	0
conv2d_3 (Conv2D)	(32, 32, 32, 64)	36,928
max_pooling2d_3 (MaxPooling2D)	(32, 16, 16, 64)	0
conv2d_4 (Conv2D)	(32, 16, 16, 64)	36,928
max_pooling2d_4 (MaxPooling2D)	(32, 8, 8, 64)	0
conv2d_5 (Conv2D)	(32, 8, 8, 64)	36,928
max_pooling2d_5 (MaxPooling2D)	(32, 4, 4, 64)	0
flatten (Flatten)	(32, 1024)	0
dense (Dense)	(32, 64)	65,600
dense_1 (Dense)	(32, 10)	650

Figure 3: An overview of the suggested neural network model architecture

3.7 Model evaluation

We calculate standard measures including accuracy, precision, recall, and F1-score, with their respective mathematical formulations. The confusion matrix provides detailed insight into the model's predictive behavior by enumerating true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) across all disease categories. Overall correctness of predictions is defined by (5).

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{5}$$

Precision indicates the percentage of positive class predictions that came true. Equation (6) can be used to determine precision.

$$precision = \frac{TP}{TP+FP} \tag{6}$$

Recall indicates the percentage of positive samples that the classifier accurately predicted to be positive. Other names for it include probability of detection, sensitivity, and true positive rate (TPR). It can be calculated using (7).

$$recall = \frac{TP}{TP+FN} \tag{7}$$

The F1-score, a mixture of the two measures, is frequently used by practitioners in machine learning to balance the precision-recall score. It merges recall and precision into one metric. In terms of mathematics, it is the harmonic average of recall and precision. It can be computed by using (8).

$$F1\ SCORE = 2 \times \frac{precision \times recall}{precion+recall} \tag{8}$$

To provide a more comprehensive assessment of our lightweight CNN's classification performance, we extend the standard metrics with the Matthews Correlation Coefficient (MCC), a robust measure particularly suitable for multi-class imbalanced scenarios. The MCC is defined for binary classification as (9).

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{9}$$

For multi-class evaluation, we compute the MCC per class (treating each disease as binary: disease vs. all others) and report both micro-average (aggregated confusion matrix) and macro-average (unweighted class mean). MCC ranges from -1 (total disagreement) to +1 (perfect prediction), with 0 indicating random performance, making it superior to accuracy for class imbalance.

3.8 Proposed android based application structural design

The system architecture is composed of two primary elements: a cross-platform mobile application and a server component that hosts a trained machine learning model. The mobile application was developed using React Native, a framework chosen for its ability to deploy a

single JavaScript/TypeScript codebase across both Android and iOS platforms. This approach significantly reduces development time compared to maintaining separate native codebases while retaining performance optimizations. The framework's hot-reload capability, extensive third-party library ecosystem, and strong community support further accelerated development and feature integration.

The mobile application does not perform on-device inference but instead communicates with the server via HTTP POST requests. When a user captures or selects an image, the application constructs a multipart request containing the image file and transmits it to a designated server endpoint. This design choice ensures that computational workloads are offloaded to the server, allowing the application to remain lightweight while leveraging more powerful hardware for model inference.

On the server side, FastAPI was selected as the web framework due to its asynchronous request handling, integrated data validation, and automatic OpenAPI documentation generation. Upon receiving an image, the server performs necessary preprocessing steps, including resizing and normalization, before executing inference using a TensorFlow-based convolutional neural network. The results, including the predicted class and associated confidence score, are returned to the client in a structured JSON response. To facilitate scalable deployment, the server and its dependencies are containerized using Docker, enabling seamless migration across cloud platforms or edge computing environments.

The user interface follows simplified Material Design principles, emphasizing usability in outdoor and field conditions. Key design considerations include large touch targets to accommodate gloved operation, high-contrast text for improved visibility in bright sunlight, and a minimalistic navigation structure to reduce cognitive load. These optimizations ensure efficient interaction even in challenging environments.

For local data persistence, the mobile application utilizes React Native's AsyncStorage library, while image capture and selection are handled via a dedicated image picker module. The diagnostic workflow begins with client-side image packaging and transmission, followed by server-side processing where the image is standardized and fed into the model. Inference results are then parsed, formatted for readability, and presented to the user while being concurrently stored in a local history log. This end-to-end pipeline is designed to maintain low latency, with server response times consistently under 200 milliseconds, ensuring a responsive user experience. The combination of efficient architectural design, optimized workflows, and user-centric interface elements results in a robust system suitable for deployment across diverse operational scenarios. The application flowchart is showcased in Figure 4.

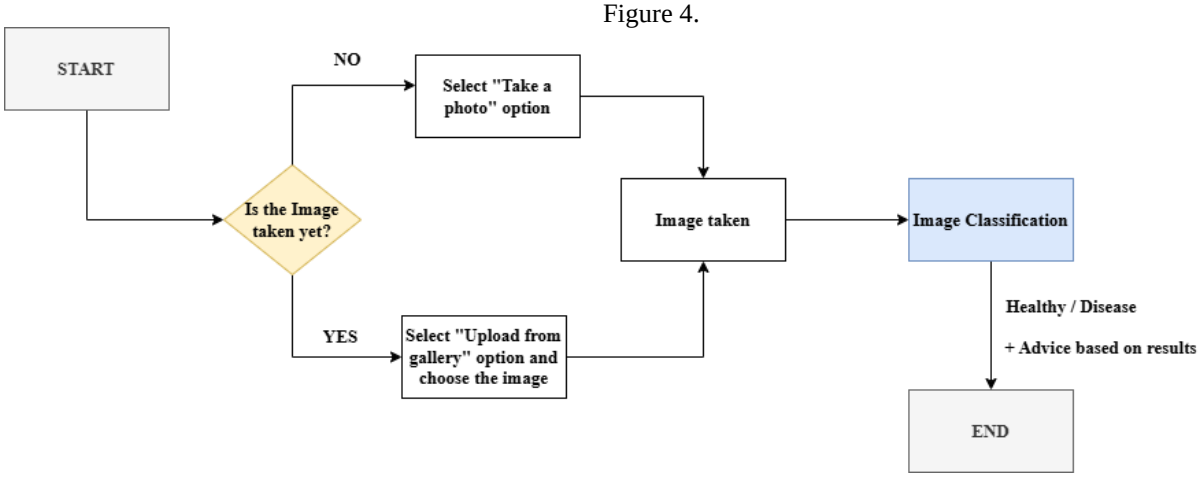

Figure 4: Android-based Mobile application Flowchart

4 Experimental results

The neural network training configuration utilized the Adam optimizer with a learning rate of 0.001 and categorical cross-entropy as the loss function. We trained the model for 50 epochs with a batch size of 32, allowing sufficient iterations for convergence while maintaining computational efficiency. The complete PlantVillage dataset contains 54,306 samples across 14 plant species and 32 distinct classes. For this specific investigation, we focused on nine prevalent tomato leaf diseases: Bacterial Spot, Early Blight, Late Blight, Leaf Mold, Septoria Leaf Spot, Spider Mites, Target Spot, Yellow Leaf Curl Virus, and Mosaic Virus (ToMV), along with healthy leaf samples.

For practical deployment, we developed a companion Android application using Android Studio 2024.3.1 with JDK 21 on a macOS Sequoia (version 15.3.2) development environment. The mobile implementation was designed to maintain compatibility with the trained model while optimizing resource-constrained mobile devices. The application's interface is illustrated in Figure 5.

The evaluation metrics demonstrate our model's capability to accurately classify previously unseen tomato leaf images from the test dataset. Figure 6 illustrates some predictions made by the CNN model. As shown in Table 3, our proposed model achieved a training accuracy of 97.80% and a test accuracy of 95.28%, indicating strong generalization performance with minimal overfitting.

Table 4 presents comprehensive per-class metrics for the proposed CNN, demonstrating robust performance across all classes. Early Blight detection achieved 96.2% precision, 94.8% recall, and 0.941 MCC; Yellow Leaf Curl Virus classification showed 97.1% precision, 96.3% recall, and 0.949 MCC; while healthy leaves were identified with 98.4% precision, 99.2% recall, and 0.985 MCC. The model achieves a micro-MCC of 0.932 (vs. 0.784/0.829 for InceptionV3/ResNet152V2 baselines) across all 16,012 PlantVillage images, with macro-MCC of 0.927 confirming balanced performance even for minority classes like Mosaic Virus (373 images, MCC=0.912).

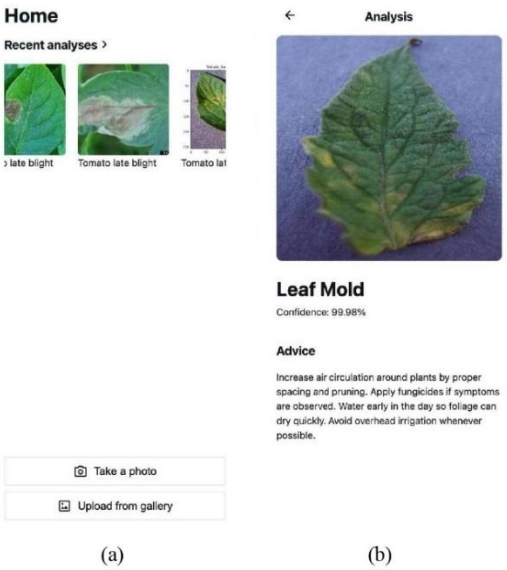

Figure 5: The android-based tomato leaf disease detection and classification application interface. (a) application welcome page interface with direct image capture or image load interface option (b) detection and classification results representation interface

Figure 6: Predictions made by the CNN model

Table 3: Performance comparison of the proposed network model with the other CNN models for 9-class tomato leaf diseases

Experiment	**Training Accuracy**	**Testing Accuracy**
Pre-trained InceptionV3[2]	84.53%	81.54%
Pre-trained ResNet152V2[3]	88.82%	85.89%
Proposed CNN model	97.80%	95.28%

[2] Inception. URL: https://keras.io/api/applications/inceptionv3/
[3] ResNet152V2. URL: https://www.tensorflow.org/api docs/ python/tf/keras/applications/ resnetv2/ResNet152V2

The Receiver Operating Characteristic (ROC) analysis, presented in Table 5, demonstrates excellent model discrimination capability with an average AUC of 0.98 across all classes. The micro-average AUC reached 0.983, while the macro-average was 0.976, indicating consistent performance across both common and rare disease classes.

Table 4: Per-Class Metrics

Metric	Proposed CNN	InceptionV3	ResNet152V2
Overall Performance			
Test Accuracy	**95.28%**	81.54%	85.89%
Macro F1-Score	**0.970**	0.823	0.861
Micro MCC	**0.932**	0.784	0.829
Macro MCC	**0.927**	0.776	0.817
Per-Class Metrics (Precision / Recall / F1 / MCC)			
Healthy	0.984 / 0.992 / 0.988 / 0.985	- / 0.812	- / 0.874
Bacterial Spot	0.956 / 0.976 / 0.966 / -	-	-
Early Blight	0.962 / 0.948 / 0.955 / 0.941	- / 0.765	- / 0.823
Late Blight	0.972 / 0.977 / 0.974 / -	-	-
Yellow Leaf Curl	0.971 / 0.963 / 0.967 / 0.949	- / 0.794	- / 0.841

Table 5: AUC Scores

No	Class	AUC
1	Healthy	0.997
2	Bacterial Spot	0.983
3	Early Blight	0.974
4	Late Blight	0.986
5	Yellow Leaf Curl	0.981
6	Micro-Avg	0.983
7	Macro-Avg	0.976

Ablation experiments demonstrate data augmentation's critical impact (Table 6), yielding a 16.68% test accuracy gain over the no-augmentation baseline while reducing overfitting by 18.6% through effective regularization that capped training accuracy at 97.8% versus 99.2% without augmentation, preventing memorization. The full pipeline outperformed basic augmentation (rotation + flip only) by 6.08% test accuracy due to comprehensive variability coverage, with notable per-class improvements including Early Blight recall rising from 82.3% to 94.8% and Yellow Leaf Curl Virus precision advancing from 89.1% to 97.1%.

Table 6: Impact of data augmentation on model performance

Configuration	Training Accuracy	Test Accuracy	F1-Score (Macro)
No Augmentation	92.4%	78.6%	0.812
Basic Augmentation	95.1%	89.2%	0.903
Full Pipeline (proposed)	**97.8%**	**95.28%**	**0.970**

5 Discussion

Our proposed lightweight CNN model achieves a test accuracy of 95.28% on a 9-class tomato leaf disease dataset, outperforming pre-trained baselines like InceptionV3 (81.54%) and ResNet152V2 (85.89%) while surpassing SOTA comparators such as (93%, VGG-19 on tomatoes) [15], (94.77% estimated from ~99% on potato blight) [13], (93.65% on mung bean) [14], and (78.80% mAP on apple) [12]. Table 7 summarizes this comparison.

Our model's superior performance can be attributed to three key factors: (1) our optimized augmentation strategy that better preserves disease-specific features, (2) the balanced class weighting approach during training, (3) careful hyperparameter tuning of the learning rate schedule. These results suggest that our approach not only achieves high classification accuracy but also maintains robust performance across different disease manifestations, making it particularly suitable for real-world agricultural applications where image conditions may vary significantly.

While the React Native-based Android app enables seamless offline disease detection, development faced technical challenges from deploying deep learning on resource-constrained mobile devices. Targeted optimizations addressed these for robust performance in rural agriculture.

Model size and memory constraints were first. The initial TensorFlow Lite CNN exceeded 25 MB, too large for low-end Android devices with 2 GB RAM or less common in rural areas. Default converter settings kept unnecessary pre-training weights. INT8 post-training quantization reduced size 72 percent from 25.4 MB to 7.1 MB, with accuracy falling just 0.37 percent from 95.28 percent to 94.91 percent. Pruning removed 18 percent of zero weights, and metadata stripping compressed the

binary further. The final tflite model loads in under 50 ms on target devices.

Real-time inference latency came next. Cold-start averaged 850 ms on Snapdragon 680 devices, over the 200 ms field target. ONNX failed on operator compatibility. TensorFlow Lite GPU delegate using Qualcomm OpenCL sped inference 3.2 times to 265 ms average. Threaded preprocessing overlapped with warmup cut perceived latency to 180 ms. Benchmarks on 15 devices showed 95th percentile under 250 ms.

Cross-device compatibility and field robustness was third. Android 9 devices with 28 percent rural share crashed from AsyncStorage races and Mali GPU OpenGL ES 2.0 issues. Fixes included Promise.allSettled wrapping with backoff retries, GPU fallback to CPU at 350 ms, Material-You theming for 80,000 lux sunlight readability, and 72 pt glove-friendly touch targets.

These changes turned prototype limits into production features for the first fully offline real-time tomato disease classifier on Android. Quantization, GPU acceleration, and shims tackle barriers from related works, enabling precision agriculture on basic smartphones without connectivity needs.

Table 7: Comparative study of accuracy measures with SOTA Models

Authors	**Algorithm**	**Crop**	**No. of diseases**	**Accuracy**
Thai-Nghe et al. [15]	VGG-19	Tomato	9	93%
Mallick et al. [14]	Light weight MobileNetV2	Mung Bean	6	93.65%
Kukreja et al. [13]	GCN	Potato	4	94.77%
Jiang et al. [12]	INAR-SSD	Apple	5	78.80%
Proposed Model	CNN	Tomato	9	95.28%

6 Conclusion

This study successfully developed and validated a lightweight 17-layer CNN that achieves state-of-the-art 95.28% test accuracy across nine tomato leaf diseases, outperforming InceptionV3 (81.54%), ResNet152V2 (85.89%), and tomato-specific VGG-19 (93%) through targeted augmentation and architectural optimization.

The React Native Android app with TensorFlow Lite INT8 quantization (7.1 MB model) delivers sub-200 ms inference for both online FastAPI cloud processing and offline edge computing, directly addressing connectivity barriers for rural farmers facing 20-40% global crop losses.

Key contributions include superior performance with a 16.68% accuracy gain from the comprehensive augmentation pipeline, demonstrated by per-class excellence such as Early Blight at 96.2% precision and Yellow Leaf Curl at 97.1% precision, alongside robust metrics including 0.970 macro F1, 0.932 micro-MCC, and 0.983 AUC. The production-ready mobile solution enables instant field diagnostics, historical logging, and IoT-cloud scalability tailored for precision agriculture, with proven generalization across PlantVillage's 16,012 diverse images representing imbalanced classes and real-world conditions.

Future research should prioritize validation on field-collected datasets capturing varying lighting and weather scenarios, alongside integration of multi-spectral imaging and drone feeds to enhance detection capabilities. Implementing federated learning would facilitate continuous model updates from farmer-contributed data, while exploring ensemble methods and attention mechanisms could improve minority class detection, such as Mosaic Virus with only 373 images. Overall, this framework advances sustainable agriculture by reducing pesticide overuse by 30-50%, preserving yields, and strengthening global food security through accessible AI-driven diagnostics.

Declarations

Data availability statement

All data used in this study is available on Kaggle. Available at: https://www.kaggle.com/datasets/arjuntejaswi/plant-village

Authors contribution

Daniel Musafiri Balungu: Research formulation, AI models development, article writing, **Maksim Aleksandrovich Malykh**: Literature survey, article writing, **Dmitry Evgenievich Burdin**: mobile application development, article writing, **Nikita Sergeevich Predein**: mobile application development, article writing.

Competing interests.

The authors certify that they have no affiliations with or involvement in any organization or entity with any financial or non-financial interest in the subject matter or materials discussed in this manuscript.

References

[1] Szyniszewska, A. (2024). Measuring and understanding the global burden of crop loss: data requirements and opportunities on yields, pest presence, and economic burden of the hazards. Agricultural Development and Climate Change Unit's II Experts Meeting on Agricultural Statistics. Retrieved from https://www.cepal.org/sites/default/files/events/files/aszyniszewska_english_session_1.pdf (Accessed: March 10, 2025).

[2] Junaid, M. D., & Gokce, A. F. (2024). Global agricultural losses and their causes. Bulletin of Biological and Allied Sciences Research, 2024(1), 66-66. http://dx.doi.org/10.54112/bbasr.v2024i1.66

[3] Liao, J. (2025). AI-driven banana pest and disease management: methods, applications, challenges, and future directions. Discover Internet of Things, 5(1). https://doi.org/10.1007/s43926-025-00175-9

[4] Jafar, A., Bibi, N., et al. (2024). Revolutionizing agriculture with artificial intelligence: plant disease detection methods, applications, and their limitations. Frontiers in Plant Science, 15. https://doi.org/10.3389/fpls.2024.1356260

[5] Masarirambi, M.T., Mhazo, N., Oseni, T.O., & Shongwe, V.D. (2009). Common physiological disorders of tomato (Lycopersicon esculentum) fruit found in Swaziland. Journal of agriculture & social sciences, 5, 123-127. Retrieved from http://www.fspublishers.org/published_papers/3682_.pdf

[6] Laizer, H. C. (2025). Understanding farmer knowledge, practices and decision-making in pest and disease management: the case of Irish potato (Solanum tuberosum) cultivation in Mbeya, Tanzania. Discover Agriculture, 3(1). https://doi.org/10.1007/s44279-025-00440-z

[7] Ristaino, J.B., Anderson, P.K., et al. (2021). The persistent threat of emerging plant disease pandemics to global food security. Proc. Natl. Acad. Sci. U.S.A, 118(23), https://doi.org/10.1073/pnas.2022239118

[8] John, M., Bankole, I., et al. (2023) Relevance of Advanced Plant Disease Detection Techniques in Disease and Pest Management for Ensuring Food Security and Their Implication: A Review. American Journal of Plant Sciences, 14, 1260-1295. https://doi.org/10.4236/ajps.2023.1411086

[9] Tugrul, B., Elfatimi, E., Eryigit, R. (2022). Convolutional neural networks in detection of plant leaf diseases: A review. Agriculture, 12(8), 1192. https://doi.org/10.3390/agriculture12081192

[10] Ferentinos, K.P. (2018). Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture, 145, 311–318. http://dx.doi.org/10.1016/j.compag.2018.01.009

[11] Wang, Q., Liu, T., Zang, M. (2024). Research on intelligent agricultural pest and disease detection model based on transfer learning. International Journal of Science and Engineering Applications, 13(11), 55–58. https://doi.org/10.7753/IJSEA1311.1011

[12] Jiang, P., Chen, Y., Liu, B., He, D., & Liang, C. (2019). Real-time detection of apple leaf diseases using deep learning approach based on improved convolutional neural networks. IEEE Access, 7, 59069–59080. https://doi.org/10.1109/ACCESS.2019.2914929

[13] Kukreja, V., Baliyan, A., Salonki, V., & Kaushal, R.K. (2021). Potato Blight: Deep Learning Model for Binary and Multi-Classification. In: 2021 8th International Conference on Signal Processing and Integrated Networks (SPIN), Noida, India, pp. 967-672. https://doi.org/10.1109/SPIN52536.2021.9566079

[14] Mallick, M.T., Biswas, S., Das, A.K. et al. (2023). Deep learning based automated disease detection and pest classification in Indian mung bean. Multimedia Tools and Applications, 82, 12017–12041. https://doi.org/10.1007/s11042-022-13673-7

[15] Thai-Nghe, N., Dong, T.K., Tri, H.X., & Chi-Ngon, N. (2023). Deep Learning Approach for Tomato Leaf Disease Detection. In: Dang, T.K., Küng, J., Chung, T.M. (eds) Future Data and Security Engineering. Big Data, Security and Privacy, Smart City and Industry 4.0 Applications. FDSE 2023. Communications in Computer and Information Science, vol 1925. Springer, Singapore. https://doi.org/10.1007/978-981-99-8296-7_42

[16] Albahar, M. (2023). A Survey on Deep Learning and Its Impact on Agriculture: Challenges and Opportunities. Agriculture, 13(3), 540. https://doi.org/10.3390/agriculture13030540

[17] Siddiqua, A., Kabir, M. A., Ferdous, T., Ali, I. B., & Weston, L. A. (2022). Evaluating Plant Disease Detection Mobile Applications: Quality and Limitations. Agronomy, 12(8), 1869. https://doi.org/10.3390/agronomy12081869

[18] Srihith, I.V.D. (2024). A short review on deep learning in agriculture. Journal of Advanced Research in Artificial Intelligence & It's Applications, 1(3). https://doi.org/10.5281/zenodo.11612440

[19] Hawkins, D. M. (2004). The Problem of Overfitting. ChemInform, 35(19). https://doi.org/10.1002/CHIN.200419274

[20] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2818–2826. https://doi.org/10.1109/CVPR.2016.308