[1] Zahra Akeshteh

[2*]Parvaneh Asghari

[3]Seyyed Hamid Haji Seyyed Javadi

[4]Hamidreza Navidi Ghaziani

An Automatic Plants Leaf Disease Identification Method Based on A New Proposed Optimized Convolutional Neural Networks Using Symbiotic Organism Search Algorithm

Abstract: - This study presents a novel method for accurately and efficiently classifying plant leaves based on different diseases and health conditions. The approach combines deep learning (DL) and optimization techniques. The model's hyperparameters are optimized using the Symbiotic Organism Search (SOS) algorithm, while a convolutional neural network (CNN) is employed for disease classification. CNNs are effective in identifying distinctive features that aid in distinguishing various classes. The Plant Village dataset is utilized, and data augmentation is performed to enhance the model's performance. The optimized CNN, which does not require segmentation, achieves a classification accuracy of 97.29% for the test set (8,777 images) and 99.89% for the training set (70,295 images). Precision and recall values of 97.33% and 97.32% are attained, respectively. The proposed method is compared to primary CNN models on the Leaf disease database, as well as two well-known pre-trained networks (VGG-16 and VGG-19), using five performance measures. The results demonstrate the superior performance and effectiveness of the developed method. This research contributes to a fast and accurate approach for leaf disease classification, with potential applications in other image classification tasks and benefits for agriculture, orchards, and plant disease identification. Furthermore, a comparison with three similar studies reveals that the proposed network achieves higher accuracy with fewer trainable parameters and computational requirements.

Keywords: Plant Leaves; Diseases; Classification; Convolutional Neural Network; Symbiotic Organism Search Algorithm.

I. INTRODUCTION

One of the factors that affect the production of food and agricultural products is plant disease. Plant disease can reduce the economic productivity of agriculture and in some cases, it can also prevent the continuation of the activity. With early diagnosis of plant diseases and dealing with them properly, production losses can be reduced and sustainable agriculture can be carried out. Accurate identification of plant diseases and appropriate information can reduce the damage caused by them [1]. According to Miller et al. [2], traditional methods for identifying plant diseases are limited in their ability to cover large areas and provide essential information for decision-making. Precision agriculture, on the other hand, leverages image acquisition, image processing, image segmentation, feature extraction, and machine learning techniques to accurately identify plant diseases from images. This system provides information about the diagnosed disease to the farmer. Automation of the disease detection system is necessary to accelerate product diagnosis [3-5]. The traditional approaches to machine learning involve several steps in building a model, including mathematical modeling, data collection and cleaning, feature extraction and selection, model training, optimization, and evaluation. In contrast, deep learning consolidates multiple processing stages into a single unit, reducing the need for costly feature extraction and design. Deep learning requires sufficient data and appropriate hyperparameters to effectively model complex data and extract features through processing layers. This makes deep learning applicable to various fields, including pattern recognition, language modeling, signal processing, and optimization, as it can handle different types of data such as text, images, and sound.

The progress in neuroscience, high-performance computing technology, and the availability of big data have facilitated the application of deep learning methods in artificial intelligence, enabling them to learn and perform intricate computations. These methods extract features from both structured and unstructured data and learn to represent those using supervised and unsupervised techniques. While deep learning has shown significant advancements in research and practical applications, it still faces challenges in the design and configuration of hyperparameters. In most cases, deep neural networks require human intervention to design their architecture and

[1] Department of Computer Engineering, Borujerd Branch, Islamic Azad University, Borujerd, Iran.

2Department of Computer Engineering, Central Tehran Branch, Islamic Azad University, Tehran, Iran.

3 Department of Computer Engineering, Shahed University, Tehran, Iran.

4Department of Mathematics and Computer Science, Shahed University, Tehran, Iran.

*Corresponding author

fix deficiencies. Training these networks necessitates a substantial volume of labeled data. This data costs a lot and consumes a lot of resources. In addition, the parameters of this type of network should be set by researchers. In recent decades, the demand to find innovative solutions based on deep learning has increased. This demand is in areas such as e-commerce, industrial control, bioinformatics [6], robotics [7], and healthcare [8].

Deep learning, as the extendable capabilities of artificial intelligence networks, will create better learning capabilities and are now established. These types of networks can automatically extract features by using a significant quantity of non-linear filters before the decision-making stage [9, 10]. One of the most popular artificial intelligence techniques is artificial neural networks. This particular neural network architecture draws inspiration from the structure and functioning of the human brain, making it suitable for modeling and analyzing large datasets to identify intricate relationships [9, 11, 12]. Convolutional Neural Networks (CNN) is widely recognized as one of the most widely adopted deep learning architectures. These networks are mostly used to classify images. The extraction of image patterns is done automatically in this method and there is no need to manually extract the features. This automatic extraction has made CNN popular. Another advantage of this method is its high accuracy and flexibility. The previously created model can be used for retraining and new diagnosis. This creates an optimal model architecture, which can recognize objects. A CNN may have hundreds of layers. The layers within CNNs can identify and understand patterns within images. Information extraction is achieved through the utilization of filters, which are applied to each training image. The output of these filters serves as the input for subsequent layers. In summary, CNNs typically comprise an input layer, an output layer, and multiple hidden layers [9].

The hyperparameters, including batch size, learning rate, and dropout rate, play a crucial role in the formation of a neural network and must be defined before its creation. The selection of these hyperparameters significantly influences the performance of a deep learning model. However, determining the optimal structure and hyperparameter values is a challenging and time-intensive task. Considering the importance of determining hyperparameters in deep learning algorithms, performing this step is an important task [13]. There are various methods such as manual search [14], random networks, and Bayesian to optimize hyperparameters. In the past, when dealing with smaller datasets and less complex neural networks, manual search methods were sufficient for determining hyperparameters. However, with the advent of big data, the complexity of neural networks has increased, necessitating the use of more sophisticated models and techniques. For this reason, it was possible to determine hyperparameters manually by experts. As big data has become more prevalent, manual determination of hyperparameters has become impractical, leading to the emergence of a new research field focused on the automatic determination of these parameters. [15]. Lorachelle used a network search algorithm [16]. The network search algorithm is a computational approach that aims to strike a balance between system performance and computational efficiency by determining hyperparameters. However, when dealing with large datasets, there is a challenge of reduced search efficiency due to the problem of exponential explosion[13, 17]. Bergstra [18] suggested an easy method for a random search. However, this approach lacks adaptability and is considered inefficient in terms of effectiveness. In sequential models[19, 20], Bayesian optimization [21-24] is considered a traditional approach for setting hyperparameters This method can use all the information from the previous searches [25], but it only samples around the optimal point. Therefore, it can be used in complex problems and performs better than manual and random searches. Heuristic algorithms work based on experience [25]. These algorithms can provide a solution with an acceptable cost by considering space and location calculations. These methods have been widely used to adjust hyperparameters. Sun [26] has used the Simulated Annealing high-speed algorithm for neural network clustering. Zhang [27] utilized the Genetic Algorithm to enhance the performance of the deep learning network, resulting in a high convergence rate. However, a common issue with heuristic algorithms is their inability to effectively navigate locally optimal points [13].

In this study, the features are acquired through an automated process using a CNN rather than manual detection. This approach enables efficient classification of leaf diseases, resulting in time and cost savings, along with promising accuracy and predictive performance. To further analyze the effectiveness of various CNN architectures and datasets, it is necessary to compare their performance differences and evaluate the generalization capability and training errors of different methods. This study describes a new metaheuristic optimization algorithm namely, Symbiotic Organisms Search (SOS), which replicates the symbiotic interaction strategies employed by organisms to thrive in their respective ecosystems. Based on the search by the authors of this study, no previous studies on plant leaf disease detection using CNN and SOS were found that dealt with simultaneous

detection in different plants. The only study found in this field was related to Pandey et al. [28], the researchers developed a novel CNN module named AResNet-50, which effectively detects leaf diseases by incorporating the attention residual learning (ARL) strategy into the standard ResNet-50 CNN model. To mitigate the risk of accuracy decline resulting from improper selection of training meta-parameters, they utilized the opposition-based symbiotic search algorithm (OSOS) to optimize the learning rate and momentum values during the implementation training phase. The study involved analyzing RGB images of citrus, guava, mango, and eggplant leaves captured in real-world or practical environments. The available case was limited to four plants including citrus, guava, mango, and eggplant. The difference between this study and the mentioned study is that the number of plants examined in the present study is much more than in the previous study. Also, the SOS algorithm has been used directly to adjust the hyperparameters. While in the mentioned study, the OSOS algorithm was used, which can affect the results. A major benefit of the SOS algorithm, in comparison to other metaheuristic algorithms, is that it does not necessitate any specific algorithm parameters.

The rest of this paper is organized as follows: in Section 2, a comprehensive review of relevant literature is presented. Section 3 outlines the utilized CNN method and provides an overview of the Plant Village database. Experimental results and analysis are presented in Section 4. The findings of this study are discussed in Section 5, along with concluding remarks.

II. RELATED WORK

To increase productivity and reduce losses, detecting diseases in plant leaves is an important challenge in smart agriculture. Automatic detection of plant diseases can simplify disease control and increase crop production. Some previous studies to use machine learning techniques for this purpose are reviewed in this section.

In [29], the authors suggested the use of a transfer learning technique on a pre-trained network. Their model is called TL-ResNet 50. The proposed model can diagnose plant disease with high accuracy. Also, in the study [30], attention-based CNNs were used to classify tomato diseases. In this study, the authors compared the results with standard transfer learning techniques.

In [31], the authors introduced a model called "PlantXViT" that combines Convolutional Neural Networks with Vision Transformers to effectively identify various plant diseases across multiple crops. This proposed model has a lightweight architecture with only 0.8 million trainable parameters, making it well-suited for IoT-based smart agriculture applications. The performance of PlantXViT is evaluated on five publicly available datasets, demonstrating an average accuracy surpassing 93.55% for Apple, 92.59% for Maize, and 98.33% for Rice in plant disease recognition.

In [32], the objective was to identify and categorize diseases that may impact olive leaves. The findings were promising and demonstrated the successful integration of CNN and vision transformer models. According to the experimental results presented in their research, the proposed model exhibited superior performance compared to other models, achieving an accuracy of approximately 95% for classification.

In [33], the authors introduced an enhanced method for computing features using Squeeze and Excitation Networks in combination with Capsule Networks (CapsNet) for plant disease classification. They have utilized two SE networks, one based on AlexNet and another on ResNet, to estimate the severity of Late Blight disease in Tomato crops using leaf images from the PlantVillage dataset. The images were categorized into four severity stages: healthy, early, middle, and end. These images were downscaled, enhanced, and used as input for the SE networks. The feature maps generated by the two networks were then separately fed into the Capsule Network for classification. The performances of these models were compared with the original CapsNet using two different image sizes: 32X32 and 64X64. The results showed that SE-Alex-CapsNet achieved the highest accuracy of 92.76%, while SE-Res-CapsNet achieved the highest accuracy of 94.4% with the 64X64 image size, outperforming the original CapsNet's accuracy of 85.53%. Based on these findings, the proposed techniques can be utilized for assessing disease severity in other crops and can be extended to other applications such as plant species classification and weed identification.

In [34], a lightweight Convolutional Neural Network (CNN) called "VGG-ICNN" was introduced for the identification of plant diseases using leaf images. VGG-ICNN consists of approximately 6 million parameters, which is significantly less than most high-performing deep learning models. The model's performance is evaluated

on various public datasets that encompass a wide range of agricultural products. These datasets include apple, maize, and rice, each with four, four, and five categories, respectively. Experimental results demonstrate that this approach achieves better accuracy in identifying diseases in apple, maize, and rice compared to some recent deep-learning approaches, with accuracies of 94.24%, 91.36%, and 96.67%, respectively. Furthermore, the model consistently performs well on all the datasets used in this study, outperforming some recent lightweight CNN models.

In [35], the authors introduced a model that improves the accuracy and efficiency of disease identification in plant leaves compared to existing approaches. The study trained standard models like AlexNet, VGG, and GoogleNet, along with a proposed model, using the Night shed plant leaf dataset available in the plant village. The dataset consists of 9 categorical classes representing diseases and healthy plant leaves. Various parameters, including batch size, dropout, learning rate, and activation function, were examined to evaluate the models' performance. The proposed model achieved an impressive disease classification accuracy rate ranging from 93% to 95%. The accuracy tests indicate that the suggested model shows promise and could significantly enhance the speed and accuracy of identifying disease-infected leaves.

The authors in [36], utilized three pre-trained deep learning models, namely MobileNetV2, EfficientNetB6, and NasNet, for transfer learning on a dataset of bean leaf images. The dataset consists of 1295 images categorized into three different classes. Additionally, various optimization techniques are employed to evaluate the performance of different CNN models. The experimental results reveal that EfficientNetB6 outperforms the other models with an accuracy of 91.74%. This finding highlights the significance of selecting an appropriate model for disease classification. Moreover, the study demonstrates the impact of different optimizers on CNN models, providing valuable insights for future research in this field.

In [37], the proposed CNN models underwent fine-tuning, followed by hyperparameter optimization using the particle swarm optimization algorithm (PSO). The weights of these architectures were further optimized through the grid search method, resulting in the creation of triple and quintuple ensemble models. The datasets were classified using five-fold cross-validation. The experimental results highlight the exceptional performance of the proposed ensemble models, demonstrating fast training and testing times and achieving a superior classification accuracy of 96.87%. This research contributes to the early detection of plant diseases simply and efficiently, aiding experts in preventing the spread of infections.

Table 1 presents a summary of relevant previous studies that focused on using neural networks for the identification of plant leaf diseases.

Table 1. The summary of relevant previous studies.

Author	Year	Subject	Dataset	Plant	Accuracy	Network Type
(Bhujel et al.[30]	2022	Tomato Leaf Disease Classification	Plant Village	Tomato	98.74	CNN-Standard ResNet50
H.-C. Chen et al.[38]	2022	Tomato Leaf Disease Classification	Kaggle	Tomato	96	AlexNet CNN
Poornima Singh Thakur[31]	2022	Plant disease identification	PlantVillage and other	Apple, Maize, Rice	93.55%(Apple), 92.59% (Maize), 98.33% (Rice)	Combined CNN and ViT
Hamoud Alshammari [32]	2022	Olive disease classification	Self Dataset	Olive	0.95	Transformer (ViT)
Shradha Verma[33]	2022	Plant disease severity	aka Solanum Lycopersicum	Tomato	0.94	Capsule networks
Poornima Singh Thakur[34]	2023	Crop disease identification	PlantVillage and other	Multiple	0.96	VGG-ICNN
Barkha M. Joshi[35]	2023	Leaf disease classification	PlantVillage	Multiple	0.95	Night-CNN

Vimal Singh[36]	2023	Classification of plant leaf disease	Self Dataset	Beans	0.91	Fine Tuned CNN
Hasan Uluta¸s[37]	2023	Detection of Tomato Leaf Disease	PlantVillage	Tomato	0.96	CNN

The subject of this study is in the category of new studies and due to the technology used in it, it has been the focus of researchers in the last few years. In 2022, in [39], A comprehensive review was conducted on 100 studies conducted within the past 5 years, specifically in the domain of plant disease detection utilizing CNNs. In his study, approaches, problems, databases, study limitations, model performance, and their comparison have been investigated. According to his research, the number of studies conducted on this topic has increased greatly in 2022. According to the search of the authors by performing a keyword search on journal articles published in ScienceDirect, Springer, MDPI, and Google Scholar databases in 2023 (until the date of preparation of this work), 24 researches related to leaf disease and using CNN have been found. The number of studies in different years is shown in Figure 1.

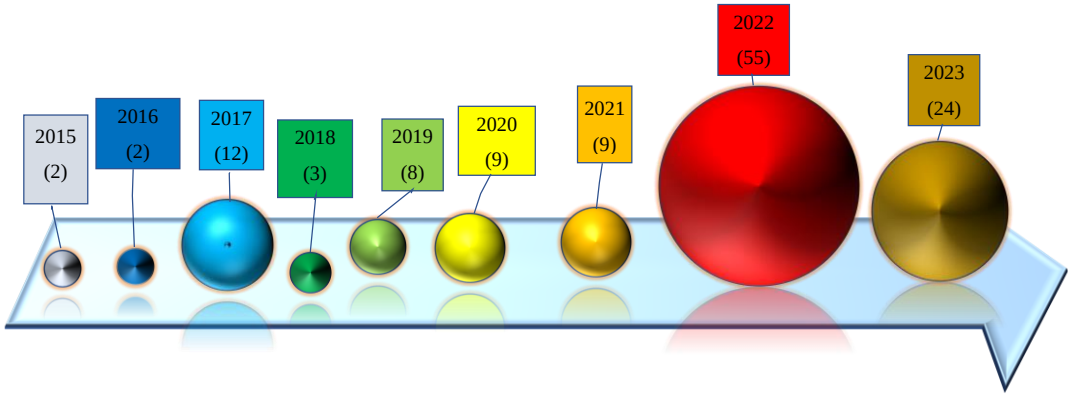


Figure 1. Distribution of the year in studies.

III. THE PROPOSED METHOD

The effectiveness of feature-based models in classification performance is heavily reliant on the manual extraction of features from the dataset. The expertise and supervision of the evaluator play a crucial role in extracting these features. Machine learning techniques are mostly unsuccessful in finding useful information among high-dimensional, irrelevant data [40]. On the other hand, deep learning models like CNN exhibit superior generalization capabilities and tolerance to noise. Unlike feature-based models, CNNs do not require human supervision for feature extraction, making them more reliable for unbiased classification. CNNs employ convolution and fusion operations to automatically extract features from input data. This study presents a CNN-based approach for the automatic classification of leaf diseases. Additionally, an optimization strategy is employed to optimize the hyperparameters of the CNN, facilitating faster convergence towards the optimal solution. Figure 2 illustrates the schematic of plant leaf disease image classification using the optimized CNN.

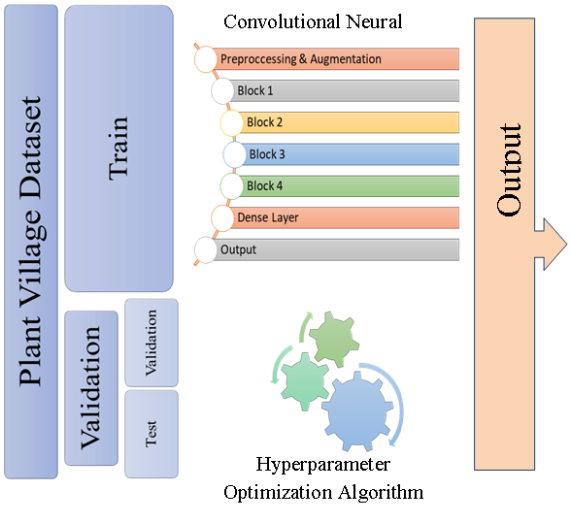


Figure 2. Schematic of leaf disease image classification.

Each of the steps of the model and its details are explained in the following sections. In this model, after dividing the data into several categories, pre-processing and data augmentation operations have been performed on them. In the next step, the training of the network is done using the train's data. Optimization of hyperparameters is done with a meta-heuristic algorithm and using validation data. Finally, the output of different models is evaluated. Moreover, it enhances the precision and dependability of the hyperparameters employed within the CNN framework. The outlined framework comprises of four primary stages: database preprocessing and augmentation, design of the CNN architecture, optimization of hyperparameters, and classification using the proposed model. The effectiveness of the optimized CNN is assessed through the computation of diverse metrics, and the outcomes are juxtaposed with those obtained from fundamental deep learning models as well as two renowned pre-trained networks (VGG-16 and VGG-19) to evaluate the model's performance.

Plant Village dataset

The first step to good agricultural yield is protecting the plants from diseases. Early plant disease recognition and prevention is the first step in this regard. However manual disease recognition is time-consuming and costly. This is one of those use cases, where deep learning can be used proactively for great benefit. Using deep learning, plant diseases can be recognized very effectively. Large-scale plant disease recognition using deep learning can cut costs to a good extent. This dataset is recreated using offline augmentation from the original dataset. The dataset used in this study contains approximately 87,000 RGB images of crop leaves, which are classified into 38 distinct categories representing both healthy and diseased crops. To ensure proper training and validation, the entire dataset is partitioned into separate training and validation sets, maintaining an 80/20 ratio. During this partitioning process, the directory structure of the dataset is preserved [41]. The data available in Plant Village consists of two parts: Train and Validation. Each of these sections has 38 subcategories for leaf diseases (healthy leaves for each plant are also considered as a category). For the final evaluation of the obtained models, the validation data were divided into two categories. The first category, named the Test, contains 10% of the total data (about half of the validation data). These data are not used in any of the stages of training and optimization of the network and are used only after the finalization of the network to evaluate it and compare it with other networks under equal conditions. The number of images in the second category, which is called the Validation, is the same as in the previous category. These data were used for the validation and optimization of hyperparameters. In addition, in the evaluation of the networks, the performance of each of them will be used for all three categories of data, Train, Test, and Validation, and the results will be evaluated for all of them. In the optimization of the hyperparameters stage, the goal is to obtain the maximum value for accuracy. Before utilizing the Plant Village database, researchers often augment the dataset due to its limited or inadequate training data. Augmentation serves to expand the training dataset, enabling its effective utilization. In the context of classification tasks, this involves processing high-dimensional inputs, such as images, to generate relevant outputs. A proficient classifier exhibits strong performance across diverse scenarios. CNN, as a widely recognized framework for image data, possesses the ability to identify pertinent subtle features within an image while remaining resilient to significant uncorrelated

alterations [42]. For image datasets, this enhancement can be performed by changing a few pixels in the images to improve generalizability and prevent the model from memorizing the training data.

When it comes to enhancing data, various transformations can be employed, such as scaling, zooming, and rotating the image in multiple directions. Augmentation plays a crucial role in improving the accuracy of classification, regardless of the image's size, position, or level of distortion. In the next step, preprocessing and data augmentation are performed on the input images. For this purpose, rescale, shear, flip and zoom are used. The values of these methods are shown in Table 2.

Table 2. Preprocessing and augmentation data.

No.	Method	Value
1	Rescale	1/255
2	Shear range	0.2
3	Zoom range	0.2
4	Horizontal flip	True

When adopting the augmentation approach, it is important to exercise caution to avoid altering the correct class through improper transformations [43].

Resized images possess three dimensions, namely height, width, and color depth, which are also referred to as the number of channels. The color depth represents the three colors of an RGB image: red, green, and blue. To convert the RGB channel to grayscale, a monochrome channel with gray levels is utilized. To handle the computational workload of networks, it is common to run the model on graphics processing units (GPUs) as CPUs may experience slowdowns or struggle with the task. GPUs offer the advantage of having a large memory bandwidth. Originally designed for graphics applications in video games, GPU hardware was later adapted for use in network computing. The specific hardware specifications consist of 12.681GB RAM and an NVIDIA GeForce RTX 3090 GPU.

Convolutional neural networks

Different types of neural networks exist, such as fully connected networks (FCNs) and semi-connected networks. In FCNs, every input neuron in a layer is connected to all neurons in the subsequent layer. However, the downside of FCNs is their high computational cost. When additional layers are added, the number of network parameters increases rapidly. On the contrary, convolutional neural networks (CNNs) have sparse connections in their structure, making them more efficient. CNNs are specifically designed for datasets with network structure topologies, commonly used for two-dimensional images, and have been successfully applied in tasks like visual classification, pattern recognition, segmentation, and related areas. These networks excel at identifying spatial and temporal relationships within an image, focusing on the most relevant information for processing. Like other neural networks, CNNs learn by adjusting weights and biases. However, a key advantage of CNNs is their ability to recognize learned patterns regardless of their location in the image. In contrast, an artificial neural network would need to relearn a pattern if its location changes [44].

Convolutional neural networks (CNNs) possess a distinctive attribute compared to traditional machine learning approaches, namely the incorporation of convolutional layers. Unlike dense layers in machine learning, which learn global patterns encompassing all pixels in the input feature space, convolutional layers specialize in acquiring local patterns within an image. These layers partition the image into smaller regions to capture details like edges, textures, and other local features. CNNs adopt a sparse structure, enabling them to diminish less significant features while retaining meaningful ones, such as edges that occupy fewer pixels. This reduction in unnecessary computations and memory usage significantly bolsters the computational efficiency of the model. Moreover, CNNs excel at deeply comprehending intricate visual concepts by uncovering a partial hierarchy of patterns. For instance, the initial convolutional layer may grasp small local patterns like edges, while subsequent convolutional layers grasp larger patterns composed of the features extracted in preceding layers, thus establishing a hierarchical progression [44].

Convolution operation

Convolution refers to a mathematical operation that involves the combination of two functions by summing or integrating them, while one function is shifted over the other. In the context of one-dimensional convolution, it can be envisioned as a square pulse moving along the x-axis towards another square pulse. The convolution is calculated at each point where the leading edge of the moving pulse aligns with the stationary pulse. Before performing the convolution operation, one of the functions is flipped concerning the independent variable. Convolutional layers in neural networks utilize the integration of local information from neighboring pixels, resulting in a reduction in the dimensionality of the feature map. This property makes convolutional layers particularly effective for classification tasks. The output of a convolutional layer is a single feature vector that contains predictions for each class. Images encompass various features and information, including edges, textures, shapes, and objects. Geometric filters, such as those detecting edges, circles, or squares, require deeper networks in subsequent layers to identify specific objects. In a convolutional network, the input is represented as a matrix where each element corresponds to an individual pixel in the image. This matrix is then passed through a convolutional layer, where a selected filter convolves across specific regions of pixels. The stride, determining the number of pixels the filter is shifted in each step, governs the movement of the filter. Each block of activated outputs in the convolutional layer undergoes an activation function computation. The resulting predictions are subsequently subsampled in pooling layers, automatically reducing the dimensions of the input data.

Max pooling operation

Pooling layers are responsible for reducing the dimensions of feature maps to enhance the efficiency of learning computations within the network. While the convolutional layer generates the feature map, the pooling layer summarizes the features within specific regions of the map. This pooling process increases the model's robustness to variations in the positional placement of features within the input image. In CNNs, the Max pooling operation aggressively downsamples the feature maps, similar to stride convolutions. This downsampling helps reduce the number of features that need to be processed. As a result, successive convolution layers can cover a broader region of the image, creating spatial filter hierarchies. By encoding patterns over different tiles, feature maps can better explore the maximal presence of various features, as opposed to the average presence. This approach prevents the loss or dilution of feature information within the map.

The proposed CNN architecture

The framework used in this study is Keras. In this study, a CNN network is designed from scratch. For this purpose, it is inspired by the VGG-19 network. Simonian and Zisserman [45], introduced and developed Visual Geometry Group Network (VGG) based on CNN architecture at Oxford Robotics Institute[46]. In a global image recognition competition, VGG19 won first place in 2014 [45, 47]. The distinguishing feature of this network architecture is the connection of multiple convolutional layer modules to three fully connected layers[48]. One notable aspect of this network design is its inclusion of multiple convolutional layer modules that are linked to three fully connected layers[49]. Considering the advantages of this network, for the architecture of a network in the diagnosis of plant leaf disease, this structure has been inspired [50].

The input images of the designed network have dimensions of 64*64*3. Applying the dimensions of the input image to the network depends on its execution speed. The use of high image dimensions makes more features recognizable, while reducing the size of the image, may cause several features to be lost. On the other hand, the high dimensions of the input images cause the number of calculations to increase and the network execution speed to decrease. Therefore, the appropriate selection of the dimensions of the input image is determined according to the conditions and experience.

The proposed network has 4 blocks and each one has different layers. The specifications of the blocks are shown in Table 3. Lastly, the network incorporates an output layer consisting of a dense layer that employs a SoftMax function for feature classification and disease-type prediction. The Adam optimizer, with a learning rate of 0.001, is utilized in this network. The loss function employed is categorical cross-entropy.

Table 3. Parameter values of blocks in the designed CNN.

Parameter	Block 1	Block 2	Block 3	Block 4

Number of filters	128	256	256	512
Size of the kernels	2×2	2×2	2×2	2×2
Activation function	Relu	Relu	Relu	Relu
Padding	Same padding	Same padding	Same padding	Same padding
Stride	2	1	1	1
Size of Maxpooling	2×2	2×2	2×2	-
Dropout rate	0.1	0.1	0.1	0.5
Batch normalization	No	No	No	Yes
Flatten	No	No	No	Yes
Dense layer size	-	-	-	512

Hyperparameter optimization

Hyperparameter optimization is a process for finding the best values of hyperparameters. The selection of appropriate hyperparameters serves as a crucial component in defining the model's architecture. The behavior of the model is intrinsically linked to the process of solving a search problem to identify the optimal parameter combination. The optimum hyperparameters lead to achieving maximum performance on the data processing in a reasonable amount of time. This process has a vital role in the prediction accuracy of a deep learning algorithm. Hence, hyperparameter optimization is widely acknowledged as the most intricate aspect of constructing deep learning models. Many deep learning algorithms provide default values for their hyperparameters, but these defaults may not yield optimal performance across all projects involving deep learning. This is to need to optimize them, to get the right combination for the best performance. To automatically determine the optimal architecture, a variety of options are explored to minimize the predefined loss function using the provided independent data. The model parameters are not manually set and can be learned from the dataset. These parameters are internal and crucial.

The model hyperparameters cannot be estimated from the data. Hyperparameters refer to adjustable parameters that determine the final values of parameters in a network and dictate their respective values. In this study, the number of filters, size of the kernels, type of activation function, size of pooling in all blocks except the last block, and the dropout rate, dense layer size, and activation function in the output layer are considered. The values of hyperparameters are established before the initiation of the learning process as they are not updated during the learning phase. Although various techniques exist for tuning hyperparameters, such as trial and error, metaheuristic optimization algorithms, and Bayesian optimization, this study employs a metaheuristic approach for hyperparameter tuning. This approach will be discussed in the following section. Specific values for each of them have been used to optimize hyperparameters. These ranges are determined using previous studies and experience in determining hyperparameters. The ranges are shown in Table 4.

Table 4. The hyperparameters range.

Hyperparameter	Range/ Type
Number of filters	2^n , n= in range 2 and 7, with stride=1
Size of the kernels	2,3,4,5,7,9
Activation function	tanh, sigmoid, Relu, elu
Pooling size	2,3,4
Dropout rate	n/100 , n= in range 0 and 100, with stride=3
Dense layer size	2^n , n= in range 2 and 13, with stride=1

Symbiotic Organisms Search (SOS)

Organisms employ symbiotic interaction strategies as a means of survival and reproduction within ecosystems. In 2014, Cheng and Prayugo[51], introduced the SOS algorithm as a novel optimization technique, incorporating it into their algorithm. This approach is population-based and offers several benefits, including straightforward implementation, efficient computational time, and rapid convergence. Cheng and Prayugo[51], The SOS algorithm was assessed for its efficacy in solving various numerical mathematical problems through the solution

of numerous mathematical problems. Belonging to the category of meta-heuristic algorithms, the SOS algorithm draws inspiration from natural phenomena. Similar to other meta-heuristic algorithms, it leverages interactions observed in symbiotic relationships between organisms to identify suitable solutions. The algorithm begins by selecting a population of candidate solutions within the search space, which constitutes the initial ecosystem. This ecosystem is generated by randomly placing a group of organisms in the search space, with each organism representing a potential solution to the problem at hand. The quality of an organism's solution is determined by its fitness value. The algorithm proceeds by iteratively applying a sequence of operations to the initial solutions, generating new solutions in each iteration. This iterative process mirrors the working mechanism of other meta-heuristic algorithms. The SOS algorithm replicates the three-phase biological interactions observed between organisms in an ecosystem, namely mutualism (beneficial for both organisms), commensalism (beneficial for one organism and negligible effect on the other), and parasitism (beneficial for one organism while actively harming the other). These interactions occur randomly in each iteration. The process continues until the termination criteria are met, necessitating repetition of the algorithm [52].

3.4.1.1. The mutualism phase

X_i represents the i^{th} organism within the ecosystem, while X_j is randomly chosen from the ecosystem to interact with X_i. Both organisms have the potential to engage in a mutually beneficial relationship, aimed at enhancing their survival prospects. The calculation of new candidate solutions for X_i and X_j, based on this mutually beneficial interaction, is performed using Equation (1) and Equation (2), respectively:

$$X_i^{(k+1)} = X_i^{(k)} + rand(0,1)\left(X^{best} - \left(\frac{X_i^{(k)} + X_j^{(k)}}{2} \right) \times BF_1 \right) \quad (1)$$

$$X_j^{(k+1)} = X_j^{(k)} + rand(0,1)\left(X^{best} - \left(\frac{X_i^{(k)} + X_j^{(k)}}{2} \right) \times BF_2 \right) \quad (2)$$

Where k expresses the iteration number; Benefit factors of organisms X_i and X_j, are BF_1 and BF_2, respectively; X^{best} refers to the best solution found by the algorithm at that specific iteration k and $(0, 1)$ a random number is typically generated at each iteration of an algorithm. This random number is used to introduce randomness and diversity into the algorithm's operations, helping to explore different regions of the search space and potentially improve the chances of finding better solutions. If BF_1 is larger than BF_2, the first organism gets a more beneficial advantage than second organism. As a result, the benefit factors are assigned arbitrarily as either 1 or 2. The second term in Equations (1) and (2) represents the collaborative effort between the organisms to maximize their survival benefits. If the new fitness of the candidate solutions surpasses their previous fitness values, they are replaced with the current solutions.

3.4.1.2. The commensalism phase

This phase is similar to the previous one. Initially, a random organism X_j is chosen from the ecosystem to interact with X_i. In this interaction, X_i strives to benefit while X_j remains unaffected. This relationship is mathematically represented by Equation (3) as follows:

$$X_i^{(k+1)} = X_i^{(k)} + rand(-1,1)\left(X^{best} - X_j^{(k)} \right) \quad (3)$$

The second term in the aforementioned equation represents the advantageous contribution made by X_j to help X_i enhance its survival advantage in the ecosystem, specifically towards reaching the highest level achieved by the current best organism (represented as X^{best}). X_i is replaced with its new value only if its new fitness surpasses its previous fitness obtained before the interaction.

3.4.1.3. The parasitism phase

During this phase, organism X_i creates an artificial parasite known as the parasite vector (P_V) within the search space. The P_V is generated by duplicating organism X_i and randomly adjusting certain dimensions using a random number. Another organism, X_j, is selected from the ecosystem to serve as the host for the P_V. The P_V

attempts to replace X_j in the ecosystem, and the fitness of both organisms is calculated. If the P_V demonstrates a more favorable fitness value, it replaces X_j in the ecosystem. However, if X_j displays a better fitness value, it remains immune to the parasite, and the P_V is removed from the ecosystem. These three phases are repeated until a termination criterion is met.

To configure the SOS algorithm, the population size is set to 10, the maximum number of iterations is 50, and the number of dimensions is equal to the number of selected hyperparameters, which are 18. Additionally, the lower bound is set to zero, and the upper bound is determined based on a specific vector.

[5, 5, 3, 2, 5, 5, 3, 2, 5, 5, 3, 2, 5, 5, 3, 33, 11, 3].

Objective fitness uses the forward network for the evaluation of fitness. Based on the obtained values of hyperparameters, a fitness function to test the efficiency of the configured CNN architectures for plant leaf disease detection is considered. The fitness can be calculated by Equation (4) as follows:

$$Fitness = 1 - Accuracy_{Validation} \quad (4)$$

Which $Accuracy_{Validation}$ is the last 5 values of accuracy in the runs.

Optimized hyperparameters

The network's results are influenced by the variation of hyperparameters, as indicated in numerous studies. Discovering the best network performance can be achieved through various approaches. However, relying solely on expert knowledge in this field can be a time-consuming process. To address this, the SOS algorithm was employed in this research to identify the optimal values for the hyperparameters. Once the optimal hyperparameter values were determined, the network was run for 100 epochs. The optimized hyperparameters are presented in Table 5, which were subsequently utilized in the CNN. A total of 18 parameters were optimized in this study, including the number of filters, kernel size, activation function type, pooling size in blocks 1 to 3, dropout rate, dense layer size, and activation function in the output layer. The optimization ranges for each parameter were established based on previous studies and expert knowledge. The SOS algorithm is known for its ability to quickly identify the optimal solution, thereby reducing computational burden. On average, the SOS algorithm took approximately 10 minutes to complete 100 epochs.

Table 5. Initial and optimized hyperparameters value.

Block/output	Hyperparameter	Initial Value	Optimized Value
Block 1	Number of filters	128	128
	Size of the kernels	2	7
	Activation function	Relu	Relu
	Size of pooling	2	2
Block 2	Number of filters	256	128
	Size of the kernels	2	3
	Activation function	Relu	Relu
	Size of pooling	2	2
Block 3	Number of filters	256	128
	Size of the kernels	2	3
	Activation function	Relu	sigmoid
	Size of pooling	2	2
Block 4	Number of filters	512	128
	Size of the kernels	3	2
	Activation function	Relu	tanh
Output	Dropout rate	0.5	0.33
	Dense layer size	512	512
	Activation function	softmax	sigmoid

IV. EXPERIMENTAL RESULTS AND ANALYSIS

The CNN takes as input a training set consisting of images depicting various types of plant leaf diseases. By assigning weights to different concepts and objects within the images, the CNN can distinguish between the different disease and leaf categories. In the initial layer, the CNN acquires knowledge of basic features such as edges, dots, and variations in brightness. As the network consists of multiple layers, subsequent layers are capable of recognizing more complex objects, with the third layer being able to identify specific details like stains. To assess the performance of the CNN, the entire set of images is divided into three subsets: 70,295 images (80%) are allocated for training, 8,795 images (10%) for validation, and 8,777 images (10%) for testing.

Accuracy and loss plots

The accuracy and loss plots are generated during the training and testing processes, depicting the performance of the model over epochs. The accuracy curves offer insights into the achieved accuracy for both the training and testing datasets. If there is a significant difference between the accuracy of the training and testing phases, it suggests the presence of overfitting. The magnitude of this difference serves as an indicator of the extent of overfitting. Figure 3 illustrates that, in most instances, the training curve closely follows the test curve. However, after the initial few epochs, the test accuracy tends to be slightly lower than the training accuracy, indicating the possibility of mild overfitting. To address this, a dropout rate, determined during the optimization stage, is incorporated to mitigate the overfitting issue.

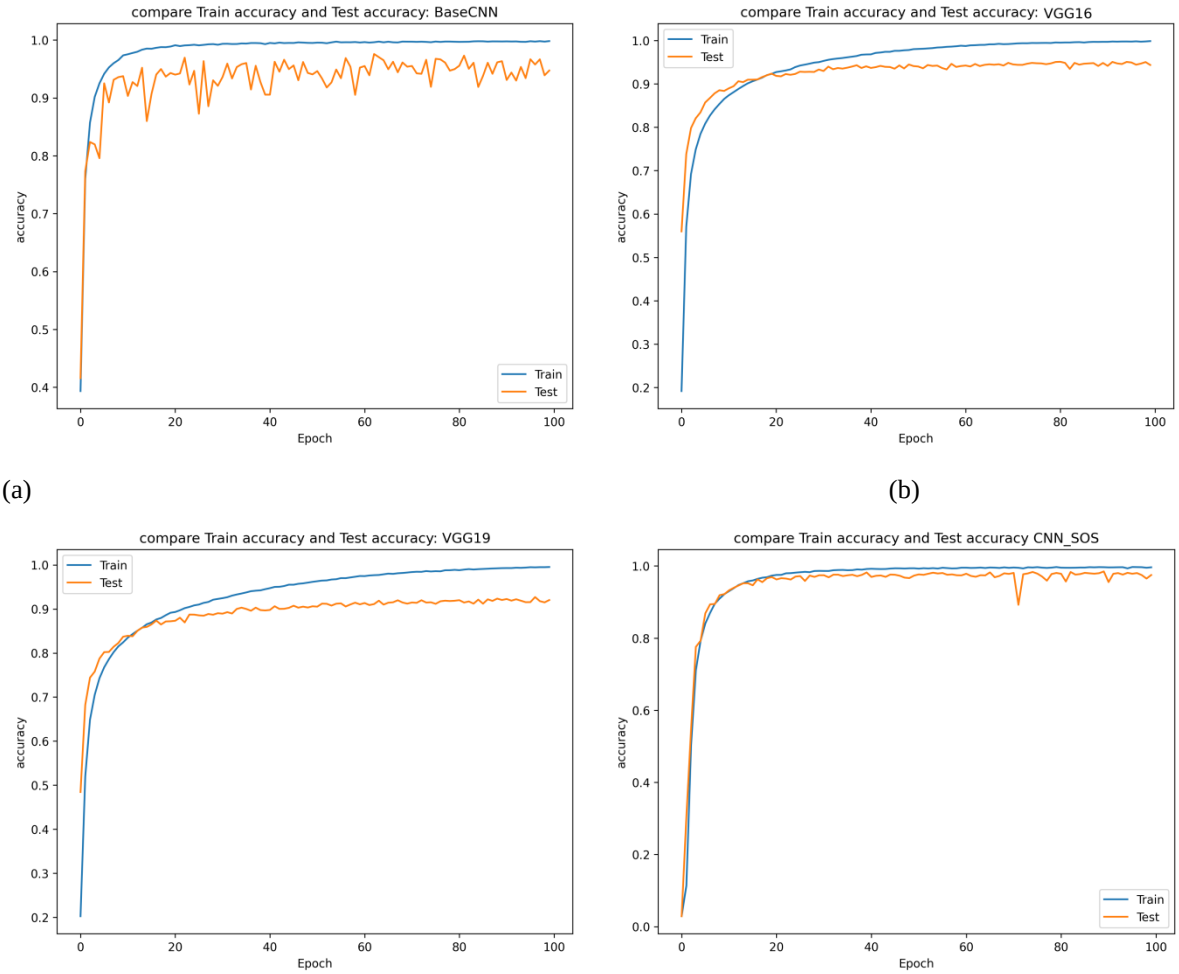


Figure 3. Graph of accuracy vs. epochs for the training and test sets (a) CNN Base, (b) VGG-16, (c) VGG-19 and (d) Optimized CNN.

In the CNN model, the accuracy of the training set steadily increased with the number of epochs and reached 99.6%, which is very close to 1. Similarly, the test accuracy reached 94.7% by the 100th epoch. These relatively high percentages demonstrate the model's capability to accurately identify various conditions in real-world scenarios.

The loss function is employed to assess the performance of the network on the training set. It indicates whether the network is progressing in the right direction. In this case, a categorical cross-entropy loss function is utilized to classify images into multiple classes, specifically identifying different types of diseases or healthy leaves.

To update the network parameters, the Adam optimization algorithm is employed, which utilizes the gradient of the loss function. During each epoch, the loss function is computed by evaluating the batches of data during the forward pass. Figure 4 depicts the improvement in the model's loss curve after each iteration of optimization, with the loss decreasing over time or after multiple iterations.

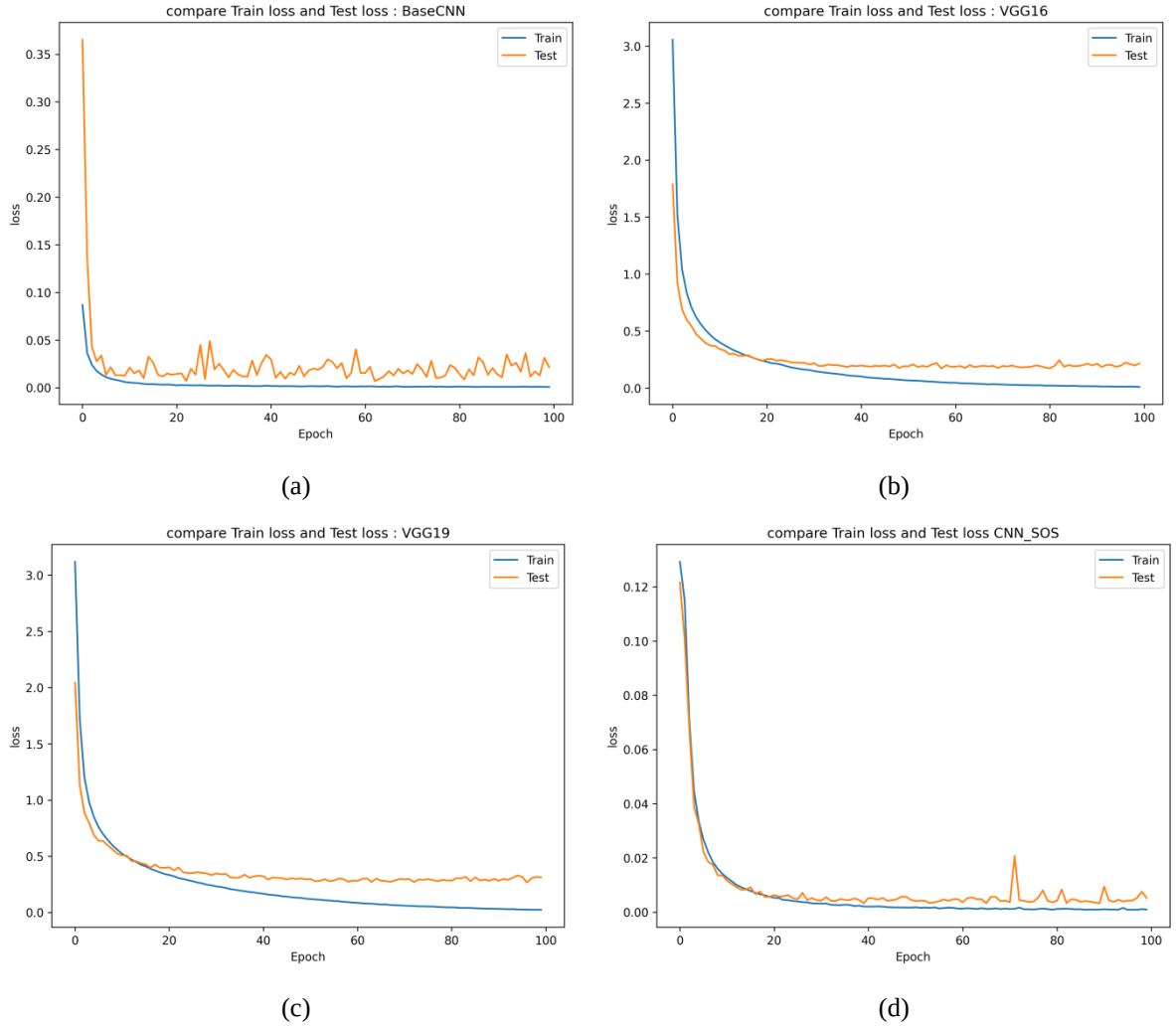


Figure 4. Graph of losses vs. epochs for the training and test sets (a) CNN Base, (b) VGG-16, (c) VGG-19 and (d) Optimized CNN.

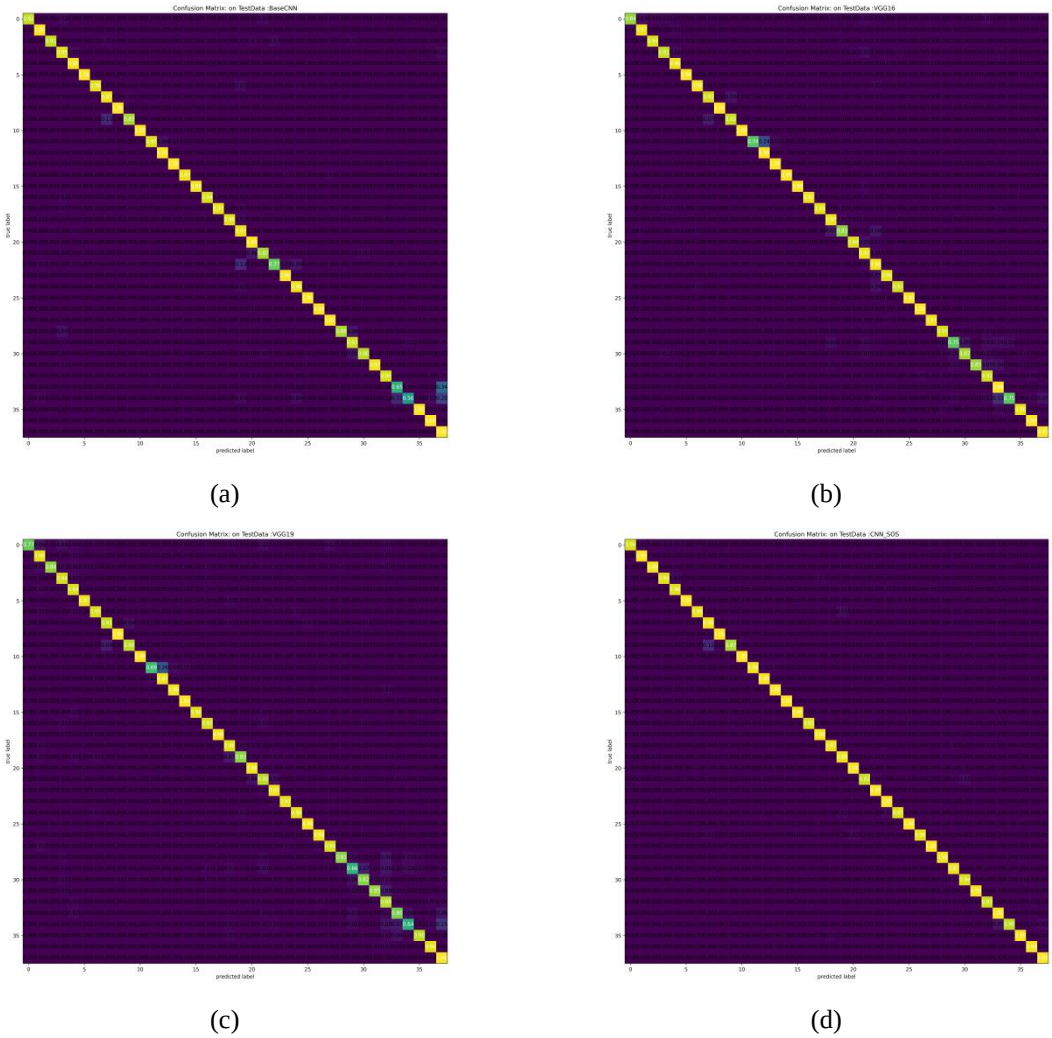
Typically, during the initial epochs, both the training and test sets exhibit higher loss values. However, as the number of epochs increases, the training loss continues to decrease, while the loss plot of the test set initially decreases before eventually showing fluctuations in both the CNN base and optimized network.

With an increase in the number of epochs, the test set experiences higher loss, suggesting the presence of potential overfitting. Overfitting occurs when the model becomes too familiar with the training set, capturing random noise and variations specific to that set. Although the model performs well on the training set, its ability to generalize to new and unseen features in images may slightly decrease, resulting in test errors. This issue can arise when the

model has more capacity or flexibility than necessary for the given problem. In the case of the CNN model, which directly processes images, it achieves an accuracy of 94% on the test set, compared to the 99% accuracy on the training set in the CNN base. This difference in accuracy may be attributed to downsizing the images, which leads to a loss of information. However, enlarging the image size is not feasible due to the limited memory capacity of the CNN network.

Confusion matrix and performance metrics

The confusion matrix, as depicted in Figure 5, provides a visual representation of the CNN's performance in accurately detecting different disease types in images, as well as any instances of misclassification. This matrix helps identify instances where there may be confusion between different disease types.



(a) (b)

(c) (d)

Figure 5. Confusion matrix for the test sets (a) CNN Base, (b) VGG-16, (c) VGG-19 and (d) Optimized CNN.

Table 6 provides a summary of important performance measures derived from the confusion matrix. These metrics are used to evaluate the diagnostic capability of the classifier. Key metrics such as accuracy and F1-score are calculated to assess the classifier's ability to distinguish between different classes.

The accuracy percentage represents the proportion of test data for which the network correctly assigns the appropriate class labels. This can be estimated by calculating the error rate, which indicates the proportion of images for which the model predicted an incorrect class. On the other hand, recall measures the number of true positive (TP) samples out of the total positive samples correctly identified by the model. Precision, on the other hand, indicates the rate of relevant results in comparison to irrelevant results. Recall and precision are competing measures, as maximizing precision often leads to lower recall values, and vice versa.

The F1-score is a comprehensive performance measure that considers class imbalance in the dataset. It combines precision and recall into a single metric. To quantitatively evaluate the performance of the proposed CNNs, average values of accuracy, sensitivity, specificity, and other metrics listed in the table are calculated.

The accuracy measure reflects the CNN's ability to accurately differentiate between different types of leaf diseases. The F1-score percentage confirms the effectiveness of the classifier in terms of recall and precision for diagnosing the diseases.

Table 6. Performance measures of Networks on the test set.

	CNN Base	VGG-16	VGG-19	Optimized CNN
Accuracy	0.94	0.9352	0.9068	0.9729
Precision	0.947	0.9397	0.9123	0.9733
Recall	0.94	0.9362	0.9073	0.9732
F1_score	0.94	0.9354	0.9064	0.9728
Time	7	29	37	10

To visually represent the performance of the SOS algorithm, the convergence curve is utilized to illustrate the search process for an optimal solution of a function. Figure 6 provides a visual depiction of how the SOS algorithm progresses in its search for the most optimal solution.

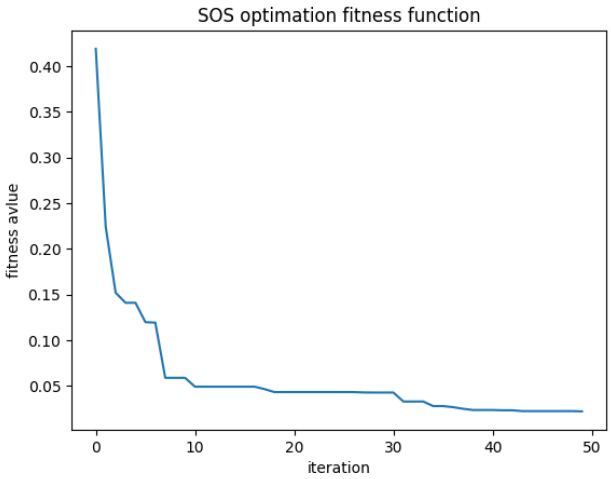


Figure 6. The convergence curve of fitness value vs. iterations.

The convergence pattern depicted in this figure reveals that the SOS algorithm effectively avoids being trapped in local optima, leading to a notable decrease in classification error. It is clear from the graph that the algorithm demonstrates significant improvement throughout the span of 50 iterations.

4.3. Comparison with similar works

Table 7 displays a comparison of the performance of the proposed model with other related works on the PlantVillage dataset.

Table 7. Performance comparison of the proposed method and other works.

Author	Year	Accuracy	Precision	Recall	F1score	Trainable parameters
Zhao et al.[53]	2022	0.9728	0.9749	0.9706	0.9727	6.7 M
Shradha Verma[33]	2022	0.9375	0.9286	0.9274	0.928	16.4 M

Barkha M. Joshi[35]	2023	0.9523	0.94	0.94	0.93	26 M
Hasan Uluta¸s[37]	2023	0.9545	0.956	0.954	0.96	5.1 M
This work	2023	0.9729	0.9733	0.9732	0.9728	5.8 M

This method shows an improved accuracy of about 1% to 5% compared to other state-of-the-art methods. For this study, the optimized CNN has 5.791 million trainable parameters, which is less than most previous studies. In the case where the number of trainable parameters is less (Hasan Uluta¸s et al.[37]), the proposed model is less accurate than the present study, and this comparison shows the superiority of this model. Also, compared to the study of Zhao et al.[53], the proposed model with 900,000 fewer parameters has obtained similar and slightly better results.

V. DISCUSSION AND CONCLUSION

By decreasing the number of parameters in the image classification model, the processing time for each image was expedited, and the convergence to the optimal solution occurred in fewer epochs. The optimized CNN model outperformed baseline CNNs, demonstrating its efficiency for agricultural applications. When creating a new model, it is crucial to consider the time required for disease classification, in addition to factors like computational power and expertise limitations. The duration of classification significantly impacts the treatment process and its eventual outcome, which is influenced by the availability of training data.

This study provides an overview of various approaches to classify plant leaf disease images and evaluates the performance of recent CNN models within this domain. The proposed hybrid approach, which utilizes SOS for hyperparameter optimization, achieved an accuracy of 99.89% for the training dataset and 97.29% for the test dataset. Precision and recall values were recorded at 99.89% and 99.9% respectively. The optimized CNN architecture demonstrated exceptional accuracy and classification capability while simultaneously reducing processing time and computational usage. Unlike traditional image classification methods, this CNN model does not necessitate separate feature extraction and plant leaf image segmentation steps. The feature selection process is seamlessly integrated into the CNN architecture, where features are extracted using convolution and pooling layers. A fully connected layer is then employed to classify the input image data. In contrast to some recent deep learning approaches for plant disease classification, this model incorporates an optimization method to identify the most suitable set of hyperparameters, thereby enhancing the model's robustness and efficiency.

Statements and Declarations

- **Ethical Approval**
- *Not applicable*
- **Funding**

We (all authors) declare that no funds, grants, or other support were received during the preparation of this manuscript

- **Competing Interests**

We (all authors) have no relevant financial or non-financial interests to disclose.

- **Author Contributions**

We (all authors) contributed to the study's conception and design. Material preparation, data collection, and analysis were performed by [Zahra Akeshteh] and [Parvaneh Asghari]. The first draft of the manuscript was written by [Zahra Akeshteh] and [Parvaneh Asghari]. [Hamid Haj Seyyed Javadi] and [Hamidreza Navidi Ghaziani] commented on the manuscript. We (all authors) read and approved the final manuscript.

- **Data Availability**

The required data are available from the corresponding author upon reasonable request.

- **Note**

We hereby submit a manuscript entitled "An Automatic Plants Leaf Disease Identification Method Based on A New Proposed Optimized Convolutional Neural Networks Using Symbiotic Organism Search Algorithm" by Zahra Akeshteh, Parvaneh Asghari, Seyyed Hamid Haji Seyyed Javadi, and Hamidreza Navidi Ghaziani to be considered for publication as a research article in your precious journal .

We declare that all the authors listed in the manuscript are not employed by a government agency that has a primary function other than research and/or education. All the authors are employed at the university and all of them just involve in research activities .

We also declare that this manuscript has not been submitted as an official representative or on behalf of the government.

VI. REFERENCES

[1] 1. Mohanty, S.P., D.P. Hughes, and M. Salathé, Using deep learning for image-based plant disease detection. Frontiers in plant science, 2016. **7**: p. 1419.

[2] 2. Miller, S.A., F.D. Beed, and C.L. Harmon, Plant disease diagnostic capabilities and networks. Annual review of phytopathology, 2009. **47**: p. 15-38.

[3] 3. Bhanu, K., H. Jasmine, and H. Mahadevaswamy. Machine learning implementation in IoT-based intelligent system for agriculture. in the 2020 International Conference for Emerging Technology (INCET). 2020. IEEE.

[4] 4. Sharma, A., et al., Machine learning applications for precision agriculture: A comprehensive review. IEEE Access, 2020. **9**: p. 4843-4873.

[5] 5. Zamani, A.S., et al., Performance of machine learning and image processing in plant leaf disease detection. Journal of Food Quality, 2022. **2022**: p. 1-7.

[6] 6. Nigam, A., et al. Augmenting Genetic Algorithms with Deep Neural Networks for Exploring the Chemical Space. 2019. arXiv:1909.11655 DOI: 10.48550/arXiv.1909.11655.

[7] 7. Shukla, P., H. Kumar, and G.C. Nandi Robotic Grasp Manipulation Using Evolutionary Computing and Deep Reinforcement Learning. 2020. arXiv:2001.05443 DOI: 10.48550/arXiv.2001.05443.

[8] 8. Yuan, Y., et al., Low-frequency robust control for singularly perturbed system. IET Control Theory & Applications, 2015. **9**(2): p. 203-210.

[9] 9. Alamri, N.M.H., M. Packianather, and S. Bigot, Deep learning: Parameter optimization using proposed novel hybrid bees Bayesian convolutional neural network. Applied Artificial Intelligence, 2022. **36**(1): p. 2031815.

[10] 10. Singh, A.K., et al., Deep Learning for Plant Stress Phenotyping: Trends and Future Perspectives. Trends in Plant Science, 2018. **23**(10): p. 883-898.

[11] 11. De Filippis, L.A.C., et al., ANN modelling to optimize manufacturing process. Advanced applications for artificial neural networks, 2018: p. 201-226.

[12] 12. Li, B.-h., et al., Applications of artificial intelligence in intelligent manufacturing: a review. Frontiers of Information Technology & Electronic Engineering, 2017. **18**: p. 86-96.

[13] 13. Fan, Y., et al., A Hybrid Sparrow Search Algorithm of the Hyperparameter Optimization in Deep Learning. Mathematics, 2022. **10**(16): p. 3019.

[14] 14. Van Rijn, J.N. and F. Hutter. Hyperparameter importance across datasets. in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2018.

[15] 15. Wang, Z. and J. Xuan, Intelligent fault recognition framework by using deep reinforcement learning with one dimension convolution and improved actor-critic algorithm. Advanced Engineering Informatics, 2021. **49**: p. 101315.

[16] 16. Larochelle, H., et al. An empirical evaluation of deep architectures on problems with many factors of variation. in Proceedings of the 24th international conference on Machine learning. 2007.

[17] 17. Lerman, P., Fitting segmented regression models by grid search. Journal of the Royal Statistical Society Series C: Applied Statistics, 1980. **29**(1): p. 77-84.

[18] 18. Bergstra, J. and Y. Bengio, Random search for hyper-parameter optimization. Journal of machine learning research, 2012. **13**(2).

[19] 19. Hutter, F., H.H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. in Learning and Intelligent Optimization: 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers 5. 2011. Springer.
[20] 20. Talathi, S.S. Hyper-parameter optimization of deep convolutional networks for object recognition. in 2015 IEEE international conference on image processing (ICIP). 2015. IEEE.
[21] 21. Cui, J., et al., A novel framework of graph Bayesian optimization and its applications to real-world network analysis. Expert Systems with Applications, 2021. **170**: p. 114524.
[22] 22. Lee, M., J. Bae, and S.B. Kim, Uncertainty-aware soft sensor using Bayesian recurrent neural networks. Advanced Engineering Informatics, 2021. **50**: p. 101434.
[23] 23. Kong, H., et al., Energy management strategy for electric vehicles based on deep Q-learning using Bayesian optimization. Neural Computing and Applications, 2020. **32**: p. 14431-14445.
[24] 24. Jin, N., et al., Highly accurate energy consumption forecasting model based on parallel LSTM neural networks. Advanced Engineering Informatics, 2022. **51**: p. 101442.
[25] 25. del Rio Chanona, E.A., et al., Real-time optimization meets Bayesian optimization and derivative-free optimization: A tale of modifier adaptation. Computers & Chemical Engineering, 2021. **147**: p. 107249.
[26] 26. Sun, L.-X., et al., Cluster analysis by simulated annealing. Computers & Chemistry, 1994. **18**(2): p. 103-108.
[27] 27. Zhang, Y. and G. Huang, Traffic flow prediction model based on deep belief network and genetic algorithm. IET Intelligent Transport Systems, 2018. **12**(6): p. 533-541.
[28] 28. Pandey, A. and K. Jain, Plant leaf disease classification using deep attention residual network optimized by opposition-based symbiotic organisms search algorithm. Neural Computing and Applications, 2022. **34**(23): p. 21049-21066.
[29] 29. Ali, S., et al., FF-PCA-LDA: intelligent feature fusion based PCA-LDA classification system for plant leaf diseases. Applied Sciences, 2022. **12**(7): p. 3514.
[30] 30. Bhujel, A., et al., A lightweight Attention-based convolutional neural networks for tomato leaf disease classification. Agriculture, 2022. **12**(2): p. 228.
[31] 31. Thakur, P.S., et al., Explainable vision transformer enabled convolutional neural network for plant disease identification: PlantXViT. arXiv preprint arXiv:2207.07919, 2022.
[32] 32. Alshammari, H., et al., Olive disease classification based on vision transformer and CNN models. Computational Intelligence and Neuroscience, 2022. **2022**.
[33] 33. Verma, S., et al., SE-CapsNet: Automated evaluation of plant disease severity based on feature extraction through Squeeze and Excitation (SE) networks and Capsule networks. Kuwait Journal of Science, 2022. **49**(1).
[34] 34. Thakur, P.S., T. Sheorey, and A. Ojha, VGG-ICNN: A Lightweight CNN model for crop disease identification. Multimedia Tools and Applications, 2023. **82**(1): p. 497-520.
[35] 35. Joshi, B.M. and H. Bhavsar, Deep Learning Technology based Night-CNN for Nightshade Crop Leaf Disease Detection. International Journal of Intelligent Systems and Applications in Engineering, 2023. **11**(1): p. 215-227.
[36] 36. Singh, V., A. Chug, and A.P. Singh, Classification of Beans Leaf Diseases using Fine Tuned CNN Model. Procedia Computer Science, 2023. **218**: p. 348-356.
[37] 37. Ulutaş, H. and V. Aslantaş, Design of Efficient Methods for the Detection of Tomato Leaf Disease Utilizing Proposed Ensemble CNN Model. Electronics, 2023. **12**(4): p. 827.
[38] 38. Chen, H.-C., et al., AlexNet convolutional neural network for disease detection and classification of tomato leaf. Electronics, 2022. **11**(6): p. 951.
[39] 39. Tugrul, B., E. Elfatimi, and R. Eryigit, Convolutional neural networks in detection of plant leaf diseases: A review. Agriculture, 2022. **12**(8): p. 1192.
[40] 40. O'Mahony, N., et al. Deep learning vs. traditional computer vision. in Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1. 2020. Springer.
[41] 41. Bhattarai, S., New Plant Diseases Dataset. 2019, kaggle.
[42] 42. Shifat-E-Rabbi, M., et al., Cell image classification: a comparative overview. Cytometry Part A, 2020. **97**(4): p. 347-362.
[43] 43. Goodfellow, I., Y. Bengio, and A. Courville, Deep learning. 2016: MIT press.

[44] 44. Chollet, F., Deep learning with Python. 2021: Simon and Schuster.
[45] 45. Simonyan, K. and A. Zisserman, Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.
[46] 46. Hemdan, E.E.-D., M.A. Shouman, and M.E. Karar, Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images. arXiv preprint arXiv:2003.11055, 2020.
[47] 47. Sukegawa, S., et al., Deep neural networks for dental implant system classification. Biomolecules, 2020. **10**(7): p. 984.
[48] 48. Wan, X., X. Zhang, and L. Liu, An improved VGG19 transfer learning strip steel surface defect recognition deep neural network based on few samples and imbalanced datasets. Applied Sciences, 2021. **11**(6): p. 2606.
[49] 49. Arshad, H., et al., A multilevel paradigm for deep convolutional neural network features selection with an application to human gait recognition. Expert Systems, 2022. **39**(7): p. e12541.
[50] 50. Karacı, A., VGGCOV19-NET: automatic detection of COVID-19 cases from X-ray images using modified VGG19 CNN architecture and YOLO algorithm. Neural Computing and Applications, 2022. **34**(10): p. 8253-8274.
[51] 51. Cheng, M.-Y. and D. Prayogo, Symbiotic organisms search: a new metaheuristic optimization algorithm. Computers & Structures, 2014. **139**: p. 98-112.
[52] 52. Hamidian, A.H., et al., Simulation and optimization of nanomaterials application for heavy metal removal from aqueous solutions. Inorganic and Nano-Metal Chemistry, 2019. **49**(7): p. 217-230.
[53] 53. Zhao, Y., et al., RIC-Net: A plant disease classification model based on the fusion of Inception and residual structure and embedded attention mechanism. computers and Electronics in Agriculture, 2022. **193**: p. 106644.