

Comparison among search algorithms for hyperparameter of support vector machine optimization

Nguyen Ba Nghien¹, Cuong Nguyen Cong², Nhung Nguyen Thi¹, Vuong Quoc Dung¹

¹Department of Information Technology, School of Information and Communication Technology, Hanoi University of Industry, Hanoi, Vietnam

²Department of Renewable Energy Engineering Technology, School of Electrical and Electronic Engineering, Hanoi University of Industry, Hanoi, Vietnam

Article Info

Article history:

Received May 18, 2024

Revised Jun 11, 2025

Accepted Jul 10, 2025

Keywords:

Chemical reaction optimization

Genetic algorithm

Grid search

Random search

Support vector machine

ABSTRACT

Support vector machine (SVM) is widely used in machine learning for classification and regression tasks, but its performance is highly dependent on hyperparameter tuning. Therefore, fine-tuning these parameters is key to improving accuracy and generality. Recently, many researchers have focused only on applying different algorithms to optimize these parameters. There is a shortage of studies that compare the performance of these methods. Hence, research is needed to compare the performance of these algorithms for the hyperparameters of the SVM optimization problem. This paper compares five optimization algorithms for tuning SVM hyperparameters: grid search (GS), random search (RS), Bayesian optimization (BO), genetic algorithm (GA), and the novel chemical reaction optimization (CRO) algorithm. Experimental results on benchmark datasets such as iris, digits, wine, breast cancer Wisconsin, and credit card fraud demonstrate that CRO consistently outperforms other methods in terms of classification scoring metrics and computational time. It achieves improvements in accuracy, precision, recall, and F1-score of up to 1% on balanced datasets and up to 10% on highly imbalanced datasets such as credit card fraud. It also reduces computation time by up to 50% compared to GS, BO, and RS. These findings suggest that CRO is a promising approach for hyperparameter optimization (HPO) of SVM models.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Nguyen Ba Nghien

Department of Information Technology, School of Information and Communication Technology

Hanoi University of Industry

Hanoi, Vietnam

Email: nguyenbanghien_cntt@hau.edu.vn

1. INTRODUCTION

Support vector machine (SVM) was proposed by Cortes and Vapnik in 1995 [1]. This model is based on statistical learning and is widely used for classification and regression problems. Many practical applications use SVM models, such as face detection and recognition [2]–[7], disease diagnosis [8]–[15], text recognition [16]–[21], sentiment analysis [22]–[24], intrusion detection systems [25]–[29], and plant disease detection [30]–[34]. The default parameter configurations of the SVM model might not consistently yield optimal outcomes for the given dataset, potentially leading to inadequate performance, overfitting, or underfitting. Consequently, parameter optimization becomes essential to refine the SVM model and enhance its predictive accuracy. The hyperparameter of the SVM consists of C, gamma, and kernel type. Optimizing them has several benefits, including:

- Improved predictive accuracy: optimization of the SVM model leads to better predictive performance, enabling more accurate predictions on new data points.
- Enhanced generalization: optimal hyperparameters help prevent overfitting or underfitting, leading to a model that generalizes well to unseen data. This ensures that the model's performance remains consistent across different datasets.
- Efficient resource utilization: an adequately optimized SVM model requires fewer computational resources for training and prediction, resulting in faster execution times and reduced computational costs.
- Increased robustness: a well-tuned SVM model is less sensitive to variations in the dataset and noise, making it more robust and reliable in real-world applications.

In general, improperly selected SVM hyperparameters can significantly degrade model performance, leading to misclassification, increased computational costs, or poor generalization. For example, an improperly tuned SVM can misclassify patients in medical diagnostics, leading to inaccurate diagnoses and potentially harmful treatments. In financial forecasting, an SVM model with suboptimal hyperparameters can fail to detect market trends accurately, resulting in significant economic losses for investors. Similarly, choosing an inappropriate kernel function in image recognition can result in poor object classification in computer vision tasks. In addition, our experiment on the wine dataset, which exhibits a significant feature variance of up to 46,546.424, demonstrates that an SVM model with improperly chosen hyperparameters results in low prediction accuracy, attaining only 66.34%. In contrast, the tuned SVM achieves an accuracy of up to 98.67%. For highly imbalanced datasets such as credit card fraud detection (CCFD), the default SVM model attains 0% in precision, recall, and F1-score metrics.

Logically, the SVM model can be tuned manually using a trial-and-error approach. However, this method is infeasible to use in actual-world applications due to computational constraints and the complexity of the data set. Therefore, optimization algorithms help solve this problem by systematically exploring the hyperparameter space and significantly improving model accuracy while reducing human effort and subjectivity.

Given the advantages of hyperparameter optimization (HPO) in SVM models and the limitations of manually tuning them, numerous studies have employed various optimization algorithms to identify the optimal hyperparameters for these models. Mantovani *et al.* [35] utilized random search (RS) techniques to optimize the hyperparameters of the SVM model, comparing the results with those obtained using grid search (GS). The experimental results demonstrate that a RS found better parameters than the default configuration and performed comparably to a GS. Guido *et al.* [36] employed genetic algorithms (GA) and a combination of GA and a decision tree for HPO of the SVM model. Research outcomes indicate that the novel approach achieves better or equal performance compared to other methods utilized in the literature for tuning the hyperparameters of the SVM model. Wenzhuo and Shuo [37] used an advanced whale optimization algorithm (WOA) to find optimal hyperparameters of the SVM model. Abdulraheem *et al.* [38] proposed to enhance the cat swarm optimization algorithm and applied it to discover the optimal hyperparameters of the SVM model. The experimental outcomes demonstrate that their proposed method achieves an average classification accuracy of 91.2% across 15 benchmark datasets. Ramasamy *et al.* [39] employed three heuristic search algorithms, including cuckoo search optimization (CSO), ant lion optimizer (ALO), and polar bear optimization (PBO), for optimizing the hyperparameters of the SVM model. The experimental results indicate that the performance of the three methods is comparable. Specifically, CSO, ALO, and PBO achieved average classification accuracies of 86.86%, 86.26%, and 84.44%, respectively. Lessmann *et al.* [40] utilized the GA to identify the optimal hyperparameters of the SVM model. They encoded the kernel function parameters as chromosome genes and then ran the GA to find the best individual, achieving the highest classification accuracy. The research outcomes demonstrate that the fusion of the GA and SVM achieves better classification accuracy than SVM alone, albeit requiring more time to find the optimal parameters.

The hyperparameter of the SVM model consists of the C, gamma, and kernel types. However, almost all researchers have optimized some of them, such as in [35]–[37], authors only optimized the C and gamma, and in [38]–[40], authors only optimized kernel type. Therefore, research is needed to optimize all parameters for the SVM model to improve its performance. In this article, we propose to use the chemical reaction optimization (CRO) algorithm to find all hyperparameters of the SVM model and investigate the hypothesis that the novel CRO algorithm for HPO of the SVM model will achieve the best performance compared to other popular algorithms, such as the GS, the RS, and the GA.

The CRO is a novel optimization method proposed by Lam and Li [41] in 2012. The algorithm simulates a chemical reaction chain to convert a less stable substance (with large kinetic energy (KE)) into a stable substance (with small KE) by releasing energy into the environment. In the context of hyperparameter tuning, molecules represent candidate hyperparameter sets that initially may be far from optimal. Chemical reactions correspond to search operations such as decomposition or synthesis, allowing exploration of different hyperparameter combinations. Energy levels are analogous to model performance metrics, such as

accuracy. Unstable molecules, analogous to poor hyperparameter choices, undergo reactions until they reach a lower energy state, analogous to better performance. The system converges to a stable state, similar to how an optimized hyperparameter set improves model accuracy over iterations. It integrates the strengths of both the GA and simulated annealing (SA). CRO preserves energy conservation in a manner similar to the Metropolis algorithm in SA, while its decomposition and synthesis processes resemble the crossover and mutation operations found in GA. Hence, the CRO performs both local and global searches.

Numerous optimization challenges have been tackled successfully using CRO. Cong *et al.* [42] used CRO to tune the proportional integral (PI) controller to fast-track the active and reactive power of the doubly fed induction generator (DFIG) with maximum power point tracking (MPPT) when the wind speed changes. The experimental results demonstrate that CRO outperforms GA and PSO regarding control error. Ba *et al.* [43] utilized CRO to find optimal PID controller parameters. The simulation results show that CRO achieves the best performance compared to GA, PSO, WOA, and teaching learning-based optimization (TLBO). Research by Yu *et al.* [44], CRO is used to train neural networks, and the neural networks trained with CRO exhibit the lowest testing error rate compared to several other representative evolutionary methods. Ba *et al.* [45] used CRO to optimize the hyperparameters of a deep learning model, such as the number of filters in each convolutional layer and the number of neurons in the fully connected layer. The experimental results demonstrate that the proposed optimized deep learning model outperforms VGG-16 and MobileNet in plant leaf disease recognition. The structure of this manuscript is organized as follows: the next section presents the methodology, followed by the experimental results. Finally, the last section includes the conclusion and future work.

2. METHOD

2.1. Support vector machine

SVM is a power supervisory learning technique applied widely in classification problems. SVM seeks to identify the optimal hyperplane that best separates the data into two classes, maximizing the margin—the distance between the hyperplane and the nearest data points. Figure 1 shows the hyperplane and its corresponding margin.

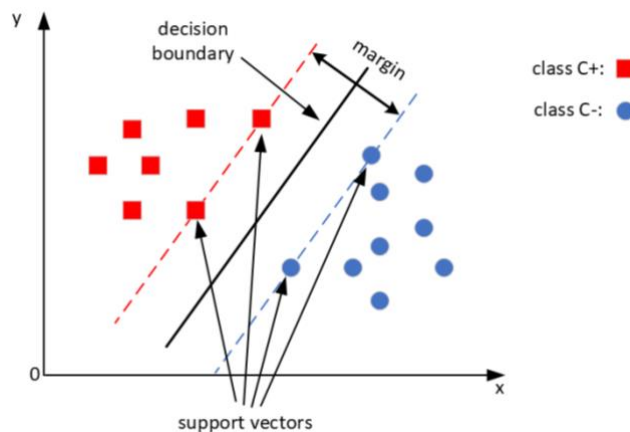


Figure 1. Illustrate the key features of the SVM

Data classification problems are based on some given samples. Specifically, given m samples x_1, x_2, \dots, x_m corresponding to the outputs $y_1, y_2, \dots, y_m \in \Delta$, where Δ is a fixed discrete set. Here, we focus on the binary classification problem, meaning $\Delta = \{-1, 1\}$. Then x_i is classified into a class C^+ if $y_i = 1$, otherwise, it is classified into a class C^- . Therefore, it is necessary to find a classification function $f(x)$ defined as follows:

$$f: R^n \rightarrow R, f(x_i) = y_i, \forall i \in M = \{1, 2, \dots, m\} \quad (1)$$

If there exists a function g such that $g(x_i) > 0$ for all x_i in C^+ and $g(x_i) < 0$ for all x_i in C^- then the classification function can be chosen as $f(x) = \text{sgn}(g(x))$ and the surface $S_g = \{x \in R^n: g(x) = 0\}$ is

called the decision surface. S_g divides R^n into two regions, corresponding to two classes C^+ and C^- . Especially when g is an affine function $g(x) = w \cdot x + b$, then S_g is a hyperplane and is called a linear decision surface. Then there exists a pair $(w, b) \in R^n \times R$ such that:

$$\begin{cases} w \cdot x_i + b \geq 1, \forall x_i \in C^+ \\ w \cdot x_i + b \leq -1, \forall x_i \in C^- \end{cases} \quad (2)$$

and $f_{w,b}(x) = \text{sgn}(w \cdot x + b)$, $S_{w,b} = \{x \in R^n : w \cdot x + b = 0\}$. Finding the optimal hyperplane with maximum margin requires solving the following optimization problem:

$$\begin{cases} \varphi(w, b) = \frac{1}{2} \|w\|^2 \rightarrow \min \\ 1 - y_i(w \cdot x_i + b) \leq 0, i \in M_i \end{cases} \quad (3)$$

The quadratic programming algorithms can solve this problem. However, to be able to extend to the case of non-linearly separable data, the Lagrange multiplier method is introduced:

$$L(w, b, \beta) = \frac{1}{2} \|w\|^2 + \sum_{i=1}^m \beta_i (1 - y_i(w \cdot x_i + b)) \quad (4)$$

where, $\beta = (\beta_1, \beta_2, \dots, \beta_m)$, $\beta_i \in R^+$, $\forall i \in M$. The Lagrange dual function is defined:

$$h(\beta) = \min_{w, b} L(w, b, \beta) \quad (5)$$

Substitute $\frac{\partial L(w, b, \beta)}{\partial w} = 0$, and $\frac{\partial L(w, b, \beta)}{\partial b} = 0$ into (4) we obtain:

$$h(\beta) = \sum_{i=1}^m \beta_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \beta_i \beta_j y_i y_j (x_i x_j) \quad (6)$$

Subject to:

$$\beta_i \geq 0, \forall i \in M \quad (7)$$

$$\sum_{i=1}^m \beta_i y_i = 0 \quad (8)$$

When perfect separation is unattainable, slack variables ρ are introduced for samples within the margin and penalty parameter C . It manages the balance between maximizing the margin and minimizing classification errors during training. The optimization problem can then be redefined as:

$$\begin{cases} \varphi(w, b, \rho) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \rho_i \rightarrow \min \\ 1 - y_i(w \cdot x_i + b) - \rho_i \leq 0, \forall i \in M \end{cases} \quad (9)$$

The Lagrange function is defined:

$$L(w, b, \rho, \beta, \gamma) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \rho_i + \sum_{i=1}^m \beta_i (1 - y_i(w \cdot x_i + b) - \rho_i) - \sum_{i=1}^m \gamma_i \rho_i \quad (10)$$

The Lagrange dual function is specified as follows:

$$h(\beta, \gamma) = \sum_{i=1}^m \beta_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \beta_i \beta_j y_i y_j (x_i x_j) \quad (11)$$

with constraints:

$$\begin{cases} \sum_{i=1}^m \beta_i y_i = 0 \\ 0 \leq \beta_i \leq C, \forall i \in M \end{cases} \quad (12)$$

The classification function is obtained:

$$f(x) = \text{sgn}(\sum_{i=1}^m \beta_i y_i (x_i x) + b) \quad (13)$$

When the samples are not linearly separable, the transformation function $\phi(x)$ that maps input space to higher dimension feature space. The kernel function is defined as follows:

$$K(x, y) = \phi(x) \cdot \phi(y) \quad (14)$$

Finally, the classification function is obtained:

$$f(x) = \text{sgn}(\sum_{i=1}^m \beta_i y_i K(x, x_i) + b) \quad (15)$$

2.2. Hyperparameter optimization formulation

In the design phase of machine learning models, efficiently exploring the hyperparameter space using optimization techniques helps identify the HPO for the models. As described in [46], HPO consists of four key components: an estimator with its objective function, a search space (or configuration space), a search algorithm for discovering tuning parameters, and a scoring function to evaluate the performance of different configurations. The primary goal of HPO is to locate an optimal point within the search space.

$$s^* = \arg \min_{\{s \in S\}} f(s) \quad (16)$$

Here, $f(s)$ presents the objective function to be minimized, s^* is an optimal point in search space S . For the SVM model, the search space can include the kernel type, penalty parameter C , and kernel gamma coefficient. Each tuning parameter has a discrete, continuous, or categorical value domain with several options m_i in the corresponding search space S_i . Therefore, the search space can be represented as follows:

$$S = \begin{matrix} S_1 & S_{1,1} & S_{1,2} & \cdots & S_{1,m_1} \\ S_2 & S_{2,1} & S_{2,2} & \cdots & S_{2,m_2} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ S_n & S_{n,1} & S_{n,2} & \cdots & S_{n,m_n} \end{matrix} \quad (17)$$

2.3. Grid search

GS is an exhaustive method that explores a predefined range of hyperparameters. It aims to discover the optimal combination that delivers the highest performance. This technique generates a grid encompassing all feasible hyperparameter values and then assesses each combination through cross-validation to determine its efficacy.

2.4. Random search

RS provides a more efficient alternative to GS by randomly sampling hyperparameters within a specified range. These sampled values are then assessed through cross-validation. This method decreases the computational expense of GS while maintaining exploration across a broad scope of hyperparameters.

2.5. Bayesian optimization

The principle of Bayesian optimization (BO) is to use a probabilistic model to find the extreme values of a function that is complex, expensive to evaluate, or has an unknown form. It is a highly effective global optimization method, particularly well-suited for hyperparameter tuning in machine learning. BO builds a probabilistic model (typically a Gaussian process) to approximate the objective function. It then uses this model to predict and select the next evaluation point to reach the global optimum with the fewest possible evaluations.

2.6. Genetic algorithm

GA simulates natural evolution to solve optimization problems. The process begins with an initial population of the potential solutions, which undergoes evolution influenced by basic operators such as natural selection, crossover, and mutation, resulting in a refined set of solutions. Ultimately, the optimal solution is found.

In a GA, each solution is represented by an individual, with chromosomes comprising genes. The number and value of genes vary based on the optimization problem being addressed. For the HPO problem of the SVM model, each chromosome consists of three genes representing parameters C , gamma, and kernel function type.

GA rely on fitness functions to evaluate individuals within a population. Individuals with higher adaptive values are more likely to be selected into new populations. Randomly selected individuals undergo crossover and mutation processes. During crossover, a pair of parents create two new offspring, inheriting a

portion of the dad's and mom's genes each. These offspring replace their parents in the new population. Mutation involves changing the value of a gene at a randomly selected position in an individual. Figure 2 illustrates the GA flow diagram.

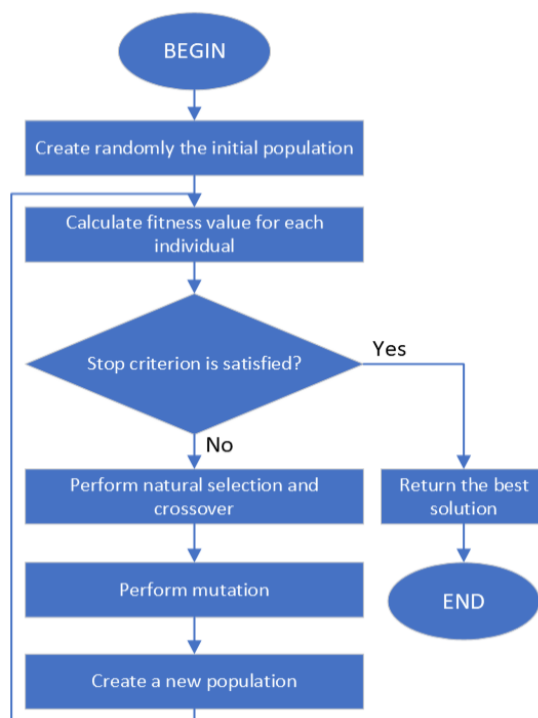


Figure 2. The genetic algorithms flow diagram

2.7. Chemical reaction optimization

The CRO algorithm is a modern stochastic search technique designed for solving optimization problems. It draws inspiration from the behavior of loosely coupled chemical reactions, simulating the dynamic transformations that occur within such systems. In a chemical reaction system, substances interact with one another and with their environment, each possessing potential energy and KE, while the environment itself is modeled as a central energy buffer [40]. As reactions proceed toward equilibrium, molecules stabilize at minimal potential energy states.

CRO mimics this equilibrium-seeking behavior by converting potential energy into KE and gradually releasing excess energy into the environment. The algorithm is structured around four fundamental reaction operators: on-wall ineffective collision, decomposition, inter-molecular ineffective collision, and synthesis. The two ineffective collision types enable local search, while decomposition and synthesis support global search. By combining these mechanisms, CRO efficiently balances exploration and exploitation to locate the global optimum within a feasible region. CRO incorporates the advantages of both SA and GA. It upholds energy conservation principles similar to the Metropolis algorithm used in SA, and its decomposition and synthesis operations resemble the crossover and mutation mechanisms in GA.

In CRO, each molecule is defined by a molecular structure (ω), which represents a candidate solution, along with its associated potential energy (evaluated via the fitness function) and KE (indicating the molecule's capacity to withstand energy increases). The algorithm simulates its four core reactions using three primary operators: neighbor, decomposition, and synthesis. The neighbor operator is applied during collision events to produce a new solution by randomly modifying elements of an existing one, thus enabling localized search for improvement. Compared to traditional GAs based on mutation and crossover operators, which may rapidly converge to a good set of individuals in the early stages, this leads to a loss of diversity in the population, prone to getting stuck at a local extreme point in high-dimensional spaces, such as SVM hyperparameter tuning. CRO provides a more adaptive and energy-aware search process. It dynamically adjusts the search path based on energy levels, which can help escape local optima more efficiently. Similarly, while SA performs a probability-reducing search process through temperature cooling, CRO combines this annealing-like behavior with molecular interactions that promote diversity and convergence. This hybrid nature allows CRO to balance exploration and exploitation more effectively, which is especially

suitable for complex optimization situations such as those encountered during SVM parameter selection. Algorithm 1 depicts the pseudocode of this operator.

Algorithm 1: Neighbor (ω)

Input: a molecular structure ω

Output: a neighbor of the ω is ω'

```

1. Clone  $\omega$  to  $\omega'$ 
2. Create a random integer  $i$  that is less than the total elements in  $\omega$ .
3. IF  $\omega'(i)$  is discrete THEN
4.    $\omega'(i) = s_i, s_i \in S_i$ 
5. ELSE IF  $\omega'(i)$  is continuous THEN
6.    $\omega'(i) = \omega(i) + \rho_i$ 
7. ENDIF
8. Return  $\omega'$ 

```

The decomposition operator is used in the decomposition reaction. This operator creates two new solutions ω'_1, ω'_2 from the given 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)

Input: a molecular structure ω

Output: two new molecular structure ω'_1, ω'_2

```

1. Copy  $\omega$  to  $\omega'_1$ 
2. Random change 50% of the element of  $\omega'_1$  by replacing  $\omega'_1(i)$  with  $s_i \in S_i$  for  $\omega'_1(i)$  is a discrete value or update  $\omega'_1(i) = \omega'_1(i) + \rho$  for  $\omega'_1(i)$  is a continuous value.
3. Repeat steps 1 and 2 for the  $\omega'_2$  in a similar manner.
4. Return  $\omega'_1, \omega'_2$ 

```

The synthesis operator simulates the synthesis reaction. This operator generates a new solution, ω' , by combining the two given molecules, ω_1 and ω_2 . The procedure entails randomly selecting components from two molecules with equal probability to form a new molecule. The pseudocode outlining this operation is presented in Algorithm 3.

Algorithm 3: Synthesis (ω_1, ω_2)

Input: Two molecules ω_1 and ω_2

Output: A new molecule ω'

```

1. FOR  $i = 1$  TO  $n$  DO
2.   Generate a random number  $r$  in a range of 0 and 1.
3.   IF  $r > 0.5$  THEN
4.      $\omega'(i) = \omega_1(i)$ 
5.   ELSE
6.      $\omega'(i) = \omega_2(i)$ 
7.   ENDIF
8. ENDFOR
9. Return  $\omega'$ 

```

The pseudocode for the CRO is presented in Algorithm 4. The initial parameters of the algorithm include the number of molecules (solutions), where Mole_Coll determines whether a mono-molecular collision or an inter-molecular collision occurs. Additionally, 'buffer' represents the initial energy of the environment, while 'InitialKE' denotes the initial KE of each molecule. δ and θ are parameters that control the decomposition and synthesis reactions, respectively.

Algorithm 4: CRO

```

1. Initial parameters of the CRO algorithm.
2. Create PopSize of molecules.
3. DO
4.   Create a random number  $k$  between 0 and 1.
5.   IF  $k > \text{MoleColl}$  THEN
6.     Select randomly one molecule from the population
7.     IF  $\text{numHit} - \text{minHit} > \delta$  THEN
8.       Decomposition is performed
9.     ELSE
10.      On-wall ineffective collision has taken place.
11.    ENDIF
12.  ELSE
13.    Select randomly two or more molecules from the population.
14.    IF  $\text{KE} < \theta$  THEN
15.      Synthesis operator is performed
16.    ELSE

```

```

17. Inter-molecular ineffective collision.
18. ENDIF
19. IF a new min solution is found THEN
20. Update the best solution
21. ENDIF
22. WHILE the stop condition is not met
23. RETURN the best solution

```

2.8. The proposed method

The proposed method is demonstrated in Figure 3. Firstly, the search space and objective function are defined. Next, the search algorithms presented above will be applied sequentially to find the C, gamma, and kernel types of the SVM model. We apply a 5-fold CV when finding optimal hyperparameters. These parameters are encoded as genes of chromosomes in the GA algorithm and molecules in the CRO algorithm. After finding optimal hyperparameters, the dataset will be divided into a training dataset and a testing dataset. We train the optimal SVM model with a training set and test by test set.

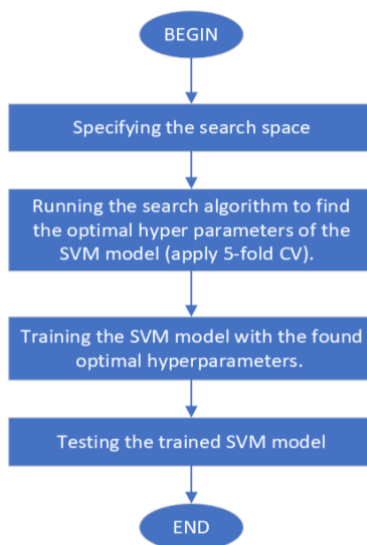


Figure 3. The principle of the proposal method

3. RESULTS AND DISCUSSION

To compare the performance of four algorithms presented above. Four benchmark datasets, including iris plants, handwritten digits, wine, breast cancer Wisconsin, and CCFD data sets, are used for an experiment. The CCFD dataset contains transactions that occurred over two days in September 2013, including 492 fraudulent cases out of a total of 284,807 transactions. Unfortunately, due to hardware limitations, the authors restricted their experiment to using only 10% of the original dataset while still maintaining the proportion of fraudulent transactions at 0.172% of the total. The features of each dataset are summarized in Table 1.

Table 1. Summary of the experimented dataset

Dataset	Number of samples	Number of features	Number of classes	Number of samples per class
Iris plants	150	4	3	50
Handwritten digits	1797	64	10	Nearly 180
Wine	178	13	3	59, 71, 48
Breast cancer Wisconsin	569	30	2	212, 357
Credit card fraud detection	28480	30	2	49 (frauds), 28431

The search space and configuration for each algorithm are summarized in Table 2. The range of each hyperparameter is chosen based on practical usage. If the range is too small or too large, the SVM model may become overfitted or underfitted, resulting in poor performance during testing and deployment. Several scores are used to evaluate each method's performance. The accuracy, precision, recall, and F1-score are defined in the following:

$$accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (18)$$

$$precision = \frac{TP}{TP+FP} \quad (19)$$

$$recall = \frac{TP}{TP+FN} \quad (20)$$

$$F1 - score = \frac{2 \times TP}{2 \times TP + FP + FN} \quad (21)$$

where: TP, or true positive, represents the number of correctly classified instances in the positive class. TN, or true negative, presents the number of correctly classified instances in the negative class. FP, or false positive, represents the number of samples incorrectly classified as positive when they belong to the negative class. FN, or false negative, represents the number of instances incorrectly classified as negative when they belong to the positive class.

Table 2. Summary of the search space and algorithm's configuration

Algorithm	C	Gamma	Kernel type	Configuration
GS	[0.1, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]	[0.0001, 0.0002, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]	[sigmoid, linear, rbf, poly]	5-fold CV
RS	Continuous range from 0.1 to 100	Continuous range from 0.0001 to 1.0	[sigmoid, linear, rbf, poly]	Number iteration: 50, seed is default, 5-fold CV
BO	Continuous range from 0.1 to 100	Continuous range from 0.0001 to 1.0	[sigmoid, linear, rbf, poly]	Number iteration: 50, seed is 42, 5-fold CV
GA	Continuous range from 0.1 to 100	Continuous range from 0.0001 to 1.0	[sigmoid, linear, rbf, poly]	Number iteration: 50, pop size: 10, crossover probability: 0.5, mutation probability: 0.2, 5-fold CV
CRO	Continuous range from 0.1 to 100	Continuous range from 0.0001 to 1.0	[sigmoid, linear, rbf, poly]	Number iteration: 50, pop size: 10, 5-fold CV

The C, gamma, and kernel types are encoded as genes in the chromosome. The fitness function is determined by the mean accuracy prediction of the 5-fold cross-validation training process. The mutation operator is performed as follows: it randomly replaces one chromosome gene with a random value within its domain. The parameters of the GA are set as follows: the number of individuals is 10, the crossover probability is 0.5, the mutation probability is 0.2, and the number of generations is 50.

For using the CRO algorithm, the parameters C, gamma, and kernel types are encoded within the molecular structure. The KE of each molecule is determined by one minus the mean accuracy prediction of the 5-fold cross-validation training process. The initial parameters of the CRO algorithm are chosen according to practical suggestions in [41]: $\delta = 300$, $\theta = 500$, initial KE is 100, KE lost rate is 0.1, mole collision is 0.1, number of molecules is 10, and number of generations is 50.

The experiment is implemented by using the Python programming language running on Intel Core i5 6300U, 3 GHz, DDR4 16 GB. The GS, BO, and RS algorithms are supported by a scikit-learn framework. The GA is supported by the distributed evolutionary algorithms in Python (DEAP) framework. The CRO algorithm is implemented by us. The HPO and score metrics are presented in Table 3.

After obtaining the optimal hyperparameters, the SVM model is trained on 80% of the samples from each dataset. It is then tested on the remaining 20%. Figures 4 to 8 illustrate a heat map of the confusion, and performance metrics generated by the SVM model for classifying handwritten digits and wine data sets, providing an example.

The experiment results demonstrate that the CRO method performs best over five benchmark datasets. This method not only increases the mean classification accuracy during the training phase using 5-fold cross-validation, but also achieves better accuracy in predictions during the testing phase, and decreases computation time. This is achieved because the CRO algorithm inherits the advantages of both GA and SA. This hybrid nature allows CRO to balance exploration and exploitation more effectively. The experiment results also reveal that when the variance of features is small (3.896 for the Iris dataset, and 36.201 for the handwritten dataset), the SVM with default hyperparameters ($C = 1$, kernel type is a radial basic function and gamma is a scale) achieves reasonable accuracy classification of 96.67% and 96.33% for the Iris and Handwritten Digits dataset, respectively. In this case, the fine-tuning SVM model improves the performance by about 2 percent. On the contrary, when the variance of features is extensive. Specifically, the variance of features is 46546.424 and 52119.705 for the wine and breast cancer data sets, respectively. The SVM model with default parameters gives low prediction results, especially with the wine data set, the

prediction accuracy is only 66.34 percent. In this case, the tuned model gets a much higher prediction accuracy, improving by about 30%. Furthermore, on highly imbalanced datasets such as credit card fraud, the default SVM model can even yield 0% in precision, recall, and F1-score metrics, whereas the SVM model with hyperparameters optimized by the CRO algorithm achieves the best performance across all evaluation metrics. Therefore, the CRO algorithm is well-suited for tuning SVM models, especially when applied to complex and highly imbalanced datasets.

Table 3. Summary of the HPO and score matrix for each method

Dataset	Method	C	Gamma	Kernel type	Mean accuracy (%)	Mean precision (%)	Mean recall (%)	Mean F1-score (%)	Time spent (s)
Iris plants	Default	1.00	-	rbf	96.67	96.85	96.67	96.66	0.015
	hp.parameters	1.00	0.70	linear	98.00	98.18	98.00	97.99	14
	Grid search								
	Random search	0.46	0.77	linear	98.00	98.00	98.18	97.99	3
	Bayesian opt.	0.22	0.68	linear	97.33	96.29	96.67	96.28	7
	Genetic algorithm	0.57	0.99	linear	98.67	98.79	98.67	98.66	4
Handwritten digits	CRO algorithm	0.52	1.00	linear	98.67	98.79	98.67	98.66	1
	Default	1.00	-	rbf	96.33	96.62	96.33	96.30	0.278
	hp.parameters	7.00	0.0002	rbf	96.94	97.13	96.93	96.91	269
	Grid search								
	Random search	8.74	0.0002	rbf	96.88	97.06	96.87	96.86	119
	Bayesian opt.	37.51	0.95	poly	95.21	96.86	96.83	96.88	21
Wine	Genetic algorithm	9.56	0.0003	rbf	97.27	97.43	97.27	97.26	120
	CRO algorithm	10.0	0.0004	rbf	97.38	97.55	97.38	97.38	117
	Default	1.00	-	rbf	66.34	60.13	62.87	60.35	0.015
	hp.parameters	0.10	0.001	linear	96.11	96.44	96.31	96.20	120
	Grid search								
	Random search	2.12	0.32	linear	96.67	96.99	97.06	96.72	16
Breast cancer Wisconsin	Bayesian opt.	5.90	0.86	poly	96.67	93.75	93.75	93.75	21
	Genetic algorithm	1.47	0.24	linear	96.67	96.99	97.06	96.72	120
	CRO algorithm	2.03	0.006	linear	96.67	96.99	97.06	96.72	5
	Default	1.00	-	rbf	91.22	92.66	88.92	90.12	0.022
	hp.parameters	9.00	0.90	linear	95.43	95.36	94.94	95.09	994
	Grid search								
Credit card fraud detection	Random search	9.41	0.91	linear	95.43	96.45	96.06	96.23	504
	Bayesian opt.	4.17	0.13	linear	95.36	92.86	94.43	93.60	674
	Genetic algorithm	9.12	0.95	linear	95.25	95.11	94.80	94.90	6660
	CRO algorithm	9.01	0.91	linear	95.43	95.36	94.94	95.09	203
	Default	1	-	rbf	99.73	0.00	0.00	0.00	20
	hp.parameters	0.1	0.0001	linear	99.80	66.67	40.00	50.00	9315
	Grid search	83.82	4.50	linear	99.91	60.00	50.00	54.54	18927
	Random search	30.09	0.032	linear	99.91	69.23	90.00	78.26	1174
	Bayesian opt.	21.52	0.165	linear	99.92	75.00	90.00	81.00	20713
	Genetic algorithm	8.79	0.1308	linear	99.94	76.92	100.00	86.95	960
	CRO algorithm								

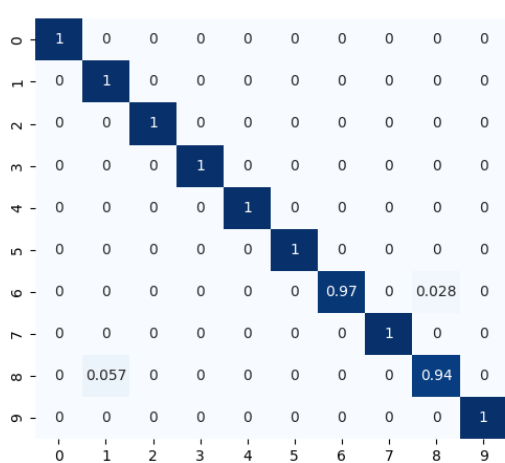


Figure 4. Performance of the GS algorithm

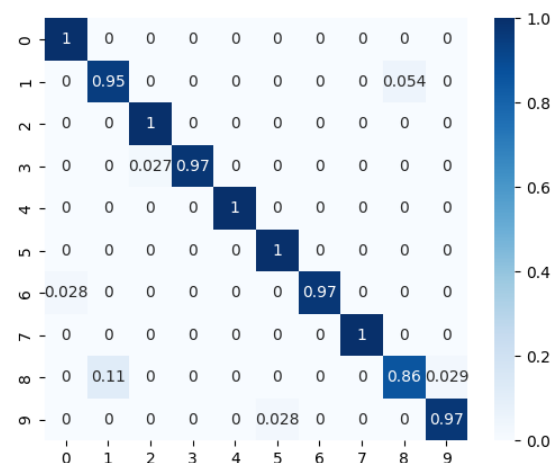


Figure 5. Performance of the RS algorithm

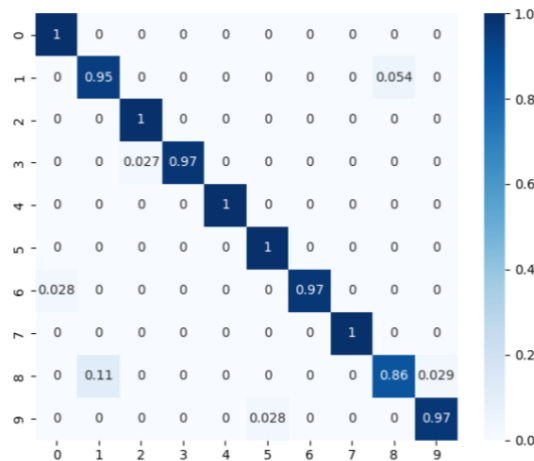


Figure 6. Performance of the GA

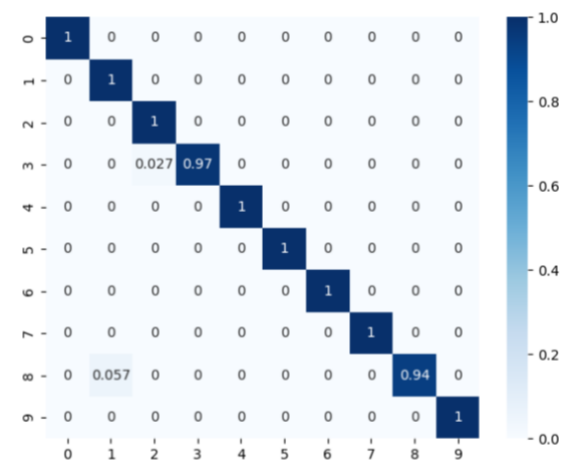


Figure 7. Performance of the CRO algorithm

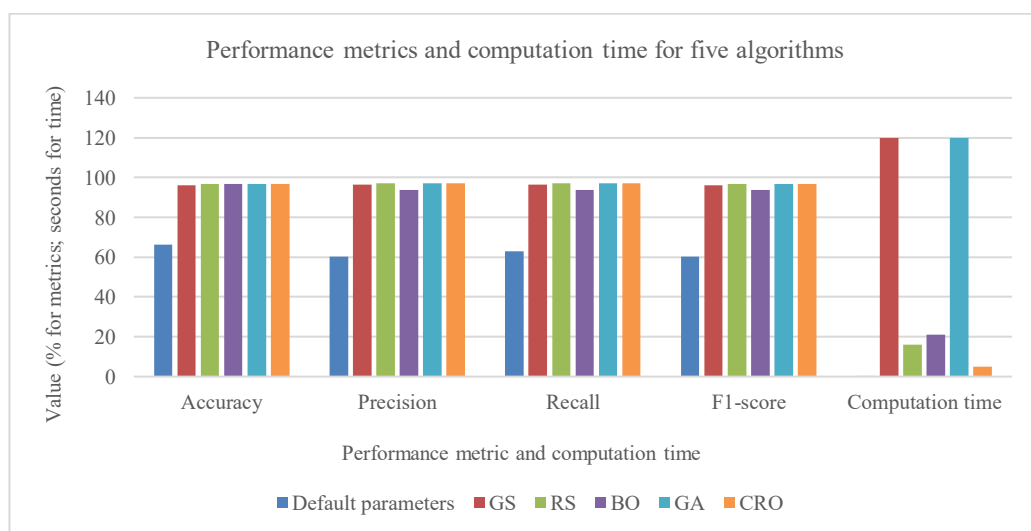


Figure 8. Compare the performance metrics and computation times for five algorithms

4. CONCLUSION

This manuscript presents five algorithms for hyperparameters of the SVM model optimization. The GS algorithm utilizes exhaustive techniques, combining all possible parameters to find the optimal set. Consequently, this algorithm is suitable only for cases where the parameter's value domain is discrete and contains few values. When the parameter's value domain is extensive, the search time can become significantly lengthy. The RS algorithm randomly selects combinations of hyperparameters from within the defined search space. It doesn't follow any specific pattern or order; instead, it randomly samples combinations. The search space can be discrete and continuous. Therefore, the RS algorithm can be more efficient than the GS algorithm because it doesn't exhaustively search through all possible combinations of hyperparameters. Instead, it explores a randomly selected subset of the search space. The GA mimics natural evolution using three operators: natural selection, crossover, and mutation. Each potential solution is represented as a set of parameters, often called a chromosome or an individual. After several generations, the best individual with the highest fitness value becomes the solution to an optimization problem. The CRO algorithm is a metaheuristic optimization algorithm inspired by the principles of chemical reactions and the behavior of molecules in a chemical system. It is based on four basic reactions: on-wall ineffective collision, decomposition, inter-molecular ineffective collision, and synthesis. In CRO, each potential solution is represented as a set of parameters, often called a molecule. After a series of chemical reactions, the best molecule with the smallest KE is found, representing the solution to an optimization problem. Five

algorithms are applied for hyperparameters of the SVM model optimization. The study has discovered that the dataset's feature variance is large, and using the default model, SVM will achieve low performance for prediction. Furthermore, the results from this study suggest that the CRO algorithm provides the best classification accuracy in the training and testing phases for all datasets in the experiment and decreases computational time significantly. This achievement is obtained because the CRO combines both advantages of the GA algorithm and SA and is energy-aware of the chemical reaction process. This research can also be applied to optimize CNN hyperparameters, such as the filter size in each convolutional layer and the number of neurons in fully connected layers, aiming to simplify the network architecture while maintaining high prediction accuracy. In the future, the authors plan to experiment with more complex datasets to extract valuable insights, test CRO with more complex models like deep learning algorithms and apply the CRO to other types of machine learning tasks such as regression or unsupervised learning.

FUNDING INFORMATION

This work was supported by Hanoi University of Industry under Grant Number: 2024/QCCTNB-DHCN.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Nguyen Ba Nghien	✓	✓	✓	✓	✓	✓		✓	✓	✓		✓	✓	
Cuong Nguyen Cong						✓				✓	✓			
Nhung Nguyen Thi		✓	✓	✓		✓		✓	✓	✓			✓	✓
Vuong Quoc Dung	✓						✓			✓	✓			

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

DATA AVAILABILITY

The data that support the findings of this study are openly available in the UCI ML at <https://doi.org/10.24432/C56C76>, <https://doi.org/10.24432/C53K8Q>, <https://doi.org/10.24432/C5PC7J>, and <https://doi.org/10.24432/C5DW2B>; and Kaggle Repository at <https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud>.

REFERENCES





- [1] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.
- [2] Q. Q. Tao, S. Zhan, X. H. Li, and T. Kurihara, "Robust face detection using local CNN and SVM based on kernel combination," *Neurocomputing*, vol. 211, pp. 98–105, 2016, doi: 10.1016/j.neucom.2015.10.139.
- [3] S. Kumar, S. Singh, and J. Kumar, "Multiple face detection using hybrid features with SVM classifier," *Advances in Intelligent Systems and Computing*, vol. 847, pp. 253–265, 2019, doi: 10.1007/978-981-13-2254-9_23.
- [4] S. Kumar, S. Singh, and J. Kumar, "Automatic live facial expression detection using genetic algorithm with haar wavelet features and SVM," *Wireless Personal Communications*, vol. 103, no. 3, pp. 2435–2453, 2018, doi: 10.1007/s11277-018-5923-y.
- [5] H. I. Dino and M. B. Abdulrazzaq, "Facial expression classification based on SVM, KNN and MLP classifiers," in *2019 International Conference on Advanced Science and Engineering (ICOASE)*, 2019, pp. 70–75, doi: 10.1109/ICOASE.2019.8723728.
- [6] G. Azzopardi, A. Greco, and M. Vento, "Gender recognition from face images using a fusion of SVM classifiers," in *Image Analysis and Recognition (ICIAR 2016)*, 2016, pp. 533–538, doi: 10.1007/978-3-319-41501-7_59.
- [7] L. Shi, X. Wang, and Y. Shen, "Research on 3D face recognition method based on LBP and SVM," *Optik*, vol. 220, 2020, doi: 10.1016/j.ijleo.2020.165157.

- [8] M. I. Al-Janabi, M. H. Qutqut, and M. Hijjawi, "Machine learning classification techniques for heart disease prediction: a review," *International Journal of Engineering & Technology*, vol. 7, no. 4, pp. 5373–5379, 2018.
- [9] N. C. Caballé, J. L. C. -Sequera, J. A. G. Pulido, J. M. G. -Pulido, and M. L. P. -Luque, "Machine learning applied to diagnosis of human diseases: A systematic review," *Applied Sciences*, vol. 10, no. 15, 2020, doi: 10.3390/app1015135.
- [10] S. Mahajan, G. Bangar, and N. Kulkarni, "Machine learning algorithms for classification of various stages of alzheimer's disease: a review," *International Research Journal of Engineering and Technology*, vol. 7, no. 8, pp. 817–824, 2020.
- [11] A. Asuntha, A. Brindha, S. Indirani, and A. Srinivasan, "Lung cancer detection using SVM algorithm and optimization techniques," *Journal of Chemical and Pharmaceutical Sciences*, vol. 9, no. 4, pp. 3198–3203, 2016.
- [12] S. Vadali, G. V. S. R. Deekshitulu, and J. V. R. Murthy, "Analysis of liver cancer using data mining SVM algorithm in MATLAB," *Advances in Intelligent Systems and Computing*, vol. 816, pp. 163–175, 2019, doi: 10.1007/978-981-13-1592-3_12.
- [13] J. Alam, S. Alam, and A. Hossan, "Multi-stage lung cancer detection and prediction using multi-class SVM classifier," in *2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2)*, 2018, pp. 1–4, doi: 10.1109/IC4ME2.2018.8465593.
- [14] A. D. Dolatabadi, S. E. Z. Khadem, and B. M. Asl, "Automated diagnosis of coronary artery disease (CAD) patients using optimized SVM," *Computer Methods and Programs in Biomedicine*, vol. 138, pp. 117–126, 2017, doi: 10.1016/j.cmpb.2016.10.011.
- [15] Z. Qiao, Q. Zhang, Y. Dong, and J.-J. Yang, "Application of SVM based on genetic algorithm in classification of cataract fundus images," in *2017 IEEE International Conference on Imaging Systems and Techniques (IST)*, 2017, pp. 1–5, doi: 10.1109/IST.2017.8261541.
- [16] H. Li, "Text recognition and classification of English teaching content based on SVM," *Journal of Intelligent and Fuzzy Systems*, vol. 39, no. 2, pp. 1757–1767, 2020, doi: 10.3233/JIFS-179949.
- [17] L. M. Francis and N. Sreenath, "TEDLESS – Text detection using least-square SVM from natural scene," *Journal of King Saud University - Computer and Information Sciences*, vol. 32, no. 3, pp. 287–299, 2020, doi: 10.1016/j.jksuci.2017.09.001.
- [18] E. A. Ismayilov, "Application of SVM and soft features to Azerbaijani text recognition," *ICTACT Journal on Image and Video Processing*, vol. 9, no. 2, pp. 1872–1875, 2018, doi: 10.21917/ijivp.2018.0265.
- [19] W. Lin, D. Ji, and Y. Lu, "Disorder recognition in clinical texts using multi-label structured SVM," *BMC Bioinformatics*, vol. 18, no. 1, 2017, doi: 10.1186/s12859-017-1476-4.
- [20] A. K. A. Hassan, B. S. Mahdi, and A. A. Mohammed, "Arabic handwriting word recognition based on scale invariant feature transform and support vector machine," *Iraqi Journal of Science*, vol. 60, no. 2, pp. 381–387, 2019, doi: 10.24996/ij.s.2019.60.2.18.
- [21] S. Sharma, A. Sasi, and A. N. Cheeran, "A SVM based character recognition system," in *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, 2017, pp. 1703–1707, doi: 10.1109/RTEICT.2017.8256890.
- [22] S. V. Wawre and S. N. Deshmukh, "Sentiment classification using machine learning techniques," *International Journal of Science and Research (IJSR)*, vol. 5, no. 4, pp. 819–821, 2016, doi: 10.21275/v5i4.NOV162724.
- [23] Z. Singla, S. Randhawa, and S. Jain, "Sentiment analysis of customer product reviews using machine learning," in *2017 International Conference on Intelligent Computing and Control (I2C2)*, 2017, pp. 1–5, doi: 10.1109/I2C2.2017.8321910.
- [24] A. M. Rahat, A. Kahir, and A. K. M. Masum, "Comparison of naive Bayes and SVM algorithm based on sentiment analysis using review dataset," in *2019 8th International Conference System Modeling and Advancement in Research Trends (SMART)*, 2019, pp. 266–270, doi: 10.1109/SMART46866.2019.9117512.
- [25] Y. Lei, "Network anomaly traffic detection algorithm based on SVM," in *2017 International Conference on Robots & Intelligent System (ICRIS)*, 2017, pp. 217–220, doi: 10.1109/ICRIS.2017.61.
- [26] A. Safi and S. Singh, "A systematic literature review on phishing website detection techniques," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 2, pp. 590–611, 2023, doi: 10.1016/j.jksuci.2023.01.004.
- [27] H. S. Emadi and S. M. Mazinani, "A novel anomaly detection algorithm using DBSCAN and SVM in wireless sensor networks," *Wireless Personal Communications*, vol. 98, no. 2, pp. 2025–2035, 2018, doi: 10.1007/s11277-017-4961-1.
- [28] H. Gharace and H. Hosseinvand, "A new feature selection IDS based on genetic algorithm and SVM," in *2016 8th International Symposium on Telecommunications (IST)*, 2016, pp. 139–144, doi: 10.1109/ISTEL.2016.7881798.
- [29] O. Yavanoglu and M. Aydos, "A review on cyber security datasets for machine learning algorithms," in *2017 IEEE International Conference on Big Data (Big Data)*, 2017, pp. 2186–2193, doi: 10.1109/BigData.2017.8258167.
- [30] U. Shruthi, V. Nagaveni, and B. K. Raghavendra, "A review on machine learning classification techniques for plant disease detection," in *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, 2019, pp. 281–284, doi: 10.1109/ICACCS.2019.8728415.
- [31] C. L. Chung, K. J. Huang, S. Y. Chen, M. H. Lai, Y. C. Chen, and Y. F. Kuo, "Detecting Bakanae disease in rice seedlings by machine vision," *Computers and Electronics in Agriculture*, vol. 121, pp. 404–411, 2016, doi: 10.1016/j.compag.2016.01.008.
- [32] M. A. Ebrahimi, M. H. Khoshtaghaza, S. Minaei, and B. Jamshidi, "Vision-based pest detection based on SVM classification method," *Computers and Electronics in Agriculture*, vol. 137, pp. 52–58, 2017, doi: 10.1016/j.compag.2017.03.016.
- [33] J. C. F. Silva, R. M. Teixeira, F. F. Silva, S. H. Brommonschenkel, and E. P. B. Fontes, "Machine learning approaches and their current application in plant molecular biology: a systematic review," *Plant Science*, vol. 284, pp. 37–47, 2019, doi: 10.1016/j.plantsci.2019.03.020.
- [34] N. N. Ba, P. P. T. Kim, H. T. Thanh, T. L. Anh, T. D. Van, and T. T. H. Phan, "Monitoring the absence of queen bee in the hive using deep learning and Hilbert Huang transform," *ASEAN Engineering Journal*, vol. 14, no. 1, pp. 113–120, 2024, doi: 10.11113/aej.V14.20163.
- [35] R. G. Mantovani, A. L. D. Rossi, J. Vanschoren, B. Bischl, and A. C. P. L. F. de Carvalho, "Effectiveness of random search in SVM hyper-parameter tuning," in *2015 International Joint Conference on Neural Networks (IJCNN)*, 2015, pp. 1–8, doi: 10.1109/IJCNN.2015.7280664.
- [36] R. Guido, M. C. Groccia, and D. Conforti, "A hyper-parameter tuning approach for cost-sensitive support vector machine classifiers," *Soft Computing*, vol. 27, no. 18, pp. 12863–12881, 2023, doi: 10.1007/s00500-022-06768-8.
- [37] Y. Wenzhuo and L. Shuo, "Optimizing parameters of support vector machines using an enhanced whale optimization algorithm," in *2023 2nd Asia Conference on Electrical, Power and Computer Engineering (EPCE)*, 2023, pp. 151–157, doi: 10.1109/EPCE58798.2023.00034.
- [38] S. A. Abdulraheem, S. Aliyu, and F. B. Abdullahi, "Hyper-parameter tuning for support vector machine using an improved cat swarm optimization algorithm," *Journal of the Nigerian Society of Physical Sciences*, vol. 5, no. 4, 2023, doi: 10.46481/jnsps.2023.1007.
- [39] L. K. Ramasamy, S. Kadry, and S. Lim, "Selection of optimal hyper-parameter values of support vector machine for sentiment analysis tasks using nature-inspired optimization methods," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 290–298, 2021, doi: 10.11591/eei.v10i1.2098.





- [40] S. Lessmann, R. Stahlbock, and S. F. Crone, "Optimizing hyperparameters of support vector machines by genetic algorithms," in *Proceedings of the 2005 International Conference on Artificial Intelligence, ICAI'05*, 2005, pp. 74–80.
- [41] A. Y. S. Lam and V. O. K. Li, "Chemical reaction optimization: a tutorial," *Memetic Computing*, vol. 4, no. 1, pp. 3–17, 2012, doi: 10.1007/s12293-012-0075-1.
- [42] C. N. Cong, R. R. -Jorge, N. N. Ba, C. T. Trong, and N. N. Anh, "Design of optimal PI controllers using the chemical reaction optimization algorithm for indirect power control of a DFIG model with MPPT," in *Web, Artificial Intelligence and Network Applications*, Cham, Switzerland: Springer, 2020, pp. 1250–1260, doi: 10.1007/978-3-030-44038-1_114.
- [43] N. N. Ba, C. N. Cong, D. V. Quoc, and N. N. Thi, "A comparison among random search algorithms for PID controller optimization," *International Review of Automatic Control*, vol. 15, no. 5, pp. 251–262, 2022, doi: 10.15866/ireaco.v15i5.22562.
- [44] J. J. Q. Yu, A. Y. S. Lam, and V. O. K. Li, "Evolutionary artificial neural network based on chemical reaction optimization," *2011 IEEE Congress of Evolutionary Computation, CEC 2011*, pp. 2083–2090, 2011, doi: 10.1109/CEC.2011.5949872.
- [45] N. N. Ba, N. N. Thi, D. V. Quoc, and C. N. Cong, "Recognition of plant leaf diseases based on deep learning and the chemical reaction optimization algorithm," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 38, no. 1, 2025, doi: 10.11591/ijeecs.v38.i1.pp447-458.
- [46] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, 2020, doi: 10.1016/j.neucom.2020.07.061.

BIOGRAPHIES OF AUTHORS







Nguyen Ba Nghien     is currently 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: nguyenbanganhien_cntt@hau.edu.vn.







Cuong Nguyen Cong     is currently a lecturer at the Electrical Engineering, Department of Renewable Energy Engineering Technology 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 a 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: nguyencongcuong@hau.edu.vn.



Nhung Nguyen Thi     is currently 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 of Technique in 2008. Her research interests focus on AI implemented in Python. She can be contacted at email: nhungnt@fit-hau.edu.vn.



Vuong Quoc Dung     is currently 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@fit-hau.edu.vn.