Indonesian Journal of Electrical Engineering and Computer Science
Vol. 38, No. 1, April 2025, pp. 447~458
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v38.i1.pp447-458

Recognition of plant leaf diseases based on deep learning and the chemical reaction optimization algorithm

Nghien Nguyen Ba[1], Nhung Nguyen Thi[1], Dung Vuong Quoc[2], Cuong Nguyen Cong[3]
[1]Faculty of Information Technology, Hanoi University of Industry, Hanoi City, Vietnam
[2]Center of Information Technology, Hanoi University of Industry, Hanoi City, Vietnam
[3]Faculty of Electrical Engineering, Hanoi University of Industry, Hanoi City, Vietnam

Article Info

Article history:

Received May 13, 2024
Revised Oct 22, 2024
Accepted Oct 30, 2024

Keywords:

Agriculture
CRO
Deep learning model
Hyperparameters optimization
Leaf diseases
SVM

ABSTRACT

Agriculture plays a crucial role in developing countries such as Vietnam, where 70 percent of the population is employed in agriculture, and 57 percent of the social labor force works in the agricultural sector. Therefore, crop productivity directly affects the lives of many people. One of the primary reasons for reduced crop yields is plant leaf diseases caused by bacteria, fungi, and viruses. Hence, there is a need for a method to help farmers identify leaf diseases early to take appropriate action to protect crops and shift to smart agricultural production. This paper proposes lightweight deep learning (DL) models combined with a support vector machine (SVM), with hyperparameters fine-tuned by chemical reaction optimization (CRO), for detecting plant leaf diseases. The main advantage of the method is the simplicity of the architecture and optimization of the DL model's hyperparameters, making it easily deployable on low hardware devices. To test the performance of the proposed method, experiments are performed on the PlantVillage dataset using Python. The superiority of the proposed method over the well-known visual geometry group-16 (VGG-16) and MobileNetV2 models is demonstrated by a 10% increase in accuracy prediction and a decrease of 5% and 66% in training time, respectively.

Corresponding Author:

Nghien Nguyen Ba
Department of Computer Engineering, Faculty of Information TechnologyHanoi University of Industry
298 Cau Dien Street, Hanoi, Vietnam
Email: nguyenbanghien_cntt@haui.edu.vn

1. INTRODUCTION

The 2023 global report on food crises shows that there is a growing need for food and livelihood assistance. The report reveals that over 250 million individuals are faced with acute hunger. In 2022, the count of individuals experiencing severe food insecurity at crisis or more severe levels reached 258 million across 58 countries and territories, a notable increase from 193 million individuals in 53 countries and territories in 2021. One of the primary reasons for reduced crop yields is plant leaf diseases caused by bacteria, fungi, and viruses. Because, plant leaf diseases often affect the chlorophyll content and structure of leaves, which are crucial for photosynthesis. Reduced photosynthesis leads to a decrease in the v, ultimately affecting its growth and yield. In addition, many leaf diseases cause symptoms such as necrosis, wilting, or premature death of leaves. This results in a reduction of the total leaf area available for photosynthesis. The loss of leaves can significantly impact the plant's ability to capture sunlight and convert it into energy. Therefore, protecting crops and increasing crop productivity is an extremely important task. Currently in Vietnam, the widespread use of pesticides leads to affecting the health of farmers, polluting land and water sources, and reducing the quality of agricultural products. In addition, the evolution of climate change is very

complicated, leading to increasingly serious diseases related to plant leaves. In particular, diseases on crop leaves are very common and seriously affect crop productivity [1].

Based on [2], plant leaf diseases pose a significant challenge in rice production, with the potential to adversely affect the crop and lead to a decrease in yield. Genitha *et al.* [3] comment farmers have difficulty detecting and classifying diseases on leaves. Relying on traditional physical observation methods for disease identification and categorization in agriculture is frequently unreliable and could lead to a considerable decline during production. Therefore, an advanced approach is necessary to assist farmers in quickly and accurately detecting plant leaf diseases. Recently, many researchers have focused on applying deep learning (DL) and machine learning (ML) to create models to recognize highly accurate identification of plant leaf diseases [4]. Ramesh *et al.* [5] use the random forest (RF) to classify the healthy and diseased leaves from their collected data set. Firstly, they used the histogram of an oriented gradient to extract image features. Then, they trained the RF model with the image features.

Sharma *et al.* [6] present image segmentation conducted to delineate leaves from the background. The segmentation process employs K-means clustering with two cluster centers, designating one for the background and the other for the foreground. Then, the pixels of the background image are changed to black to eliminate irrelevant information and enhance prediction accuracy. Finally, some ML algorithms such as logistic regression, K-nearest neighbors (KNN), and support vector machine (SVM) are implemented for the classification of plant leaf diseases. The logistic regression achieved the best classification accuracy of 66.4%. Genitha *et al.* [3] proposed a fusion method for the classification of plant leaf diseases. First, leaf images are pre-processed by transforming the color image to a grayscale image. Then, noise is removed by the median filter, edge, and direction of the plant leaf are detected by the Sobel filter, and the Gabor filter respectively. Next, the primary characteristics of the leaf image are extracted by the principal component analysis (PCA) technique. Finally, the principal characteristics fed to the SVM for classification purposes.

The experiment results demonstrate that the proposed method achieves a prediction accuracy of 90.66%. Mengistu *et al.* [7] utilize backpropagation neural networks and decision trees (DT) to detect three primary diseases affecting coffee trees. The accuracy of the proposed model reached 94.5%. Sharma *et al.* [8] utilized a blend of image processing methods and various data mining algorithms such as KNN, SVM, RF, and DT, for the timely identification of rice plant ailments. The results of the experiment demonstrate the RF algorithm achieves the highest accuracy prediction, reaching 90%. Mohanty *et al.* [9] use two well-known DL architectures, namely AlexNet and GoogLeNet, to identify plant leaf diseases. They utilize images from the PlantVillage dataset for training and testing transfer models, as well as for training models from scratch. They tested scenarios, including color images, grayscale images, and segmented images, with training and testing data percentages of 80%-20%, 60%-40%, and 50%-50%, as well as 20%-80%. The experimental findings show that GoogLeNet, with transfer learning and a training-testing set ratio of 80%-20%, achieved the highest accuracy prediction of 99.35%. Andrew *et al.* [10] conducted experiments using the PlantVillage image dataset with four common DL models: residual network (ResNet)-50, InceptionV4, visual geometry group-16 (VGG-16), and DenseNet-121, all pre-trained on the ImageNet dataset. Experimental outcomes show that the pre-trained DenseNet-121 achieved the highest classification accuracy, reaching 99.81%.

Jung *et al.* [11] proposed a method for classifying crops, detecting diseases, and categorical crop ailments. The proposed approach is executed in three stages. For the initial stage, a pre-trained model is used to classify crops. For the subsequent stage, several models, each dedicated to a specific crop, are employed to detect plant diseases from images. In the final step, a set of pre-trained models is utilized to classify diseases for each crop. Five pre-trained models, including ResNet50, AlexNet, GoogleNet, VGG19, and EfficientNet, are experimented with in each step. Leaf images of three crops bell pepper, tomato, and potato are extracted from the PlantVillage dataset utilized in the experiment. The results of the experiment indicate that the pre-trained EfficientNet model achieved the highest accuracy of 99.33% and 99.40% in the first and last steps, respectively. Additionally, the pre-trained GoogLeNet attained the highest accuracy of 100% for the bell pepper crop, VGG19 yielded the best accuracy, reaching 100% for the potato crop, and ResNet50 attained the utmost accuracy of 99.75% for the tomato crop in the second step. Hang *et al.* [12] suggested an approach that replaced the fully connected layer of the VGG16 model with the inception and squeeze-and-excitation modules. The experimental findings illustrate the superior performance of their proposed model compared to other common structures including AlexNet, GoogLeNet, VGG16, VGG19, ResNet-50, Inceptionv2, Inceptionv3, Inceptionv4, and SENet.

Sharma *et al.* [6] gathered 20,000 images featuring both healthy and diseased leaves across 19 different classes. The dataset encompasses prevalent leaf diseases like black rot, rust, bacterial spots, and others, affecting various crops such as corn, apple, potato, and tomato. Subsequently, the images underwent segmentation using the K-means algorithm to isolate the leaves and background. Finally, the researchers trained and tested several ML models and a convolutional neural network (CNNs) using the segmented dataset. They reported that the CNN architecture attained the utmost classification accuracy of 98%.

Sardogan *et al.* [13] suggested an approach that combines CNN and learning vector quantization (LVQ). The leaf images' features are extracted using CNN and then classified through LVQ. For the experiment, 500 tomato leaf images were utilized, categorized into healthy, bacterial spot, late bright, septoria spot, and yellow curved from the PlantVillage dataset. The dataset underwent division into a training set (80%) and a test set (20%). The experiment data indicate that their proposed method achieves an average classification accuracy of 86%. Nikith *et al.* [14] employed the CNN model to classify leaf diseases in the Soyabean leaf dataset. They reported a 96% classification accuracy during the training phase and 84% during the testing phase. Algani *et al.* [15] proposed a method that integrates ant colony optimization (ACO) and CNN for the detection and classification of plant leaf diseases. In this approach, ACO is utilized for feature extraction, while CNN is employed to differentiate between healthy and diseased leaves. The experimental data demonstrate that the suggested approach attained the highest accuracy in classification, reaching 99.98%, surpassing popular models like generative adversarial network (GAN) and CNN. Nawaz *et al.* [16] employed a combination of ResNet-34, the convolution block attention module (CBAM), and faster region-based convolutional neural network (faster-RCNN) to locate the region and classify leaf diseases in tomato crops. ResNet-34, in conjunction with CBAM, is responsible for extracting features from the leaf disease images. Meanwhile, faster-RCNN is pivotal in localizing and categorizing the identified leaf diseases. The experimental outcomes highlight that their suggested approach achieved an impressive prediction accuracy of 99.7%.

Albattah *et al.* [17] suggested a modified CenterNet model by replacing ResNet-101 in the original model with DenseNet-77. They utilized the modified CenterNet to detect regions and classify various leaf diseases. Experimental outcomes reported that the suggested model achieved an mean average precision (mAP) of 0.99 and a mean intersection over union (IOU) of 0.993. Anari *et al.* [18], modified the original residual block by incorporating weights for the feedback block. Additionally, the author employed multiple SVM models for classification purposes, using a portion of the PlantVillage images dataset for the experiment. The reported results indicate that the proposed model achieved a prediction accuracy of 99.1%. Balaji *et al.* [19] employed a genetic algorithm (GA) and a pre-trained CNN for the classification of plant leaf diseases. The GA was utilized for denoising and feature extraction, while the pre-trained ResNet-50 and VGG-16 models aided in classifying the identified leaf diseases. According to their report, the proposed method attained a classification accuracy of 95%. Agarwal *et al.* [20], proposed the utilization of CNN for the classification of tomato leaf diseases. The reported results indicate a classification accuracy of 91.2%. Ezhilarasan *et al.* [21] proposed a method using a DL ResNet model with parameters optimized by the jellyfish optimization algorithm for the classification of tomato leaf disease. The performance of the proposed model is evaluated based on the classification of tomato plant leaf disease. The experimental results show that the proposed method achieves a classification accuracy of 97.3%. Pujar *et al.* [22], proposed the enhanced KNN for the classification of corn leaf diseases. The reported results indicate a classification accuracy of 99.86%.

However, almost all researchers have employed the complex architecture of CNNs, which involve a large number of parameters. Consequently, previously proposed solutions are challenging to implement on hardware with limited performance, such as mobile and internet of things (IoT) devices. Additionally, the hyperparameters of CNNs are often chosen based on experience, making it difficult to achieve optimal models. In this study, a lightweight CNN model is designed for easy integration into low-performance hardware devices and high-accuracy prediction. In addition, a novel method that utilizes the chemical reaction optimization (CRO) algorithm to optimize the tunable parameters of the CNN and SVM model is introduced. Furthermore, to enhance classification accuracy, the last layer of the trained CNN is replaced by an SVM model. Initially, images depicting plant leaf diseases extracted from the PlantVillage dataset are employed to train and evaluate the proposed DL model. Subsequently, the final layer of the trained DL model is substituted with an SVM for enhanced classification accuracy. The hyperparameters of both the DL and SVM models are fine-tuned using the CRO algorithm.

The principle of the proposed method is presented in section 2. The experimental results and discussion are presented in section 3. The goal of the study is reiterated, the findings are summarized, the significance of the findings is discussed, and future work is outlined in the conclusion section.

2. METHOD

2.1. Convolution neural networks

The structure of the DL network illustrated in Figure 1 includes convolutional, pooling, and fully connected layers. As the CNN progresses through its layers, its complexity grows, allowing it to detect larger areas of the image. The initial layers concentrate on basic characteristics such as colors and edges, gradually advancing to recognize more substantial elements or shapes of the object. Ultimately, the CNN reaches a point where it successfully identifies the intended object. The convolutional layers serve as removing noise

and boundary detectors, whereas the pooling layers perform computations to obtain either average or maximum local values, reducing the size of the image. Fully connected layers perform classification tasks.

The convolutional layer performs a summarization of the element-wise product of two matrices. The first matrix represents a portion of the input, while the second matrix corresponds to the kernel. Figure 2 illustrates the calculation process of the convolutional layer. The pooling calculation is illustrated in Figure 3, where the stride determines how quickly the kernel moves both horizontally and vertically across the pixels of the input image during convolution and pooling.

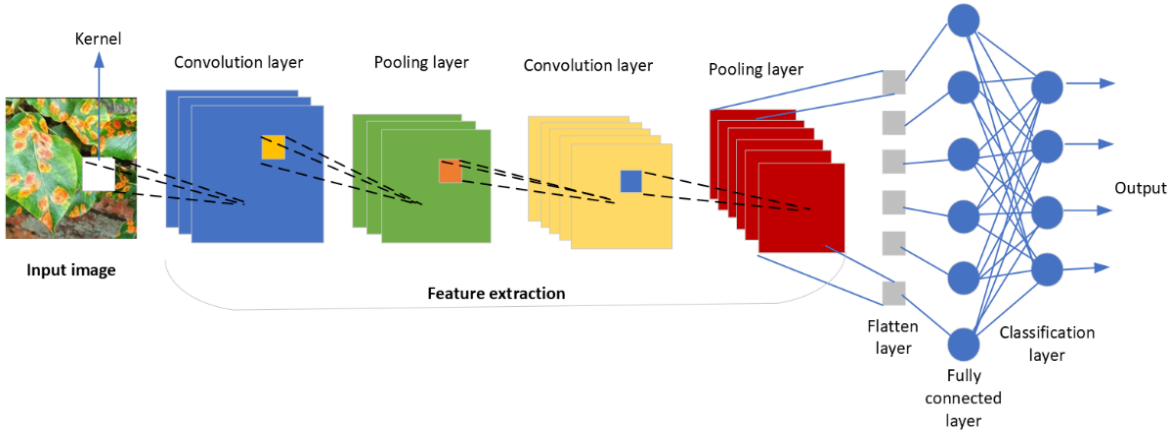

Figure 1. The general CNN structures

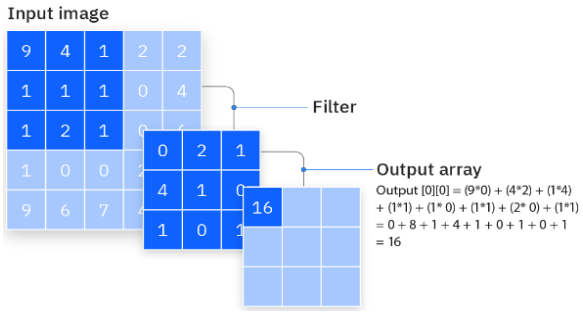

Figure 2. The principle of the convolution operation

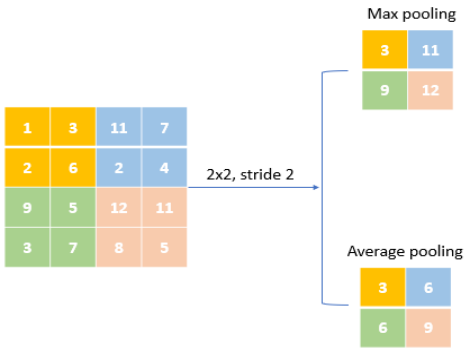

Figure 3. The principle of the pooling process

2.2. Support vector machine

SVM are widely utilized in the domain of DL for classification purposes. SVM aims to find the optimal hyperplane that effectively separates data into two categories, with the margin representing the maximum distance between the hyperplane and the closest sample. Figure 4 illustrates a hyperplane along with its associated margin.

Let's consider a training set comprising n samples $X = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where each x_i represents a vector in a d-dimensional space, and $y_i \in \{-1, 1\}$ denotes the corresponding labels. A hyperplane that partitions X into two regions is expressed by the equation $w.x + b = 0$. The objective of the SVM algorithm is to determine the values of w and b to maximize the margin. This requires solving the subsequent optimization problem [23]:

$$\min_{w,b,\xi} \left\{ \frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^{n} \xi_i \right\} \quad (1)$$

to meet the requirement (2):

$$\Omega: \begin{cases} (w, b, \xi) \in R^d \times R \times R_+^n \\ y_i(\langle w, x_i \rangle + b) \geq 1 - \xi_i, \forall\, 1 \leq i \leq n \end{cases} \quad (2)$$

where, the slack variables, denoted as ξ_i, are introduced to relax the categorization criterion, while C serves as a tuning factor. It regulates the balance between maximizing the margin and minimizing the training error. Rather than tackling the primary problem directly, it is often preferable to address its dual counterpart, which proceeds as follows:

$$\min_{\alpha} \frac{1}{2}\alpha^T H\alpha - \vec{1}\alpha \tag{3}$$

with constraints:

$$\Delta: \begin{cases} y^T\alpha = 0 \\ 0 \le \alpha_i \le C, i = 1, \dots n \end{cases} \tag{4}$$

where $y = (y_1, y_2, \dots, y_n)$, $\vec{1}$ is a unit vector, and H is a symmetric matrix specified by:

$$H_{i,j} = y_i y_j \langle \Phi(x_i), \Phi(x_j) \rangle = y_i y_j K(x_i, x_j) \tag{5}$$

here, $\langle \cdots \rangle$ presents a dot product, and $\phi(.)$ denotes a transformation from the input space to a feature space of greater dimensions, addressing situations where samples are not linearly separable. $K(\cdot)$ is referred to as the kernel function and is specified in expression (6).

$$K(x, y) = \langle \Phi(x), \Phi(y) \rangle \tag{6}$$

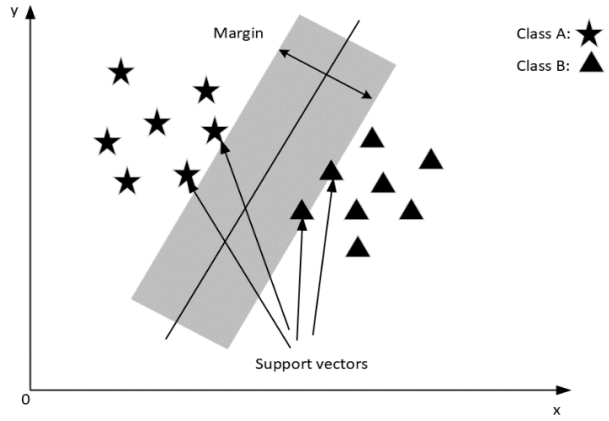

Figure 4. Sketch principle of the SVM

2.3. Hyperparameter optimization formulation

During the design phase of ML models, efficient exploration of the hyperparameter space through optimization techniques can identify the optimal hyperparameters (HPO) for these models. As outlined in the article [24], HPO comprises four essential components: an estimator comprising its objective function, a search space (also known as configuration space), a search algorithm used to uncover tuning parameters, and a scoring function for comparing the effectiveness of various tuning parameter setups. Typically, the goal of the HPO problem is to pinpoint a specific point within the search space defined by (7) [7].

$$s^* = \arg\min_{s \in S} f(s) \tag{7}$$

here, $f(s)$ presents the objective function to be minimized, s^* is an optimal point in search space *S.*

For DL models, the search space can include the number and size of filters in the convolutional layers, the activation function, the neuron count within the fully connected layer, and the initial learning rate. Suppose we need to find n optimal tuning parameters for a DL model. Each tuning parameter has a discrete or categorical value domain with several options m_i in the corresponding search space S_i. Therefore, the search space can be represented as (8).

$$S = \begin{matrix} S_1 \\ S_2 \\ \vdots \\ S_n \end{matrix} = \begin{matrix} s_{1,1} & s_{1,2} & \cdots & s_{1,m_1} \\ s_{2,1} & s_{2,2} & \cdots & s_{2,m_2} \\ \vdots & \vdots & \cdots & \vdots \\ s_{n,1} & s_{n,2} & \cdots & s_{n,m_n} \end{matrix} \quad (8)$$

Hence, $s^* = [s_1, s_2, \ldots, s_n]^T$ and $s_i \in S_i$. To find the s^*, an appropriate optimization algorithm needs to be used.

2.4. Chemical reaction optimization algorithm

The CRO algorithm is a contemporary random search employed for the problem of optimization, mimicking the dynamics of loosely coupled chemical transformations within optimization processes. In a chemical reaction system, various chemical substances interact within their environment, each possessing both potential (PE) and kinetic energies (KE), while the environment itself is represented by a central energy buffer [25]. As the chemical reaction reaches equilibrium, all substances stabilize with minimal potential energy. CRO emulates this equilibrium process by transforming potential energy into kinetic energy, slowly discharging the energy stored in chemical molecules into the surroundings. CRO is founded upon four basic transforms: on − wall ineffective collision, decomposition, inter-molecular ineffective collision, and synthesis. While two ineffective collisions facilitate local searches, the others facilitate global searches. Consequently, CRO effectively integrates these two search types to explore the global minimum within the feasible region. By amalgamating the beneficial aspects of both simulated annealing (SA) and GA, CRO maintains energy conservation akin to the metropolis algorithm used in SA, while its decomposition and synthesis operations resemble the crossover and mutation operations of GA. In CRO, every molecule is defined by a molecular structure (ω), serving as an answer to the issue, along with two types of energy: PE and KE. PE corresponds to the value of the fitness function, while KE denotes a molecule's tolerance to an increase in its energy state. To simulate four basic reactions based on three operators neighbor, decomposition, and synthesis. The neighbor operator is used in the collision reaction to create a new solution from a particular one. The new solution is created by changing randomly selected elements in the current solution. The goal of this operator is to conduct a local exploration for a better solution. The pseudocode of this operator is shown by Algorithm 1.

Algorithm 1. Neighbor (ω)

1. Copy ω to ω'
2. Generate a random integer i smaller than the total number of elements in ω
3. Assign $\omega'(i) = s_i, s_i \in S_i$

The decomposition operator is used in the decomposition reaction. This operator creates two new solutions ω_1', ω_2' from the specified solution ω. This operator helps to escape local minima by half the total change. The pseudocode of this operator is demonstrated by Algorithm 2.

Algorithm 2. Decomposition (ω)

1. Copy ω to ω_1'
2. Randomly change 50% of the elements of ω_1' by selecting randomly s_i from set S_i
3. Repeat steps 1 and 2 for ω_2' in a similar manner

The synthesis operator is used in the synthesis reaction. This operator creates a new solution ω' from the two given solutions ω_1, ω_2. The process involves randomly selecting components of two molecules with similar chances to create a new molecule. The pseudocode for this operation is provided in Algorithm 3. The CRO algorithm is started by an initial population. The pseudocode for creating this population is presented in Algorithm 4.

Algorithm 3. Create synthesis (ω_1, ω_2)

1. FOR i = 1 TO m DO
2. Generate a random number r between 0 and 1
3. IF r>0.5 DO
4. $\omega'(i) = \omega_1(i)$
5. *ELSE*
6. $\omega'(i) = \omega_2(i)$
7. ENDIF
8. ENDFOR

Algorithm 4. Create the initial population

```
FOR i = 1 TO PopSize DO
FOR j = 1 TO m DO
Select randomly s_j ∈ S_j for discrete or categorical domains. Generate a random number s_j ∈ [l_j, u_j] for the continuous domain.
ω_i(j) = s_j
ENDFOR
Train the CNN or SVM model with the given ω_i
Calculate the objective function PE(ω_i) = f(ω_i) = 1 − accuracy(test set)
ENDFOR
```

2.5. Proposed method

The principle of the proposed approach involves a combination of a DL network and an SVM model. The DL network is responsible for extracting features from images of diseased leaves, while the SVM serves as the classifier. Initially, images of diseased leaves are extracted from the PlantVillage database. Subsequently, these images are augmented by rotating them at specified angles. The augmented images then undergo feature extraction through the proposed lightweight CNN network. Finally, the image features are classified using the SVM model. The performance of DL networks mainly depends on their architecture and hyperparameters. In this paper, the CNN structure is designed to balance simplicity with the ability to extract all necessary features. This architecture is suitable to be deployed on devices with low hardware configurations, such as embedded computers or mobile phones. Additionally, the optimal tuning parameters for both the CNN network and SVM model are determined by using the CRO algorithm. The structure of the proposed model includes five convolutional layers and two fully connected layers, as illustrated in Figure 5, here, ki, mi, fi, and n represent the number of filters, size of the filter, activation function, and the number of neurons, respectively. These are tuning parameters that require optimization.

To enhance performance in classification problems, A fusion of CNN and SVM is proposed. Initially, CNN functions as the feature extraction component, after that the feature vectors extracted by the CNN serve as inputs for the SVM. This fusion is expected to take the strengths of both models: CNN efficiently extracts features from images, while SVM demonstrates high categorical accuracy when the input data is effectively preprocessed. Figure 6 depicts the integration of these two models.

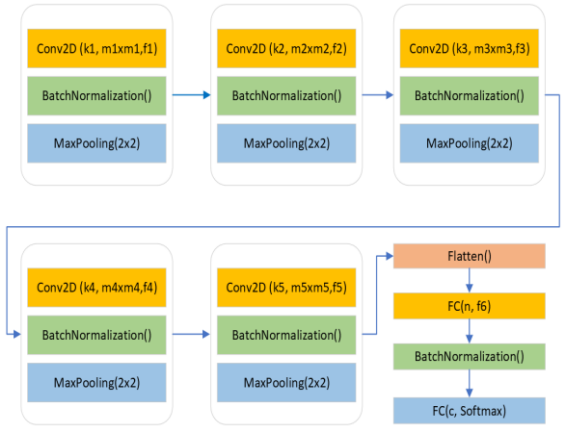

Figure 5 The architecture of the proposed CNN

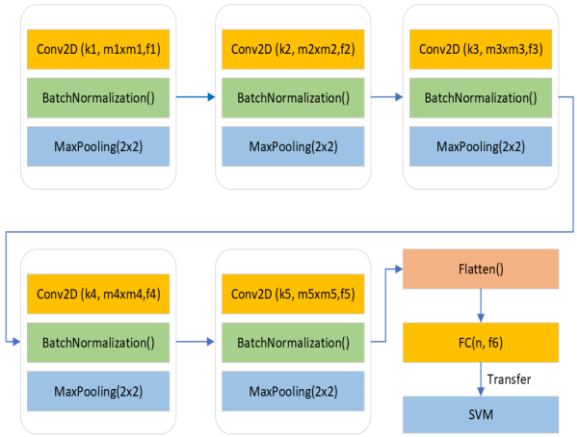

Figure 6. The combination of the proposed CNN and SVM

The tuning parameters of the proposed CNN such as $k1$, $k2$, $k3$, $k4$, $k5$, $m1$, $m2$, $m3$, $m4$, $m5$, $f1$, $f2$, $f3$, $f4$, $f5$, $f6$, and n or C, σ and kernel type of the SVM model are encoded as a molecule structure (solution), and then the CRO algorithm is applied to find the global minimum of the objective function. In this case, the objective function is defined as 1 minus accuracy or 1 divided by accuracy. The proposed algorithm for finding the best tuning parameters of the CNN and SVM models are presented in Algorithms 5 and 6, respectively.

Algorithm 5. The proposed algorithm for a CNN's hyperparameter optimization

Input: The objective function f = 1 − accuracy(test dataset), search space S, image dataset, stopping criterion.
Output: Optimal hyperparameters

1. Initial parameters of the CRO algorithm such as PopSize, KELossRate, MoleColl, Initial KE, δ, θ, and buffer.
2. Create the initial population according to Algorithm 4 and find the current best solution s^*.
3. **REPEAT**
   - 3.1. Generate a random number b∈ [0,1]
   - 3.2. **IF** b>MoleColl **THEN**
   - 3.3. Random select one solution from the population denoted by s
   - 3.4. **IF** s.numBit − s.minHit > δ **THEN**
   - 3.5. Perform decomposition reaction according to the operation illustrated by algorithm 2
   - 3.6. **ELSE**
   - 3.7. Perform onWallIneffectiveCollision reaction according to the operation illustrated by algorithm 1
   - 3.8. **ENDIF**
   - 3.9. **ELSE**
   - 3.10. Select randomly two solutions from the population denoted by s_1 and s_2
   - 3.11. **IF** $s_1.KE < \theta$ **AND** $s_2.KE < \theta$ **THEN**
   - 3.12. Perform Synthesis reaction according to Algorithm 3
   - 3.13. **ELSE**
   - 3.14. Perform InterMolecularIneffectiveCollision reaction according to the operation illustrated by algorithm 1
   - 3.15. **ENDIF**
   - 3.16. Train and test the DL model with a new solution by using the 5-fold CV technique for calculating the objective function
   - 3.17. **IF** a better solution is found **THEN**
   - 3.18. Update the best solution
   - 3.19. **ENDIF**
4. **ULTIL** Stop criterion is met
5. Obtain s^* from a molecule of the best solution
6. Train and test the DL model with the s^*

Algorithm 6. The proposed algorithm for the SVM's hyperparameter optimization

Input: Trained DL, search space S, image dataset, the objective function f.
Output: SVM's Optimal hyperparameters

1. Initial parameters of the CRO algorithm such as PopSize, KELossRate, MoleColl, Initial KE, δ, θ, and buffer.
2. Create the initial population according to Algorithm 4 and find the current best solution s^*.
3. featureSet = {}
4. **FOREACH** img IN imageSet DO
   - 4.1. Calculate the output of the FC(n,f6) layer of the trained model with the img input as a feature vector.
   - 4.2. Append feature vector into featureSet
5. **ENDFOR**
6. **REPEAT**
   - 6.1. Generate a random number b∈ [0,1]
   - 6.2. **IF** b>MoleColl **THEN**
   - 6.3. Random select one solution from the population denoted by s
   - 6.4. **IF** s.numBit − s.minHit > δ **THEN**
   - 6.5. Perform decomposition reaction according to the operation illustrated by algorithm 2.
   - 6.6. **ELSE**
   - 6.7. Perform onWallIneffectiveCollision reaction according to the operation illustrated by algorithm 1
   - 6.8. **ENDIF**
   - 6.9. **ELSE**
   - 6.10. Select randomly two solutions from the population denoted by s_1 and s_2
   - 6.11. **IF** $s_1.KE < \theta$ **AND** $s_2.KE < \theta$ **THEN**
   - 6.12. Perform Synthesis reaction according to Algorithm 3
   - 6.13. **ELSE**
   - 6.14. Perform InterMolecularIneffectiveCollision reaction according to the operation illustrated by algorithm 1
   - 6.15. **ENDIF**
   - 6.16. Train and test the SVM model on the featureSet with a new solution by using the 5-fold CV technique for calculating the objective function
   - 6.17. **IF** a better solution is found **THEN**
   - 6.18. Update the best solution
   - 6.19. **ENDIF**
7. **ULTIL** Stop criterion is met
8. Obtain s^* from a molecule of the best solution
9. Train and test the SVM model with the s^* on the featureSet.

3. EXPERIMENT RESULTS AND DISCUSSION

The proposed solution has been experimented on the PlantVillage dataset, collected by Hughes and Salathe [26]. This dataset is highly renowned, encompassing more than 50,000 images, and has been extensively utilized by experts for plant disease diagnosis. The PlantVillage comprises 38 image classes depicting disease symptoms on leaves and 1 image class without leaves. Due to hardware limitations, the authors restricted their experiment to using only 100 images per class. Some images of plant leaf diseases are illustrated in Figure 7. Details on the dataset used in the experiment are summarized in Table 1.

Figure 7. Some images of plant leaf diseases

Table 1. Summary of the experimented dataset

Number of crop type	Number of plant leaf diseases	Number of samples	Number of classes	Image size
14	26	3899	39	128×128×3

The search spaces, S1 for the hyperparameters of the proposed CNN and S2 for those of the SVM model in the experiment, are set up as follows:

$$S_1 = \begin{matrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ K_5 \\ M \\ F \\ N \end{matrix} = \begin{matrix} 32 & 64 & 128 & 256 \\ 64 & 128 & 256 & 512 \\ 128 & 256 & 512 & 1024 \\ 256 & 512 & 1024 & 2048 \\ 512 & 1024 & 2048 & 4096 \\ 1 & 3 & 5 & 7 \\ ELU & ReLU & Sigmoid & Tanh \\ 128 & 256 & 512 & 1024 \end{matrix} \quad (9)$$

$$S_2 = \begin{matrix} C \\ Sigma \\ F \end{matrix} = \begin{matrix} [1, 10] \\ [0.0001, 1.0] \\ rbf \quad linear \quad poly \quad sigmoid \end{matrix} \quad (10)$$

The accuracy prediction metric, defined in (11), is used to evaluate the performance of the proposed model.

$$Accuracy = \frac{Number\ of\ correct\ prediction}{Total\ number\ of\ prediction} \quad (11)$$

Hyperparameters $k_i \in K_i, m_i \in M, f_i \in F, n \in N, C \in C, \sigma \in Sigma, and\ kernel \in F$ are found by the CRO algorithm. The initial parameters of the CRO algorithm are set as PopSize=10, KELossRate=0.1, MoleColl=0.1, Initial KE=100, δ =500, θ=10, and buffer=0.

The suggested algorithm is implemented using Python programming language on Google Colab. The proposed DL is implemented by TensorFlow and Keras flatform. The SMV model is implemented using the Sklearn framework. The authors implemented the CRO algorithm. The proposed CNN is trained using EarlyStoping, ReduceLROnPlateau, and the 5-fold CV techniques. The SVM model is trained using the 5-fold CV technique. The tuning parameters of the CNN model were found by running the CRO algorithm

implemented by algorithm 5 are m1 = m2 = m3 = m4 = m5 = 1, k1 = 32, k2 = 64, k3 = 128, k4 = 256, k5 = 512, n = 512, f1 = f2 = f3 = f4 = f5 = f6 = ReLU (rectified linear unit).

This finding is reasonable because it is well-suited to numerous DL frameworks and hardware accelerators, which are tailored for optimizing power-of-two dimensions. In addition, gradually increasing the number of filters in deeper layers enables the network to capture increasingly intricate features and patterns. Furthermore, the proposed CNN model, with the identified configuration, converged quickly, achieving 100% accuracy in the training phase after approximately 30 epochs. This is demonstrated in Figures 8 and 9, which illustrate the accuracy and loss function values over each epoch during the training phase.

The tuning parameters of the SVM were determined by applying the CRO algorithm implemented by the Algorithm 6 are C=10, σ 0.76065, and the kernel is linear. To assess the effectiveness of the proposed model in comparison to others, the authors conducted experiments using two widely recognized models for computer vision: VGG16 and MobileNetV2, both pre-trained on the ImageNet dataset for transfer learning. Table 2 presents the comparison results for the accuracy prediction metric in the training and testing phases, and Figure 10 graphically displays this performance metric. The results show that the combination of CNN and SVM in the proposed model has improved the classification accuracy from 92.81% to 99.18% compared to using only the CNN model. Furthermore, the simple CNN architecture with hyperparameters is optimized to attain better results than complex CNN architecture such as VGG16 and MobileNetV2. In addition, the proposed model is a custom-built lightweight CNN, resulting in shorter training times compared to VGG16 and MobileNetV2. The training time of the proposed CNN model is 1397.04 seconds, whereas the VGG16 and MobileNetV2 models took 1469.13 and 2329.75 seconds, respectively.

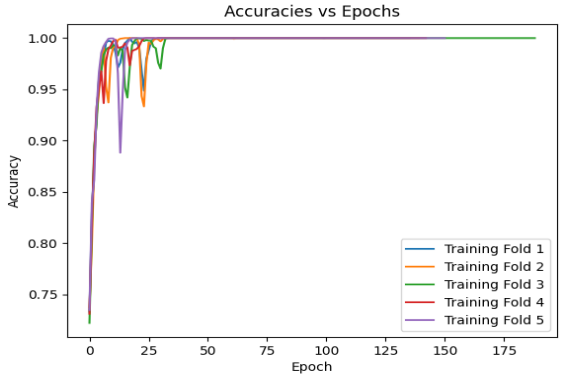

Figure 8. Accuracy classification over epoch

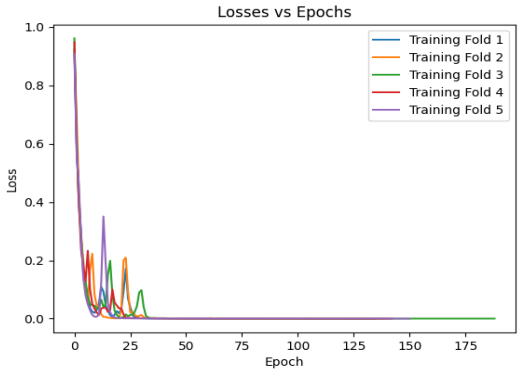

Figure 9. Error over epoch

Table 2. The model's classification accuracy

Structure	Training sample (%)	Test sample (%)
The proposed CNN	100.00	92.81
The combination of the proposed CNN and SVM	100.00	99.18
VGG16	99.63	89.76
MobileNetV2	98.27	90.53

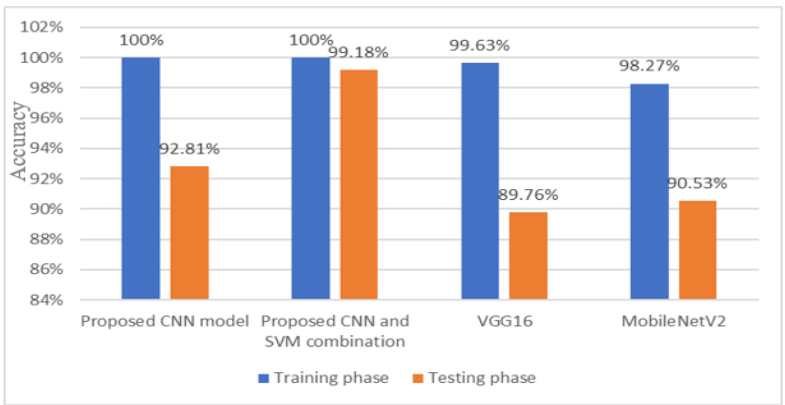

Figure 10. The accuracy prediction in the training and testing phase

4. CONCLUSION

This study presents the fusion of CNN, SVM, and CRO algorithms for the classification of various plant leaf diseases. CNN excels at extracting features from images, SVM outperforms a single neural network layer for classification tasks, and CRO is effective at solving optimization problems. The study aims to propose a lightweight and highly accurate prediction model for classifying plant leaf diseases. This simple model can be easily deployed on low-hardware devices such as mobile phones and embedded computers. The study has discovered that a simple CNN architecture with hyperparameter optimization can achieve better results than a complex CNN architecture for plant leaf disease classification. Furthermore, the results from this study suggest that replacing the last layer of the trained CNN with the optimized hyperparameters of an SVM model can attain better results than the original CNN model. This work has raised several questions that require further investigation. Future work must be done to ensure the proposed method works well on other image datasets.

ACKNOWLEDGEMENTS

The authors thank the Hanoi University of Industry for supporting this research.

REFERENCES

[1] C. Carroll, C. Carter, R. Goodhue, and C.-Y. C. L. Lawell, "Crop disease and agricultural productivity," Cambridge, MA, Jun. 2017. doi: 10.3386/w23513.

[2] S. Mathulaprangsan, K. Lanthong, D. Jetpipattanapong, S. Sateanpattanakul, and S. Patarapuwadol, "Rice diseases recognition using effective deep learning models," in *2020 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON)*, Mar. 2020, pp. 386–389, doi: 10.1109/ECTIDAMTNCON48261.2020.9090709.

[3] H. Genitha C., E. Dhinesh, and A. Jagan, "Detection of leaf disease using principal component analysis and linear support vector machine," in *2019 11th International Conference on Advanced Computing (ICoAC)*, Dec. 2019, pp. 350–355, doi: 10.1109/ICoAC48765.2019.246866.

[4] C. Jackulin and S. Murugavalli, "A comprehensive review on detection of plant disease using machine learning and deep learning approaches," *Measurement: Sensors*, vol. 24, p. 100441, Dec. 2022, doi: 10.1016/j.measen.2022.100441.

[5] S. Ramesh *et al.*, "Plant disease detection using machine learning," in *2018 International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C)*, Apr. 2018, pp. 41–45, doi: 10.1109/ICDI3C.2018.00017.

[6] P. Sharma, P. Hans, and S. C. Gupta, "Classification of plant leaf diseases using machine learning and image preprocessing techniques," in *2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Jan. 2020, pp. 480–484, doi: 10.1109/Confluence47617.2020.9057889.

[7] A. D. Mengistu, S. G. Mengistu, and D. M. Alemayehu, "An automatic coffee plant diseases identification using hybrid approaches of image processing and decision tree," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 9, no. 3, pp. 806–811, Mar. 2018, doi: 10.11591/ijeecs.v9.i3.pp806-811.

[8] M. Sharma, C. J. Kumar, and A. Deka, "Early diagnosis of rice plant disease using machine learning techniques," *Archives of Phytopathology and Plant Protection*, vol. 55, no. 3, pp. 259–283, Feb. 2022, doi: 10.1080/03235408.2021.2015866.

[9] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, Sep. 2016, doi: 10.3389/fpls.2016.01419.

[10] J. Andrew, J. Eunice, D. E. Popescu, M. K. Chowdary, and J. Hemanth, "Deep learning-based leaf disease detection in crops using images for agricultural applications," *Agronomy*, vol. 12, no. 10, p. 2395, Oct. 2022, doi: 10.3390/agronomy12102395.

[11] M. Jung *et al.*, "Construction of deep learning-based disease detection model in plants," *Scientific Reports*, vol. 13, no. 1, p. 7331, May 2023, doi: 10.1038/s41598-023-34549-2.

[12] J. Hang, D. Zhang, P. Chen, J. Zhang, and B. Wang, "Classification of plant leaf diseases based on improved convolutional neural network," *Sensors*, vol. 19, no. 19, p. 4161, Sep. 2019, doi: 10.3390/s19194161.

[13] M. Sardogan, A. Tuncer, and Y. Ozen, "Plant leaf disease detection and classification based on CNN with LVQ algorithm," in *2018 3rd International Conference on Computer Science and Engineering (UBMK)*, Sep. 2018, pp. 382–385, doi: 10.1109/UBMK.2018.8566635.

[14] B. V. Nikith, N. K. S. Keerthan, M. S. Praneeth, and D. T. Amrita, "Leaf disease detection and classification," *Procedia Computer Science*, vol. 218, pp. 291–300, 2023, doi: 10.1016/j.procs.2023.01.011.

[15] Y. M. A. Algani, O. J. M. Caro, L. M. R. Bravo, C. Kaur, M. S. Al Ansari, and B. K. Bala, "Leaf disease identification and classification using optimized deep learning," *Measurement: Sensors*, vol. 25, p. 100643, Feb. 2023, doi: 10.1016/j.measen.2022.100643.

[16] M. Nawaz *et al.*, "A robust deep learning approach for tomato plant leaf disease localization and classification," *Scientific Reports*, vol. 12, no. 1, p. 18568, Nov. 2022, doi: 10.1038/s41598-022-21498-5.

[17] W. Albattah, M. Nawaz, A. Javed, M. Masood, and S. Albahli, "A novel deep learning method for detection and classification of plant diseases," *Complex & Intelligent Systems*, vol. 8, no. 1, pp. 507–524, Feb. 2022, doi: 10.1007/s40747-021-00536-1.

[18] M. S. Anari, "A hybrid model for leaf diseases classification based on the modified deep transfer learning and ensemble approach for agricultural AIoT-based monitoring," *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–15, Apr. 2022, doi: 10.1155/2022/6504616.

[19] V. Balaji *et al.*, "Deep transfer learning technique for multimodal disease classification in plant images," *Contrast Media & Molecular Imaging*, vol. 2023, pp. 1–8, May 2023, doi: 10.1155/2023/5644727.

[20] M. Agarwal, A. Singh, S. Arjaria, A. Sinha, and S. Gupta, "ToLeD: tomato leaf disease detection using convolution neural network," *Procedia Computer Science*, vol. 167, pp. 293–301, 2020, doi: 10.1016/j.procs.2020.03.225.

[21] G. Ezhilarasan, S. R. Ranganathan, L. S. Mani, and S. Kadry, "An intelligent deep residual learning framework for tomato plant leaf disease classification," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 14, no. 3, pp. 3168–3176, 2024, doi: 10.11591/ijece.v14i3.pp3168-3176.

[22] P. Pujar, A. Kumar, and V. Kumar, "Efficient plant leaf detection through machine learning approach based on corn leaf image classification," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 13, no. 1, pp. 1139–1148, 2024, doi: 10.11591/ijai.v13.i1.pp1139-1148
[23] V. N. Vapnik, *The nature of statistical learning theory*. Springer, 1995.
[24] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, Nov. 2020, doi: 10.1016/j.neucom.2020.07.061.
[25] A. Y. S. Lam and V. O. K. Li, "Chemical reaction optimization: a tutorial," *Memetic Computing*, vol. 4, no. 1, pp. 3–17, Mar. 2012, doi: 10.1007/s12293-012-0075-1.
[26] D. P. Hughes and M. Salathe, "An open access repository of images on plant health to enable the development of mobile disease diagnostics," *arXiv preprint 1511.08060*, 2015, [Online]. Available: http://arxiv.org/abs/1511.08060.

BIOGRAPHIES OF AUTHORS

Nghien Nguyen Ba currently is a lecturer at the Faculty of Information Technology of Hanoi University of Industry. He received a diploma in electronic and computer engineering from Hanoi University of Science and Technology in 1999, a master's degree in Information Processing and Communication from Hanoi University of Science and Technology in 2004, and Ph.D. degree in System Control and Informatics from Czech Technical University in Prague in 2014. His research interests focus on parameter optimization for PID and PI controllers by soft computing techniques, environmental quality monitoring by IoT systems, and AI and IoT applications for smart agriculture. He can be contacted at email: nguyenbanghien_cntt@haui.edu.vn.

Nhung Nguyen Thi currently is a lecturer at the Faculty of Information Technology of Hanoi University of Industry. She received a diploma in information technology from Hanoi University of Science and Technology in 1999, and a master's degree in Information Systems from Military Academy Technique in 2008. Her research interests focus on AI implemented in Python. She can be contacted at email: nhungnt@haui.edu.vn.

Dung Vuong Quoc currently is a lecturer at the Information Center of Hanoi University of Industry. He received a diploma in Information technology from Hanoi University of Science and Technology in 1987 and a master's degree in Information Technology in 1992. His research interests focus on deep learning models. He can be contacted at email: dungvq@haui.edu.

Cuong Nguyen Cong currently is a lecturer at the Electrical Engineering of Hanoi University of Industry. He received a diploma in Electrical Engineering from Hanoi University of Science and Technology in 2007, a master's degree in Electrical Engineering from Czech Technical University in Prague in 2013, and Ph.D. degree in Electrical Engineering from the University of Mining and Geology in 2022. His research interests focus on parameter optimization for PID and PI controllers by soft computing techniques and renewable energy. He can be contacted at email: cuongnc@haui.edu.vn.