Evaluating Knowledge Distillation for Plant Leaf Disease Classification

David J. Richter
Dept. of A. I. Convergence
Chonnam National University
Gwangju, South Korea
0000-0001-5413-6710

Md Ilias Bappi
Dept. of A. I. Convergence
Chonnam National University
Gwangju, South Korea
0009-0000-7616-3074

Kyungbaek Kim
Dept. of A. I. Convergence
Chonnam National University
Gwangju, South Korea
0000-0001-9985-3051

***Abstract*—Plant leaf diseases lead to large losses in crop harvest yields every year worldwide. Preventing the spread of such diseases is therefore essential to guarantee stable food production and as a result to feed as many people as possible. Manual field scouting can be tedious, costly, laborious, time consuming and error prone. As a result, artificial intelligence solutions have been explored in recent years, to help overcome these issues and improve field scouting to assure better protection of plant diseases. Such artificial intelligence models usually take in images and return whether or not the plant is diseased. Image based models such as this can be rather large and computationally heavy though, which can lead to performance issues or even prevent large models to be used in field, where only mobile end devices are really feasible. To generate more lightweight models that can be used in field scenarios, knowledge distillation will be utilized in this work to train a lightweight student model based on the knowledge stored in a large scale teacher model that was trained on the plant leaf disease data previously. The resulting small student model, which is only has about 6.5% of the large scales teachers parameters and only about 17.3% of the FLOPS, still achieves 84.5% accuracy.**

***Index Terms*—Plant Leaf Disease Classification, Plant Pathology, CNN, Deep Learning, Knowledge Distillation**

I. Introduction

Plant leaf disease identification is highly important to stop the spread of these plant diseases in crop fields in order to minimize losses. Crop production worldwide is constantly negatively affected by such plant diseases, which leads to large numbers of potential harvest being lost to them [1]. This results in valuable food resources, that could have helped combat the food shortage issue that is still present to this day [2], being lost. Crops are essential to the food supply all over the world, being the most common source of calorie and protein supply worldwide [3], which has led to even further increased crop production in recent years [3] and in years to come [1]. Plant diseases, as mentioned above, are the cause for huge losses in crop production [1]. To contain the spread of diseases and the losses that come from them, staff need to identify diseases as soon as possible and react accordingly to minimize the spread. To spot diseases early, field scouting is often employed. Field scouting is the traditional manual process of sending out staff into the field, where they sample plants in field for disease symptoms multiple times during the growth process of the crops, to spot diseases as early, fast, and reliable as possible. This process is, however, laborious, costly, time consuming, and potentially error prone.

One way of aiding farmers and staff in disease spotting is artificial intelligence (AI). Using AI models staff can take images of the plant, which the AI models then analyze before returning the disease it predicted. Models can do this with very high accuracy [4], which can aid in the process of correctly identifying diseases from healthy plants.

Convolutional neural networks (CNN) are a powerful AI solution that excels in this task [5]. CNN are deep learning (DL) neural networks (NN) that are specifically designed to receive and handle image data. Such plant leaf disease classification models based on CNN have seen a lot of attention in recent years [4], [6], [7]. CNN come in many different shapes and sizes, with many different architectures introducing new methods and advanced methods [5].

Many new CNN models, while very powerful, tend to be bigger and bigger, with more parameters and floating point operation (FLOP) counts. Such large models are indeed highly capable, but at the cost of performance and accessibility. Large scale models with advanced blocks and methods can only be run on very powerful GPU machines or even clusters, making them unable to be run on small and mobile end devices. In field, however, access to GPU clusters is not a given, and such smaller mobile end devices like smartphones are more feasible for usage.

To ensure that in-field usage of highly capable CNN models is given, training smaller models that are more lightweight and less computationally demanding is necessary, ideally without too much loss in performance. While large scale models are more capable, smaller models can be trained in a teacher student setting via knowledge distillation (KD) [8].

KD trains a large scale model on the data first. The large scale model, benefiting from the large number of parameters to store information, will ideally manage to learn to understand the data in a deep and nuanced fashion, learning meaningful representation of the features needed to complete the given task. This large scale model, while sufficiently powerful, might however be too heavy to be used in real-world use-cases. This is where the small scale student model comes into play. The previously trained large scale model will be used as a teacher to train the smaller student. The student is trained on the same

data as the teacher, but in addition to the labels provided by the dataset, the student will also be trained on labels generated by the teacher. Where the dataset labels are hard absolute (0 and 1), the teacher provided labels are soft (logits) and contain additional more nuanced information such as class similarity. This combination allows the smaller model, which might otherwise not be powerful enough to learn this detailed information on its own, to obtain that information through the teacher and perform better than it would otherwise.

Like this the smaller scale model can benefit from the knowledge stored within the pre-trained larger model and learn to better understand the target data. Through this methodology of knowledge distillation, smaller models can be trained to understand plant leaf disease images with better understanding of the nuances like a larger model would, while being small and lightweight enough to be usable in less computationally capable environments, as they can be found in field.

So, in this paper we present:

- A reduced DenseNet model called DenseNet-mini that only has 6.5% of the parameters and only 17.3% of the Giga FLOPS (GFLOPS) of DenseNet201
- A knowledge distillation approach for rice plant leaf disease classification using DenseNet201 as the teacher and DenseNet-mini as the student
- Comparative results of the student DenseNet-mini to the results of DenseNet-mini being trained without the help of the teacher model alongside further comparison to other models

The remainder of the paper is structured as follows: In Section II a brief review of related works is presented, before Section III describes the methodology used in this paper. Section IV presents the results generated in the experiments and Section V discusses the results, limitations and future work, before Section VI concludes this work.

II. Related Works

Recent work in plant leaf disease identification using AI has seen different approaches and methods.

In [9] Huang et al. present a KD approach for plant leave disease detection using the plantDoc dataset. By nature of this being a detection approach, the teacher and student models are a variant of the YOLO architecture. 4 different students were trained (YOLOR-Light-v1, YOLOR-Light-v2, Mobile-YOLOR-v1, and Mobile-YOLOR-v2), with the teacher being the larger YOLOR model. KD was carried out in a multi-stage process, with the backbone, the neck and the head being distilled. Results were promising, showcasing the validity of the approach.

Dong et al. [10] focus on apple leaf disease classification in their paper. Their proposed student model, was trained with ConvNeXt-Base acting as the teacher. The dataset utilized images of multiple datasets, combining them into a single large scale dataset to train the models on. While the teacher did outperform the student in their results, the student did achieve high accuracy scores, while reducing the model size significantly.

TABLE I: Dataset information for the Paddy Doctor [14] Dataset used in this work.

Parameter	Value
Image Size	(224, 224, 3)
Total Images	8,323
Diseases	Healthy, Bacterial Leaf Streak, Tungro, Brown Spot, Dead Heart, Hispa, Downy Mildew, Bacterial Panicle Blight, Blast, Bacterial Leaf Blight
Number of Classes	10
Plants	Rice Paddy
Train-Test Split	80-20

Ghofrani et al. [11] present a KD set of experiments with the lab data only dataset PlantVillage. The teacher was a Xception model, with the student being a tiny version of the MobileNet architecture. The dataset came with 39 classes and roughly 54,000 images. The teacher did outperform the student, but the student, which is much more lightweight, did manage to post high accuracy.

Hu et al. [12] make use of channel wise KD. Channel wise KD taps into the backbone prior to the output layers and compares embeddings channel by channel, while keeping the width and height dimensionality intact. The loss is calculated per channel per layer and summed up to achieve the total loss of the KD portion of training. This is said to better take into account the channel information stored in the layers.

In [13] Zhang et al. showcase their results of combining knowledge distillation with mix augmentation techniques to train a model that is lightweight yet capable on diverse data in a field where large datasets are sparse. Their ablation study shows that combining the two methods manages to outperform the pure baseline ResNet34, the baseline with only KD and the baseline with only mix augmentation.

III. Methodology

A. Dataset

To facilitate the KD experiments a sufficient dataset is needed. In this work, the Paddy Doctor [14], [15] dataset was used to train the teacher and student networks. The dataset is captured in field and contains 10 classes. In the experiments the train and validation subsets are used, which total 8,323 images which are train-validation split with a ratio of 80-20 (see Table I for information).

B. Models

The networks used as teacher and student are based on the DenseNet architecture [16] which have proven to work very well in the field of plant leaf disease classification [4], [5]. DenseNet uses so called dense blocks with appended residual connections to carry over the information of earlier convolutional layers in the block to later ones, which can benefit from this extra information, resulting in stronger performance (see Figure 1 for a detailed architectural overview of the DenseNet models used in this work). Regular DenseNet201 was used as

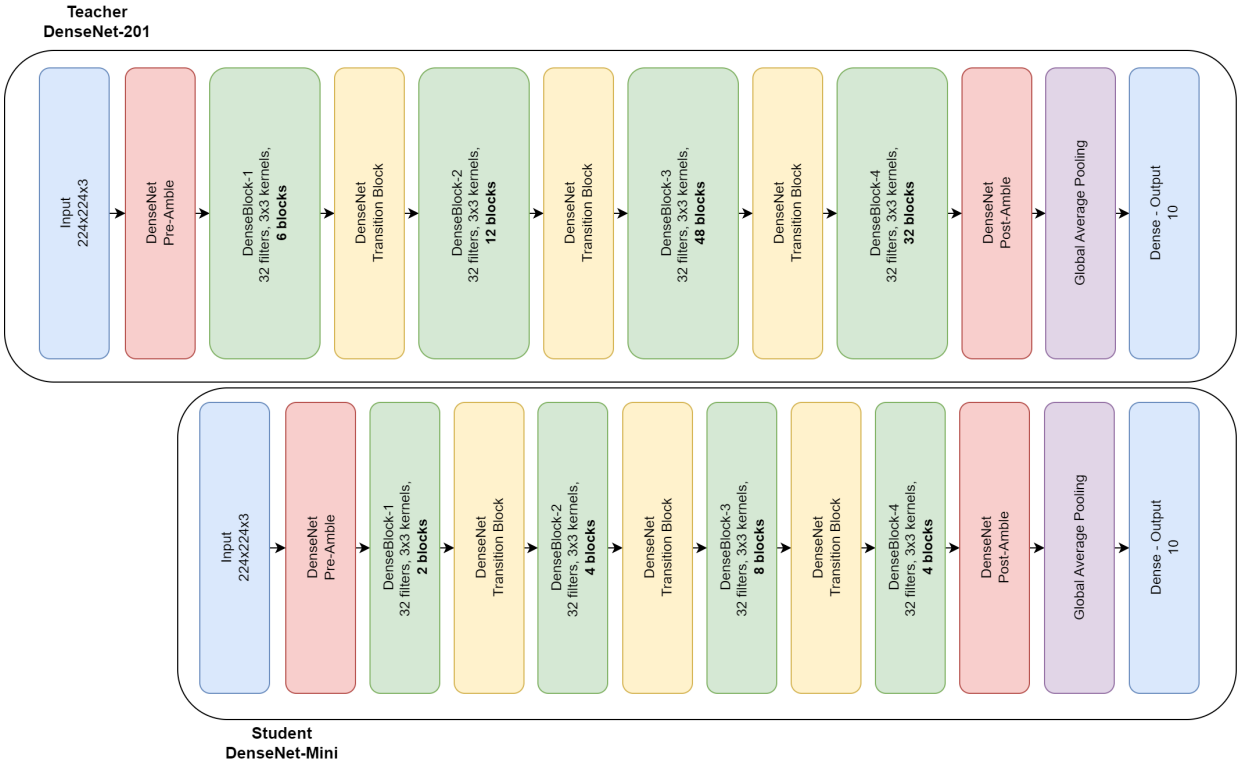

Fig. 1: The model architectures that were used as the teacher and student models in the experiments.

TABLE II: Parameter and GFLOP counts of the models used in this work.

Model	Parameters	GFLOPS
DenseNet201	18,341,194	8.63
DenseNet-mini	1,196,138	1.49
MobileNetV3-Small	1,540,218	0.12
NASNetMobile	4,280,286	1.15
EfficientNet-B0	4,062,381	0.79

the teacher, with a much smaller custom layer count DenseNet (DenseNet-mini) being used as the student (see Figure 1 for both models). The models are compared to EfficientNet-B0 [17], MobileNetV3-Small [18], and NASNetMobile [19]. All models and their parameter counts as well as GFLOPS can be seen in Table II. It becomes apparent that the DenseNet-mini model used and proposed as the student model in this work is both small and computationally lightweight, as can be seen by the parameter and GFLOP counts. Note that only very small and efficient models were used in this comparison besides the rather large teacher model. The student model is only 6.5% of the size of the teacher in terms of parameters and only about 17.3% in terms of FLOPS.

Then the teacher was trained on the dataset first, before aiding the student during knowledge distillation training (see Figure 2), where the student is trained with the teachers output as soft labels and the dataset labels as hard labels, which are combined in the loss calculation (see (1), (2) and (3)). The hyperparameters used can be seen in Table III. The training of the teacher is conducted just like regular supervised learning for classification CNN, here the teacher was trained for 100 epochs, but with early stopping enabled with a patience of 10. The trained teacher is then used as a teaching model during the training of the student. The student receives 2 sets of labels during training, one are the regular labels contained within the dataset, which are called hard labels, since labels are absolute (100% probability for the actual class of the image and 0% for all others). The other labels are called soft labels and come from the teacher. The teacher receives the same image as the student, but while the student is learning, the teacher is in inference mode. It predicts the logits for the image it received, which become the soft labels (here the class prediction is a probability distribution that is not 100% one single class, but the slightly imperfect prediction of the teacher, which contains class similarities which can help the student model learn more nuanced relations than only hard labels do).

$$\mathcal{L}_{\text{KD}} = \tau^2 \cdot \text{KL}\left(\text{softmax}\left(\frac{\hat{y}_{\text{teach.}}}{\tau}\right), \text{softmax}\left(\frac{\hat{y}_{\text{stud.}}}{\tau}\right)\right) \quad (1)$$

$$\mathcal{L}_{\text{stud.}}(y, \hat{y}_{\text{stud.}}) = -\sum_{i=1}^{C} \mathbf{1}_{[y=i]} \cdot \log\left(\text{softmax}(\hat{y}_{\text{stud.}})_i\right) \quad (2)$$

$$\mathcal{L}_{\text{total}} = \alpha \cdot \mathcal{L}_{\text{stud.}}(y, \hat{y}_{\text{stud.}}) + (1-\alpha) \cdot \mathcal{L}_{\text{KD}}(\hat{y}_{\text{teach.}}, \hat{y}_{\text{stud.}}) \quad (3)$$

The hard labels generate loss through simple sparse cross entropy (see (2)) with the predicted student labels, while the

TABLE III: List of Hyperparameters used.

Parameter	Value
Image Size	(224, 224, 3)
Batch Size Teacher Training	64
Batch Size Student Training	32
Loss	Categorical Crossentropy
Optimizer	Adam
Learning Rate	0.001
Temperature	1
Alpha	0.99
Epochs Teacher	100
Early Stopping Teacher	Patience 10
Epochs Student	10

TABLE IV: System Configuration used during Development.

Component	Specification
CPU	AMD Ryzen 7 5800X 8-Core @ 3.80 GHz
RAM	64GB
GPU	NVIDIA GeForce RTX 3090 24GB
Storage	500GB

teacher's soft labels are compared to the student output via KL-Divergence (see (1)), where τ is the temperature which can further soften the labels. Then both losses are combined through Equation (3), where α also denotes the factor that weights both functions. Both values can be found in table III. All experiments were carried out on the GPU box listed in Table IV. The 3 other models, as well as the DenseNet-mini model with no teacher help are trained for comparison. These models use the same hyperparameters as the student, but without KD loss, soft labels, or any of the other KD specific methods, and rather just use regular supervised learning with cross entropy loss.

IV. Results

After training the teacher, the student, as well as the comparison model results are listed in Table V. Here we can see that the teacher model manages the highest accuracy scores, as one would expect, with the model being so much larger than all the other models and with the teacher being given more time to train. When looking at the small scale models, however, we can see that the proposed DenseNet-mini student model trained through KD manages to outperform the other models as well as the DenseNet-mini model which was trained without the teacher. KD trained DenseNet-mini outperforms the DenseNet-mini without KD by over 9%, while managing to outperform the EfficientNet-B0 model by over 2.5%, while being considerably smaller in terms of parameters. Both NASNetMobile and MobileNetV3-Small heavily overfit to the data and never managed to learn any proper class representations in our experiment runs with our setup.

These results showcase that the lightweight KD trained student DenseNet-mini manages to perform within under 7% of the teacher, while outperforming all other models at the same time, beating the non KD DenseNet-mini by over 9%. This proves that KD can be a viable option in training small scale models for plant leaf disease classification and that the teacher's soft labels can benefit the student during training.

TABLE V: Results obtained. Teacher and Student training settings can be found in Table III. All other models were run with the same settings as the student.

Model	Val. Acc.
DenseNet201 - Teacher	91.29%
DenseNet-mini - Student	84.50%
DenseNet-mini - Standalone	75.48%
MobileNetV3-Small	16.47%
NASNetMobile	15.75%
EfficientNet-B0	81.91%

V. Discussion

The methods and results presented in this paper show that KD can be of benefit for the training of small scale models in the field of plant leaf disease classification, with the students improvement over the regularly trained model observed during training. These smaller models are much more applicable to real world usage in field, as these smaller models are much more lightweight, allowing them to run quicker and to run on lower-end end-devices, which can then be employed in field, while large scale models require lots of time and computational power.

In future work, the methods presented in this work could be tested in more challenging environments, with more and different models as the teacher, student and for comparison. Other data could also be used to train these models, to see how robust they are to different scenarios. Different variations and methods for KD could also be tested and compared, as well as different parameters and setups. Experiments could also be run more extensively than they were in this work to generate more robust and general results, to verify how well KD fits the field more broadly and generally.

VI. Conclusion

In this work, we presented a knowledge distillation approach to rice plant leaf disease classification, reducing the teacher models size by 93.5% in terms of parameters and by 82.7% in terms of GFLOPS, while only losing 6.79% in accuracy performance, with the student still reaching 84.50%. The student model trained through knowledge distillation manages to outperform the same model architecture trained without knowledge distillation by 9.02% accuracy. These results showcase that the knowledge distillation approach of reducing the final models size, and allowing it to run on weaker end-devices which will help in field usage, does have merit in the field and can effectively allow the training of smaller, yet still capable models.

Acknowledgements

This work was supported by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry(IPET) through the Agriculture and Food Convergence Technologies Program for Research Manpower development, funded by Ministry of Agriculture, Food and

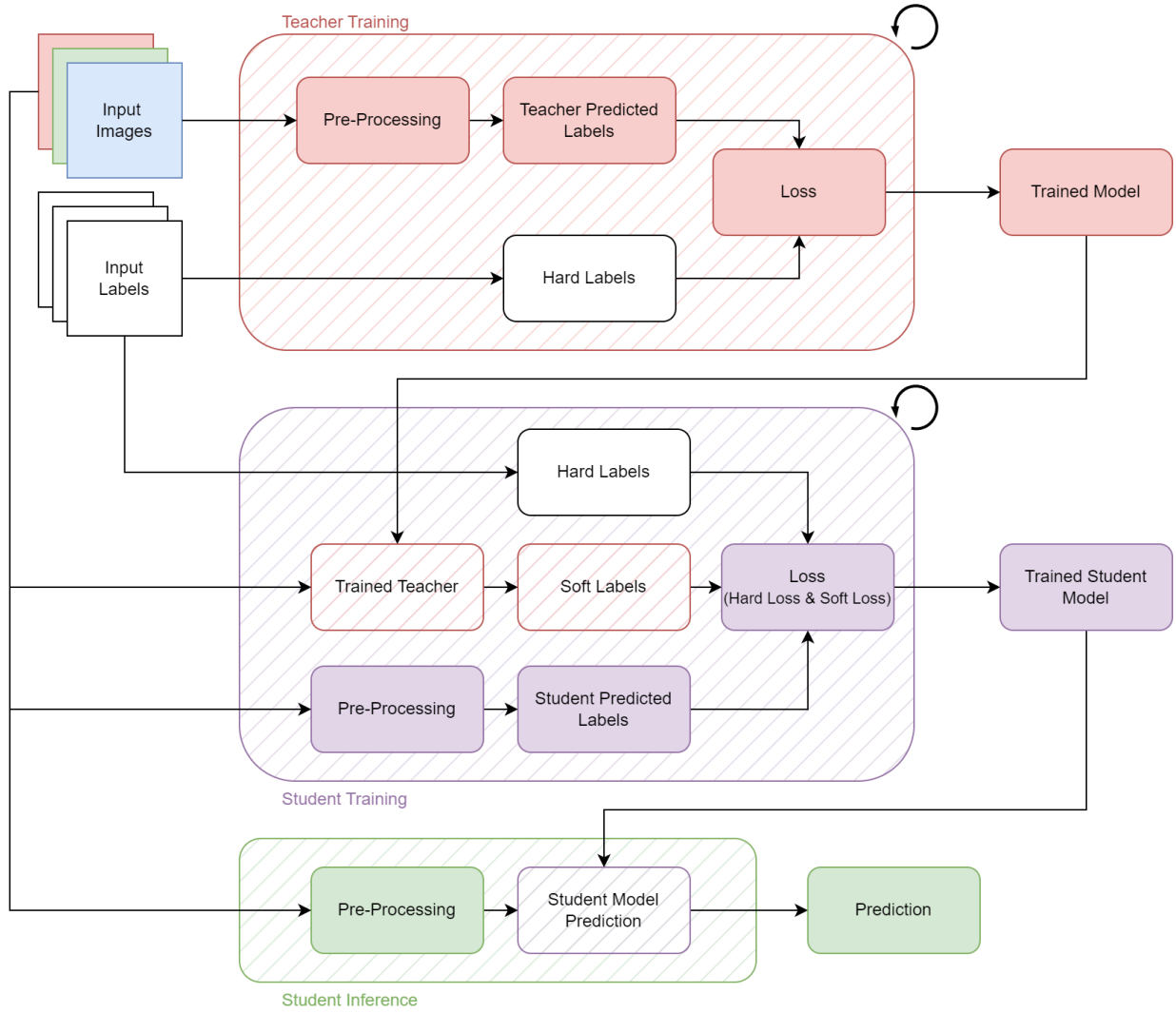

Fig. 2: The workflow of knowledge distillation, as it was used in this work, visualized.

Rural Affairs(MAFRA)(project no. RS-2024-00397026, 34%). This work was supported by the Institute of Information & Communications Technology Planning & Evaluation(IITP)-Innovative Human Resource Development for Local Intellectualization program grant funded by the Korea government(MSIT)(IITP-2025-RS-2022-00156287, 33%). This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) under the Artificial Intelligence Convergence Innovation Human Resources Development (IITP-2025-RS-2023-00256629, 33%) grant funded by the Korea government(MSIT).

REFERENCES

[1] FAO, *The Future of Food and Agriculture: Trends and Challenges*, 1st ed. Rome, Italy: FAO, 2017. [Online]. Available: https://openknowledge.fao.org/handle/20.500.14283/i6583e

[2] FAO, IFAD, UNICEF, WFP and WHO, *The State of Food Security and Nutrition in the World 2024 – Financing to end hunger, food insecurity and malnutrition in all its forms*, ser. The State of Food Security and Nutrition in the World (SOFI). Rome, Italy: FAO, IFAD, UNICEF, WFP, WHO, 2024, no. 2024.

[3] FAO, *World Food and Agriculture – Statistical Yearbook 2023*, ser. FAO Statistical Yearbook – World Food and Agriculture. Rome, Italy: FAO, 2023, no. 2023.

[4] D. J. Richter, M. I. Bappi, S. S. Kolekar, and K. Kim, "A systematic review of the current state of transfer learning accelerated cnn-based plant leaf disease classification," *IEEE Access*, 2025.

[5] D. J. Richter and K. Kim, "Assessing the performance of domain-specific models for plant leaf disease classification: a comprehensive benchmark of transfer-learning on open datasets," *Scientific Reports*, vol. 15, no. 1, pp. 1–31, 2025.

[6] C. Sarkar, D. Gupta, U. Gupta, and B. B. Hazarika, "Leaf disease detection using machine learning and deep learning: Review and challenges," *Applied Soft Computing*, vol. 145, p. 110534, 2023.

[7] W. B. Demilie, "Plant disease detection and classification techniques: a comparative study of the performances," *Journal of Big Data*, vol. 11, no. 1, p. 5, 2024.

[8] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[9] Q. Huang, X. Wu, Q. Wang, X. Dong, Y. Qin, X. Wu, Y. Gao, and G. Hao, "Knowledge distillation facilitates the lightweight and efficient plant diseases detection model," *Plant phenomics*, vol. 5, p. 0062, 2023.

[10] Q. Dong, R. Gu, S. Chen, and J. Zhu, "Apple leaf disease diagnosis based on knowledge distillation and attention mechanism," *IEEE Access*, vol. 12, pp. 65 154–65 165, 2024.

[11] A. Ghofrani and R. Mahdian Toroghi, "Knowledge distillation in plant disease recognition," *Neural Computing and Applications*, vol. 34, no. 17, pp. 14 287–14 296, 2022.

[12] Y. Hu, G. Liu, Z. Chen, J. Liu, and J. Guo, "Lightweight one-stage maize

leaf disease detection model with knowledge distillation," *Agriculture*, vol. 13, no. 9, p. 1664, 2023.

[13] H. Zhang and M. Wang, "Mixkd: Mix data augmentation guided knowledge distillation for plant leaf disease recognition," in *International Conference on Green, Pervasive, and Cloud Computing*. Springer, 2022, pp. 169–177.

[14] Petchiammal, B. Kiruba, Murugan, and P. Arjunan, "Paddy doctor: A visual image dataset for automated paddy disease classification and benchmarking," in *Proceedings of the 6th Joint International Conference on Data Science & Management of Data (10th ACM IKDD CODS and 28th COMAD)*, 2023, pp. 203–207.

[15] Petchiammal and P. Arjunan, "Paddy disease classification," https://kaggle.com/competitions/paddy-disease-classification, 2022, kaggle.

[16] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[17] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114. [Online]. Available: https://proceedings.mlr.press/v97/tan19a.html

[18] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.

[19] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.