FMDB *Transactions on Sustainable Health Science Letters*

Enhanced Sunflower Leaf Disease Identification Using YOLO and CNN-Based Feature Extraction with Optimizer-Driven Classification

M. Arunadevi Thirumalraj[1,*], M. Mathumathi[2], R. Kannan[3], Rahul Panakkal[4]

[1]Department of Computer Science and Engineering, Karunya Institute of Technology and Science, Coimbatore, Tamil Nadu, India.
[1,3]Department of Computer Science and Business Management, Saranathan College of Engineering, Tiruchirappalli, Tamil Nadu, India.
[2]Department of Computer Science and Engineering, K. Ramakrishnan College of Technology, Tiruchirappalli, Tamil Nadu, India.
[4]Department of Computer Science and Engineering, University of Illinois at Urbana, Champaign, Illinois, United States of America.
aruna.devi96@gmail.com[1], mathumathim.cse@krct.ac.in[2], kannan7137@saranathan.ac.in[3], rahulp8@illinois.edu[4]

Abstract: Essential oilseed crops grown all around the globe include sunflowers (Helianthus annuus). Nevertheless, rust, powdery mildew, downy mildew, and Alternaria leaf blight are only a few of the leaf diseases that drastically reduce crop output. Lower yields and economic losses result from these diseases' effects on plant health, which diminish photosynthetic efficiency. Conventional approaches to disease detection rely on human observation, which can be laborious, biased, and prone to errors due to both the observer and their surroundings. A fast, accurate, and reliable disease-detection system is required to detect and categorise sunflower leaf illnesses in real time. This study utilises the YOLO (You Only Look Once) object detection network and a pre-trained CNN for feature extraction and classification to develop a deep learning-based process for disease diagnosis in sunflower leaves. An enhanced YOLO-based detection model is used for real-time classification, with a pre-trained CNN extracting deep spatial features from sunflower leaf images. To further improve detection accuracy while reducing computational complexity, an optimiser-based network is also used. The suggested method outperforms conventional machine learning techniques in identifying a range of diseases affecting sunflower leaves, achieving impressive recall and precision.

Keywords: Sunflower Leaves Disease Detection; Convolutional Neural Network; Feature Extraction; Machine Learning Methods; Alternaria Leaf Blight; Modified Wiener Filter.

Received on: 10/02/2025, **Revised on:** 15/04/2025, **Accepted on:** 23/06/2025, **Published on:** 06/12/2025

Journal Homepage: https://www.fmdbpub.com/user/journals/details/FTSHSL

DOI: https://doi.org/10.69888/FTSHSL.2025.000513

Cite as: M. A. Thirumalraj, M. Mathumathi, R. Kannan, and R. Panakkal, "Enhanced Sunflower Leaf Disease Identification Using YOLO and CNN-Based Feature Extraction with Optimizer-Driven Classification," *FMDB Transactions on Sustainable Health Science Letters*, vol. 3, no. 4, pp. 246–255, 2025.

1. Introduction

The sunflower leaf is used in numerous sectors, including animal husbandry. Diseases that affect sunflower leaves are numerous; however, this thesis focuses on four specific diseases: downy mildew, Verticillium wilt, Alternaria leaf blight, and

[*]Corresponding author.

Phomab light. Sunflowers and their leaves are beneficial and useful in many ways [1]. Because of its many uses, including those of humans, animals, and the environment, their contamination would be devastating. Saving the sunflower plant and its leaves—and with them, the environment and a wealth of nutrients—depends on early disease detection and recognition [2]. In the early stages of disease classification or recognition, both traditional and modern computer vision approaches are available. On the other hand, computer vision, often called machine learning, has enabled faster classification and more accurate results than the traditional method. Even though ML algorithms provide respectable results, they are not up to snuff; specifically, they failed to achieve very accurate picture identification [3]. The development of agriculture is vital to our country's economy, and one of the most pressing needs is the early identification and forecasting of plant diseases. It helps keep people fed and the economy afloat [4]. Researchers can save plants and prevent losses if they spot them earlier. Classification and outcome forecasting are common applications of deep learning algorithms [5].

CNNs are the primary classifiers used in active deep learning approaches for plant disease classification. No matter the dataset size, CNNs are among the top models for image-based classification and recognition [6]. It is unnecessary to manually construct the feature extraction function and classifier when using deep learning, as the model dynamically learns structured representations. Many models, such as AlexNet, GoogLeNet, and LeNet, are built on a simple CNN architecture. For image recognition and classification, CNNs are highly recommended and widely regarded as the best systems [7]. When tested for plant diseases, it does quite well. It is the most basic method for identifying objects. Depending on the needs, any feature extractor can be matched with any neural architecture. For models to function properly, data must be preprocessed. The presence of overlapping symptoms makes the diagnosis of many diseases, whether fungal or viral, challenging [8]. There has been a lot of buzz about automated disease detection methods due to advances in deep learning and artificial intelligence. An excellent approach for large-scale agricultural disease surveillance is object detection models such as YOLO, which provide real-time, high-accuracy disease identification. This research aims primarily to:

- To develop an automated sunflower leaf disease finding system using deep learning techniques, particularly YOLO, combined with a pre-trained CNN for feature extraction.
- To enhance detection accuracy and classification performance by integrating an optimiser-based network to refine disease identification and reduce false positives.
- To enable real-time disease detection by optimising the model’s computational efficiency, making it suitable for large-scale deployment in agricultural fields.

To provide a scalable and efficient solution that assists farmers in early disease detection and intervention, minimising crop losses and improving yield quality.

2. Related Works

An approach to illness detection using few-shot learning and diffusion generative models was suggested by Zhou et al. [9]. An end-to-end outline for disease diagnosis has been developed by combining the benefits of few-shot learning with the feature-generation capabilities of diffusion models. In addition, the model's granular characteristics were improved, and the quality of illness feature representations was substantially raised by incorporating attention processes. Parez et al. [10] introduced LeafNet, a lightweight CNN optimised for devices with limited resources. While the block-wise VGG19 architecture served as inspiration for LeafNet, the latter made numerous optimisations that enabled it to achieve comparable accuracy, including smaller size, faster inference time, and fewer parameters. The findings show that the proposed method performs as well as state-of-the-art methods across accuracy, FPR, runtime, model size, and practicality, suggesting its promise for practical use. In their study, Pavithra and Aishwarya [11] employed three optimisation strategies to identify leaf-disease-causing agents. During the pre-processing stage of the proposed methodology, the input image is processed with the Modified Wiener Filter (MWF) to remove noise. Compared to current methods, the proposed methodology achieves 98.53% accuracy. Using deep learning models, Ünal and Dudak [12] attempted to categorise the illnesses observed in sunflower flowers and leaves. Researchers started by classifying it using two pre-trained CNN models, ResNet101 and ResNext101.

Then, to add pressure to these networks, researchers compared the results. Grey diseases that can harm sunflower crops; this study employed a dataset that includes this information. In our study, researchers classified sunflower disorders using the original deep learning models. In this study, researchers implemented the deep model-based plant disease identification and classification methodology proposed by Patil and Borse [13]. At first, the necessary picture is retrieved from the internet databases. In the segmentation step, the Adaptive Attention-aided Mask Region-based CNN (AAM-RCNN) is employed, with the collected images serving as input. The parameters of the AAM-RCNN are improved using the Boosted Random Parameter-based Golden Tortoise Beetle Optimiser (BRP-GTBO) to improve segmentation performance. The HC-2D/1D-MDEB7 model yields the final detected and categorised results. To demonstrate the effectiveness of the suggested framework, experimental verification is executed.

3. Proposed Methodology

In this section, the proposed deep learning model detects sunflower plant disease, as shown in Figure 1.

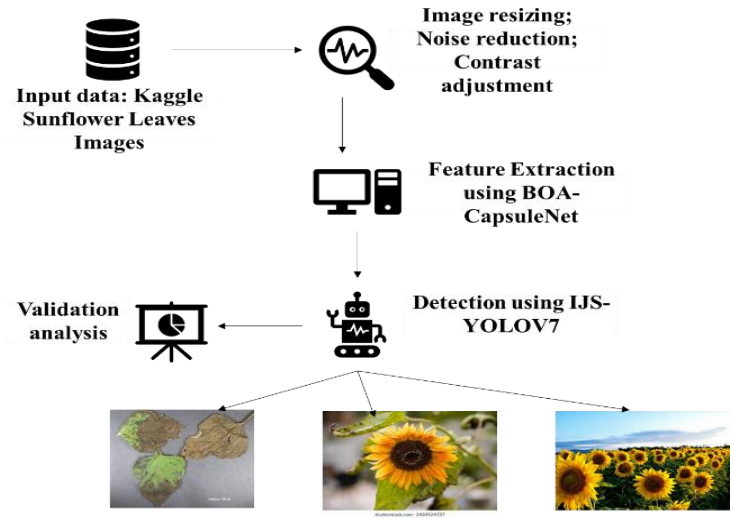

Figure 1: Workflow of the research model

3.1. Dataset Description

The dataset consists of 467 original images and 1,668 augmented sunflower leaves and flowers categorised into four classes: Grey Mould, Downy Mildew, Leaf Scars, and Fresh Leaves [14]. Figure 2 shows examples from the dataset, including diseased and healthy leaves.

Table 1: Circulation of the dataset by class

The sum of Original Imageries	The sum of the Images augmented	Class Name
72	398	Gray Mold
120	470	Downy Mildew
141	509	Leaf Scary
134	491	Fresh Leaves
467	1668	Total

The augmentation technique is applied to this sample dataset, including rotation, scaling, and shearing, as shown in Table 1.

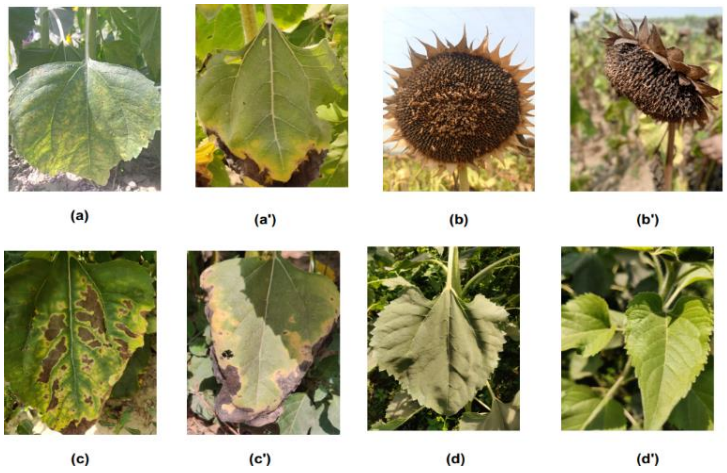

Figure 2: Trial dataset— (a, a) downy mildew, (b, b') grey mould, (c, c') leaf scars, besides (d, d') fresh leaf

3.2. Pre-Processing

The following procedure is applied to the input sample images to enhance classification accuracy:

- Images resized to 640 × 640 pixels.
- Noise is reduced using Gaussian Blur and denoising functions.
- Brightness and contrast adjusted for clarity.

3.3. Feature Extraction Using Capsule Network

The input images are treated as such; in addition, the preprocessing described in "Preprocessing" is applied to improve the accuracy of the results by increasing image contrast and removing noise. The 512 × 512 high-resolution images produced by preprocessing are used as inputs to the architecture.

3.3.1. Convolution Layer

In the convolution layer, the first layer, each neuron is linked to a receptive field—a specific area of the brain that receives input. The convolution layer's primary function is to extract picture features. As an element of functions deciphered, the created function provides the integral of functions. The syntax for a Keras convolution layer is as follows. Conv2D with parameters [input shape, filters, kernel size, strides, padding, activation='relu']. The convolutional layer is said to take as input the preprocessed image with dimensions 640 × 640. All four of the convolutional layers—Conv1, Conv2, Conv3, and Conv4—are part of the convolutional architecture. With each filter being 9 × 9 and having a stride of 1, 256 filters are used across all four layers to convolve the features. A 9 × 9 kernel with a stride of 1 is used with an input of size 512 × 512 for the chief internal layer (Conv1). The feature map produced has dimensions [504 × 504, 256], which is the same as [(512 − 9)/1 + 1], and is used as input to the subsequent internal layer (Conv2). This feature map, which is an input to the following internal layer (Conv3), has size [(504 − 9)/1 + 1] or [496 × 496, 256]. The output feature map from Conv3 is [488 × 488, 256], which is the input for the fourth internal layer (Conv4), and its size is [(496 − 9)/1 + 1]. The layer, the next layer in the proposed structure, is fed the final output size of the convolution layer, which is [480×480,256], or [(488 − 9)/1 + 1]. Every internal layer requires 20,737 parameters, calculated as 256 × (9 × 9 + 1).

3.3.2. Primary Capsule Layer

Following the convolution layer is the main capsule layer. Convolution, function, and the squash function are the three separate operations that make it up. The main capsule layer uses the Squash function to achieve this goal. This method preserves the position data while ensuring the vector's length is between 0 and 1. This layer receives as input the feature map with dimensions [480 × 480, 256] from the convolution layer. A convolution with a 5 × 5 filter, with a stride of 2, is applied to the input feature map. The final dimension of the feature map is [(238 × 238, 256)], which is equal to [(480 − 5)/2 +1]. One hundred and forty-four primary capsules follow. The main capsules' job is to extract the most important characteristics from convolution and generate all possible permutations of those features. Like the convolutional layer, this layer's 64 "primary capsules" perform similar functions. The 480 × 480 × 256 input volume is processed by four 5 × 5 × 256 kernels (with a stride of 2) per capsule, resulting in an output volume of 238 × 238 × 4. Using 64 such capsules in the suggested design, the final output volume is 238 × 238 × 4 × 64. There are 3,625,216 trainable parameters in this layer, with a formula of 238 × 238 × 64.

3.3.3. Class Capsule Layer

In the suggested approach, the class capsule layer stands in for CNN's max-pooling layer, which is substituted by dynamic routing-by-agreement. There are three distinct kinds of capsules with 4, 8, and 16 dimensions present in this layer of the suggested methodology for each class label. The output of the capsule in the ith layer (l), which is denoted by the symbol v_i, is utilised as an input for the capsules in the layer that comes after it (l+1). Within that layer, the jth capsule receives input v_i and multiplies it by the appropriate weight. (w_{ij}) among the ith capsule besides the jth capsule. The subsequent is signified as $v_{j|i}$ which contributes to the influence of the ith capsule in the (l + 1) layer:

$$v_{j|i} = W_{ij} * v_j \quad (1)$$

where W_{ij} In this case, vi is the input value for the ith node. V_{ij} is the output value from the ith capsule. The phrase represents the weight of the jth node of the l+1 layer. What follows is an application of the squash function to calculate a weighted total of all the main capsule predictions for class capsule j:

$$S_j = \sum_{i=1}^{N} C_{ij} * v_{j|i} \quad (2)$$

where C_{ij} is the result of the softmax capsule, $v_{j|i}$ is input besides S_j is the total weighted sum of all the nodes contained within the capsule. It is necessary to apply the squashing function, which is computed in the following manner, to guarantee that this result is between 0 and 1:

$$SQ_j = \frac{\|S_j\|^2 * \|S_j\|}{1+\|S_j\|^2 * \|S_j\|} \quad (3)$$

The class capsule layer sum of limits. The trainable limits as C_{ij} are computed by multiplying the sum of vectors in the capsule layer by the number of vectors required as output, for 4D and 8D, and for 8D and 16D.

= 420, 525, 056 × 4 × 8 + 580 × 8 × 16
= 13, 456, 801, 792 + 72, 240
= 13, 456, 874, 032.

Hence, the entire trainable limits are

$= 420,525,636 + 13,456,874,032$= 13, 877, 399, 668.

3.3.4. Softmax Layer

In the output layer, SoftMax is typically used. The nodes in the SoftMax layer are identical to the labels it outputs. Consequently, in the proposed method, the SoftMax layer is assumed to have nine nodes, corresponding to the disease characteristics affecting sunflower leaves. It is possible to implement the SoftMax as a full SoftMax, meaning the probability calculation is performed for each possible feature. The parameters of the suggested feature extraction model are fine-tuned using BOA in this study [15].

3.4. Classification

In this section, sunflower disease detection is performed using an optimised YOLO model, as described in the upcoming sub-section.

3.4.1. Detection Using Yolov7

Key components of the YOLOv7 model's backbone network include E-ELAN, CBS, and MP-1. To extract and merge features across different levels of the backbone, YOLOv7's head network utilises the FPN architecture and E-ELAN. With the convolutional spatial pyramid (CSP) design and the SPP structure, researchers can improve feature extraction and reduce computational costs while extracting attributes at multiple scales. To increase the network's perceptual range, to combine the two approaches to create the SPPCSPC. The ELAN-W layer provides additional improvements to the feature extraction procedure. With the addition of two more output channels, the MP-2 block becomes functionally identical to the MP-1 block. A 1 × 1 convolution is employed to estimate the anchor frame, and the Rep modifies the number of image channels used in the features produced by the head network. Using a unique residual architecture inspired by RepVGG, the Rep structure reduces network complexity without sacrificing predictive performance. In actual estimations, this design convolution.

3.4.2. Attention Mechanism Module

One mechanism that attends to both channel and spatial dimensions is the CBAM module. The SAM has two separate components. The SAM focuses on locations with picture-wide contextual information, whereas the CAM highlights the image's foreground and relevant areas. First, 1D channel attention (the dark grey box), besides second, 2D spatial attention (the dark purple box), are the two halves of the CBAM attention process. A feature map is generated by feeding the channel spatial attention module's output into the CBAM. The initial step is to create an H × W × 1 tensor using global methods. When these feature maps are joined, their dimensionality is reduced. After that, the feature map is passed through a sigmoid activation function to produce the spatial feature. After applying this spatial attention feature to the input feature map, the resultant feature map is computed. This operation can be characterised by Equation (5):

$$M_c(F) = \sigma(\mathrm{MLP}(\mathrm{AvhPool}(F)) + \mathrm{MLP}(\mathrm{MaxPool}(F))) \quad (4)$$

$$M_s(F) = \sigma(f^{7\times7}(\mathrm{AvhPool}(F)) + \mathrm{MLP}(\mathrm{MaxPool}(f))) \quad (5)$$

If the input photos contain a wide range of item sizes, the CBAM attention technique can be applied. Extracting zone, which involves removing noise and honing in on the important objects, could enhance the detection presentation of the CBAM.

3.4.3. SPPF+

Based on the principles of feature reuse from both SPPF and SPPCSPC, the research proposed an SPPF+ structure, an updated version of SPPF. By encouraging feature reuse and creating dense links, researchers strengthened the SPPF in this study. After acquiring the SPPF module, researchers reduced the feature information loss associated with max pooling. Implementing the SPPF+ module enabled enhanced long-term retention of global data across small target areas. To detect tiny targets in impacted regions, the SPPF+ module efficiently stores global information.

3.4.4. BiFPN

The goal of multiscale data is to acquire data at different temporal and spatial scales. An instance of this would be when a collection of features of different sizes, referred to as $P^{in} = \left(P_{l_1}^{in}, P_{l_2}^{in}, \ldots, \right)$, where $P_{l_i}^{in}$ characterises the feature at the level l_i, to a mapping function of $P^{out} = f(P^{in})$. Pin by $P^{in} = (P_3^{in}, \ldots P_7^{in})$, where P_i^{in} is the image, and 1/2 is 640 × 640 pixels, for example, P_3^{in} corresponds to level 3 ($640/2^3 = 80$) resolution of 80 × 80 pixels, while P_7^{in} the same as feature level 7, with a 5 × 5-pixel dimension. Typical FPN combines multiscale features from the top down in the following way:

$$P_7^{out} = \mathrm{Conv}(P_7^{in}) \quad (6)$$

$$P_6^{out} = \mathrm{Conv}(P_6^{in} + \mathrm{Resize}(P_7^{out})) \quad (7)$$

$$P_3^{out} = \mathrm{Conv}(P_3^{in} + \mathrm{Resize}(P_4^{out})) \quad (8)$$

After the data sets are combined, the box may be predicted using the box networks. The weights utilised by the box, in addition to class networks, are proportional across all feature levels. Using learnable weights, Bi-FPN enhances YOLOv7 by enabling faster, more convenient top-down and feature fusions. Compared to PANet, Bi-FPN is superior at disease identification, as it uses fewer constraints while maintaining comparable accuracy.

3.4.5. Decoupled Head

To enhance detection accuracy, YOLOx's decoupled head divides the classification and localisation procedures. Compared with the YOLOv7 linked head, the YOLOx decoupled head has more convolutional layers, incorporates a 1 × 1 convolutional layer to reduce the channel dimension at each feature level, and then uses two 3 × 3 convolutional layers in tandem to form two branches. An extra 1 × 1 layer is present in both branches. There is an IoU branch in the regression tree as well. Despite having additional parameters, the decoupled head enhances the convergence speed, as previously stated. To calibrate the YOLO model's parameters—including batch size, epochs, learning rate, momentum, and weight decay—this study employs the enhanced jellyfish optimiser [16].

4. Results and Discussion

Research for the projected model was conducted using Python's deep learning toolbox and Google Colab. With 8 GB of RAM, the NVIDIA Quadro P4000 served as the GPU for both testing and training. To have utilised a 10-fold cross-validation technique to assess the suggested models. The benchmark datasets were split into training and test sets for the model.

4.1. Analysis of Proposed Feature Extraction

Figure 3 presents a graphical study of the proposed FE using diverse metrics for multi-class diseases of sunflower leaves. A comparative analysis of segmentation performance across various diseased leaf categories, Verticillium Wilt, and Healthy, using standard metrics. Among the disease categories, Phoma Blight achieved the highest F1-score of 0.8334, closely followed by Healthy leaves (0.8305) and Downy Mildew (0.8295), reflecting a strong segmentation of these classes. The results collectively suggest that the segmentation model demonstrates consistent and reliable performance across disease types, with Phoma Blight and Healthy leaves showing the most optimal segmentation outcomes, both in positive detection and in avoiding false alarms.

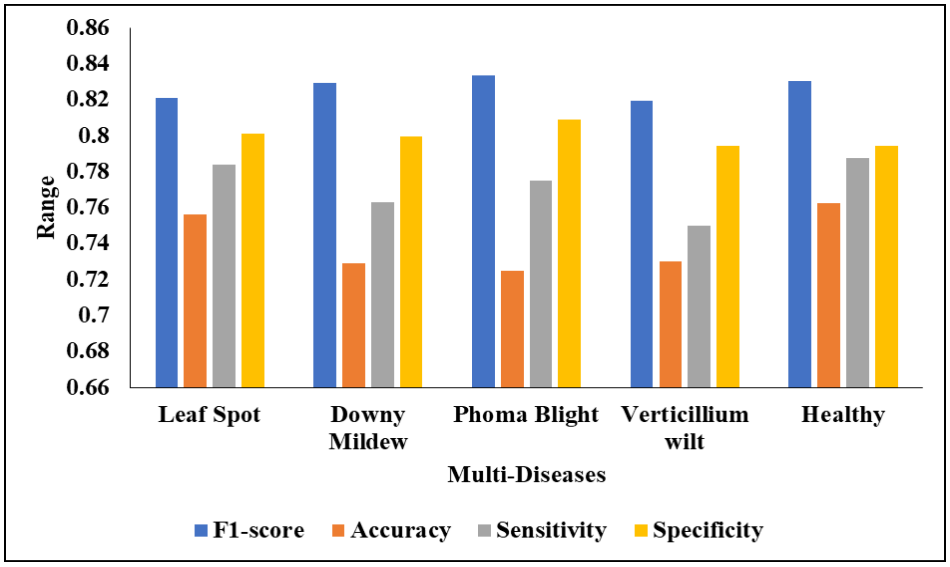

Figure 3: Analysis of proposed feature extraction on multi-class disease

Figure 4 presents a comparative analysis of projected extraction with existing procedures across diverse metrics. A comparative evaluation of segmentation performance between the proposed CapsuleNet-BOA model and several existing architectures, including CapsuleNet, AlexNet, VGGNet, and ResNet, using four core metrics: Sensitivity, Specificity, F1-score, and Accuracy.

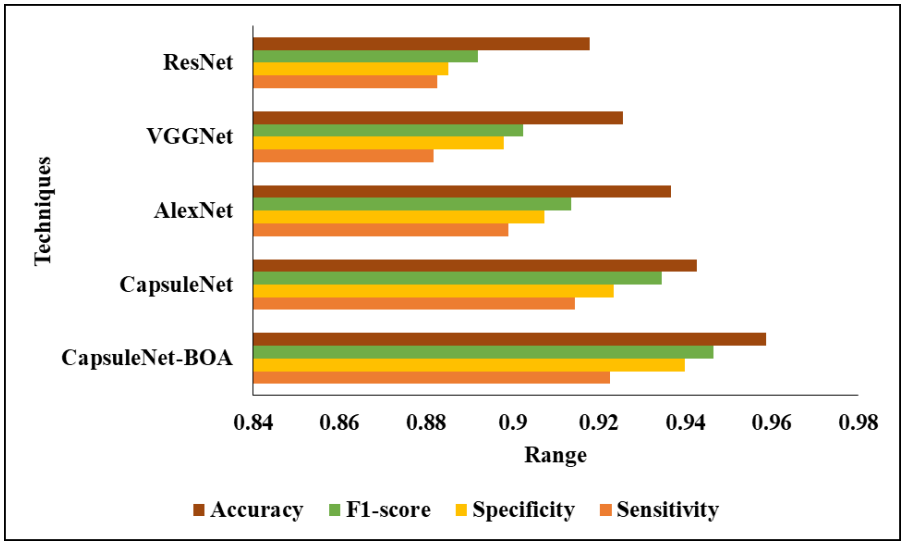

Figure 4: Study of different feature extraction

The CapsuleNet-BOA model consistently outperforms all other classifiers across all metrics, achieving the highest Sensitivity (0.9226), indicating superior ability to correctly identify diseased regions. Among the baseline models, CapsuleNet (without BOA) outperforms traditional CNNs, achieving an F1-score of 0.9345 and an accuracy of 0.9427, demonstrating the advantage of capsule-based architectures. AlexNet, VGGNet, and ResNet follow, with AlexNet slightly ahead across all metrics. VGGNet and ResNet achieve competitive yet lower accuracies of 0.9255 and 0.9178, respectively. These findings affirm that CapsuleNet-BOA significantly enhances segmentation performance, particularly through BOA integration, making it an excellent choice for disease classification in agricultural image analysis.

4.2. Validation Analysis of Projected Classifier

Figure 5 presents a comparative study of projected procedures across diverse metrics, and Figure 6 provides a visual analysis of the proposed fine-tuning optimiser.

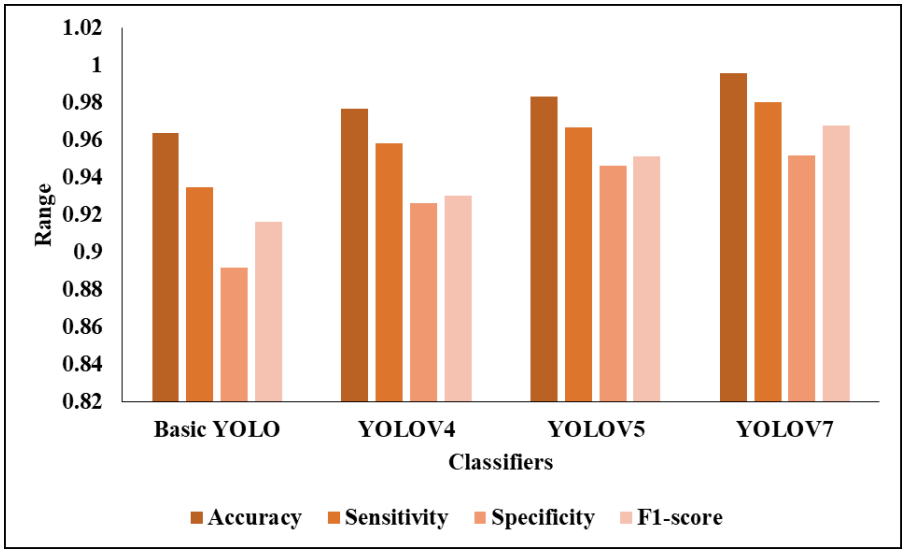

Figure 5: Analysis of projected classifier

A classification performance comparison of various YOLO-based models—Basic YOLO, YOLOv4, YOLOv5, and YOLOv7—across four key metrics: Accuracy, Sensitivity, Specificity, and F1-score. Among all models, YOLOv7 achieves the highest overall performance, with a remarkable accuracy of 0.9956, sensitivity of 0.9804, specificity of 0.9519, and F1-score of 0.9677. This indicates that YOLOv7 is highly effective in properly classifying both positive (e.g., disease-infected) and negative (e.g., healthy) instances with minimal misclassifications. Following YOLOv7, YOLOv5 also demonstrates strong classification performance, achieving 0.9834 in accuracy, 0.9667 in sensitivity, and 0.9513 in F1-score, confirming its balanced performance across detection and precision. YOLOv4 performs with a moderate accuracy of 0.9769 and an F1-score of 0.9304, while Basic YOLO lags comparatively, though still maintains a decent accuracy of 0.9637, besides an F1-score of 0.9163. These results clearly establish YOLOv7 as the superior model for object classification in the tested scenario, owing to its refined architecture and better trade-offs between speed and precision. The consistent improvement across YOLO versions also highlights the progressive enhancements in detection mechanisms and optimisation strategies.

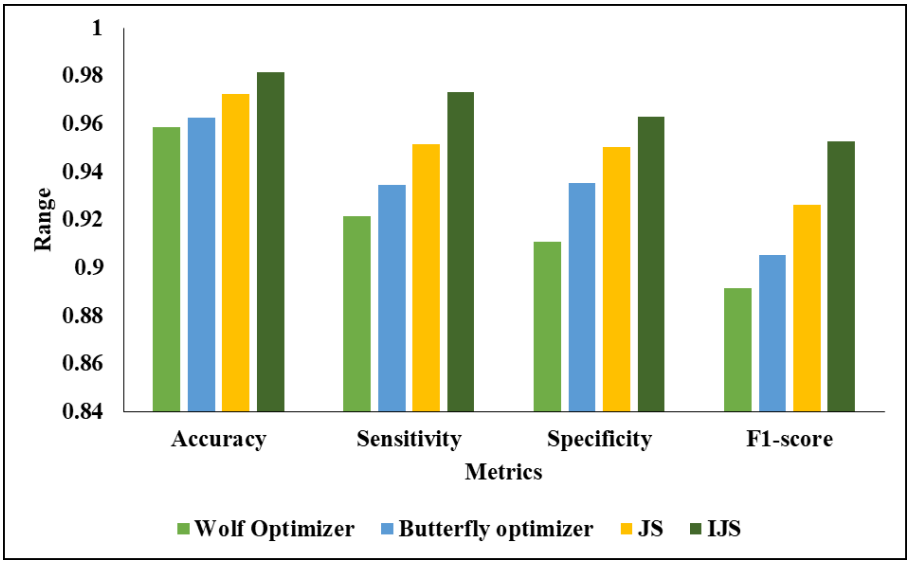

Figure 6: Visual analysis of the proposed optimiser

A comparative analysis of various optimisation algorithms—Wolf Optimiser, Butterfly Optimiser, JS (Jaguar Search), and IJS (Improved Jaguar Search)—based on four classification metrics: score. The Improved Jaguar Search (IJS) optimiser significantly outperforms all others, achieving the highest accuracy of 0.9816, a sensitivity of 0.9734, a specificity of 0.9628, and an F1-score of 0.9527. This indicates that IJS enables more precise learning and classification, yielding better negative detection with minimal error. Wolf Optimiser shows the least favourable performance, with the lowest accuracy (0.9587) and

F1-score (0.8916), highlighting relatively lower consistency in optimising model predictions. Overall, this comparison clearly establishes IJS as the most effective optimiser, demonstrating its ability to fine-tune model parameters with high efficiency, thereby improving detection robustness across all evaluated metrics.

5. Conclusion

The purpose of this research was to use the Kaggle sunflower leaf disease dataset to create a framework for efficient and reliable disease identification in sunflower leaves. The suggested method maximises performance by combining the YOLOv7 object detection architecture's characteristics with a Capsule Network's enhanced feature extraction capabilities and a hybrid metaheuristic approach. Furthermore, the BOA was used to fine-tune key hyperparameters, including learning rate, momentum, and batch size. This enabled a balanced exploration and exploitation of the hyperparameter space, ensuring optimal training dynamics. To further enhance detection accuracy and localisation in real-world scenarios, these elements were subsequently incorporated into the YOLOv7 model. To improve YOLOv7's detection process and optimise network parameters and anchor boxes using the IJSA. Both the convergence time and the model's generalizability were much improved by this adjustment. The inference speed of the suggested hybrid model was all advanced than that of traditional detection methods, according to the experimental data. The model successfully distinguished between several disease types, regardless of lighting or background conditions, thanks to the Capsule Network and the dual-optimisation method. Finally, by utilising IJSA and BOA to enhance the YOLOv7-CapsuleNet framework, a robust, scalable approach for real-time disease sensing in sunflower leaves has been provided. This opens the door to automated monitoring systems in precision agriculture. Deploying the system on edge devices and doing real-time field monitoring to aid in early disease diagnosis and intervention could be explored in future research.

Acknowledgment: The authors acknowledge the support and guidance provided by Karunya Institute of Technology and Science, Saranathan College of Engineering, K. Ramakrishnan College of Technology, and the University of Illinois at Urbana throughout this research. The contributions and encouragement from these institutions were instrumental in the successful completion of the study.

Data Availability Statement: The data supporting the findings of this study are available from the corresponding author upon reasonable request. Due to privacy and ethical considerations, some data may be restricted or require additional permissions. All requests will be reviewed in accordance with institutional guidelines and data-sharing policies.

Funding Statement: This research was conducted without any external financial support or funding.

Conflicts of Interest Statement: The authors declare that there are no conflicts of interest associated with this work.

Ethics and Consent Statement: The study was conducted following established ethical standards and received approval from the relevant institutional review boards of all participating institutions. Informed consent was obtained from all participants prior to their involvement.

References

1. J. Joshi, M. Aeri, V. Kukreja, and S. Mehta, “Revolutionizing in agriculture: Federated CNN models for sunflower leaf diseases,” *in Proc. 2024 11th Int. Conf. Reliability, Infocom Technol. Optim. (ICRITO)*, Noida, India, 2024.
2. V. Kiyan, A. Smagulova, Y. Kukhar, T. Savin, A. Bekenova, and R. Uakhit, “Morphological and molecular characterization of bacterial pathogens associated with leaf mottle of sunflower in Northern Kazakhstan,” *Plant Disease*, vol. 108, no. 2, pp. 264–269, 2024.
3. L. Centorame, A. Ilari, A. D. Gatto, and E. F. Pedretti, “A systematic review on precision agriculture applied to sunflowers: The role of hyperspectral imaging,” *Comput. Electron. Agric.*, vol. 222, no. 7, p. 109097, 2024.
4. A. Dolatabadian, T. X. Neik, M. F. Danilevicz, S. R. Upadhyaya, J. Batley, and D. Edwards, “Image-based crop disease detection using machine learning,” *Plant Pathology*, vol. 74, no. 1, pp. 18–38, 2024.
5. R. K. Dubey and D. K. Choubey, “Reliable detection of blast disease in rice plant using optimized artificial neural network,” *Agron. J.*, vol. 116, no. 3, pp. 1099–1111, 2023.
6. E. Saraswathi and J. F. Banu, “A novel probabilistic intermittent neural network (PINN) and artificial jelly fish optimization (AJFO)-based plant leaf disease detection system,” *J. Plant Dis. Prot.*, vol. 131, no. 2, pp. 587–600, 2024.
7. O. Yeni, M. Şen, S. Hasançebi, and N. T. Kara, “Optimization of loop-mediated isothermal amplification assay for sunflower mildew disease detection,” *Sci. Rep.*, vol. 14, no. 1, p. 23224, 2024.

8. H. Sharma, V. Kukreja, S. Mehta, N. Chandran, and A. Garg, “Plant AI in agriculture: Innovative approaches to sunflower leaf disease detection with federated learning CNNs,” *in Proc. 2024 5th Int. Conf. Emerging Technol. (INCET)*, Belgaum, India, 2024.
9. H. Zhou, W. Li, P. Li, Y. Xu, L. Zhang, X. Zhou, Z. Zhao, E. Li, and C. Lv, “A novel few-shot learning framework based on diffusion models for high-accuracy sunflower disease detection and classification,” *Plants*, vol. 14, no. 3, p. 339, 2025.
10. S. Parez, N. Dilshad, and J. W. Lee, “A channel attention-driven optimized CNN for efficient early detection of plant diseases in resource-constrained environment,” *Agriculture*, vol. 15, no. 2, p. 127, 2025.
11. P. Pavithra and P. Aishwarya, “Plant leaf disease detection using hybrid grasshopper optimization with modified artificial bee colony algorithm,” *Multimedia Tools Appl.*, vol. 83, no. 8, pp. 22521–22543, 2024.
12. Y. Ünal and M. N. Dudak, “Deep learning approaches for sunflower disease classification: A study of convolutional neural networks with squeeze and excitation attention blocks,” *Bitlis Eren Univ. Sci. J.*, vol. 13, no. 1, pp. 247–258, 2024.
13. M. P. Patil and I. S. Borse, “Comprehensive Review on Plant Disease Detection and Identification,” *Journal of Electrical Systems*, vol. 20, no. 3, pp. 732-746, 2024.
14. U. Sara, A. Rajbongshi, R. Shakil, B. Akter, S. Sazzad, and M. S. Uddin, “An extensive sunflower dataset representation for successful identification and classification of sunflower diseases,” *Data in Brief*, vol. 42, no. 6, p. 108043, 2022.
15. S. Arora and S. Singh, “Butterfly optimization algorithm: A novel approach for global optimization,” *Soft Comput.*, vol. 23, no. 3, pp. 715–734, 2019.
16. D. D. Yuan, M. Li, H. Y. Li, C. J. Lin, and B. X. Ji, “Wind power prediction method: Support vector regression optimized by improved jellyfish search algorithm,” *Energies*, vol. 15, no. 17, p. 6404, 2022.