**INTERNATIONAL JOURNAL OF ADVANCED RESEARCH**
*International Peer-Reviewed Research Journal*
**ISSN (Online): 3106-7980 | ISSN (Print): 3106-7972**
**Volume 3 • Issue 7 • November 2026**

# SEQUENTIAL DENSE NEURAL NETWORK FOR TOMATO LEAF DISEASE DETECTION USING BATCH NORMALIZATION AND DUAL REGULARIZATION

**Tahreem Sharif[1], Dr. Muhammad Ahmad Niaz[*2], Hafiz Muhammad Irfan[3], Rizwan Saleem[4], Imran Mahmood[5]**

[1] *Department of Computer Science, The Islamia University of Bahawalpur, Bahawalpur, Pakistan*

[1]tahreemsharif123@gmail.com, [*2]ahmad.niaz@iub.edu.pk, [3]irfansyal@gmail.com, [4]rizwanmcs09@gmail.com, [5]emranmehmood47@gmail.com

DOI: https://doi.org/10.5281/zenodo.24157988

**Keywords**
*convolutional neural network, batch normalization, dropout, L2 regularization, model verification, precision agriculture.*

**Article History**
*Received: 05 August 2026*
*Accepted: 19 October 2026*
*Published: 04 November 2026*

**Copyright @Author**
**Corresponding Author: ***
**Dr.Muhammad Ahmad Niaz**

**Abstract**
*Tomato leaf diseases are difficult to distinguish reliably when symptoms overlap or images are acquired under inconsistent conditions. This study presents a compact Sequential Dense Neural Network, or SDNN, for classifying nine tomato diseases and healthy foliage. The experimental record contains 8,043 images resized to 224 × 224 pixels. The network combines three convolutional blocks with batch normalization, max pooling, dense classification layers, dropout, and L2 regularization. It was trained with Adam and categorical cross-entropy for a maximum of 100 epochs. The recorded run reports 96.50% training accuracy, 96.67% validation accuracy, 96.71% precision, 96.68% recall, and a 96.69% F1-score. The accompanying figures also report an 18 MB model footprint and a training-time value of 2.5. These outcomes suggest a favorable accuracy–efficiency balance, but internal checks identify unresolved evidence issues. The displayed confusion matrix contains more observations than the stated dataset, all class-wise ROC curves show AUC = 1.00 despite non-zero classification errors, the training-time unit is missing, and the stated flattened dense design appears inconsistent with an 18 MB float32 model. The results are therefore retained and interpreted as reported outputs rather than independently reproduced measurements. A publication-grade rerun should use a frozen split, saved predictions, a fully specified averaging rule, an unregularized baseline, component ablations, and hardware-normalized timing. The model remains a promising screening architecture, but its quantitative claims require reproducible verification before they can support field deployment.*

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH
*International Peer-Reviewed Research Journal*
**ISSN (Online): 3106-7980 | ISSN (Print): 3106-7972**
**Volume 3 • Issue 7 • November 2026**

## INTRODUCTION

### 1.1 Background

Tomato production is exposed to fungal, bacterial, viral, and pest-related disorders whose visible symptoms may overlap. Late or inconsistent recognition can delay treatment and increase crop losses. This matters at both farm and food-system levels because plant disease remains a major constraint on yield and supply stability [1], [2], [3]. Visual scouting is indispensable, yet it is labor intensive and difficult to scale. Image-based screening can support the first stage of diagnosis by producing a consistent probability-ranked assessment while leaving treatment decisions to agronomic experts [4], [5]. Deep learning has shifted plant-disease recognition from manually selected color and texture descriptors to representations learned directly from images. Convolutional neural networks can capture lesion edges, color transitions, spot geometry, and higher-level symptom patterns. Early work on PlantVillage demonstrated the value of learned features, while later studies examined transfer learning, attention, synthetic augmentation, compact networks, and real-time detection [6], [7], [8], [9]. Tomato-focused systems have reported strong repository-based performance, but backgrounds, illumination, repeated images, geographic shift, mixed symptoms, and class imbalance continue to limit field transfer [10], [11], [12].

### 1.2 Research problem and objectives

The design challenge is not simply to maximize accuracy. A useful agricultural classifier must also remain small enough for local deployment, stable enough to train consistently, and transparent enough for its errors to be inspected. Large pretrained backbones provide rich features but may impose unnecessary storage and computation. A smaller custom network is easier to deploy, yet it can memorize dataset-specific patterns unless the data pipeline and regularization strategy are controlled. The study asks whether an SDNN that combines batch normalization with dropout and L2 regularization can distinguish ten tomato-leaf conditions while maintaining a compact resource profile. Its objectives are to develop the model, evaluate complementary predictive metrics, compare the recorded outcomes with benchmark CNNs, interpret class-level errors, and identify the evidence required before the architecture is treated as deployment ready.

### 1.3 Contributions

The study contributes a transparent three-block architecture, an accuracy–efficiency comparison, and a technical audit of the reported evidence. The audit is important because strong validation accuracy alone does not establish reproducibility. By retaining the recorded figures while separating observation from inference, the paper makes clear which conclusions are supported by the present run and which require a controlled rerun.

## 2 RELATED WORK

### 2.1 From handcrafted features to deep representations

Traditional systems segmented diseased regions and described them with color, shape, and texture statistics. Their interpretability was useful, but their performance depended heavily on illumination and preprocessing. CNNs reduced this dependence by learning hierarchical features from pixels. The modern design space builds on convolutional representation learning, deep image classification, batch normalization, dropout, residual and dense connectivity, lightweight depthwise convolution, multiscale processing, Adam optimization, data augmentation, and visual explanations [13], [14], [15], [16], [17], [18], [19], [20], [21], [22].

### 2.2 Tomato-leaf classification

Tomato research follows four broad directions. Custom CNNs provide task-specific baselines [23], [24], [25]. Transfer-learning studies reuse ResNet, VGG, DenseNet, MobileNet, and Inception representations [26], [27], [28], [29]. Attention, feature fusion, hybrid classification, and synthetic augmentation aim to separate visually similar conditions [30], [31], [32], [33],

[34]. Field-oriented work emphasizes localization, portable inference, and robustness outside curated repositories [35], [36], [37], [12].

**Table :1 Principal Research Direction And Unsolved Limitations**

| Research direction | Representative evidence | Strength | Persistent limitation |
|---|---|---|---|
| Custom CNNs | Agarwal et al. (2020a), Tm et al. (2018), Kibriya et al. (2021) | Task-specific features | Sensitivity to split and image domain |
| Transfer learning | Rangarajan et al. (2018), Nawaz et al. (2022), Attallah (2023) | Strong pretrained representation | Storage and computational cost |
| Attention and hybrid models | Karthik et al. (2020), Zhao et al. (2021), Sunil et al. (2023) | Focused and fused features | Additional tuning and complexity |
| Field and lightweight systems | Fuentes et al. (2017), Wang and Liu (2024), Jelali (2024) | Portability and localization | Domain shift and device constraints |

**Figure 1. Representative images from the ten tomato-leaf classes**

### 2.3 Research gap

The unresolved gap is an evidence problem as much as an architecture problem. Many papers report high accuracy on controlled images, while deployment requires stable performance across cameras, locations, seasons, and symptom stages. A compact model must therefore be evaluated with a leakage-safe split, repeated runs, calibrated probabilities, device-level measurements, and an explicit baseline. The proposed SDNN addresses the architectural side of this gap. The present technical audit addresses the reporting side.

## 3 MATERIAL AND METHODS

### 3.1 Dataset and preprocessing

The experimental record contains 8,043 JPEG images obtained mainly from the PlantVillage tomato subset and supplemented by open agricultural images. Ten labels were used. Nine represent Bacterial Spot, Early Blight, Late Blight, Leaf Mold, Septoria Leaf Spot, Spider Mites, Target Spot, Tomato Mosaic Virus, and Tomato Yellow Leaf Curl Virus. The tenth represents healthy foliage. Images were converted to RGB, resized to 224 × 224 pixels, and scaled to the interval [0, 1]. Labels were one-hot encoded. Training-time augmentation included rotation, horizontal and vertical flipping, zoom, rescaling, and brightness variation. The record states that stratified training, validation, and test partitions were created and that augmentation was restricted to the training pipeline. However, the split proportions, exact class counts, random seed, duplicate-removal procedure, and provenance of the supplementary images are not recorded. These items are required to reproduce the evaluation and should be fixed before the final experimental rerun.

### 3.2 SDNN architecture

A 224 × 224 × 3 image enters three convolutional blocks with 32, 64, and 128 filters. Each convolution uses a 3 × 3 kernel and ReLU activation. Batch normalization is applied to intermediate feature maps to stabilize optimization, and 2 × 2 max pooling reduces spatial resolution. The resulting features are flattened and passed to dense layers with 512 and 256 units. Dropout at 0.30 randomly deactivates units during fitting, while L2 regularization penalizes large weights. The final ten-unit softmax layer returns class probabilities. The three

stabilizing mechanisms should not be treated as interchangeable. Batch normalization controls activation scale during optimization [15]. Dropout reduces co-adaptation in the dense layers [16]. L2 regularization constrains weight magnitude. Their joint effect may improve generalization, but that contribution can only be demonstrated through an ablation study in which each component is removed under the same data split, seed, optimizer, and training budget.

**Figure 2. Recorded SDNN architecture and its three regularization components.**

### 3.3 Training procedure.

The recorded configuration used Python 3.9, TensorFlow 2.4, and Keras in Google Colab with a Tesla P100 GPU. Adam was configured with a learning rate of 0.001, categorical cross-entropy, and a batch size of 32. Training was allowed to continue for 100 epochs, with early stopping after ten validation epochs without improvement and checkpointing of the best validation state. A reproducible run should additionally report the random seeds, package build, CUDA and cuDNN versions, augmentation parameters, L2 coefficient, weight initialization, checkpoint selection metric, and whether the held-out test set remained untouched until the final evaluation.

### 3.4 Evaluation protocol

Accuracy, precision, recall, and F1-score were used to summarize predictive performance. For multiclass reporting, the averaging rule must be declared as macro, micro, or support weighted because the same predictions can yield different aggregate values. A confusion matrix describes class-specific errors, while one-versus-rest ROC curves evaluate ranking discrimination. Model size and execution time describe deployment cost only when serialization format, numerical precision, hardware, warm-up procedure, and timing unit are stated. The evidence hierarchy used in this paper separates three levels. A recorded result is a value appearing in the supplied experimental document. A verified result is recomputed from saved labels and probabilities on a frozen held-out set. A comparative conclusion requires all candidate models to be trained or measured under the same split, hardware, preprocessing, and reporting protocol. The present paper retains the recorded outputs but does not relabel them as independently verified.

### 3.5 Baseline and ablation design required for attribution

A normal-model control should preserve the same convolutional and dense structure but omit batch normalization, dropout, and L2 regularization. Four further ablations should add batch normalization alone, dropout alone, L2 alone, and the full combination. All variants should use the same frozen split, seed schedule, augmentation, optimizer, early-stopping rule, and epoch ceiling. Mean and standard deviation across at least five runs would show whether the full SDNN improves generalization consistently rather than benefiting from a favorable initialization. Because these controlled runs are absent from the supplied record, the current results support performance description but not causal attribution to any single component.

## 4. Results

### 4.1 Aggregate performance

The recorded run reports 96.50 training accuracy and 96.67 validation accuracy. Precision is 96.71, recall is 96.68%, and F1-score is 96.69%. The 0.17 percentage-point difference between training and validation accuracy is small, and the three aggregate error metrics differ by only 0.03 percentage points. Within the recorded run, this pattern is consistent with balanced classification rather than dominance by a single error type. The interpretation remains provisional because the averaging rule, class supports, and independently held-out test result are not provided.

Table 2. Reported Comparative Classification Performanc

| Model | Accuracy (%) | Precision (%) | Recall (%) | F1-score (%) |
|---|---|---|---|---|
| Sequential Dense NN | 96.67 | 96.71 | 96.68 | 96.69 |
| ResNet50 + CBAM | 94.50 | 94.30 | 94.10 | 94.20 |
| VGG16 + transfer learning | 93.20 | 93.00 | 92.80 | 92.90 |
| MobileNetV2 | 92.80 | 92.50 | 92.30 | 92.40 |
| DenseNet121 | 91.50 | 91.30 | 91.00 | 91.10 |
| InceptionV3 | 90.80 | 90.50 | 90.20 | 90.30 |

The SDNN is 2.17 percentage points above the recorded ResNet50 + CBAM accuracy and 5.87 points above InceptionV3. This is a descriptive ordering only. Without a shared split, repeated runs, and paired statistical analysis, the differences cannot establish that the SDNN is intrinsically superior to the benchmark architectures.

Figure 3. Recorded accuracy comparison between the SDNN and benchmark architectures.

**4.2 Learning behavior**

Training and validation accuracy rise rapidly, level off between approximately epochs 10 and 30, and increase again between epochs 30 and 45. The two curves remain close thereafter and finish in the 95% to 98% range. The absence of sustained divergence in the displayed run is consistent with controlled fitting, although the step-like changes suggest that the underlying history should be exported and checked against the learning-rate schedule and checkpoint log.

Figure 4. Recorded training and validation accuracy across 100 epochs.

Training and validation loss decline in parallel from roughly 1.0 to 1.1 at initialization to approximately 0.15 at the final epoch. Parallel loss curves do not show the usual visual signature of severe overfitting in this run. They do not, however, rule out leakage, duplicate images, or domain-specific memorization because those problems can affect training and validation similarly.

Figure 5. Recorded training and validation loss across 100 epochs.

**4.3 Class-level outcomes**

The displayed confusion matrix is strongly diagonal. Healthy leaves and Tomato Mosaic Virus each show 950 correct classifications, while Early Blight has the lowest diagonal value at 880. Errors are distributed among several visually related categories, including Early Blight and Late Blight. Such confusion is plausible because early lesions may share shape, color, and boundary characteristics.

Figure 6. Recorded confusion matrix for the ten tomato-leaf classes.

The matrix cannot yet be reconciled with the stated dataset size. Its ten diagonal values alone sum to 9,165, which already exceeds the complete dataset of 8,043 images, before off-diagonal errors are added. It therefore cannot represent the held-out partition described in Section 3.1. Possible causes include plotting demonstration counts, mixing augmented images into evaluation, using predictions from another run, or assigning the wrong matrix to the manuscript. The figure is preserved as part of the supplied record, but its counts must not be used for prevalence or class-wise performance claims until regenerated from the frozen evaluation set.

The ROC plot reports AUC = 1.00 for every class. Perfect ranking can coexist mathematically with a small number of thresholded classification errors, but ten identical perfect AUC values alongside a 96.67% aggregate accuracy warrant

verification. The curves should be regenerated from the saved one-hot labels and softmax probabilities using a documented one-versus-rest procedure. Per-class support, macro AUC, micro AUC, and confidence intervals should be reported.

### 4.4 Computational profile

**Table 3. Recorded Training-Time Values And Model Size**

| Model | Training-time value | Model size (MB) |
|---|---|---|
| SDNN | 2.5 | 18 |
| ResNet50 | 5.0 | 98 |
| VGG16 | 4.5 | 528 |
| MobileNetV2 | 3.2 | 14 |
| DenseNet201 | 6.0 | 77 |

**Figure 7. Recorded one-versus-rest ROC curves. The AUC values require reproduction from saved predictions.**

The supplied comparison records the SDNN at 18 MB and assigns it the lowest training-time value of 2.5. MobileNetV2 is smaller at 14 MB, while ResNet50, DenseNet201, and VGG16 are larger. The timing comparison is not publication ready because the unit, hardware, batch count, number of epochs, and measurement procedure are missing.

The model-size claim also conflicts with the written architecture. After three 2 × 2 pooling operations, a 224 × 224 input produces approximately 28 × 28 × 128 features. Flattening yields 100,352 values, and the following 512-unit dense layer alone contains about 51.4 million weights. Stored as float32, those weights require roughly 206 MB before the remaining layers are counted. An 18 MB model would require a materially different implementation, such as global average pooling, aggressive quantization, a much smaller tensor, or another unreported change. The architecture summary and the serialized file should therefore be reconciled before deployment claims are made.

## 5. Discussion

### 5.1 What the reported results do support

Taken at face value, the recorded accuracy, precision, recall, and F1-score describe a strong repository-level classifier. Their close agreement suggests that the reported model did not trade recall against precision excessively. The training and validation curves also track one another, which is compatible with stable optimization. Batch normalization, dropout, L2 regularization, and augmentation provide a technically plausible explanation for this behavior, but plausibility is not the same as demonstrated causation.

The comparison with pretrained networks is most useful as an engineering hypothesis. If the SDNN can reproduce 96.67% accuracy with a genuinely compact file, it could offer a practical balance between discrimination and local inference. This would matter for devices with limited storage or intermittent connectivity. The present tables, however, do not yet isolate architecture quality from different data splits, preprocessing, hyperparameter budgets, or hardware conditions.

### 5.2 Candidate-model mistakes and their consequences

The first mistake is the absence of a normal unregularized SDNN baseline. Without that control, the study cannot determine whether batch normalization, dropout, L2 regularization, augmentation, or ordinary convolutional learning produced the observed gain. The second mistake is incomplete dataset accounting. Missing split

counts and a confusion matrix larger than the entire dataset prevent independent reconstruction of the evaluation. The third mistake is using validation accuracy as the headline result without a separately reported test metric. Repeated model selection against the validation set can make that number optimistic.

The fourth mistake concerns metric provenance. AUC = 1.00 for every class is not automatically impossible, but it is insufficiently supported when the same evaluation reports numerous thresholded errors. The fifth is an incomplete computational comparison. A time value without units and a model size inconsistent with the stated flattening operation cannot support an efficiency claim. The sixth is the lack of repeated-run uncertainty. A single seed can favor one candidate, especially when early stopping and augmentation are involved.

These issues do not require the recorded results or graphs to be discarded. They require the claims to be narrowed. The figures are evidence of what the experimental record reported, while the text identifies which values need regeneration. This distinction protects the scientific contribution from being overstated and provides a clear path to verification.

### 5.3 What should be done before final quantitative claims

The corrected experiment should begin with file-level deduplication before splitting so that near-identical leaves do not appear in multiple subsets. A fixed, stratified split should then be saved with image identifiers and class counts. Augmentation must remain inside the training loader. The unregularized baseline and all ablations should be trained across the same seed set. The best checkpoint from each run should be evaluated exactly once on the held-out test set.

For every test image, the true label, predicted label, and ten softmax probabilities should be exported. Those records should generate the confusion matrix, class-wise precision and recall, macro and weighted F1, ROC curves, AUC values, and confidence intervals. The model summary should be archived with total and trainable parameter counts. File size should be measured from the submitted serialization format. Training duration should be reported in minutes or hours on named hardware, while inference latency should use warm-up iterations and a fixed batch size on the intended device.

Finally, the statistical question should match the claim. If the claim concerns generalization, repeated-run test performance and an external field set are required. If it concerns efficiency, all models must be trained and timed under the same environment. If it concerns the benefit of dual regularization, the baseline and ablation comparisons are essential. This alignment between claim and evidence is more important than adding further architectures to the comparison.

### 5.4 Relation to prior work

The recorded validation accuracy lies within the strong range reported for tomato-leaf classification, but direct ranking across studies is unreliable because datasets and protocols differ. Custom CNNs and transfer-learning systems established the feasibility of learned representations [38], [26], [39], [40]. Later studies used synthetic images, attention, feature fusion, and hybrid learners to improve discrimination [32], [41], [42], [34]. Recent work increasingly emphasizes compact inference, explainability, real-field imagery, and portable monitoring [43], [37], [44], [45]. The SDNN is best positioned within this literature as a transparent candidate architecture rather than a universal state-of-the-art model. Its value lies in the possibility of achieving useful accuracy with a simpler design. That proposition becomes publishable only when the model file, parameter count, split, saved predictions, and timing protocol agree with the manuscript.

### 5.5 Practical implications and limitations

A verified SDNN could support mobile or edge-based screening. A grower could photograph a leaf and receive a ranked suggestion with a confidence score. Such output should assist rather than replace agronomic judgment because

nutrient stress, environmental injury, and mixed infection can resemble disease. A deployable system would need image-quality checks, uncertainty thresholds, referral of ambiguous cases, model updates, and monitoring for domain drift.

The current evidence is limited by predominantly controlled imagery, missing split details, no external geographic test, no repeated-run intervals, unresolved figure provenance, and no hardware-normalized latency. The perfect ROC display and the inconsistency between the stated architecture and model size are especially important. Until corrected, the conclusions apply only to the supplied experimental record.

## 6. Conclusion

This study presents a three-block SDNN for classifying nine tomato-leaf diseases and healthy foliage. The supplied run reports 96.50% training accuracy, 96.67% validation accuracy, 96.71% precision, 96.68% recall, and 96.69% F1-score. Its learning curves suggest stable fitting, and the intended combination of batch normalization, dropout, and L2 regularization is technically coherent.

The same record contains inconsistencies that prevent unconditional acceptance of the quantitative claims. The confusion matrix exceeds the stated dataset size, all ROC curves show AUC = 1.00, the time unit is absent, and the dense-layer parameter count appears incompatible with an 18 MB float32 model. The appropriate conclusion is therefore measured. The SDNN is a promising and testable candidate for resource-conscious screening, while its reported performance must be reproduced through a frozen split, saved predictions, a normal-model baseline, component ablations, repeated runs, and device-specific measurements. Completing these steps would convert the present strong preliminary result into defensible evidence for publication and field evaluation.

## 7 Declarations

Data availability. The study uses publicly accessible tomato-leaf images obtained mainly from PlantVillage and supplemented by open agricultural image collections, as described in Section 3.1. The paper introduces no newly collected observations.

Ethics statement. The work analyzes plant images and involves no human participants, personal data, or live animals. Ethics approval and informed consent are not applicable.

Conflict of interest. The author declares no conflict of interest.

Funding. No external funding was reported for this study.

## REFERENCE

R. N. Strange and P. R. Scott, "Plant disease: A threat to global food security," Annual Review of Phytopathology, vol. 43, p. 83–116, 2005.

S. Savary, A. Ficke, J. N. Aubertot and C. Hollier, "Crop losses due to diseases and their implications for global food production losses and food security," Food Security, vol. 4, p. 519–537, 2012.

S. Panno and e. al., "A review of the most common and economically important diseases that undermine the cultivation of tomato crop in the Mediterranean Basin," Agronomy, vol. 11, no. 11, 2021.

F. Martinelli and e. al., "Advanced methods of plant disease detection. A review," Agronomy for Sustainable Development, vol. 35, pp. 1-25, 2015.

C. H. Bock, G. H. Poole, P. E. Parker and T. R. Gottwald, "Plant disease severity estimated visually, by digital photography and image analysis, and by hyperspectral imaging," Critical Reviews in Plant Sciences, vol. 29, no. 2, p. 59–107, 2010.

K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," Computers and Electronics in Agriculture, vol. 145, p. 311–318, 2018.

E. C. Too, L. Yujian, S. Njuki and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," Computers and Electronics in Agriculture, vol. 161, p. 272–279, 2019.

M. H. Saleem, S. Khanchi, J. Potgieter and K. M. Arif, "Image-based plant disease identification by deep learning meta-architectures," Plant, vol. 9, no. 11, 2020.

J. G. A. Barbedo, "Factors influencing the use of deep learning for plant disease recognition," Biosystems Engineering, vol. 172, p. 84–91, 2018.

M. Gehlot, R. K. Saxena and G. C. Gandhi, "Tomato-Village: A dataset for end-to-end tomato disease detection in a real-world environment," Multimedia Systems, vol. 29, 3305–3328.

M. Jelali, "Deep learning networks-based tomato disease and pest detection: A first review of research studies using real field datasets," Frontiers in Plant Science, vol. 15, 2024.

Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, p. 2278–2324, 1998.

A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems, 2012, p. 1097–1105.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception architecture for computer vision," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," Journal of Machine Learning Research, vol. 15, p. 1929–1958, 2014.

K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

G. Huang, Z. Liu, L. van der Maaten and K. Q. Weinberger, "Densely connected convolutional networks," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv preprint, 2017.

D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in International Conference on Learning Representations (ICLR), 2015.

C. Short and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," Journal of Big Data, vol. 6, 2019.

R. R. Selvaraju and e. al., "Grad-CAM: Visual explanations from deep networks via gradient-based localization," International Journal of Computer Vision, vol. 128, p. 336–359, 2020.

M. Agarwal, S. K. Gupta and K. K. Biswas, "Development of efficient CNN model for tomato crop disease identification," Sustainable Computing: Informatics and Systems, vol. 28, 2020.

P. Tm, A. Pranathi, K. SaiAshritha, N. B. Chittaragi and S. G. Koolagudi, "Tomato leaf disease detection using convolutional neural networks," in 11th International Conference on Contemporary Computing (IC3), 2018.

H. Kibriya, R. Rafique, W. Ahmad and S. M. Adnan, "Tomato leaf disease detection using convolution neural network," in 18th International Bhurban Conference on Applied Sciences and Technologies (IBCAST), 2021.

A. K. Rangarajan, R. Purushothaman and A. Ramesh, "Tomato crop disease classification using pre-trained deep learning algorithm," Procedia Computer Science, vol. 133, p. 1040–1047, 2018.

S. Z. Zaki, M. A. Zulkifley, M. M. Stofa, N. A. Kamari and N. A. Mohamed, "Classification of tomato leaf diseases using MobileNet V2," IAES International Journal of Artificial Intelligence, vol. 9, no. 2, p. 290–296, 2020.

M. Nawaz and e. al., "A robust deep learning approach for tomato plant leaf disease localization and classification," Scientific Reports, vol. 12, 2022.

O. Attallah, "Tomato leaf disease classification via compact convolutional neural networks with transfer learning and feature selection," Horticulturae, vol. 9, no. 2, 2023.

R. Karthik, M. Hariharan, S. Anand, P. Mathikshara, A. Johnson and R. Menaka, "Attention embedded residual CNN for disease detection in tomato leaves," Computing, vol. 86, 2020.

Q. Wu, Y. Chen and J. Meng, "DCGAN-based data augmentation for tomato leaf disease identification," DCGAN-based data augmentation for tomato leaf disease identification, vol. 8, p. 98716–98728, 2020.

A. Abbas, S. Jain, M. Gour and S. Vankudothu, "Tomato plant disease detection using transfer learning with C-GAN synthetic images," Computers and Electronics in Agriculture, vol. 187, 2021.

S. Zhao, Y. Peng, J. Liu and S. Wu, "Tomato leaf disease diagnosis based on improved convolution neural network by attention module," Agriculture, vol. 11, no. 7, 2021.

C. K. Sunil, C. D. Jaidhar and N. Patil, "Tomato plant disease classification using multilevel feature fusion with adaptive channel spatial and pixel attention mechanism," Expert Systems with Applications, vol. 228, 2023.

A. Fuentes, S. Yoon, S. C. Kim and D. S. Park, "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition," Sensors, vol. 17, no. 9, 2017.

J. Liu and X. Wang, "Tomato diseases and pests detection based on improved YOLO v3 convolutional neural network," Frontiers in Plant Science, vol. 11, 2020.

X. Wang and J. Liu, "An efficient deep learning model for tomato disease detection," Plant Methods, vol. 20, 2024.

M. Brahimi, K. Boukhalfa and A. Moussaoui, "Deep learning for tomato diseases: Classification and symptoms visualization," Applied Artificial Intelligence, vol. 31, no. 4, p. 299–315, 2017.

M. Agarwal, A. Singh, S. Arjaria, A. Sinha and S. Gupta, "ToLeD: Tomato leaf disease detection using convolution neural network," Procedia Computer Science, vol. 167, p. 293–301, 2020.

V. Maeda-Gutiérrez and e. al., "Comparison of convolutional neural network architectures for classification of tomato plant diseases," Applied Sciences, vol. 10, no. 14, 2020.

M. S. A. M. Al-Gaashani, F. Shang, M. S. A. Muthanna, M. Khayyat and A. A. Abd El-Latif, "Tomato leaf disease classification by exploiting transfer learning and feature concatenation," IET Image Processing, vol. 16, p. 913–925, 2022.

M. S. Islam and e. al., "Multimodal hybrid deep learning approach to detect tomato leaf disease using attention-based dilated convolution feature extractor with logistic regression classification," Sensors, vol. 22, no. 16, 2022.

International Journal of Advanced Research

N. Nigar, H. M. Faisal, M. Umer, O. Oki and J. M. Lukose, "Improving plant disease classification with deep-learning-based prediction model using explainable artificial intelligence," IEEE Access, vol. 12, p. 100005–10001, 2024.

A. Batool and e. al., "An enhanced lightweight T-Net architecture based on convolutional neural network for tomato plant leaf disease classification," PeerJ Computer Science, vol. 10, 2024.

H. Ghosh and e. al., "Advanced neural network architectures for tomato leaf disease diagnosis in precision agriculture," Discover Sustainability, vol. 6, 2025.

M. Kaur and e. al., "Visual-language transformer-based tomato leaf disease detection for portable greenhouse monitoring device," Plant Methods, vol. 21, 2025.

International Journal of Advanced Research