

Adaptive Strategies for Multi-Class Disease Detection in *Azadirachta Indica* Using Deep Learning with Few-Shot Adaptation (DL-FSA)

H. A. Vidya

Department of CSE, BMS Institute of Technology and Management, Visvesvaraya Technological University, Belagavi, India
vidya.ha@gmail.com (corresponding author)

M. S. Narasimha Murthy

Department of Information Science and Engineering, BMS Institute of Technology and Management, Visvesvaraya Technological University, Belagavi, India
narasimhamurthys@bmsit.in

A. Muthu Kumar

Forest Protection Division, ICFRE-IWST, Bangalore, India
iwst.muthukumar@gmail.com

Received: 24 October 2025 | Revised: 10 November 2025 | Accepted: 21 November 2025

Licensed under a CC-BY 4.0 license | Copyright (c) by the authors | DOI: <https://doi.org/10.48084/etasr.15752>

ABSTRACT

These days, plant disease detection is a vital aspect because plants are the primary sources of nutrients for living organisms. Medicinal plants provide natural healing properties. *Azadirachta indica* (neem) is a versatile medicinal with numerous benefits, supporting skin health and immune function and offering wellness from root to leaf. However, it is susceptible to various diseases, making the identification and characterization of these threats essential to protect its invaluable contributions. The dataset for this study was collected in real time, and a few healthy neem samples from a public dataset were used for the healthy class. The study employs Generative Adversarial Networks (GANs) to generate synthetic dataset images. The dataset contains seven classes, one healthy class and six diseased classes: *Alternaria*, bacterial infection, defoliator, Dieback, irregular yellowing, and leaf blotch. The main objective of this work is to classify multiple diseases in *Azadirachta indica* (neem) leaves using a hybrid model, Deep Learning (DL) with Few-Shot Adaptation (DL-FSA), which integrates Few-Shot Learning (FSL), Convolutional Neural Networks (CNNs), Artificial Neural Networks (ANNs), and Deep Neural Networks (DNNs). The hybrid model also employs a weighted average fusion technique to combine the probabilities from FSL and DNN to produce the final classification output. The model achieves an accuracy of 96% with a limited dataset, outperforming the baseline FSL model by 1.43%, demonstrating enhanced generalization and classification efficiency for neem leaf disease detection.

Keywords-disease detection; *Azadirachta indica*; data augmentation; Generative Adversarial Network (GAN); Few-Shot Learning (FSL); Convolutional Neural Network (CNN); Artificial Neural Network (ANN); Deep Neural Network (DNN)

I. INTRODUCTION

Disease refers to abnormal functioning in humans, animals, and plants. Plants are the source of food and habitat for most living organisms on Earth. Some plants possess medicinal properties and are categorized as medicinal plants, which are used in treating various ailments. Many modern medicines rely on plant extracts, and neem is one of the medicinal plants having numerous medicinal benefits compared to others. The scientific name of neem is *Azadirachta indica*. It has diverse

medicinal properties, including antibiotic, antiviral, antibacterial, anticancer, antidiabetic, and antiseptic effects [1], and, in addition to these therapeutic values, it has been utilized in environmentally friendly biofertilizers [2]. Neem is affected by numerous diseases, including bacterial infection, yellowing, *Alternaria*, Dieback, leaf spot, termite, leaf blotch, leaf blight, defoliator, and powdery mildew [3]. Among these, Dieback is the major disease severely affecting neem plantations in India, caused by *Phomopsis azadirachtae* [4]. As a result, a disease incidence can reduce crop yield, decrease the quality of neem

and its byproducts, and thus compromise the efficiency of medicinal products derived from neem, ultimately impacting the economy.

Advances in Artificial Intelligence (AI), particularly in Machine Learning (ML) and Deep Learning (DL), have enabled broad applications across modern software ecosystems. Plant disease detection is the major area where DL models are applied to detect, classify, and predict the type of diseases. DL models necessitate a significant amount of data for training the models and obtain accurate results, and data availability may be a bottleneck in many cases [5]. In an effort to mitigate this challenge, a hybrid DL model combining Few-Shot Learning (FSL) with DL models is proposed in this study. The hybrid model uses two sub-models: FSL with meta-learning and metric learning, and a DL module combining Convolutional Neural Networks (CNNs), Artificial Neural Networks (ANNs), and Deep Neural Networks (DNNs). The class probabilities from both FSL and DNN are combined using weighted average fusion to produce the final output.

In AI, Generative Adversarial Networks (GANs) excel at producing realistic images from random noise. This study also uses GANs for synthetic data generation, similar to previous work [6]. The GAN architecture is delineated by two key units: the generator and the discriminator. The generator synthesizes noise distributions and maps them into a realistic image. The discriminator evaluates whether the images produced by the generator are genuine or synthetic, and provides feedback to the generator, guiding it to improve image generation [7, 8].

Numerous studies have explored DL applications. Traditional DL generative models faced challenges in estimating model parameters and activation functions, often performing better in the discriminator than in the generator. GANs were developed to overcome these difficulties [9]. The GAN was developed by Goodfellow in 2014 and has many variations, such as DCGAN, Cycle GAN, Conditional GANs, and Style GAN, as explained and compared in [10]. Authors in [11] demonstrated the use of GANs for generating synthetic brain images for detecting Alzheimer's disease, where the generator used a multilayer perceptron and the discriminator used a CNN architecture.

FSL is another emerging area that addresses the issue of data scarcity for ML and DL models [12]. FSL uses support and query datasets containing a few samples of data for training and evaluation. The query set includes unseen samples to evaluate the efficiency of the model. This approach is often described as n-shot, k-way learning, where n-shots represent the number of samples in support and query sets, and k-ways represent the number of classes [13, 14]. Authors in [15] applied FSL for plant disease detection using CNN for feature extraction on the PlantVillage dataset and a Support Vector Machine (SVM) classifier for six disease categories. Various FSL types exist, such as prototypical networks, meta-learning, metric learning, and Siamese networks. Existing studies often use a single approach, e.g., [16] uses meta-learning, [17] uses metric learning, and [18] combines meta-learning and metric learning for improved results. Authors in [19] explored performance enhancements in FSL models. While typical FSL models achieve up to 90% accuracy, the present work

combines meta-learning and metric learning to enhance FSL performance.

DL is widely applied in AI, particularly for image processing [20]. CNNs extract features automatically and classify data using layered architectures [21]. CNN performance can improve when combined with other models, e.g., with the Long Short-Term Memory (LSTM) model and the Recurrent Neural Network (RNN) [22]. ANN provides the foundation for advanced DL models, and its integration with other models can maximize its performance [23]. The combination of DNN and CNN achieved 86% accuracy in predicting plant diseases [24]. Studies indicate that CNN can achieve up to 99% accuracy, ANN up to 95% on simple datasets, and DNN up to 98%, outperforming ANN. Numerous pre-trained models based on CNN are also widely used in many DL-based applications [25]. Fusing multiple models is a new approach to maximize performance. Weighted-average fusion combines outputs from multiple models to enhance overall efficiency, with weights determined by model accuracy and similarity [26, 27].

Earlier studies using CNN and DNN achieved high accuracy but required large and balanced datasets. FSL methods learn from limited data yet often suffer from weak feature generalization and sensitivity to class imbalance. Transformer- and attention-based models have recently gained popularity for plant disease detection; however, they demand high computation and extensive training data. The proposed DL-FSA model with weighted fusion addresses these gaps by combining complementary features from multiple networks, enhancing generalization under data scarcity while maintaining high accuracy with moderate computational cost.

In this work, disease classification in *Azadirachta indica* leaves is performed by combining FSL with CNN, ANN, and DNN models using a small real-time dataset. The contributions of this work are as follows.

- Collection of real-time data for *Azadirachta indica*.
- Validation of the real-time dataset with the help of plant pathologists.
- Generation of synthetic data using a GAN for the collected real-time data.
- Design of a hybrid DL model that integrates FSL with CNNs, ANNs, and DNNs, fusing the outputs from FSL and DNN with weighted average fusion to classify diseases in *Azadirachta indica* (neem) leaves.
- Demonstration of superior performance of the hybrid model compared to existing FSL models and their variations.
- Quantitative performance comparison of the hybrid model with other DL models.

The proposed hybrid model improves classification accuracy for various diseases in *Azadirachta indica* using a small dataset. It is scalable and can be used in the early diagnosis of plant diseases, reducing crop loss.

II. PROPOSED METHOD

A. Materials and Methods

The experiments were conducted in Google Colab using a GPU-accelerated environment (NVIDIA Tesla T4/P100, 16 GB RAM) running Ubuntu 20.04 LTS. The setup used Python 3.10 with TensorFlow and Pytorch frameworks for model training and evaluation.

The dataset consists of healthy and diseased leaf images of *Azadirachta indica*, classified into seven different categories, one healthy and six types of diseases: Alternaria, bacterial infection, defoliator, Dieback, irregular yellowing, and leaf blotch. The data were collected in real time from the local areas of Tiptur, Chikkasandra Kaval forest area of Tumkur, and Mysuru, Karnataka, India, as documented by the location images taken using a GPS camera, shown in Figure 1.



Fig. 1. Location of real-time data collection in the Chikkasandra Kaval forest area, Tumkur, Karnataka, India.

A few images that belong to the healthy class were taken from the medicinal plant datasets publicly available on the Mendeley platform [28, 29]. The number of real-time samples within each category is as follows: 880 healthy, 300 bacterial infection, 340 defoliator, 325 Alternaria, 500 Dieback, 300 irregular yellowing, and 440 leaf blotch. Real-time data collection takes more time and effort and the data samples are often not sufficient for training the DL models. Hence, we employed an image augmentation technique to increase the number of samples per category and balance the dataset. We generated synthetic images from the real-time data using GANs, as shown in Figure 2.

To handle domain differences, GAN-based augmentation was applied to all classes of *Azadirachta indica*. Normalization, including contrast and size adjustment, was applied to maintain uniform visual quality across all samples. The experiment used a generator model that was trained up to 8,000 epochs and generated a batch of 25 images, arranged in a 5x5 grid. GANs typically generate images with pixel values in the range [-1, 1], but for visualization purposes, these values were rescaled to [0, 1]. The scaling made the images suitable to display using the standard Matplotlib library. After augmentation, the total number of images per class was: 1,821 for healthy, 1,538 for Alternaria, 1,559 for defoliator, 1,859 for Dieback, 1,152 for irregular yellowing, 816 for bacterial infection, and 1,336 for leaf blotch, totaling 10,081 images. The dataset was partitioned

into 80% training and 20% validation data. Although cross-validation was not used, random shuffling was applied to reduce sampling bias.

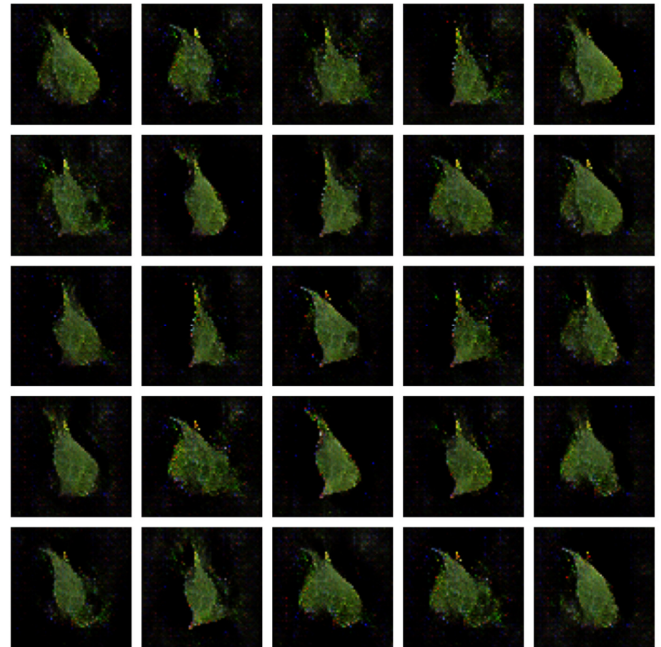


Fig. 2. Sample augmented neem leaf images generated using a GAN.

Table I presents the class-wise image counts before and after GAN-based augmentation. The augmentation step balanced the dataset, increased sample diversity, reduced bias, and contributed to improved performance.

TABLE I. SUMMARY OF THE NEEM LEAF IMAGE DATASET SHOWING THE NUMBER OF SAMPLES BEFORE AND AFTER AUGMENTATION

Class	Original images	After augmentation
Healthy	880	1,821
Alternaria	325	1,538
Bacterial infection	300	816
Defoliator	340	1,559
Dieback	500	1,859
Irregular yellowing	300	1,152
Leaf blotch	440	1,336
Total	3,085	10,081

B. Methodology

The conceptual framework of the proposed work is illustrated in Figure 3. Input neem images are augmented using GAN to enhance the quantity of images within each class. The generated synthetic data are normalized and resized in the data preparation step, and the dataset is then split into training and validation sets. The hybrid model learns from the training data and is evaluated with the validation set. If the input is correctly classified, the process completes.

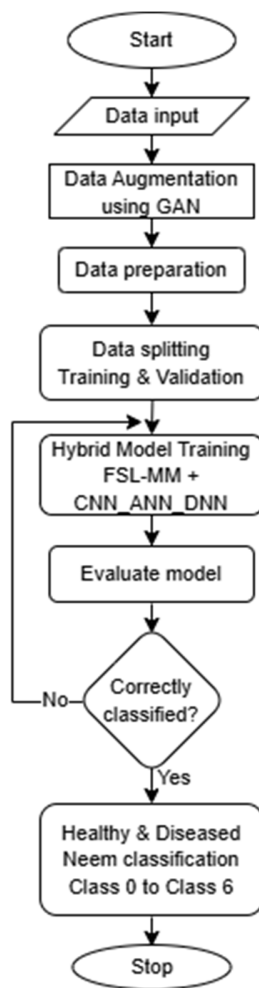


Fig. 3. Flowchart of the proposed DL-FSA approach for neem leaf disease classification.

The hybrid model combines the strengths of the FSL module with meta-learning and metric learning, and the DL module with CNN, ANN, and DNN networks. It is designed to extract features and classify them effectively by combining the probabilities of both FSL and DNN using weighted average fusion. The model successfully classifies inputs into the seven target classes considered in the experiment. The architecture of the proposed hybrid model is depicted in Figure 4.

Figure 5 shows the detailed view of the FSL module. The training dataset is divided into a support set and a query set. The support set is intended to train the model, and the query set is used for classification. The FSL module is designed for $n_shots=5$ (support set samples per class), $n_query=5$ (query set samples per class), and $n_classes=7$ (number of classes). These sets are processed in batches during the FSL training process.

In meta-learning the model learns to generalize to new tasks with limited data, whereas in metric learning, it calculates the similarity between support and query image features. The model then classifies the query images to seven classes based on learned feature distances. The FSL module consists of a

feature extractor, flattened layer, fully connected layer, meta-learning and metric learning modules, and a final classifier. The input image dimensions were $224 \times 224 \times 3$. ResNet-50 performs feature extracts and produces a $7 \times 7 \times 2048$ feature map. The final layer of ResNet-50 was removed as it is used as a feature extractor here. The flattened layer converts the 3D feature map into a 1D feature vector of size 2048. The fully connected layer reduces the dimensionality of this feature vector from 2048 to 256 by applying linear transformation. This reduces the computational complexity, reduces overfitting, and improves the overall performance of the meta-learning process.

During the inner loop of the meta-learning process, the support set parameters are updated using the Stochastic Gradient Descent (SGD) optimizer, whereas during the outer loop, the meta-parameters of the query set images are updated using the Adam optimizer. The metric learning module employs the Euclidean distance function to measure the similarity between the input samples [16] and is guided by a contrastive cross-entropy loss function for discriminative feature learning. The final classification layer produces the probabilities for the seven target classes, reducing dimensionality from 256 to 7.

Figure 6 shows the detailed view of the DL module. Its functional specifications are as follows:

- Input layer: It serves as the starting point for the model, where image data are fed for processing. The input corresponds to 300×300 resolution RGB images.
- CNN: Extracts spatial features and detects patterns such as textures, edges, and shapes. It consists of a three-layer convolutional block with filter sizes set to 32, 64, and 128, respectively. Each convolution operation uses a 3×3 kernel size and ReLU activation for non-linearity. The increasing filter capacity allows the model to capture diverse spectrum of image features. Spatial downsampling of feature maps is accomplished through the application of 2×2 max pooling layers subsequent to each convolutional layer. This performs spatial reduction, making the model computationally efficient and reducing overfitting.
- ANN: A feature learning block containing two dense layers with 512 and 256 neurons, using ReLU activation for high-level feature learning. These layers help in learning high-level abstractions and representations from the features. It also contains dropout layers to prevent overfitting with a rate of 0.5.
- DNN: A decision-making block that performs classification based on learned features. A 128-neuron dense layer with ReLU refines the ANN-learned features.
- Output layer: The classification task is completed by a dense layer with 7 output units, each corresponding to a class. Softmax activation is used to produce a probability vector over the 7 classes, making the model suitable for multi-class categorization.

The summary of the DL module used in the DL-FSA model is presented in Table II.

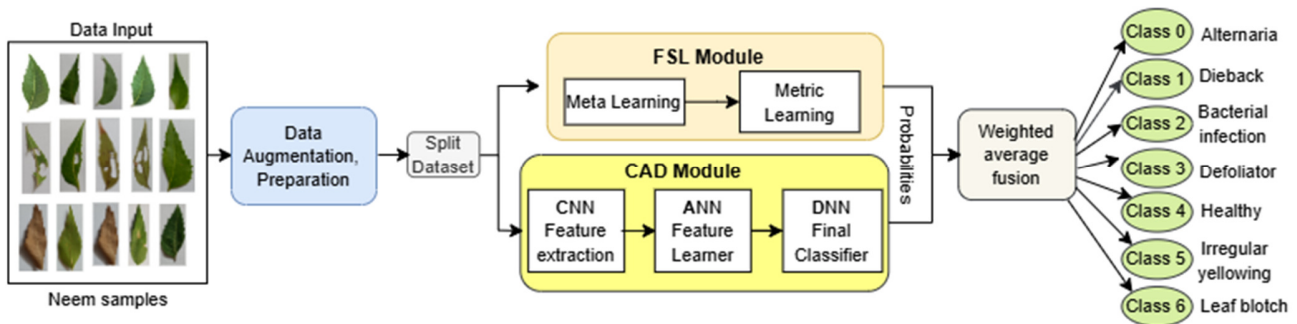


Fig. 4. Architecture of the proposed hybrid DL-FSA model integrating FSL with CNN, ANN, and DNN sub-networks and weighted fusion.

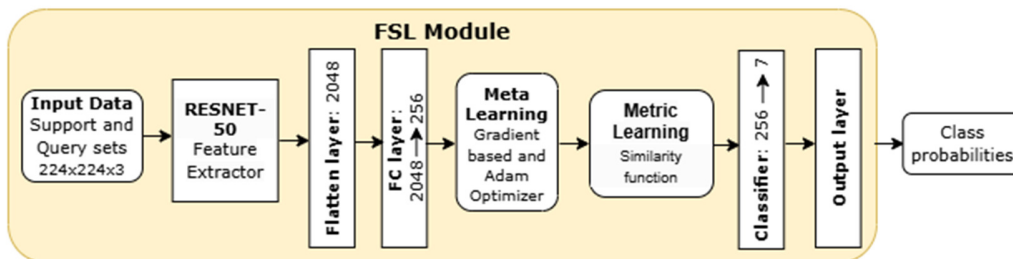


Fig. 5. FSL module of the proposed hybrid DL-FSA model with meta-learning and metric learning methods.

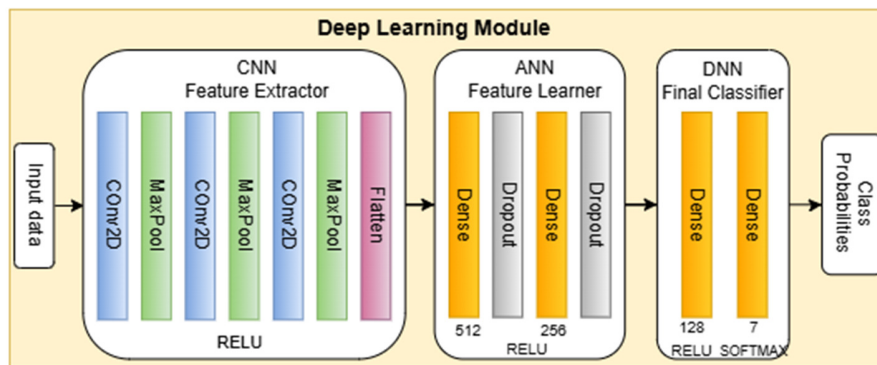


Fig. 6. DL Module of the proposed hybrid DL-FSA model.

TABLE II. SUMMARY OF THE DL MODULE IN THE HYBRID DL-FSA MODEL

Layer (type)	Output Shape	Param #
Conv2d (Conv2d)	(None, 32, 222, 222)	896
MaxPool2d (MaxPool2d)	(None, 32, 111, 111)	0
Conv2d_1 (Conv2d)	(None, 64, 109, 109)	18,496
MaxPool2d_1 (MaxPool2d)	(None, 64, 54, 54)	0
Conv2d_2 (Conv2d)	(None, 128, 52, 52)	73,856
MaxPool2d_2 (MaxPool2d)	(None, 128, 26, 26)	0
Flatten (Flatten)	(None, 128 * 26 * 26)	0
Dense_1 (Linear)	(None, 512)	1,049,088
Dropout_1 (Dropout)	(None, 512)	0
Dense_2 (Linear)	(None, 256)	131,328
Dropout_2 (Dropout)	(None, 256)	0
Dense_3 (Linear)	(None, 128)	105,062
Dense_4 (Linear)	(None, 7)	257
Total params		1,274,921
Trainable params		1,274,921
Non-trainable params		0

To expedite learning, the model's parameters are tuned via the Adam optimizer, whereas categorical cross-entropy quantifies the divergence between predicted and actual class distributions. Training is performed in batches of 32 images over 20 epochs, with callbacks applied to optimize training time and prevent overfitting. During compilation, the model tracks classification accuracy for both training and validation sets. The class probabilities from both the FSL and DL modules are combined using the weighted average fusion technique to get the final classification output. Both modules of the hybrid architecture undergo training utilizing the training dataset. The validation dataset is employed to optimize the fusion weights and estimate final performance. The optimal weights are selected by iterating over the validation set. For each batch, the probabilities $P_{fsl}[b, c]$, and $P_{dnn}[b, c]$ are calculated, where $P_{fsl}[b, c]$ represents the probability of image b categorized as class c according to the FSL model, and $P_{dnn}[b, c]$ represents the probability of image b categorized as class c according to the DL module. These probabilities are combined by considering the weighted averages w_{fsl} and

w_{ann} , and evaluation metrics are computed by comparing the predictions to the actual labels in the validation set. Weights are adjusted according to the SGD optimization procedure, which is a more sophisticated approach than manual or grid search. Our findings are consistent with the observations made in [30].

FSL significantly enhances the capabilities of CNNs, ANNs, and DNNs within the hybrid model by enabling effective feature extraction, representation learning, and robust classification from limited data. This synergy allows the model to perform well in scenarios where traditional DL approaches might struggle due to data scarcity. The hybrid model's classification capabilities are significantly augmented by integrating FSL, resulting in improved efficiency, adaptability, and performance, demonstrated through higher reliability and accuracy. In FSL, for each class k and support images per class S , the prototype p_k is calculated as:

$$p_k = \left(\frac{1}{S}\right) * \sum_{s=1}^S f(x_{\{k,s\}}) \quad (1)$$

where:

- p_k denotes the prototype vector representing class k .
- S is the number of support images in class k .
- $f(x_{\{k,s\}})$ is the feature embedding of the s -th support image of class k .

The Euclidean distance between a query image embedding and each class prototype is calculated by:

$$d(f(x_q), p_k) = \text{sqrt}\left(\sum_{i=1}^d (f(x_q)_i - p_{\{k,i\}})^2\right) \quad (2)$$

where:

- $d(f(x_q), p_k)$ is the Euclidean distance between the query sample feature $f(x_q)$ and the prototype vector p_k of class k .
- $f(x_q)_i$ and $p_{\{k,i\}}$ are the i -th elements of the respective vectors.
- d is the total number of feature dimensions.

The probability output of the FSL module, representing the likelihood that a query image belongs to class k , is calculated using (3), which is the ratio of the dissimilarity between the query image and the prototype of class k to the total dissimilarity across all class prototypes.

$$P_{fst}[q, k] = \frac{\exp(-d(f(x_q), p_k))}{\sum_{j=1}^K \exp(-d(f(x_q), p_j))} \quad (3)$$

where:

- $P_{fst}[q, k]$ represents the predicted probability that query sample x_q belongs to class k .
- $d(f(x_q), p_k)$ is the Euclidean distance between the query embedding and the class prototype.
- $f(x_q)$ is the feature representation of the query image.

- K is the total number of classes.

The formula for feature extraction using CNN is given by:

$$\text{Output}[i, j, k] = \sum_m \sum_n \sum_c \text{Input}[i + m - 1, j + n - 1, c] * \text{Kernel}[m, n, c, k] + \text{Bias}[k] \quad (4)$$

where:

- $\text{Output}[i, j, k]$ is the k -th feature response at spatial index (i, j) .
- $\text{Input}[i + m - 1, j + n - 1, c]$ denotes the activation of the c -th input channel at position $(i + m - 1, j + n - 1)$.
- $\text{Kernel}[m, n, c, k]$ represents the weight of the kernel at spatial location (m, n) for the c -th input channel and k -th output channel.
- $\text{Bias}[k]$ is the bias for the k -th output channel.
- $\sum_m \sum_n \sum_c$ is the summation of the kernel dimensions and input channels.

The core of the ANN layer uses the ReLU activation function given by:

$$y = \text{ReLU}(W * x + b) \quad (5)$$

where:

- y denotes the output activation of the ANN layer.
- x denotes the input activation to the ANN layer.
- W denotes the weight matrix.
- b is the bias vector.
- $\text{ReLU}(z) = \max(0, z)$, with $z = W * x + b$.

The dense layer output from the final linear layer has the shape (batch size, 7), corresponding to the seven class logits z . The softmax activation function is applied to convert these logits into probabilities using (6):

$$p_i = \frac{\exp(z_j)}{\sum_j \exp(z_j)} \quad (6)$$

where:

- p_i is the probability for class i .
- z_i is the logit for class i .
- $\sum_j(\cdot)$ denotes the summation of all classes j .

The output from the DNN layer is the likelihood that image b belongs to class c , and it is calculated by the softmax function as:

$$P_{ann}[b, c] = \frac{\exp(z[b, c])}{\sum_{j=0}^6 \exp(z[b, j])} \quad (7)$$

where:

- $\exp(\cdot)$ is the exponential function to ensure that the values are positive.
- $z[b, c]$ is the logit for image b and class c .

- $z[b, j]$ represents the logits for image b and other classes j .

Weighted average fusion combines the predictions from the FSL and DNN modules using weights that are dynamically adjusted based on the performance of each model. The process involves calculating a fused probability, defining a loss function, performing gradient computation for the loss, updating the weights, and ensuring that the weights remain valid (sum up to 1). This method enhances performance by leveraging the strengths of multiple approaches. Empirical tuning based on validation accuracy yielded an optimal ratio of 0.4 for FSL and 0.6 for DNN in the fusion block, with internal weights of 0.5:0.3:0.2 for CNN, DNN, and ANN, respectively. This configuration achieved the best balance between feature adaptability and classification accuracy. The fused probability for image b and class c is computed as:

$$P_{final}[b, c] = w_{fsl} * P_{fsl}[b, c] + w_{dnn} * P_{dnn}[b, c], \quad \forall b, c \quad (8)$$

where:

- $P_{final}[b, c]$ is the fused probability for image b and class c .
- w_{fsl} is the weight assigned to FSL probabilities.
- $P_{fsl}[b, c]$ is the probability for image b and class c from the FSL module.
- w_{dnn} is the weight assigned to DNN probabilities (equal to $1 - w_{fsl}$).
- $P_{dnn}[b, c]$ is the probability for image b and class c from the DNN module.

The loss function for the fused output is:

$$L = -\left(\frac{1}{batch_size}\right) * \sum_{b=1}^{batch_size} \sum_{c=0}^{num_classes-1} y[b, c] * \log(P_{final}[b, c]) \quad (9)$$

where:

- L is the average cross-entropy loss for the batch.
- $batch_size$ represents the cardinality of the current image batch.
- $y[b, c]$ indicates the true label (0 or 1) for image b and class c .
- $P_{final}[b, c]$ is the fused probability computed in (8).

The derivative of the loss with respect to w_{fsl} is:

$$\frac{\partial L}{\partial w_{fsl}} = -\sum_{b=1}^{batch_size} \sum_{c=0}^{num_classes-1} y[b, c] * \frac{(P_{fsl}[b, c] - P_{dnn}[b, c])}{P_{final}[b, c]} \quad (10)$$

where $\frac{\partial L}{\partial w_{fsl}}$ is the partial derivative of the loss L concerning the fusion weight w_{fsl} .

The weight update rule is:

$$w_{fsl_new} = w_{fsl_old} - learning_rate * \frac{\partial L}{\partial w_{fsl}} \quad (11)$$

where:

- $learning_rate$ specifies the step size for parameter updates during the optimization process.
- w_{fsl_new} is the updated weight for FSL.
- w_{fsl_old} is the old weight for FSL.

Finally, the DNN weight is constrained as follows:

$$w_{dnn} = 1 - w_{fsl_new} \quad (12)$$

The novelty of this approach lies in the fusion step. We take the probability distributions produced by each module and combine them using a weighted average. Equation (8) represents the fusion strategy and is the core of the hybrid approach, allowing the model to integrate the knowledge learned by the two different pathways.

The performance of the proposed DL-FSA model is evaluated using the classification metrics: accuracy, precision, recall, and F1-score:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (13)$$

$$Precision = \frac{TP}{TP+FP} \quad (14)$$

$$Recall = \frac{TP}{TP+FN} \quad (15)$$

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (16)$$

where TP are the true positives, TN the true negatives, FP the false positives, and FN the false negatives.

To assess whether the improvement of the model was statistically significant, a two-proportion z-test was applied using the following equation:

$$z = \frac{p_1 - p_2}{\sqrt{p(1-p)\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}} \quad (17)$$

where:

- p_1 and p_2 are the accuracies of the proposed and baseline models, respectively.
- p is the pooled proportion.
- n_1 and n_2 denote the number of validation samples.

III. RESULTS AND DISCUSSION

The results obtained from the hybrid model are presented below. The hybrid model achieved a training accuracy of 98.17% and a loss of 0.0632, a validation accuracy of 95.78%, and a loss of 0.1904, showing the effectiveness of combining FSL, CNN, ANN, and DNN architectures. The model was run for 20 epochs and other metrics like precision, recall, and F1-score were also computed. Figure 7 describes the number of images per class utilized in the dataset. Figure 8 shows the model's training and validation performance over 20 epochs.

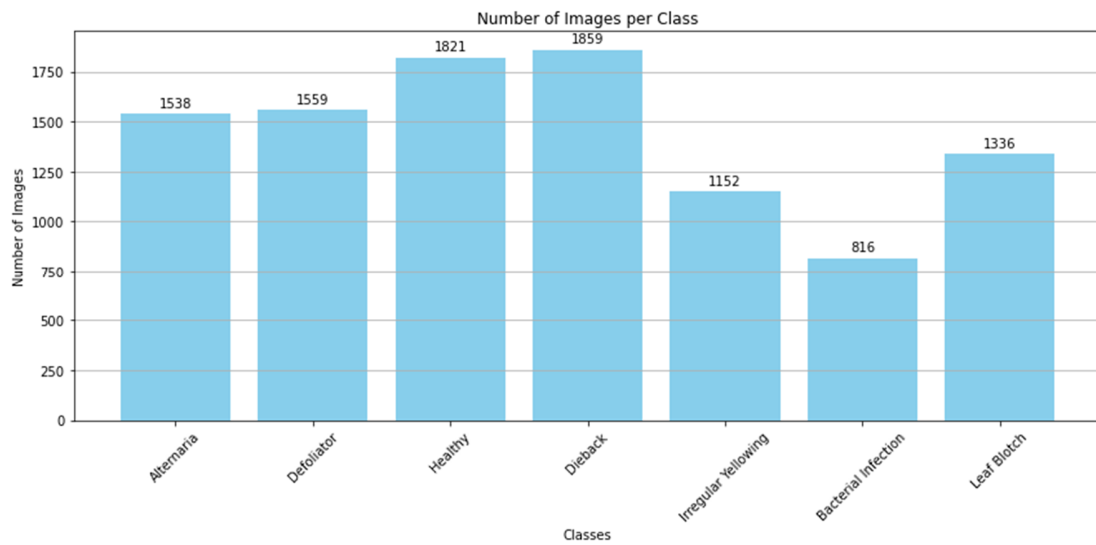


Fig. 7. Class-wise distribution of neem leaf images in the dataset.

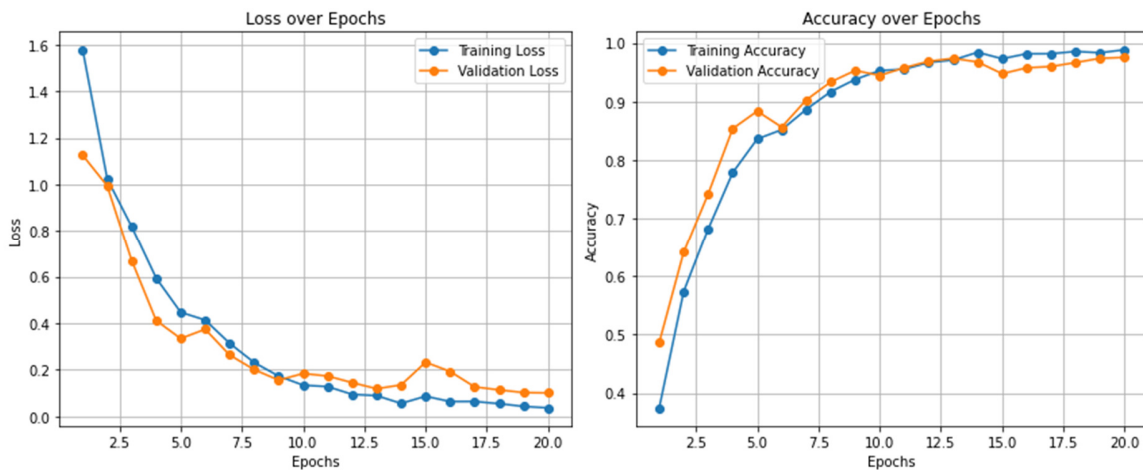


Fig. 8. Model efficacy during training and validation across epochs.

Figure 9 presents the Receiver Operating Characteristic (ROC) curve for the DL-FSA model across multiple classes, highlighting the true positive and false positive rate trade-off. It is used for binary classification but here it is plotted for a multiclass classification problem with 7 classes (Class 0 to Class 6). The image shows separate class-specific ROC curves, indicating the performance of the model in classifying each class. The Area Under the Curve (AUC) per class is shown in the legend. The dashed black line serves as a baseline for comparison. All the ROC curves per class are closer to 1.0 in the top left corner, indicating excellent classification performance.

A detailed analysis of the model's performance for each class is provided through the confusion matrix in Figure 10. The diagonal values indicate correct classifications, whereas the non-diagonal values show the misclassified instances. For example, 302 instances were correctly predicted as Alternaria (true positive), 0 instances were incorrectly predicted as Dieback, bacterial infection, defoliator, and irregular yellowing (false negatives), 2 instances were incorrectly classified as

healthy, and 3 instances were incorrectly classified as leaf blotch (false positive). Similarly, other classes can be analyzed.

The classification report in Figure 11 indicates strong model performance with an overall accuracy of 96%. Dieback achieved perfect scores (precision, recall, F1-score = 1.00), whereas defoliator and leaf blotch also showed high consistency with values above 0.97. Alternaria recorded precision and recall of 0.96 and 0.98, respectively. Slightly lower recall values for bacterial infection (0.86) and irregular yellowing (0.85) suggest minor misclassifications. The macro- and weighted-average F1-scores of 0.95 and 0.96 confirm the model's balanced and reliable classification across all disease categories.

Statistical validation of model performance was conducted using a two-proportion z-test, confirming the proposed hybrid model's improvement over the best baseline was significant ($p=0.025$). Although multiple training runs were not performed, the accuracy consistency was verified across training epochs, showing fluctuations within $\pm 1.2\%$.

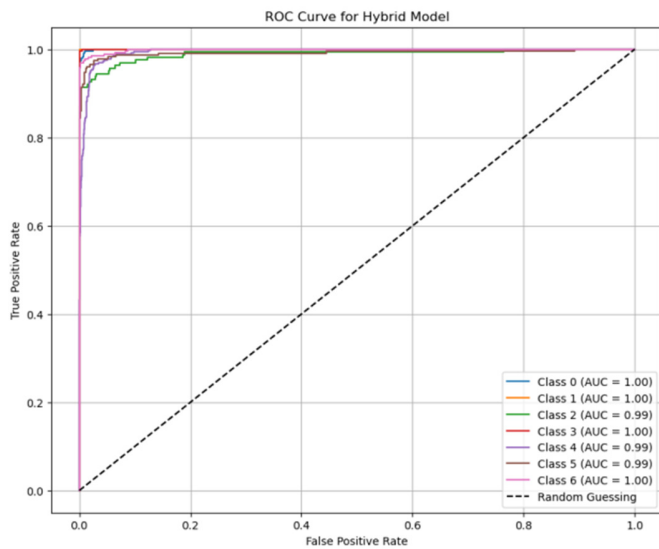


Fig. 9. ROC curve of the proposed hybrid DL-FSA model.

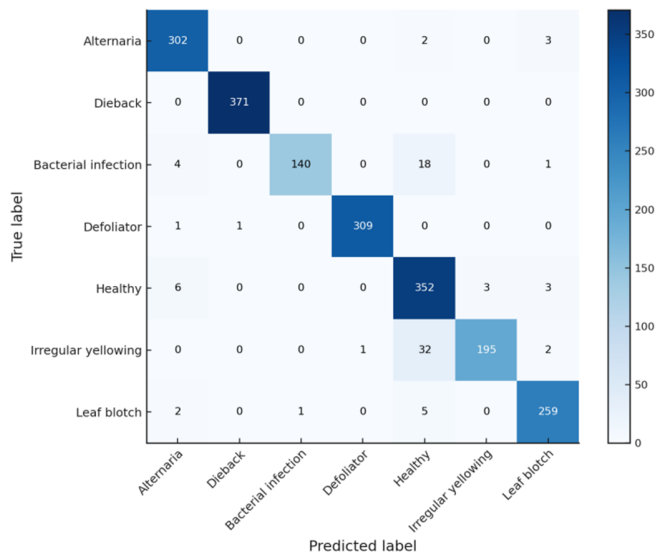


Fig. 10. Confusion matrix of the proposed hybrid DL-FSA model for neem leaf disease classification across seven categories.

Classification Report:

	precision	recall	f1-score	support
Alternaria	0.96	0.98	0.97	307
Dieback	1.00	1.00	1.00	371
bacterial_infection	0.99	0.86	0.92	163
defoliator	1.00	0.99	1.00	311
healthy	0.86	0.97	0.91	364
irregular_yellowing	0.98	0.85	0.91	230
leaf_blotch	0.97	0.97	0.97	267
accuracy			0.96	2013
macro avg	0.97	0.95	0.95	2013
weighted avg	0.96	0.96	0.96	2013

Fig. 11. Classification report of the proposed hybrid DL-FSA model showing precision, recall, F1-score, and support values for each disease class.

The heatmap visualization in Figure 12 shows class-wise prediction strengths of the hybrid model, with the diagonal values close to 1.00 indicating strong class specific accuracy, whereas lighter off-diagonal areas represent limited confusion among visually similar diseases.

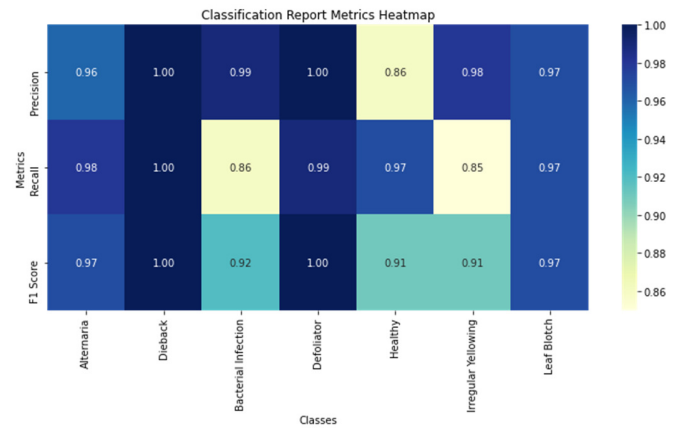


Fig. 12. Heatmap visualization showing class-wise classification performance.

The hybrid model was evaluated against several baseline approaches. A CNN-only model with two convolutional layers and dense layers achieved 91.90% validation accuracy but showed limited generalization due to its simplicity. Transfer learning with VGG16, using frozen base layers to retain pre-trained features and custom fully connected layers, achieved 90.96%. FSL with meta-learning and metric learning alone attained 94.57%, showing a moderate improvement. By combining the strengths of FSL and DNNs, the proposed hybrid model further improved performance, reaching 96% validation accuracy. In this architecture, the neural network module refines predictions through richer feature representation, whereas FSL supports effective learning from limited samples. Table III and Figure 13 summarize the comparative results.

TABLE III. PERFORMANCE COMPARISON OF THE PROPOSED HYBRID DL-FSA MODEL WITH BASELINE APPROACHES

Model	Validation accuracy (%)	Training time	Complexity	Generalization
CNN-only	91.90	Faster	Lightweight	Moderate
Transfer learning with VGG16	90.96	Slower	Computationally expensive	High
FSL with meta-learning and metric learning	94.57	Faster	Lightweight	High
CNN-ANN-DNN	93.49	Slower	Computationally expensive	High
DL-FSA (proposed)	96.00	Slowest	Computationally expensive	Highest

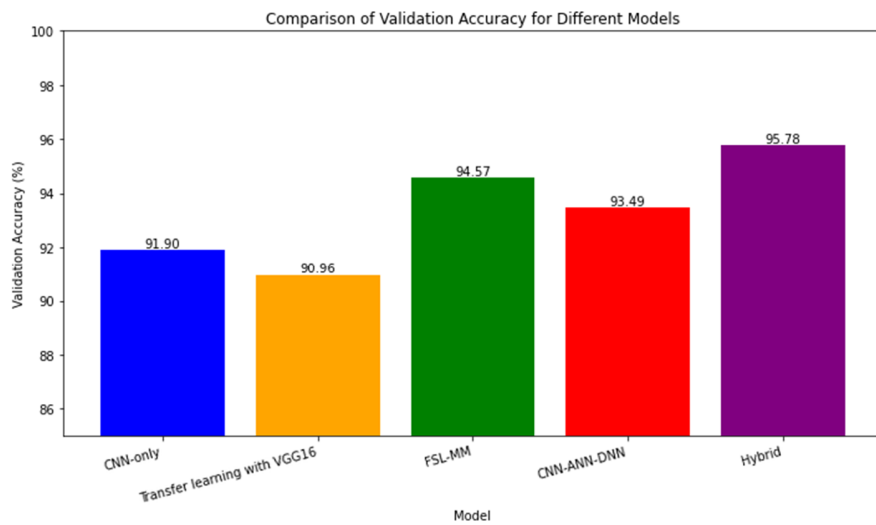


Fig. 13. Validation accuracy comparison of the proposed hybrid DL-FSA model with baseline models.

As shown in Table IV, the proposed DL-FSA model achieved superior accuracy compared with recent literature, validating its effectiveness for real-time medicinal plant disease detection.

TABLE IV. COMPARISON OF THE PROPOSED HYBRID DL-FSA MODEL WITH EXISTING FEW-SHOT AND DL STUDIES ON PLANT DISEASE CLASSIFICATION

Reference	Model	Dataset type	Accuracy (%)
[15]	FSL	Real field leaf images	91.4
[16]	Meta-learning baseline	Agricultural image set	90.8
[17]	Metric learning approach	Generic image classification datasets	89.48
[18]	TWFSL-MM	Neem leaf images	92.09
[19]	MAFDE-DNA (Few-shot with Deep Nearer Neighbor)	Public plant disease dataset	81.41
DL-FSA (proposed model)	Hybrid Few-shot + CNN-ANN-DNN	Real-time neem dataset	96

To verify the validity of the reported improvement in performance, a statistical significance analysis was conducted by comparing the validation accuracy of the proposed model (96%) with the best baseline few-shot model (94.5%). Using the validation set of 2,016 images, a two-proportion z-test yielded $z = 2.24$ with $p\text{-value} = 0.025$, confirming that the improvement is statistically significant at the 95% confidence level. This indicates that the observed accuracy gain is true and is not due to random variation.

Although the proposed hybrid few-shot model shows only a 1.43% accuracy improvement over the few-shot baseline, it offers better stability, feature representation, and class-wise consistency. The slight gain in accuracy supports stronger generalization. While the model's fusion design increases computational cost, this trade-off is acceptable for research and can be optimized for practical deployment through pruning and lightweight network adaptation.

The proposed hybrid FSL model achieved an accuracy of 96%, demonstrating strong generalization and robustness in

identifying seven neem leaf disease classes. The main strength of the model lies in its ability to extract diverse features from limited data, effectively reducing overfitting and improving stability on real-time field images. However, the model is computationally expensive due to its multi-network structure, resulting in slower training and higher resource requirements. Further validation on larger and more diverse datasets is needed to enhance scalability and confirm its applicability.

IV. CONCLUSION

The proposed hybrid model demonstrates promising results, achieving an average accuracy of 96%. This approach efficiently identifies the spatial patterns and textures specific to neem leaf diseases using neural networks. Few-Shot Learning (FSL) enhances the performance of Convolutional Neural Network (CNN), Artificial Neural Network (ANN), and Deep Neural Network (DNN) components within the hybrid model by providing a framework that allows these networks to learn effectively from limited data. The model exhibits high-level feature learning through the ANN, improved decision-making through the DNN, and robustness ensured by regularization techniques. It successfully classifies seven classes of neem leaf images, addressing a multi-class classification task. The novelty of this approach lies in the tailored design of FSL and the hybrid architecture that combines the strengths of CNN, ANN, and DNN with weighted average fusion. This is helpful in scenarios where the data are limited and efficient learning is crucial. The work also compares the hybrid model with other state-of-the-art models, highlighting its effectiveness.

The framework can be applied for automatic neem disease detection in agriculture, enabling early diagnosis of plant diseases and reducing crop loss. By leveraging this hybrid architecture, the model provides a robust and effective approach for multi-class plant disease classification. Furthermore, the model can be adapted to other plant disease datasets or classification tasks through parameter fine-tuning. Future work could explore more complex fusion approaches, such as stacking, and improve scalability by deploying the model on edge or mobile devices for real-time field detection. The

methodology can also be extended to other medicinal or endangered plant species for broader applicability.

Overall, this work contributes to the ongoing efforts in Machine Learning (ML) and Deep Learning (DL) to develop efficient and effective models that can learn from minimal data, paving the way for advancements across applications like medical imaging, wildlife monitoring, and others.

REFERENCES

- [1] M. A. Alzohairy, "Therapeutics Role of Azadirachta indica (Neem) and Their Active Constituents in Diseases Prevention and Treatment," *Evidence-Based Complementary and Alternative Medicine*, vol. 2016, no. 1, Mar. 2016, Art. no. 7382506, <https://doi.org/10.1155/2016/7382506>.
- [2] S. Gajalakshmi and S. A. Abbasi, "Neem leaves as a source of fertilizer-pesticide vermicompost," *Bioresource Technology*, vol. 92, no. 3, pp. 291–296, May 2004, <https://doi.org/10.1016/j.biortech.2003.09.012>.
- [3] "Diseases of Neem (Azadirachta indica)." TNAU Agritech Portal. https://agritech.tnau.ac.in/forestry/forest_disease_neem.html.
- [4] M. N. Nagendra Prasad, S. Shankara Bhat, V. Nivedita Dharwar, B. Shraddha Mehta, and A. Chauhan, "In vitro efficacy of plant essential oils against Phomopsis azadirachtae – the causative agent of die-back disease of neem," *Archives of Phytopathology and Plant Protection*, vol. 44, no. 5, pp. 412–418, Mar. 2011, <https://doi.org/10.1080/03235400903092941>.
- [5] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Computer Science Review*, vol. 40, May 2021, Art. no. 100379, <https://doi.org/10.1016/j.cosrev.2021.100379>.
- [6] I. H. Rather and S. Kumar, "Generative adversarial network based synthetic data training model for lightweight convolutional neural networks," *Multimedia Tools and Applications*, vol. 83, no. 2, pp. 6249–6271, Jan. 2024, <https://doi.org/10.1007/s11042-023-15747-6>.
- [7] C. Little, M. Elliot, R. Allmendinger, and S. S. Samani, "Generative Adversarial Networks for Synthetic Data Generation: A Comparative Study." arXiv, Dec. 03, 2021, <https://doi.org/10.48550/arXiv.2112.01925>.
- [8] C. Tian, H. Gao, P. Wang, and B. Zhang, "An Enhanced GAN for Image Generation," *Computers, Materials & Continua*, vol. 80, no. 1, pp. 105–118, July 2024, <https://doi.org/10.32604/cmc.2024.052097>.
- [9] I. J. Goodfellow *et al.*, "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems*, Montreal, Canada, 2014, pp. 2672–2680, <https://doi.org/10.48550/arXiv.1406.2661>.
- [10] I. Goodfellow, "NIPS 2016 Tutorial: Generative Adversarial Networks." arXiv, Apr. 03, 2017, <https://doi.org/10.48550/arXiv.1701.00160>.
- [11] J. Islam and Y. Zhang, "GAN-based synthetic brain PET image generation," *Brain Informatics*, vol. 7, no. 1, Mar. 2020, Art. no. 3, <https://doi.org/10.1186/s40708-020-00104-2>.
- [12] R. Agrawal, "Adaptive Few-Shot Learning (AFSL): Tackling Data Scarcity with Stability, Robustness, and Versatility." arXiv, Jan. 23, 2025, <https://doi.org/10.48550/arXiv.2501.13479>.
- [13] Y. Song, T. Wang, P. Cai, S. K. Mondal, and J. P. Sahoo, "A Comprehensive Survey of Few-shot Learning: Evolution, Applications, Challenges, and Opportunities," *ACM Computing Surveys*, vol. 55, no. 13, July 2023, Art. no. 271, <https://doi.org/10.1145/3582688>.
- [14] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a Few Examples: A Survey on Few-shot Learning," *ACM Computing Surveys*, vol. 53, no. 3, June 2020, Art. no. 63, <https://doi.org/10.1145/3386252>.
- [15] D. Argüeso *et al.*, "Few-Shot Learning approach for plant disease classification using images taken in the field," *Computers and Electronics in Agriculture*, vol. 175, Aug. 2020, Art. no. 105542, <https://doi.org/10.1016/j.compag.2020.105542>.
- [16] Y. Li and J. Yang, "Meta-learning baselines and database for few-shot classification in agriculture," *Computers and Electronics in Agriculture*, vol. 182, Mar. 2021, Art. no. 106055, <https://doi.org/10.1016/j.compag.2021.106055>.
- [17] X. Li, X. Yang, Z. Ma, and J.-H. Xue, "Deep metric learning for few-shot image classification: A Review of recent developments," *Pattern Recognition*, vol. 138, June 2023, Art. no. 109381, <https://doi.org/10.1016/j.patcog.2023.109381>.
- [18] H. A. Vidya and M. S. N. Murthy, "TWFSL-MM: Few-Shot Learning using Meta-Learning and Metric-Learning for Disease Detection in Azadirachta Indica," *Engineering, Technology & Applied Science Research*, vol. 15, no. 2, pp. 21129–21135, Apr. 2025, <https://doi.org/10.48084/etasr.9886>.
- [19] Y. Zhao, Z. Zhang, N. Wu, Z. Zhang, and X. Xu, "MAFDE-DN4: Improved Few-shot plant disease classification method based on Deep Nearest Neighbor Neural Network," *Computers and Electronics in Agriculture*, vol. 226, Nov. 2024, Art. no. 109373, <https://doi.org/10.1016/j.compag.2024.109373>.
- [20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, <https://doi.org/10.1038/nature14539>.
- [21] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *Journal of Big Data*, vol. 8, no. 1, Mar. 2021, Art. no. 53, <https://doi.org/10.1186/s40537-021-00444-8>.
- [22] S. F. Ahmed *et al.*, "Deep learning modelling techniques: current progress, applications, advantages, and challenges," *Artificial Intelligence Review*, vol. 56, no. 11, pp. 13521–13617, Nov. 2023, <https://doi.org/10.1007/s10462-023-10466-8>.
- [23] E. Grossi and M. Buscema, "Introduction to artificial neural networks," *European Journal of Gastroenterology & Hepatology*, vol. 19, no. 12, Dec. 2007, Art. no. 1046, <https://doi.org/10.1097/MEG.0b013e3282f198a0>.
- [24] G. Shobana, K. Vignesh, and S. Sree Dharshan, "Plant Disease Detection Using Deep Neural Network," in *2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation*, Coimbatore, India, 2023, pp. 1–6, <https://doi.org/10.1109/ICAECA56562.2023.10199940>.
- [25] V. H. Ananda, N. M. M. S. Rao, and T. D. Krishnamurthy, "Harnessing deep learning for medicinal plant research: a comprehensive study," *International Journal of Electrical and Computer Engineering*, vol. 15, no. 1, pp. 908–920, Feb. 2025, <https://doi.org/10.11591/ijece.v15i1.pp908-920>.
- [26] Z. Hong, Z. Yang, H. Wang, D. Li, W. Nai, and Y. Xing, "The Weighted Average Ensemble Learning Based on Polar Bear Algorithm with T-Distribution Parameters," in *2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference*, Chongqing, China, 2020, pp. 1902–1905, <https://doi.org/10.1109/ITAIC49862.2020.9338792>.
- [27] C. Yu, L. Chang, X. Xu, Y. Cao, and Z. Zhang, "Compatible multi-model output fusion for complex systems modeling via set operation-based focus data identification," *Journal of Computational Science*, vol. 82, Oct. 2024, Art. no. 102423, <https://doi.org/10.1016/j.jocs.2024.102423>.
- [28] P. B. R. and S. Rani, "Indian Medicinal Leaves Image Datasets." Mendeley Data, May 05, 2023, <https://doi.org/10.17632/748f8jkphb.3>.
- [29] P. Sarma and P. Aziz Boruah, "MED117_Medicinal Plant Leaf Dataset & Name Table." Mendeley Data, Jan. 18, 2023, <https://doi.org/10.17632/dtvbwrhznz.3>.
- [30] T. Dimlioglu and A. Choromanska, "GRAWA: Gradient-based Weighted Averaging for Distributed Training of Deep Learning Models." arXiv, Mar. 07, 2024, <https://doi.org/10.48550/arXiv.2403.04206>.