IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 31, No. 1, January, 2026, pp. 243-251
Journal homepage: http://iieta.org/journals/isi

Knowledge Distillation–Driven Continual Learning for Lightweight Plant Disease Detection on Edge Devices

Shruthi K. S.*, Naveen B., Naveen K. B.

Department of Electronics and Communication, BGS Institute of Technology, Adichunchanagiri University, B.G. Nagara 571448, India

Corresponding Author Email: shruthiks391@gmail.com

https://doi.org/10.18280/isi.310123

Received: 9 November 2025
Revised: 28 December 2025
Accepted: 17 January 2026
Available online: 31 January 2026

Keywords:
plant disease detection, continual learning, knowledge distillation, edge artificial intelligence, lightweight deep learning, smart agriculture, edge computing

ABSTRACT

Accurate plant disease detection is essential for improving crop productivity and ensuring global food security. Although deep learning models have demonstrated high performance in plant disease classification, deploying such models on Internet of Things (IoT) and edge devices remains challenging due to limited computational resources, memory constraints, and the inability of conventional models to adapt to newly emerging disease categories. This study proposes PlantKD-CLNet, a lightweight continual learning framework that integrates knowledge distillation with replay-based incremental learning for plant disease detection on edge devices. The proposed architecture employs a MobileNetV2 teacher network to transfer discriminative knowledge to a compact student model while incorporating a replay buffer mechanism to mitigate catastrophic forgetting during incremental learning. To enhance model robustness under real-world agricultural conditions, progressive data augmentation strategies are introduced to simulate variations in lighting, occlusion, and leaf texture. Experiments were conducted on the Tomato_10 dataset using a multi-phase incremental learning protocol. The results demonstrate that the proposed framework achieves an overall classification accuracy of 97.0% while significantly reducing model size from approximately 60 MB to 3.7 MB and decreasing inference latency from 12.3 ms to 4.3 ms. Moreover, the model maintains stable performance across incremental learning stages, effectively preserving previously acquired knowledge. These findings indicate that integrating knowledge distillation with continual learning provides an efficient and scalable solution for real-time plant disease monitoring in edge-based smart agriculture systems.

1. INTRODUCTION

Agriculture continues to be a key source of income in many developing areas, accounting for over 25% of countries' gross domestic product and over 60% of the rural workforce [1]. Much of this production comes from smallholder farmers, who generally have access to limited amounts of land, equipment, and capital. Consequently, smallholder farmers are particularly vulnerable to pests and diseases affecting their crops because these phenomena can wipe out a significant portion of months of toil and dramatically lower their earnings [2]. Globally, approximately 30% of annual crop yields are lost to plant diseases [3] threatening both food security and agricultural sustainability.

Traditional methods of diagnosing crop diseases i.e., visual identification and lab testing are often time-consuming, labour-intensive, and prohibitively expensive for many smallholder farmers [4]. Thus, they lack the knowledge necessary to implement these methods effectively [5]. Therefore, automated image-based methods of detecting and diagnosing crop diseases have received increased interest over the last couple years [6]. Advances made in utilizing Deep Learning techniques for recognizing crop diseases via image analysis e.g.; MobileNet, EfficientNet, ResNet have exhibited a high accuracy when evaluated against established datasets, such as the PlantVillage dataset [7-9].

These classification models are typically trained under laboratory conditions and implemented into real-world agricultural practices. There are significant variations can occur with respect to several environmental characteristics such as light intensity and angle at which a plant leaf is oriented with respect to the light source. The image quality enhancement is much required to predict the health of the plant using image data [10]. Due to these differing environmental attributes, models are at risk of reduced classification capacity when transitioned from training to real farming environments.

Besides the unpredictable variation in the external conditions they're exposed to, limitations in available resources also present several practical limitations on the performance of state-of-the-art convolutional neural networks on edge devices such as mobile phones, aircrafts, networked computers like a Single Board Computer (SBC), and low-powered Internet of Things (IoT) connected devices that dominate the twenty-first century in precision agriculture [11].

Furthermore, most existing Lightweight Models assume there will be no modifications or additions to their original

datasets; therefore, in instances when new species of pests occur or have been introduced to areas not previously known for infestations, smaller versions of large neural networks will be unable to support constant monitoring of the newly introduced species as it will require complete retraining of the model. In practical situations, the issues of catastrophic forgetting occur when model parameters are overwritten with data trained from a completely different class of organisms [12].

Currently, KD yields significant reductions in size for large models while providing an effective means of transferring structured knowledge of a teacher to a student model, KD implementation is performed only on datasets containing images from previously trained model classes. Adoption of KD has limited potential for retuning of the student model with new classes not contained in the training dataset.

Like KD techniques used in Continuous Learning approaches, CL techniques do not currently exist that are able to accommodate re-utilising large amounts of memory to perform the necessary training and updates for continued development of models being used in edge-based applications in the field of agriculture. In response to these shortcomings, we have proposed a unified deep learning framework called PlantKD-CLNet that leverages a knowledge distillation method coupled with a replay-buffer-based continual learning approach to provide support for evolving plant disease distribution while maintaining high levels of accuracy on low power, limited resources devices.

The model comprises a MobileNetV2 pre-trained network which serves as the teacher for the smaller student MobileNetV2, which is derived from one of its layers. To avoid catastrophic forgetting when performing incremental learning, representative samples that would otherwise not be retained will be saved in a small replay buffer to help prevent catastrophic forgetfulness through incremental learning method. The result of this combination of technologies allows the model to learn new disease categories while preserving knowledge gained from prior learning experiences.

The key contribution of this study is summarized as follows:

- Developed an integrated architecture for using knowledge distillation and replay-based continual learning to enable our models to learn about new plant diseases whilst retaining previously learned information.
- Utilises both soft-logit distillation and feature-map distillation and reduces the model size to facilitate real time inference on edge devices.
- Developed an adaptive memory replay buffer that uses a combination of stability-plasticity loss balancing to avoid catastrophic forgetting during multiple-instance learning.
- The proposed model results demonstrate substantial improvements over the knowledge distillation, continual learning and typical CNN/Transformer baselines, and our models outperform those state of the art on accuracy, robustness and incremental learning performance.

The remaining sections of the paper are organized as follows: In Section II, we describe previous work related to this subject. Section III provides a description of our methods and results. Section IV will present experimental findings and results of our analyses. and Section V summarises our results and provide recommendations for future research.

2. LITERATURE REVIEW

Recent developments in deep learning technology are having a positive impact on plant disease detection. But because nearly all current models employ very large architectures and utilize static training during the development and deployment phases, they are not well-suited for use in actual farm operations. The current review has examined past few years literatures and discusses both the progress in the area and the critical areas of improvement in terms of model effectiveness, adaptability, and continuous learning. The systematic review of plant disease detection using machine learning and deep learning was by Jackulin and Murugavalli [13]. In their comprehensive review, the authors overview a myriad of CNN architectures, data preprocessing techniques, and feature engineering methods. They also discuss how deep models outperform classical ML. The authors have also reviewed prevalent challenges, such as dataset imbalance, poor-quality field images, as well as the development of a generalizable model. However, the authors do not mention lightweight model deployment or continual learning methods needed to adapt to new plant disease classes over time, which limits their research's usability for edge-based agricultural systems in real time.

A review of deep learning-based plant disease detection methods was conducted by Pacal et al. [14]. Over 160 relevant studies were included in their systematic review, which examined temporal and spatial changes in plant disease detection model architecture; the number of datasets available for plant disease detection; augmentation techniques used; and performance benchmarking against other models. The authors have also noted critical research gaps: insufficient robustness to field settings, lack of a systematic framework for cross-domain evaluations, and a lack of lightweight architectures capable of real deployments of plant disease detection technology. Although the review provides adequate background information on constraints to model performance, it does not offer any frameworks for model compression, distillation or incremental learning that would allow for the adaptation of models to mobile or IoT-enabled devices.

Zhang et al. [15] have developed a lightweight model for identifying pests and diseases on plants using tensor feature combination through knowledge distillation. By using knowledge distillation, their method preserves performance during the training phases and allows a reduction in a large model to a smaller student model with minimal loss of accuracy. Their method also provides increased efficiency in inference on devices with limited processing capability. However, the model only addresses static classification and does not incorporate continual learning and replay techniques, restricting its application to only those scenarios where current disease types will change over time.

Wang [16] has developed a zero-exemplar deep continual learning approach to recognizing crop diseases using Vision Transformers (ViTs) while incorporating total variation attention regularization. Their method allows for the least amount of forgetting when adding new tasks and, as such, will not require the storage of exemplars. Their work offers theoretical contributions to the field of continual learning, yet the significant computational overhead associated with transformer models makes deployment on low-power edge devices very challenging. Furthermore, the lack of model compression limits the applicability of their work to real-time inference within fields.

Yang et al. [17] developed a lightweight method of detecting diseases in canopies based on direct observations and indirect feedback through the application of distillation techniques combined with Dual Scale Weighted Convolution (DSWC). The authors achieved very good accuracy as well as significant reductions in the size of deep learning models. This dual-scale design also allowed for an improvement in the detection of lesions from images of different qualities and complexities; thus, making it possible for their model to perform effectively against more complex leaf textures. The authors did however only evaluate the performance of their model using fixed data sets and static training phases. Therefore, the effect of continued learning on a distilled model in an incremental learning scenario is unknown, nor is it known how the distilled model will respond to the introduction of new diseases.

Duhan et al. [18] have produced a lightweight mobile network (RTR_Lite_MobileNetV2) that is suitable for plant disease detection in mobile devices that are highly resource-constrained. Authors reduced the number of model parameters and associated computational cost to provide an efficient inference capability. Ashurov et al. [19] have improved the performance of depth-wise CNN architectures by adding shortcut residual connections and squeeze-and-excitation block modules. The authors specifically demonstrated that their modified architecture produced improved feature representation performance, especially in low lit, noisy or occluded images.

Upadhyay et al. [20] provided a review of plant disease detection via computer vision and deep learning models. They identified major areas of interest for current researchers, including trends within the convolutional, transformer-based, and hybrid architectures. Huang et al. [21] have demonstrated that it is possible to use knowledge distillation as a method to create lightweight plant disease detection models. It resulted in a drastically reduced model size, while maintaining a relatively high level of accuracy. The resulting student model accurately simulates the teacher and therefore will be suitable for deployment onto embedded devices.

The multi-dataset, multi-domain deep learning model for plant leaf disease detection proposed by Krishna et al. [22] have demonstrated strong generalizability across very different domains; the technique employs a combination of CNN architectures trained across heterogeneous datasets that confer robustness to dataset shifts. Zhao et al. [23] have proposed a multi-task continual learning paradigm for plant disease detection that allows for the concurrent learning of disease-classification and related severity-estimation tasks. This multi-task setting reduces forgetting and improves knowledge retention.

An AI-IoT smart agriculture pivot system that blends automatic treatment with plant disease detection is developed by Ibrahim et al. [24]. This framework represents a practical field deployment of detection models in IoT nodes and a centralized server. But it totally depends on cloud processing and too little attention has been paid so far to ultra-lightweight on-device learning or continual model adaptation right at the edge.

Li et al. [25] have presented a non-exemplar class-incremental learning methodology for diagnosing plant diseases that eliminates the use of rehearsal buffers by utilizing regularization-based methods and constraints on feature space. By eliminating rehearsal buffers, they are able to create models that consume a lower overall storage footprint. Non-exemplar methods often lack the accuracy of replay training-based approaches, and thus represent a significant hindrance to plant disease models. Aftab et al. [26] published a study examining how Raspberry Pi devices and AI based on the Python programming language could be used to detect plant diseases on-site. The authors findings suggest that it is possible to deploy small CNN models onto very energy-efficient devices such as Raspberry Pi.

The design of advanced plant disease detection methods with deep learning has resulted in numerous advances. There are still some key areas that have not been successfully addressed. Current Methods use static training approaches and cannot incorporate data on new diseases without completely retraining the model. That presents significant practical challenges for agricultural applications.

Although many of the advances that have occurred in agricultural vision are based on compressed models through knowledge distillation. Few studies have combined this with a continual learning model to allow for efficient memory consumption and allow for resistance to catastrophic forgetting. Additionally, in prior studies the focus has been primarily on maximizing accuracy without due consideration of other deployment issues such as model size, inference latency, and energy consumption.

Further, the performance of the many methods that are being developed is generally evaluated only on the basis of final accuracy. Whereas evaluating continually learning performance requires consideration of average accuracy and forgetting, limiting the evaluation of long-term learning performance. Finally, most existing works only validate their methods using single crop datasets in ideal settings. Therefore, validation across crops, real-world noise, and evolving disease scenarios has yet to be achieved, thus limiting the generalization and therefore practical use of these methods.

3. PROPOSED METHOD

The proposed PlantKD-CLNet integrates Knowledge Distillation (KD), Continual Learning (CL), and Replay Buffers to build a compact model capable of adapting as plant disease pattern changes. Earlier studies mostly focus on static KD or complex ensemble models, but the proposed framework is shaped specifically to handle three challenges that arise in real agricultural environments: the steady arrival of new disease classes as field condition change, the loss of previously learned knowledge, commonly known as Catastrophic forgetting during continual learning, and the practical limits of computation and memory on IoT and edge devices. The methodology is developed through definition of the dataset and class structure, data preprocessing and augmentation, and model training pipeline.

PlantKD-CLNet was designed to help us effectively detect plant diseases in real-time on edge devices using a combination of Knowledge Distillation (KD), Continual Learning (CL) and Replay Buffers, whilst solving three common issues faced when operating in agricultural environments: the appearance of new disease classes over time, catastrophic forgetting when using incremental learning, and limited processing power available from IoT and edge device capabilities.

A four-phase structure defines the pipeline through which PlantKD-CLNet operates are also given in Figure 1:

3.1 Teacher training & student ınitialization

The teacher model MobileNetV2 performs training to create robust representations for the base disease classes. This, in turn, allows a student model to be created and developed per the supervised learning method, whilst distilling soft labels from the teacher model. The benefits of the teacher models' advanced learning are conferred on the student model, making it optimal for resource-constrained environments. The categorical cross-entropy loss of Teacher model is calculated using the Eq. (1).

$$\mathcal{L}_{CE}^{T} = -\sum_{i=1}^{C} y_i \log(P_i^T) \tag{1}$$

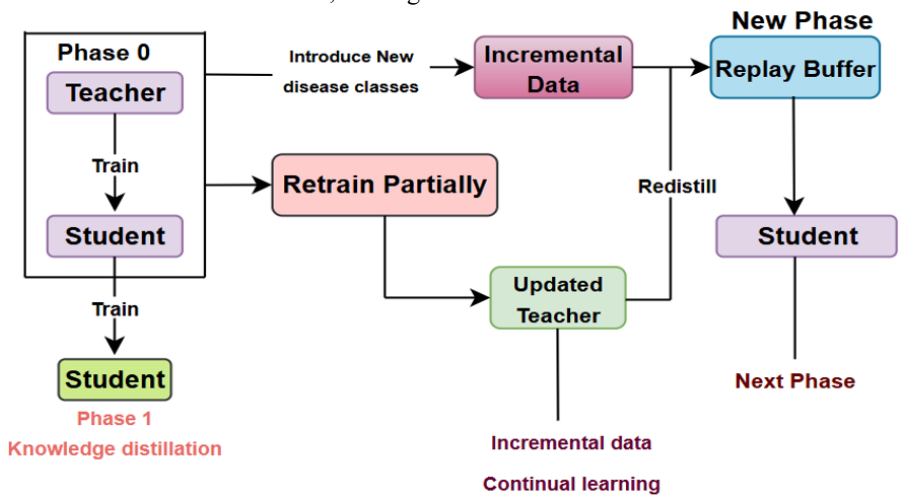

Figure 1. PhantKD-CLNet architecture

It is the Standard cross-entropy for the teacher model over C classes. Teacher Output Probability (Softmax) is calculated using the Eq. (2).

$$P_i^T = \frac{e^{z_i^T}}{\sum_{j=1}^{C} e^{z_j^T}} \tag{2}$$

Softmax applied to logits z_i^T of teacher. Teacher Weight Update is given in Eq. (3).

$$\theta_T = \theta_T - \eta \nabla_{\theta_T} \mathcal{L}_{CE}^{T} \tag{3}$$

Standard gradient descent update.

3.2 Knowledge distillation to student model

This phase uses incremental dataset to introduce the first batch of new disease classes. During this stage, students use a combination of supervised learning, replay-buffer examples and distillation from a prior snapshot taken of the student, in response to adapting to these new classes. This ensures that any past knowledge is retained without loss, thus ensuring that old and new class performance levels are maintained. Student Output Probability is calculated using Eq. (4).

$$P_i^S = \frac{e^{z_i^S}}{\sum_{j=1}^{C} e^{z_j^S}} \tag{4}$$

Softened Teacher Distribution is computed using Eq. (5).

$$P_i^{T(T)} = \frac{e^{z_i^T/T}}{\sum_{j=1}^{C} e^{z_j^T/T}} \tag{5}$$

Softened logits with temperature T for distillation. Knowledge Distillation Loss is computed using Eq. (6).

$$\mathcal{L}_{KD} = T^2 \sum_{i=1}^{C} P_i^{T(T)} \log \frac{P_i^{T(T)}}{P_i^{S(T)}} \tag{6}$$

Student Cross-Entropy Loss computed using Eq. (7).

$$\mathcal{L}_{CE}^{S} = -\sum_{i=1}^{C} y_i \log(P_i^S) \tag{7}$$

Combined Loss for Student using Eq. (8).

$$\mathcal{L}_S = \alpha \mathcal{L}_{KD} + (1-\alpha)\mathcal{L}_{CE}^{S} \tag{8}$$

Balances learning from teacher and ground truth. Typically, $\alpha = 0.5$. Student Weight Update is given as Eq. (9).

$$\theta_S \leftarrow \theta_S - \eta \nabla_{\theta_S} \mathcal{L}_S \tag{9}$$

3.3 Replay-based continual learning for new diseases

The third stage of the incremental learning process. This stage adds new data sets for training the model. In addition, the "replay buffer" is updated by collecting "herding" samples of the previously learned classes so that the representative samples from the previous class will still be present. The model continues to learn in a progressive manner by utilizing a combination of teacher distilling, supervision and regularisation; these three methods allow for the reduction of catastrophic forgetting due to the increasing number of diseases being represented by the same data set. Refinement Loss (Hard Labels Only) is calculated using Eq. (10).

$$\mathcal{L}_{refine} = \mathcal{L}_{CE}^{S} \quad (10)$$

Refined Student Weight Update is given as Eq. (11).

$$\theta_S = \theta_S - \eta \nabla_{\theta_S} \mathcal{L}_{refine} \quad (11)$$

Augmentation Consistency Loss is given as Eq. (12).

$$\mathcal{L}_{aug} = \frac{1}{N} \sum_{i=1}^{N} \| f_\theta(x_i) - f_\theta(\hat{x}_i) \|_2^2 \quad (12)$$

Here, Ensures stable predictions under augmented inputs $\hat{x}_i$. Total Refinement Loss is computed using Eq. (13).

$$\mathcal{L}_{total}^{refine} = \mathcal{L}_{refine} + \lambda \mathcal{L}_{aug} \quad (13)$$

Here, $\lambda$is a regularization weight.

3.4 Optimization

The last stage of the incremental learning process and the last set of classes in that stage. In stage 3, all data sets for training the student are incorporated for the final set of incremental disease classes, using D^3. The student is now faced with training on all classes, and hence the difficulty level of training the model has increased considerably, as the student has exemplars from each of the previous classes contained in the replay buffer. Increased levels of regularisation and feature-alignment restrictions will be employed to ensure long-term model stability. The student model effectively covers all categories of disease, while at the same time effectively reducing catastrophic forgetting. Replay Buffer Sampling is computed using Eq. (14).

$$\mathcal{B}_r = \{ (x_j, y_j) \mid j \in \text{Exemplars from previous classes} \} \quad (14)$$

Incremental Training Batch is given as Eq. (15).

$$\mathcal{B} = \mathcal{B}_{new} \cup \mathcal{B}_r \quad (15)$$

Dual-Loss for Incremental Learning is given as Eq. (16).

$$\mathcal{L}_{dual} = \beta \mathcal{L}_{CE}^{new} + (1 - \beta) \mathcal{L}_{KD}^{replay} \quad (16)$$

Combines learning new classes ($\mathcal{L}_{CE}^{new}$) and retaining old knowledge ($\mathcal{L}_{KD}^{replay}$). Exemplar Selection (Herding) is done using Eq. (17).

$$x_j^* = \arg\min_{x_j \in \mathcal{D}_c} \| \phi(x_j) - \mu_c \| \quad (17)$$

Selects representative samples closest to class mean $\mu_c$in feature space $\phi(x)$. Student Update (Incremental Phase) is dones using Eq. (18).

$$\theta_S = \theta_S - \eta \nabla_{\theta_S} \mathcal{L}_{dual} \quad (18)$$

3.5 Edge deployment & on-device adaptation

The final model used by the student is deployed to an IoT or mobile device using techniques such as model compression (quantization or pruning) to ensure that it has the smallest possible file size and will still be able to perform inference at millisecond-level latencies. Additionally, the model that is uploaded to an IoT or mobile device can receive micro-updates from the device's field data. These micro-updates allow for continuous improvements to the model without the need for large sets of training data or for full retraining on a central server. Latency-Constrained Objective function is given in Eq. (19).

$$\min_{\theta_S} \mathcal{L}_{final} \text{ Latency}(\theta_S) \leq L_{\max} \quad (19)$$

Ensures model fits edge device constraints. Compressed Model Output is given as Eq. (20).

$$f_\theta(x) = \text{Quantize}\left(f_{\theta_S}(x)\right) \quad (20)$$

Applies quantization for lightweight deployment.

In every incremental stage, training batches are created by merging the incoming task data with previously collected replay buffer samples from earlier classes. The student model will be supervised from three distinct areas at the same time: (1) hard labels from the task data, (2) replayed hard labels from earlier classes, and (3) soft targets that have been produced through a frozen teacher model for both new and replay samples. The combined loss value is determined using a combination of cross-entropy loss values based on the current hard labels versus Kullback–Leibler divergence between the soft predictions of both the student and the teacher models. Jointly optimizing multiple objectives will allow for the maintenance of previously learned knowledge while providing for the adaptability to new classes. As demonstrated in this four-phase pipeline, the PlantKD-CLNet process creates highly accurate, low-latency models that require little memory to store and run, making them well-suited for near-real-time monitoring of plant disease in rapidly changing agricultural ecosystems.

4. RESULTS AND DISCUSSIONS

The experimentation involves the Tomato 10 dataset comprising high-quality photos of ten different tomato plant disease categories. This dataset includes 10 different disease classes. To compensate for the low number of test images, we conduct a 5-Fold Cross Validation on all of the reported results in this study. The structure of each fold utilizes 80% of the entire data set for training and 10% each for validation and testing. The splits used are fixed with 10,001 training, 912 validations, 33 test images and these fixed splits are used only to provide benchmarking figures for pre-processing. The final results represent the averages of the cross-validation folds as opposed to the fixed splits. Since there is a learning process built into the design of the Tomato_10 images, they are organized based on phase increments or levels of learning Phase 0 contains only the base categories of tomato diseases, and Phases 1 through 3 contain new categories of tomatoes. To standardize the input images in a consistent manner, they have been down sampled to 224 × 224 pixels before being normed to be in the range of [0,1]. All Models have been augmented with a step-wise approach where new augmentations are introduced incrementally to improve the generalization of the Models. In Phase 0 the lightest level of augmentation is introduced through the addition of small

rotations, horizontal, vertical flips, and slight contrast adjustments. This will act as the base for future varieties in the models. Increasingly aggressive augmentations have been added in subsequent incremental Learning Phases. These augmentations reflect some of the more commonly seen effects of real-world environmental differences in conditions such as brightness, partial leaf coverage, and natural texture noise differences. It is therefore important that the Student Model is well equipped for any future impacts from these environmentally varied conditions when using or training under field conditions or photo acquisition conditions that are less than optimal.

Experiments are performed using a combination of edge devices along with some of the best performing computing resources available, an Intel Core i7 CPU with an NVIDIA RTX 3080 GPU was used for training. While deploying and performing both deployment and inference efforts, it was done on the equivalent of an NVIDIA Jetson Nano and a Raspberry Pi 4 to simulate what might be experienced in field-based setting. Python 3.10 and Pytorch 2.2, CUDA 12.0 and OpenCV 4.7 were used as the framework for these experiments with Training Hyperparameters remaining consistent through every phase of the experiment i.e., Batch Size of 32, Initial Learning Rate of 0.001 and the Adam Optimizer. The teacher training phase-0 lasts for five epochs but the training phases for student distillation and refinement. Each subsequent incremental learning phase will instead last for ten epochs with implementation of validation loss-based early stopping to mitigate overfitting issues.

The evaluation metrics used in assessing the model performance of PlantKD-CLNet include the following: Overall accuracy, which refers to the number/percentage of correctly classified images; Precision, Recall, F1-score. The size of the models is reported in megabytes for memory efficiency, with latency of inference, which indicates suitability of real time use. The effect of catastrophically remembering previous classes is quantified by comparing the accuracy of those classes when learned and when received through incremental updates as a measure of the stability of continuous learning.

To evaluate the performance of the PlantKD-CLNet method, it will be compared with other leading knowledge distillation methods only, continual learning methods only, and hybrid of the prior two using DenseNet-Composite-lightweight CNN [16], ResNet-EfficientNet [17], Plant-based Mobile Vision Transformer (PMVT) [23], LFM-CNAPS [22], Multistage KD [19]. The metrics used for all baseline comparisons are the same, which allows for comprehensive performance evaluation of each method with respect to accuracy, size, latency, and robustness to incremental learning. A thorough experimental scheme guarantees extensive testing of the presented framework's performance, efficiency, and real-time application of IoT solutions within the agriculture sector. This can be demonstrated through various experiments that utilize different test conditions, hardware platforms, etc.

The effectiveness of all the training steps of the new PlantKD-CLNet solution, along with their comparison with both the teacher network and top baseline models, shows how well the proposed solution works. In Phase 0, an initial MobileNetV2 based teacher model provided a baseline performance of 95.8% while being very large and having an inference latency of 12.3ms, which limits its usefulness for deployments to regular edge devices, i.e. on-demand systems; therefore, it requires further improvements. Analysis of the Phase 1 distilled knowledge models show that, while the teacher model provided good accuracy at 95.8%, models produced by this phase provide improved accuracy of 96.8%, as well as reduced model size of 3.8MB and inference latency of 4.7ms. These findings support the benefits of the knowledge distillation process for retaining a model's learnings while greatly improving the efficiency of deploying it for use.

Continued enhancements provided by the Phase 2 refinement phase produced results that further improved the model. Based on the results shown for the refined student model, the final result of 97% accuracy is also supported by a slightly smaller refined model with a significantly faster latency. The combination of refining distilled knowledge with KD and Replay provided a way for the student to combine the most important aspects of the teacher while minimizing the loss of knowledge when exposed to augmented data and different field conditions. Using the KD + Replay continual learning approach in Phase 3 prevented the student from experiencing catastrophic forgetting when learning from newly added disease classes. The accuracy of the student across all incremental phases remained constant at 97%, providing evidence that knowledge of prior learned classes was maintained through the use of a replay buffer and a dual-loss optimization process.

A breakdown of precision, recall, and F1-scores for each class demonstrated that overall precision, recall, and F1-score were very high. F1-scores for most classes were greater than 0.94, indicating that the models were robust enough to differentiate between visually similar diseases. Confusion matrices indicated that most misclassifications occurred for leaf disease classes with overlapping visual features, providing the potential for further development and improvement through the introduction of additional sensor data or the introduction of more complex features representing texture. The evaluation of the proposed system is performed on 4 stages by using tomato_10 dataset. Images were resized to 224×224 normalized and expanded with various augmentation including blur, rotation, noise and random erasing. Table 1 summarizes the accuracy, speed, inference time and model size across each stage. The initial model MobileNetV2 reaches 95.8% accuracy but need close to 13MB memory and roughly to 96.8% and bought inference time down to 4.7ms. After further refinement the model became even lighter without satisfactory performance. The final PlantKD-CLNet achieved 97% accuracy with 37MB usage and around 4.3ms latency. These differences are shown in Figures 2 and 3.

Table 1. Phase wise performance summary

Phase	Model	Accuracy in %	Model Size (MB)	Latency (ms)
0	Teacher (MobileNetV2)	95.8	~60	12.3
1	Student (KD)	96.8	3.8	4.7
2	Student Refinement	97.0	3.7	4.3
3	KD+Replay (PlantKD-CLNet)	97.0	3.7	4.3

The Figure 4 represents the confusion matrix for the Teacher model and gives clear picture of where the model is outperforming and where it struggled before considering any distillation steps. When looking at the KD + Replay variant in Figure 5, the pattern appears noticeably more balanced. The number of off diagonal mistakes drops and the classes with

fewer samples receive more confidential predictions. The Accuracy and the loss curves in Figure 6 settle in a steady manner without signs of overfitting. It suggests that the chosen augmentation and the early stopping settings worked reasonably well in practice. The distilled and continual learning models are quite compact, around 3.7 MB so they fit comfortably on lightweight IoT devices such as Jetson Nano or Raspberry Pi 4. With a batch-size-32, the inference time comes out to roughly 0.009s leaving a generous margin under the usual200ms limit for real-time identification even without any pruning or quantization applied. The phase wise comparison in shows the inference timing improves across the models. The Teacher required 12.3ms, the KD Student 4.7ms, and the Refined Student manages about 4.3ms. All of these fall well within the practical limits for real time field deployment, making the framework suitable for usage in agricultural settings.

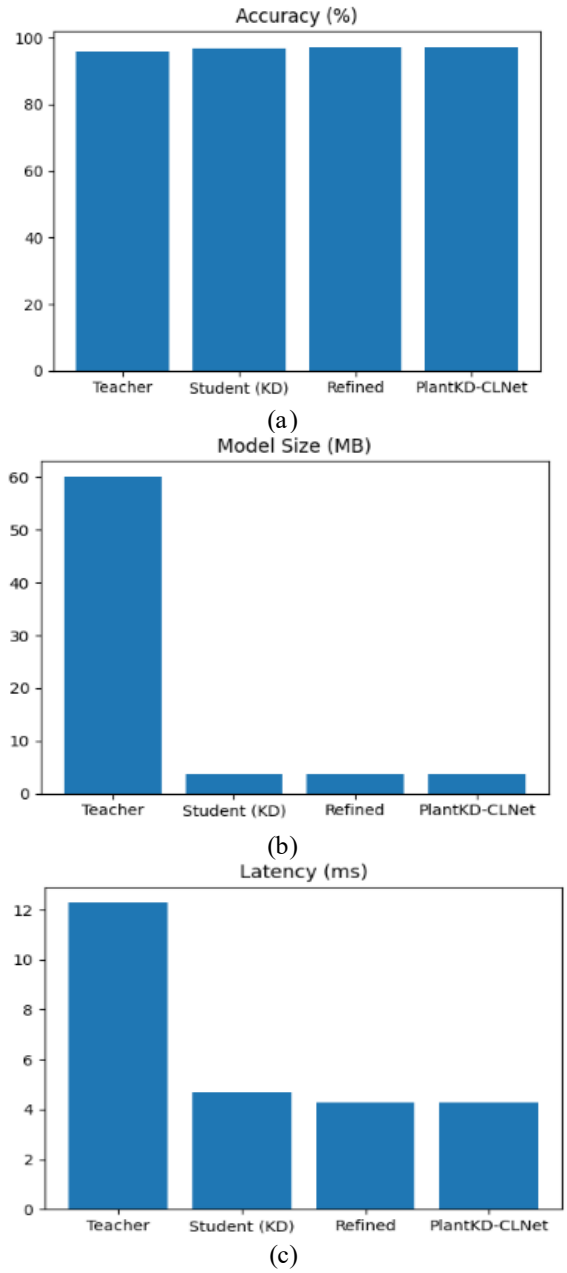

Figure 2. Highlighting the trade-off between (a)accuracy, (b)size, and (c) latency across phases

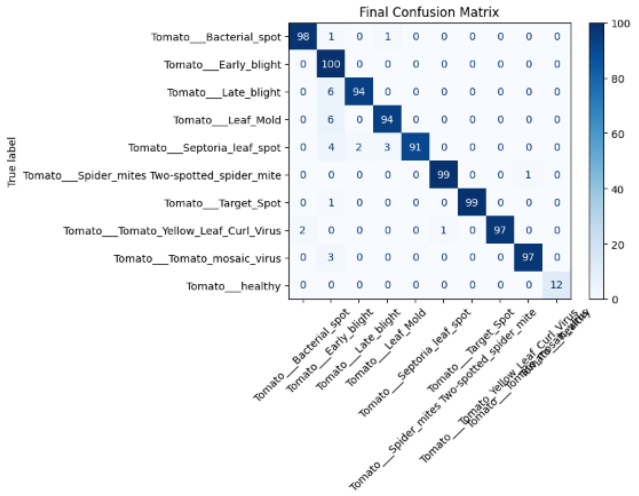

Figure 3. Confusion matrix for teacher model (Phase 0)

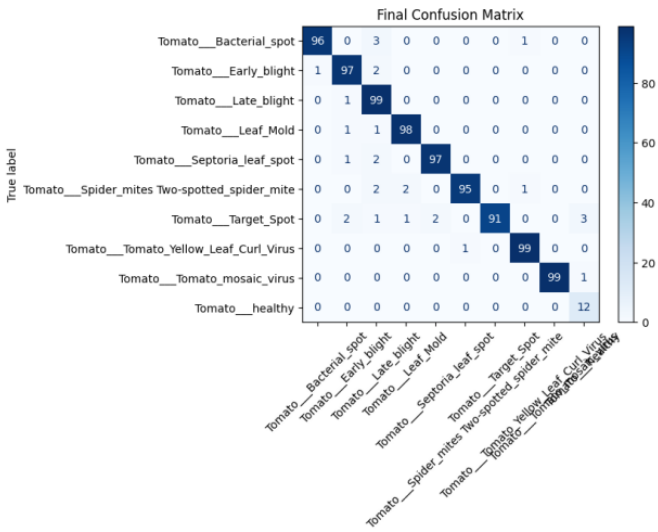

Figure 4. Confusion matrix for KD + replay student model

Note: CNN = Convolutional Neural Networks; KD = Knowledge Distillation.

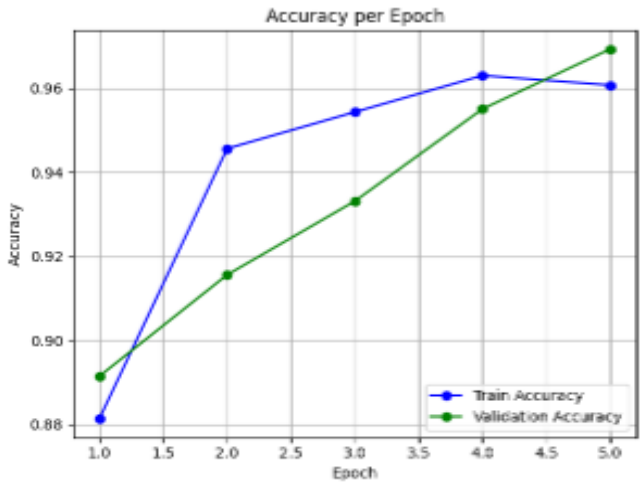

Figure 5. Training/validation accuracy

The strength of support provided by the model for each individual disease class was analysed by determining the precision, recall and F1 score values for all disease categories

and comparing those results with the aggregate results shown in Table 2.

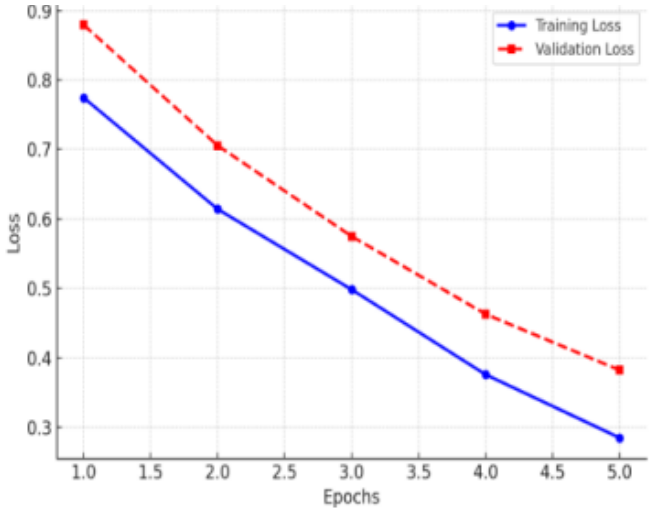

Figure 6. Loss curves across epochs

The PlantKD-CLNet model performs well on each disease class, with F1-scores above 0.94 for every disease category. However, some of these categories share visual symptoms that are very similar e.g. bacterial spots and fungal spots. The fact that only a small number of these two classes were frequently confused shows that additional feature channels or the combination of two or more sensors may benefit some models during the development process. A comparative analysis of the proposed framework against KD only, CL only, and hybrid models was performed using the results presented in Table 2.

Table 2. Comparison with existing methods

Category	Method	Accuracy / Size
KD-only	DenseNet-Lightweight CNN [16]	94.6%, ~70% reduction
KD-only	ResNet-EfficientNet [17]	98.1%, ~5.1 MB
KD-only	Deep Teacher-Shallow CNN [18]	99.4%, 90% param reduction
KD-only	Multistage KD [19]	60%mAP, 18–70 MB
CL-only	Generalized CNN [20]	96–97%
CL-only	PMVT [23]	98.9%
Hybrid	PlantKD-CLNet (Proposed)	97%, 3.7 MB

Note: CNN = Convolutional Neural Networks; KD = Knowledge Distillation; PMVT = Plant-based Mobile Vision Transformer.

KD-only models excel with static datasets but do not generalize well when a disease type is introduced. CL-only models provide incremental learning functionality but are often highly resource intensive and inefficient. PlantKD-CLNet was created to combine both fields of study and, as such, offers several benefits: plantKD-CLNet has high performance, an ultra-lightweight form factor, real-time capabilities, and is able to perform incremental learning. This makes the system more suitable for low power IoT hardware commonly used for continuous crop health monitoring. Across the different training phases, it comes clear that knowledge distillations help pass the important features from the larger model to the lighter one without causing a noticeable drop in performance. The replay buffer also plays a useful role by reducing catastrophic forgetting, helping the model learn new diseases categories while still remembering the older ones. The results in Confusion matrix shows that most errors appeared in leaf diseases classes due to similar visual patterns, suggesting the future improvements may benefit from considering stronger texture features or blending image data with additional sensors. The inference time evaluation highlights the practicality of the proposed approach, as the student model is capable of processing each image under 5ms. Maintained an accuracy level above 96%, the framework shows strong ability for deployment in real agricultural IoT settings, rather than staying only an experimental scenario.

5. CONCLUSION

This research presents PlantKD-CLNet, an efficient and effective system that uses a combination of Knowledge Distillation and Replay-based Continual Learning Techniques to enable the detection of plant diseases in real-time using mobile edge devices. The framework has been developed based on the key obstacles faced in agricultural applications: limited resources to support the deployment of complex algorithms, changing patterns of disease, and the need to prevent catastrophic forgetting when training continual learning algorithms. The student network, trained using a MobileNetV2 teacher achieved 97% accuracy while achieving a decrease in the overall size of the model. This model went from 60MB down to 3.7MB with an actual inference time of just 4.3ms. The findings from this research indicate that knowledge distillation improves both model compressibility and inference speed, whereas replay-based continual learning has a substantial impact on reducing catastrophic forgetting, both of which allow models to maintain good performance when learning new plant diseases incrementally. Future developments to the PlantKD-CLNet framework should focus on the detection of multiple crops, sensor integration, federated learning and interpretability to provide greater robustness and trust from users in ever-changing field environments.

REFERENCES

[1] FAO. (2020). The state of food and agriculture 2020. Overcoming water challenges in agriculture. Food and Agriculture Organization of the United Nations, 68-73.

[2] Convention on Biological Diversity. (2018). 2.6 billion people draw their livelihoods mostly from agriculture. CBD. https://www.cbd.int/article/biodiversityforfood-1.

[3] Gai, Y., Wang, H. (2024). Plant disease: A growing threat to global food security. Agronomy, 14(8): 1615. https://doi.org/10.3390/agronomy14081615

[4] Sun, X., Li, G., Qu, P., Xie, X., Pan, X., Zhang, W. (2022). Research on plant disease identification based on CNN. Cognitive Robotics, 2: 155-163. https://doi.org/10.1016/j.cogr.2022.07.001

[5] Ali, H., Shifa, N., Benlamri, R., Farooque, A.A., Yaqub, R. (2025). A fine tuned EfficientNet-B0 convolutional neural network for accurate and efficient classification of apple leaf diseases. Scientific Reports, 15(1): 25732. https://doi.org/10.1038/s41598-025-04479-2

[6] Srinivasan, S., Prabin, S.M., Mathivanan, S.K., Rajadurai, H., Kulandaivelu, S., Shah, M.A. (2025). Sugarcane leaf disease classification using deep neural network approach. BMC Plant Biology, 25(1): 282. https://doi.org/10.1186/s12870-025-06289-0

[7] Sutaji, D., Rosyid, H. (2022). Convolutional Neural Network (CNN) models for crop diseases classification. Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, 7(2): 1443. https://doi.org/10.22219/kinetik.v7i2.1443

[8] Li, D., Yin, Z., Zhao, Y., Li, J., Zhang, H. (2024). Rehearsal-based class-incremental learning approaches for plant disease classification. Computers and Electronics in Agriculture, 224: 109211. https://doi.org/10.1016/j.compag.2024.109211

[9] Zhan, K., Peng, Y., Liao, M., Wang, Y. (2025). Domain generalization plant leaf disease recognition: Toward from laboratory to field. Engineering Applications of Artificial Intelligence, 156: 111168. https://doi.org/10.1016/j.engappai.2025.111168

[10] Joshi, A., Shet, A.V., Thambi, A.S., R, S. (2023). Quality improvement of image datasets using hashing techniques. In 2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE), Bengaluru, India, pp. 18-23. https://doi.org/10.1109/IITCEE57236.2023.10091044

[11] Wang, Z., Yang, E., Shen, L., Huang, H. (2024). A comprehensive survey of forgetting in deep learning beyond continual learning. IEEE Transactions on Pattern Analysis and Machine Intelligence, 47(3): 1464-1483. https://doi.org/10.1109/TPAMI.2024.3498346

[12] Li, W., Xu, X., Wang, W., Chen, J. (2025). Lightweight plant disease detection with adaptive multi-scale model and relationship-based knowledge distillation. Expert Systems, 42(6): e70059. https://doi.org/10.1111/exsy.70059

[13] Jackulin, C., Murugavalli, S.J.M.S. (2022). A comprehensive review on detection of plant disease using machine learning and deep learning approaches. Measurement: Sensors, 24: 100441. https://doi.org/10.1016/j.measen.2022.100441

[14] Pacal, I., Kunduracioglu, I., Alma, M.H., Deveci, M., et al. (2024). A systematic review of deep learning techniques for plant diseases. Artificial Intelligence Review, 57(11): 304. https://doi.org/10.1007/s10462-024-10944-7

[15] Zhang, X., Liang, K., Zhang, Y. (2024). Plant pest and disease lightweight identification model by fusing tensor features and knowledge distillation. Frontiers in Plant Science, 15: 1443815. https://doi.org/10.3389/fpls.2024.1443815

[16] Wang, B. (2024). Zero-exemplar deep continual learning for crop disease recognition: A study of total variation attention regularization in vision transformers. Frontiers in Plant Science, 14: 1283055. https://doi.org/10.3389/fpls.2023.1283055

[17] Yang, X., Wang, H., Zhou, Q., Lu, L., Zhang, L., Sun, C., Wu, G. (2025). A lightweight and efficient plant disease detection method integrating knowledge distillation and dual-scale weighted convolutions. Algorithms, 18(7): 433. https://doi.org/10.3390/a18070433

[18] Duhan, S., Gulia, P., Gill, N.S., Narwal, E. (2025). RTR_Lite_MobileNetV2: A lightweight and efficient model for plant disease detection and classification. Current Plant Biology, 42: 100459. https://doi.org/10.1016/j.cpb.2025.100459

[19] Ashurov, A.Y., Al-Gaashani, M.S., Samee, N.A., Alkanhel, R., Atteia, G., Abdallah, H.A., Saleh Ali Muthanna, M. (2025). Enhancing plant disease detection through deep learning: A Depthwise CNN with squeeze and excitation integration and residual skip connections. Frontiers in Plant Science, 15: 1505857. https://doi.org/10.3389/fpls.2024.1505857

[20] Upadhyay, A., Chandel, N.S., Singh, K.P., Chakraborty, S.K., et al. (2025). Deep learning and computer vision in plant disease detection: A comprehensive review of techniques, models, and trends in precision agriculture. Artificial Intelligence Review, 58(3): 92. https://doi.org/10.1007/s10462-024-11100-x

[21] Huang, Q., Wu, X., Wang, Q., Dong, X., et al. (2023). Knowledge distillation facilitates the lightweight and efficient plant diseases detection model. Plant Phenomics, 5: 0062. https://doi.org/10.34133/plantphenomics.0062

[22] Krishna, M.S., Machado, P., Otuka, R.I., Yahaya, S.W., Neves dos Santos, F., Ihianle, I.K. (2025). Plant Leaf Disease Detection Using Deep Learning: A Multi-Dataset Approach. J-Multidisciplinary Scientific Journal, 8(1): 4. https://doi.org/10.3390/j8010004

[23] Zhao, Y., Jiang, C., Wang, D., Liu, X., Song, W., Hu, J. (2023). Identification of plant disease based on multi-task continual learning. Agronomy, 13(12): 2863. https://doi.org/10.3390/agronomy13122863

[24] Ibrahim, A.S., Mohsen, S., Selim, I.M., Alroobaea, R., Alsafyani, M., Baqasah, A.M., Eassa, M. (2025). AI-IoT based smart agriculture pivot for plant diseases detection and treatment. Scientific Reports, 15(1): 16576. https://doi.org/10.1038/s41598-025-98454-6

[25] Li, D., Yin, Z., Zhao, Y., Zhao, Y., Zhang, H. (2025). Non-exemplar class-incremental learning for continual plant diagnosis. Crop Protection, 190: 107069. https://doi.org/10.1016/j.cropro.2024.107069

[26] Aftab, S., Lal, C., Beejal, S.K., Fatima, A. (2022). Raspberry pi (python ai) for plant disease detection. International Journal of Current Research and Review, 14: 36. https://doi.org/10.31782/IJCRR.2022.14307

NOMENCLATURE

C	Number of classes in current phase
x_i	Input image
y_i	Ground truth label
z_i^T, z_i^S	Logits of teacher/student
P_i^T, P_i^S	Softmax output probabilities
T	Temperature for softening logits
θ_T, θ_S	Model parameters (teacher/student)
η	Learning rate
α, β, λ	Loss weighting hyperparameters
$\mathcal{D}^k$	Dataset of incremental phase k
$\mathcal{B}_r$	Replay buffer batch
$\mathcal{B}_{new}$	Current new class batch
$\phi(x)$	Feature representation of input x
μ_c	Feature mean of class c
$L_{\max}$	Maximum allowed latency for edge deployment
$\mathcal{L}_{CE}, \mathcal{L}_{KD}, \mathcal{L}_{dual}, \mathcal{L}_{aug}$	Loss functions as defined above