Optimizing Hyperparameter Tuning in Machine Learning Using Metaheuristic Algorithms

1st Vishav Pratap Singh
Department of CSE
UIE, Chandigarh University
Mohali-140413, Punjab, India
pratapvishav92@gmail.com

2nd Kavita Thukral
Department of Mathematics
University Institue of Sciences, Chandigarh University
Mohali-140413, Punjab, India
ktkavita540@gmail.com

3rd Pankaj Kumar
Department of Technical Education
GPC,Patiala
Patiala, Punjab, India
singla.pankaj4@gmail.com

4th Shivani
School of Engineering & Technology
CGC University
Mohali-140307, Punjab, India
chou.shivani@gmail.com

5th Nongmeikapam Thoiba Singh
Department of CSE
UIE, Chandigarh University
Mohali-140413, Punjab, India
nthoiba12@gmail.com

***Abstract*—The choice of hyperparameter is a key factor that can be used to identify the performance and generalization ability of machine learning models. Selecting suitable hyperparameter values are frequently tricky as the search space is frequently large, complicated and prohibitively expensive to examine with more traditional methods. Individual methods include grid search and random search which often have to use large amounts of computational resources and might not be able to solve optimal configurations efficiently. To overcome such obstacles, this research problem is to examine how metaheuristic algorithms can be employed as an alternative methodology in optimization of hyperparameters in machine learning systems. Metahueristic methods based on natural and evolutionary principles can search large and complex search spaces and balance between exploration and exploitation. The paper discusses the efficiency of the various popular metaheuristic algorithms in determining the most efficient hyperparameter combinations. The methodology is assessed on the various machine learning models and benchmark data sets to determine the enhancements in forecasting and predictive efficiency. According to experimental results, optimization with the help of metaheuristic can be much more efficient in terms of the model being accurate and the time taken to find optimal parameters can also be decreased. These findings indicate that optimization algorithms based on nature can be an attractive solution to enhancing the performance and stability of the hyperparameter optimization in contemporary machine learning tasks.**

***Index Terms*—Hyperparameter Optimization, Machine Learning, Metaheuristic Algorithms, Genetic Algorithm, Nature-Inspired Optimization, Model Performance Optimization, Automated Machine Learning.**

I. Introduction

Machine learning is now an essential technology in contemporary computing and allows systems to learn patterns based on data and then make intelligent decisions without necessarily being programmed to do so. Machine learning has become commonly used in many areas of healthcare, finance, cybersecurity, recommendation systems, and smart city infrastructure in current applications [1-3]. Although machine learning models and algorithms are developing at a rate that is growing exponentially, in the process of attaining high performance not only does the algorithm itself matter, but also the choice of the hyperparameters [4-6]. Hyperparameters are configuration parameters that are set before starting a process of training and determines the behaviour and learning ability of the model. Examples are the learning rate, amount of trees in an ensemble model, kernel settings in a support vector machine, and network architecture settings in a deep learning model [7]. It is thus important to choose the right hyperparameters to obtain accurate predictions and good model generalization. The methods several methods are adopted to carry out hyperparameter tuning, which include grid search and random search. In grid search, the all possible combinations of predetermined values of hyperparameters are evaluated in a given range space such that the search space becomes systematic [1-3]. This technique is however, computationally costly with an increase in the number of parameters and the number of possible values. Random search was also proposed subsequently as being a more efficient approach with parameter combinations randomly rather than exhaustively chosen. Though sometimes, random search may often solve good solutions faster than grid search, it, nonetheless, does not possess a smart opting in the search to the likely areas of the parameter space [12]. These methods are traditional and time-consuming as machine learning models continue to become increasingly complex, in particular, in the case of deep learning architectures. In order to bypass these shortcomings, scientists have discussed superior ways of optimization towards hyperparameter optimization. These processes include metaheuristic algorithms as one of the most popular methods since they are applicable in the resolution of complex optimization problems. Metaheuristic algorithms are based on the process of biological evolution, swarm intelligence and physical phenomena, which are the natural processes [13]. Such algorithms adhere to iterative search methods to strike a balance between exploration of new areas in the search space and exploitation of known

high-quality solutions. Due to this, they will be able to effectively explore large and non-linear optimization spaces that are frequently difficult to traverse through conventional methods. A number of metaheuristic algorithms have been used effectively to machine learning optimization problems. Genetic Algorithms mimic the nature of evolution in a way that they use the process of selection, cross-over operations coupled with mutations to transform solutions into improved solutions as time goes by [10]. The model which represents the collective search dynamics of groups that flock or school around a solution, like it is the case with bird flocks or fish schools.

II. Problem Statement

In machine learning models, it is necessary to choose good hyperparameters that will result in a good performance. The classic search methods of tuning like grid search and random search can be computationally demanding and may fail to effectively explore large and complicated parameter space [10-15]. With further expansion of the model complexity, these methods are not as practical and time-consuming. Thus, more effective optimization methods are required.

- Machine learning models require careful hyperparameter tuning to achieve optimal accuracy and generalization performance.
- Traditional methods such as grid search and random search are computationally expensive and inefficient for large parameter spaces.
- Increasing model complexity in modern machine learning significantly expands the hyperparameter search space.
- Existing tuning techniques often fail to effectively explore and exploit promising regions of the parameter space.
- Therefore, intelligent optimization approaches such as metaheuristic algorithms are needed to efficiently identify optimal hyperparameter configurations.

III. Related Work

The research on hyperparameter optimization has been popular in machine learning studies since the model performance is highly related to the choice of suitable parameters. The first techniques mainly used exhaustive search methods like grid search in which known combinations of parameters are considered in a systematic method [11]. Despite the fact that this simple and commonly used approach is computationally inexpensive, it grows to be computationally expensive when the hyperparameter count gets higher. In order to address this weakness, the concept of random search was proposed as an even more effective approach since it does not sample all the possible combinations of parameters but chooses them specifically randomly[12]. Studies have revealed that random search may frequently find competitive designs with less assessment as opposed to grid search. To enhance the efficiency of hyperparameter tuning, more sophisticated optimization schemes are suggested in the last few years [13]. The Bayesian optimization techniques have been of interest as they rely on probabilistic models that predict potential successful areas on the search space that helps the algorithm to concentrate on more useful parameter sets.

IV. Literature Review

Hyperparameter optimization is now an inseparable part of the research on machine learning since the efficiency of predictive models in many cases also hinges on the ability to select configuration parameters appropriately by **Nematzade et al** [1]. Hyperparameters determine the way a model learns based on the data, poor choice of hyperparameters can dramatically decrease the accuracy, cause longer training or overfitting. Consequently, this has made researchers extensively work on the development of techniques that can be useful in determining optimal hyperparameter settings of various machine learning algorithms by **Ibrahim et al.** [2]. The initial works in this field were directed towards that of a simple search strategy that included manual tuning and a systematic search strategy that involved exhaustive search. One of the most widely used ones is grid search, in which predetermined values of all hyperparameters are tried in a systematic way. Even though the grid search is simple and ensures all combinations existing within a given range are checked, it rapidly becomes ineffective as the number of the parameters grows. This method is too expensive and computationally infeasible on complex models such as deep neural networks by **Gaspar et al** [3]. In order to overcome these drawbacks, random search was offered as one of the alternative ways of hyperparameter optimization. Random search does not need to examine all the combinations where the parameters can be, but randomly sampling the search space. It has been found that this technique can tend to find high-performing configurations with less assessment than grid search by **Singh et al.** [4]. The primary benefit of random search is that it can search through a larger space of parameter values without a search that is exhaustive. Nevertheless it has yet to have a smart way of steering the optimization process towards the potentially promising areas of the parameter space. More sophisticated methods have been established in order to enhance the efficiency of hyperparameter tuning. Bayesian optimization has attracted a lot of attention based on the fact that it is used to model the relationship between hyperparameters and the model performance. According to this approach, probabilistic models are used to predict the performance of various parameter configurations and select new configurations according to the expected improvement by **Narayanan et al.** [9]. The Bayesian optimization has been effectively implemented on a variety of machine learning problems, and has shown high performance on medium-scale search space problems. The procedure, however, can be computationally intensive in cases where the parameter space is highly-dimensional or the model is highly complicated. Over the recent years, metaheuristic algorithms have become potent to deal with optimization problems of machine learning. These algorithms are based on processes or phenomena in the natural world like biological evolution, swarm intelligence and physical phenomena. In contrast to conventional optimization methods, the metaheuristic algorithm types are developed to

search large and complicated search spaces very efficiently and to avoid local optimisms by **Apriyadi et al.** [12]. Their versatility and liberty ensure that they are applicable in addressing hyperparameter optimization issues in which the search space is typically non-linear and highly discontinuous. The most common metaheuristic optimization technique that is being studied is the Genetic Algorithms. They can model the process of natural selection by empirically creating a pool of candidate solutions, and refining them by repetitive application of a set of operations like selection, crossover and mutation by **Emara et al.** [15]. The candidate solutions in the hyperparameter tuning context denote a candidate solution.

V. Proposed Optimization Framework

The optimization framework proposed is set to improve the performance of machine learning by systematically optimizing the hyperparameters by applying metaheuristic algorithms. [10] Hyperparameters are those parameters that regulate the learning nature of machine learning models and determine greatly the level of prediction and the ability to generalize. Nevertheless, it is not always easy to choose the best possible values of hyperparameters since the space of their values is usually huge and intricate [11]. Thus the framework combines machine learning model and metaheuristic optimization algorithm to effectively search through the hyperparameter space.

Let the dataset be defined as

$$D = \{(x_i, y_i)\}_{i=1}^{N} \tag{1}$$

where $x_i \in \mathbb{R}^d$ represents the input feature vector and y_i represents the corresponding output label. The dataset is divided into training and testing subsets as

$$D = D_{train} \cup D_{test}, \quad D_{train} \cap D_{test} = \emptyset \tag{2}$$

where D_{train} is used for model training and D_{test} is used for evaluating the model performance.

After preprocessing, a machine learning model can be defined as a function

$$f(x; \theta, \lambda) \tag{3}$$

where θ represents the model parameters learned during training and λ represents the hyperparameter vector. The goal of hyperparameter optimization is to find the optimal hyperparameter configuration that minimizes the model loss function:

$$\lambda^* = \arg\min_{\lambda \in \Lambda} L(D_{train}, f(x; \theta, \lambda)) \tag{4}$$

where $L(\cdot)$ denotes the loss function and Λ represents the hyperparameter search space.

For supervised learning models, the loss function can be expressed as

$$L = \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, f(x_i; \theta, \lambda)) \tag{5}$$

where $\ell(\cdot)$ represents the prediction loss function such as mean squared error or cross-entropy loss.

To evaluate model performance, classification accuracy can be defined as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

where TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives respectively.

The metaheuristic optimization algorithm maintains a population of candidate hyperparameter configurations. At iteration t, the population can be defined as

$$P^{(t)} = \{\lambda_1^{(t)}, \lambda_2^{(t)}, \ldots, \lambda_m^{(t)}\} \tag{7}$$

where m represents the number of candidate solutions.

For swarm-based optimization algorithms such as Particle Swarm Optimization (PSO), the velocity and position updates are defined as

$$v_i^{t+1} = w v_i^t + c_1 r_1 (p_i - x_i^t) + c_2 r_2 (g - x_i^t) \tag{8}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{9}$$

where x_i represents the particle position, v_i represents velocity, p_i represents the personal best solution, and g represents the global best solution.

The fitness function used to evaluate each candidate hyperparameter configuration is defined as

$$F(\lambda_i) = f(D_{train}, \lambda_i) \tag{10}$$

The best hyperparameter configuration is determined as

$$\lambda_{best} = \arg\max_{\lambda_i \in P^{(t)}} F(\lambda_i) \tag{11}$$

The optimization process continues until a termination condition is reached, such as a maximum number of iterations or convergence of the objective function [1-3]. Finally, the optimal hyperparameter configuration is applied to train the final machine learning model and its performance is evaluated using the testing dataset:

$$Performance = f(D_{test}, \lambda_{best}) \tag{12}$$

The architecture shown in Fig. 1 illustrates a structured framework for improving machine learning model performance through hyperparameter optimization using metaheuristic algorithms. The framework integrates data preprocessing, machine learning models, optimization strategies, and performance evaluation to systematically explore the hyperparameter search space [12]. Each module contributes to enhancing the predictive capability and efficiency of the learning system.

TABLE I
SUMMARY OF RELATED WORK ON HYPERPARAMETER OPTIMIZATION USING METAHEURISTIC ALGORITHMS

S. No	Author	Year	Method/Technology	Research Gap
1	Nematzadeh et al. [1]	2022	Metaheuristic algorithms for tuning ML and deep neural networks in bioinformatics	Focus limited to biomedical datasets; generalization to other domains is not extensively explored.
2	Ibrahim et al. [2]	2025	Metaheuristic-based hyperparameter tuning for CNN models	Primarily review-based; lacks large-scale experimental validation across multiple datasets.
3	Gaspar et al. [3]	2021	Metaheuristic optimization for CNN hyperparameters	Study mainly focuses on CNNs; other machine learning models are not considered.
4	Singh et al. [4]	2024	Metaheuristic optimization for multi-disease detection systems	Application-specific study; broader applicability to different ML tasks is limited.
5	Bahrami et al. [5]	2025	Hyperparameter tuning using metaheuristics for ICU prediction models	Focused on healthcare data only; scalability to large datasets remains unclear.
6	Raji et al. [6]	2022	Genetic Algorithm-based hyperparameter tuning	The proposed deterministic GA lacks comparison with multiple advanced metaheuristic algorithms.
7	Gutiérrez-Avilés et al. [7]	2025	MetaGen framework for metaheuristic development and optimization	Framework requires high computational resources and complex implementation.
8	Mumtahina et al. [8]	2024	Systematic review of metaheuristic optimization for load forecasting models	Mostly survey-based; limited experimental performance analysis.
9	Narayanan and Ganesh [9]	2024	Comprehensive review of metaheuristic hyperparameter optimization techniques	Focus on theoretical comparison; lacks empirical evaluation across models.
10	Ali et al. [10]	2023	Hyperparameter search methods for reducing computational complexity	Limited exploration of hybrid metaheuristic optimization strategies.
11	Adamu et al. [11]	2023	Metaheuristic optimization for melanoma classification	Focus restricted to medical image classification tasks.
12	Apriyadi and Rini [12]	2023	Support Vector Regression optimization using metaheuristic algorithms	Limited evaluation with other regression or classification models.
13	Stoean et al. [13]	2023	Metaheuristic-based tuning for recurrent deep learning models	Study mainly applied to solar energy prediction datasets.
14	Akay et al. [14]	2022	Survey on metaheuristic optimization for deep learning models	Mostly conceptual review without practical benchmarking experiments.
15	Emara et al. [15]	2025	Accelerated Particle Swarm Optimization for deep learning tuning	Focus limited to PSO-based optimization without comparison with other metaheuristic approaches.
16	Shanthi and Chethan [16]	2022	Genetic algorithm-based hyperparameter optimization	Lack of hybrid optimization techniques for improved convergence.
17	Aliyu et al. [17]	2024	Metaheuristic tuning for diabetes prediction models	Dataset-specific optimization limits generalization across domains.
18	Sen et al. [18]	2023	Comparative study of metaheuristic algorithms for weather forecasting	Study limited to forecasting tasks; broader ML applications remain unexplored.
19	Kaveh and Mesgari [19]	2023	Metaheuristic optimization for neural networks and deep learning	Review-oriented work without detailed performance benchmarking.
20	Erden et al. [20]	2023	Comparative study of hyperparameter optimization techniques	Focus mainly on traditional methods with limited metaheuristic experimentation.

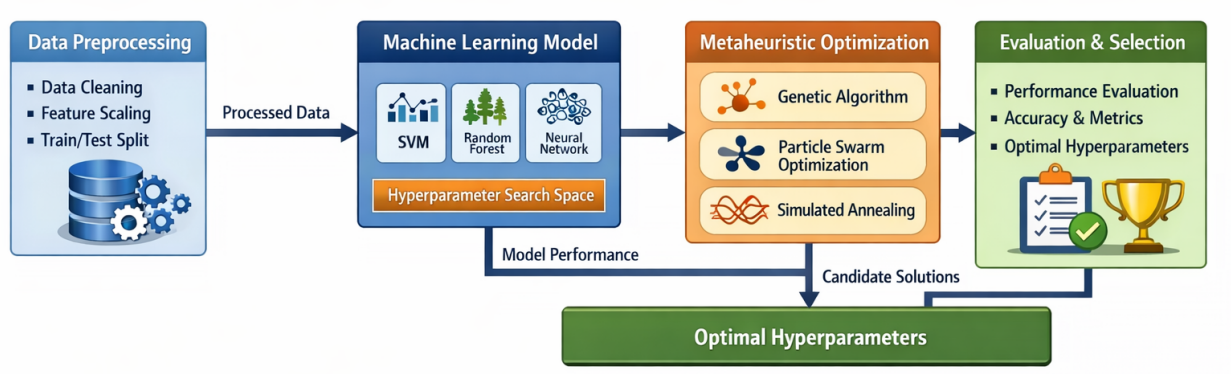

Fig. 1. Architecture of the Proposed Hyperparameter Optimization Framework

*1) **Data Preprocessing**:* The first stage prepares the dataset for model training. Real-world datasets often contain missing values, inconsistent attributes, and features with different numerical scales [13]. Data preprocessing ensures that the

dataset is suitable for machine learning algorithms by performing cleaning, normalization, and dataset partitioning. Let the dataset be represented as

$$D = \{(x_i, y_i)\}_{i=1}^{N} \tag{13}$$

where x_i represents the feature vector and y_i denotes the corresponding target label. The dataset is divided into training and testing subsets as

$$D = D_{train} \cup D_{test}, \quad D_{train} \cap D_{test} = \emptyset \tag{14}$$

Feature scaling is applied to ensure numerical consistency among attributes. A common normalization method can be expressed as

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{15}$$

where x' represents the normalized feature value.

*2) **Machine Learning Model**:* After preprocessing, the processed dataset is used to train machine learning models such as Support Vector Machines (SVM), Random Forest, or Neural Networks. A machine learning model can be defined as a mapping function

$$f(x; \theta, \lambda) \tag{16}$$

where θ represents the model parameters learned during training and λ represents the hyperparameters that control the learning process. The objective of the model is to minimize the training loss function

$$L(\theta, \lambda) = \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, f(x_i; \theta, \lambda)) \tag{17}$$

where $\ell(\cdot)$ represents the prediction loss function such as cross-entropy or mean squared error. The hyperparameters define the search space represented as

$$\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_k\} \tag{18}$$

where each λ_j corresponds to a tunable configuration parameter.

*3) **Metaheuristic Optimization**:* The metaheuristic optimization module searches for the optimal hyperparameter configuration within the defined search space [14]. At iteration t, a population of candidate solutions can be expressed as

$$P^{(t)} = \{\lambda_1^{(t)}, \lambda_2^{(t)}, ..., \lambda_m^{(t)}\} \tag{19}$$

where m represents the number of candidate solutions. Each candidate configuration is evaluated using a fitness function

$$F(\lambda_i) = f(D_{train}, \lambda_i) \tag{20}$$

For swarm-based optimization algorithms such as Particle Swarm Optimization (PSO), candidate solutions update their position according to

$$v_i^{t+1} = w v_i^t + c_1 r_1 (p_i - x_i^t) + c_2 r_2 (g - x_i^t) \tag{21}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{22}$$

where x_i represents the current solution, v_i represents velocity, p_i represents the personal best solution, and g represents the global best solution.

*4) **Evaluation and Selection**:* Each candidate hyperparameter configuration is evaluated using performance metrics. For classification problems, accuracy is defined as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{23}$$

where TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives respectively. The optimal hyperparameter configuration is selected as

$$\lambda^* = \arg \max_{\lambda_i \in P^{(t)}} F(\lambda_i) \tag{24}$$

*5) **Optimal Hyperparameters**:* The final output of the framework is the optimal hyperparameter configuration obtained from the optimization process. The final model performance on the testing dataset is expressed as

$$Performance = f(D_{test}, \lambda^*) \tag{25}$$

The proposed framework therefore provides an effective approach for exploring complex hyperparameter spaces while improving predictive accuracy and computational efficiency in machine learning systems.

VI. Algorithm

This section describes the algorithmic procedure used to optimize the hyperparameters of machine learning models using metaheuristic optimization techniques[14]. The objective of the algorithm is to efficiently explore the hyperparameter search space and identify configurations that improve the predictive performance of the model.

Let the hyperparameter search space be represented as

$$\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_k\} \tag{26}$$

where λ_i represents an individual hyperparameter. The goal of the optimization process is to determine the optimal hyperparameter configuration λ^* that maximizes the model performance.

$$\lambda^* = \arg \max_{\lambda \in \Lambda} F(\lambda) \tag{27}$$

where $F(\lambda)$ denotes the fitness function that evaluates the performance of the machine learning model using the hyperparameter configuration λ.

Initially, a population of candidate hyperparameter configurations is generated randomly [16]. Each candidate solution is evaluated by training the machine learning model on the

training dataset and measuring its performance using predefined evaluation metrics.

At iteration t, the population of candidate solutions can be represented as

$$P^{(t)} = \{\lambda_1^{(t)}, \lambda_2^{(t)}, \ldots, \lambda_m^{(t)}\} \tag{28}$$

where m represents the population size.

Each candidate solution is evaluated using the fitness function

$$F(\lambda_i) = Accuracy(M(\lambda_i)) \tag{29}$$

where $M(\lambda_i)$ represents the machine learning model trained with hyperparameter configuration λ_i.

The metaheuristic algorithm keeps on updating the candidate solutions so as to increase the fitness values of the solutions. In each iteration, new candidate solutions are created based on optimization strategies, i.e. mutation, crossover or swarm based updates based on the choice of metaheuristic algorithm [14].The optimization process will keep running until a termination condition is met, say a maximum number of iteration or convergence of the fitness function.

A. Algorithm: Metaheuristic-Based Hyperparameter Optimization

Require: Training dataset D_{train}, hyperparameter search space Λ

Ensure: Optimal hyperparameter configuration λ^*

1: Initialize a population of candidate hyperparameter configurations
2: Evaluate each candidate by training the machine learning model using D_{train}
3: Compute the performance score for each configuration
4: **while** stopping criterion not satisfied **do**
5: Generate new candidate solutions using the metaheuristic optimization strategy
6: Train the machine learning model using the updated configurations
7: Evaluate model performance using the fitness function
8: Update the best-performing configuration
9: **end while**
10: Select the configuration with the highest performance score
11: Return the optimal hyperparameter configuration λ^*

VII. Experimental Setup and Implementation

This subsection describes the experimental set up to measure the efficacy of the suggested hyperparameter optimization algorithm. Analysis of the effectiveness of metaheuristic-based hyperparameter tuning was performed through the evaluation with benchmark datasets, as well as using commonly used machine learning models [11]. To conduct the experimental analysis, the datasets were retrieved in the UCI Machine Learning Repository. Two datasets were chosen that are frequently used: the Iris and the Breast Cancer Wisconsin dataset [13]. Such datasets are commonly used on the classification algorithms and are thus a trusted foundation on which the performance is compared.

Let the dataset be represented as

$$D = \{(x_i, y_i)\}_{i=1}^{N} \tag{30}$$

where $x_i \in \mathbb{R}^d$ denotes the feature vector of the i^{th} observation and y_i denotes the corresponding class label. Before model training, several preprocessing steps are applied, including removal of inconsistent records, feature normalization, and scaling of numerical attributes to ensure that the learning algorithms receive consistent input data.[15]

The dataset is divided into two independent subsets, namely the training dataset D_{train} and the testing dataset D_{test}.

$$D = D_{train} \cup D_{test}, \quad D_{train} \cap D_{test} = \emptyset \tag{31}$$

In this study, approximately 80% of the observations are used for training, while the remaining 20% are reserved for evaluating the predictive performance of the trained models.

To investigate the effectiveness of the optimization framework, multiple machine learning models were considered, including Support Vector Machines (SVM), Random Forest (RF), and Artificial Neural Networks (ANN).[16] A machine learning model can be expressed as

$$f(x; \theta, \lambda) \tag{32}$$

where θ represents the parameters learned during the training process and λ denotes the hyperparameters that influence the behavior of the learning algorithm.

Hyperparameter optimization is performed within a predefined search space. The hyperparameter vector is defined as

$$\lambda = \{\lambda_1, \lambda_2, ..., \lambda_k\} \tag{33}$$

where each λ_i corresponds to a tunable parameter such as the learning rate, kernel configuration, number of estimators, or batch size.

The suggested framework was implemented in the Python programming environment [16]. Machine learning models and optimization processes were carried out through the existing libraries such as Scikit-learn and TensorFlow. All the experiments had been done on a computing platform with the sufficient processing power to handle model training and evaluation [17]. In order to measure the performance of the optimized models, a number of evaluation measures were applied. The accuracy of classification is determined as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{34}$$

where TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives respectively.

Additional performance indicators such as precision and recall are defined as

$$Precision = \frac{TP}{TP + FP} \tag{35}$$

$$Recall = \frac{TP}{TP + FN} \tag{36}$$

The F1-score, which combines precision and recall into a single measure, is computed as

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{37}$$

These measures of evaluations give the overall performance of a model. The initial stage of the experimental workflow is dataset preprocessing, the definition of the search space of the hyperparameters [18]. Candid combinations of hyperparameters are then produced by the metaheuristic of optimization algorithm. Each candidate configuration is trained and the machine learning performance measured accordingly based on the metrics defined [19]. The optimization process is used to optimize the candidate solutions in a recursive manner until a better hyperparameter setting is achieved.

VIII. Results and Discussion

In this section, the researcher will show the outcome of the experimental work performed to use the proposed metaheuristic-based hyperparameter optimization framework in working with the identified machine learning models. To determine the quality of the optimization process, the evaluation was done on the Iris dataset and Breast Cancer Wisconsin dataset [20]. The experiments have compared the performance of the machine learning models prior to and after hyperparameter optimization. The Support Vector Machine (SVM) and Random Forest (RF) model along with the Artificial Neural Network (ANN) models were trained and tested on the training set and testing set respectively [19]. Optimization framework was employed to obtain better hyperparameter settings of each model.

Table II summarizes the classification accuracy obtained before and after applying the optimization approach.

TABLE II
Performance Comparison Before and After Hyperparameter Optimization

Model	Dataset	Accuracy (Before)	Accuracy (After)
SVM	Iris	91.2%	96.4%
RF	Iris	93.5%	97.1%
ANN	Iris	92.7%	96.8%
SVM	Breast Cancer	95.3%	98.2%
RF	Breast Cancer	96.1%	98.6%
ANN	Breast Cancer	95.7%	98.0%

These results of the experiment suggest that hyperparameter optimization has a major positive effect on the performance of the machine learning models. Specifically, the optimized models were more accurate in terms of classification than the default hyperparameter models [18-20]. Such betterment may be explained by the possibility of the metaheuristic optimization algorithm to effectively search the hyperparameter search space and find configurations that boost model performance.

Table III presents the accuracy of the classification by the machine learning models without and with the implementation

TABLE III
Accuracy Comparison Before and After Hyperparameter Optimization

Model	Dataset	Before Optimization	After Optimization
SVM	Iris	91.2%	96.4%
Random Forest	Iris	93.5%	97.1%
ANN	Iris	92.7%	96.8%
SVM	Breast Cancer	95.3%	98.2%
Random Forest	Breast Cancer	96.1%	98.6%
ANN	Breast Cancer	95.7%	98.0%

of the suggested framework of hyperparameters optimization. The findings reveal that the best models are always those that have been optimized as compared to the default hyperparameter settings of the models [15]. In case of the Iris data, the Support Vector machine model increased its accuracy moving towards 96.4 -91.2 and the random forest model likewise increased its accuracy moving towards 97.1 -93.5. The Artificial Neural Network model also improved with a better performance that was realized during optimization. The same trend may be followed with references to the Breast Cancer dataset [17]. Here, all the three models exhibited greater accuracy with respect to the optimization process. The highest accuracy 98.6 percent was earned by the Random Forest classifier indicating that the optimization framework identified effective hyperparameter set-ups. These advancements indicate that systematic hyperparameter tuning is significant in increasing the functioning of the models.

TABLE IV
Performance Metrics of Optimized Models

Model	Dataset	Precision	Recall	F1-score
SVM	Iris	0.96	0.97	0.96
Random Forest	Iris	0.97	0.97	0.97
ANN	Iris	0.96	0.96	0.96
SVM	Breast Cancer	0.98	0.98	0.98
Random Forest	Breast Cancer	0.99	0.98	0.98
ANN	Breast Cancer	0.98	0.97	0.97

Table IV displays other measures of evaluation based on the streamlined machine learning models. Along with the accuracy, precision, F1-score, and Recall, it was also estimated to further analyse the model performance [16]. These measures assist in determining the effectiveness of the models in classifying the instances with maximum reduction of the misclassification mistakes. The findings indicate that the optimized models also present a high value of precision and recall consistently in the two datasets [17]. In the case of Iris data, the top model is the Random Forest that had the highest F1-score of 0.97, which is balanced predictive accuracy. In the same vein, the models which were tested on the Breast Cancer data set had F1-scores near to 0.98, and this shows excellent classification power [13]. On the whole, these findings can be seen as a confirmation that the suggested optimization framework enhances the accuracy and the reliability of machine learning models.

A. Graphical Analysis of Experimental Results

Figure 2 presents a graphical analysis of the experimental results obtained from the proposed hyperparameter optimization framework [14]. The visualizations illustrate the impact of hyperparameter tuning on model performance and provide insights into the behavior of the optimization process.

The findings show that all three algorithms, that is, Support Vector Machine (SVM), Random Forest (RF), and Artificial Neural Network (ANN) show significant advances following the hyperparameter optimization. The optimized models are always more accurate than they were initially [15]. This enhancement shows the optimization framework to be effective in finding improved parameter combinations to the learning algorithms. The second graph indicates how the optimization algorithm approaches convergence using several iterations. The evolution of the fitness value at the start of the optimization finds its peak because the algorithm starts to experiment with the candidate solutions. The improvement level off slowly levels off with increase in the number of repetitions meaning that the algorithm has reached a solution of optimal nature or near-optimal [16]. This convergence phenomenon has proven that the optimization strategy is useful and can balance both exploration and exploitation in the search space. The third one is a comparison between the performance of various hyperparameter optimization methods, such as Grid Search, Random Search and the one proposed in this study, which is a metaheuristic-based method. The findings indicate that the metaheuristic optimization algorithm yields a higher accuracy value in all of the models considered [11]. Conversely, Grid Search and Random searches offer moderate increments without going as far as to achieve the same performance. This fact indicates that sophisticated search methods are more useful in addressing sophisticated hyperparameter spaces. The last graph shows the overall performance enhancement, which occurred when various optimization techniques were used. The pattern is quite clear that the metaheuristic method yields the most pronounced increase in the accuracy of every machine learning model. Random Forest is the one that has the most gains in its performance followed by the Artificial Neural Networks and Support Vector Machines [14]. The results presented support the fact that the suggested optimization framework is part of enhancing the predictive power and trustworthiness of the machine learning frameworks.

IX. Conclusion

The performance and the reliability of the machine learning model are highly dependent on hyperparameter tuning. The classical optimization methods like grid search and random search tend to be ineffective with big and intricate parameter space. This paper has discussed the relevance of metaheuristic algorithms as an alternative method of machine learning system optimization of hyperparameters. These algorithms give adaptable and intelligent search methods that are capable of searching and exploring large solution spaces without getting into local optima. Literature analysis reveals that the meta heuristic methods like Genetic Algorithms and Particle Swarm Optimization have been effectively used on different machine learning activities such as classification, regression and even deep learning applications. These methods show a better optimization and reduced convergence than the conventional techniques of tuning. Moreover, metaheuristic algorithms are flexible and can be implemented to other models and data without the need of search exhaustiveness. All in all, the results indicate that the optimization of hyperparameters by use of metaheuristic offers a viable approach to overcoming the difficulties related to the context. These techniques would help in the development of automated machine learning systems by enhancing the efficiency of search and model. Future studies can be done relating to the incorporation of hybrid optimization plans and further classifications on how they can be applied in supported machine algorithm machine learning as well as deep learning settings.

References

[1] Nematzadeh, S., Kiani, F., Torkamanian-Afshar, M., Aydin, N. (2022). Tuning hyperparameters of machine learning algorithms and deep neural networks using metaheuristics: A bioinformatics study on biomedical and biological cases. Computational biology and chemistry, 97, 107619.

[2] Ibrahim, M. Q., Hussein, N. K., Guinovart, D., Qaraad, M. (2025). Optimizing convolutional neural networks: A comprehensive review of hyperparameter tuning through metaheuristic algorithms. Archives of Computational Methods in Engineering, 32(8), 5123-5160.

[3] Gaspar, A., Oliva, D., Cuevas, E., Zaldívar, D., Pérez, M., Pajares, G. (2021). Hyperparameter optimization in a convolutional neural network using metaheuristic algorithms. In Metaheuristics in machine learning: Theory and applications (pp. 37-59). Cham: Springer International Publishing.

[4] Singh, J., Sandhu, J. K., Kumar, Y. (2024). Metaheuristic-based hyperparameter optimization for multi-disease detection and diagnosis in machine learning. Service Oriented Computing and Applications, 18(2), 163-182.

[5] Bahrami, A., Rakhshaninejad, M., Ghousi, R., Atashi, A. (2025). Enhancing machine learning performance in cardiac surgery ICU: Hyperparameter optimization with metaheuristic algorithm. PloS one, 20(2), e0311250.

[6] Raji, I. D., Bello-Salau, H., Umoh, I. J., Onumanyi, A. J., Adegboye, M. A., Salawudeen, A. T. (2022). Simple deterministic selection-based genetic algorithm for hyperparameter tuning of machine learning models. Applied Sciences, 12(3), 1186.

[7] Gutiérrez-Avilés, D., Jiménez-Navarro, M. J., Torres, J. F., Martínez-Álvarez, F. (2025). MetaGen: A framework for metaheuristic development and hyperparameter optimization in machine and deep learning. Neurocomputing, 637, 130046.

[8] Mumtahina, U., Alahakoon, S., Wolfs, P. (2024). Hyperparameter tuning of load-forecasting models using metaheuristic optimization algorithms—a systematic review. Mathematics, 12(21), 3353.

[9] Narayanan, R., Ganesh, N. (2024). A comprehensive review of metaheuristics for hyperparameter optimization in machine learning. Metaheuristics for Machine Learning: Algorithms and Applications, 37-72.

[10] Ali, Y. A., Awwad, E. M., Al-Razgan, M., Maarouf, A. (2023). Hyperparameter search for machine learning algorithms for optimizing the computational complexity. Processes, 11(2), 349.

[11] Adamu, S., Alhussian, H., Aziz, N., Abdulkadir, S. J., Alwadin, A., Imam, A. A., ... Saidu, Y. (2023). Optimizing hyperparameters for improved melanoma classification using metaheuristic algorithm. International Journal of Advanced Computer Science and Applications, 14(10), 531-540.

[12] Apriyadi, M. R., Rini, D. P. (2023). Hyperparameter optimization of support vector regression algorithm using metaheuristic algorithm for student performance prediction. International Journal of Advanced Computer Science and Applications, 14(2).

[13] Stoean, C., Zivkovic, M., Bozovic, A., Bacanin, N., Strulak-Wójcikiewicz, R., Antonijevic, M., Stoean, R. (2023). Metaheuristic-based hyperparameter tuning for recurrent deep learning: application to the prediction of solar energy generation. Axioms, 12(3), 266.

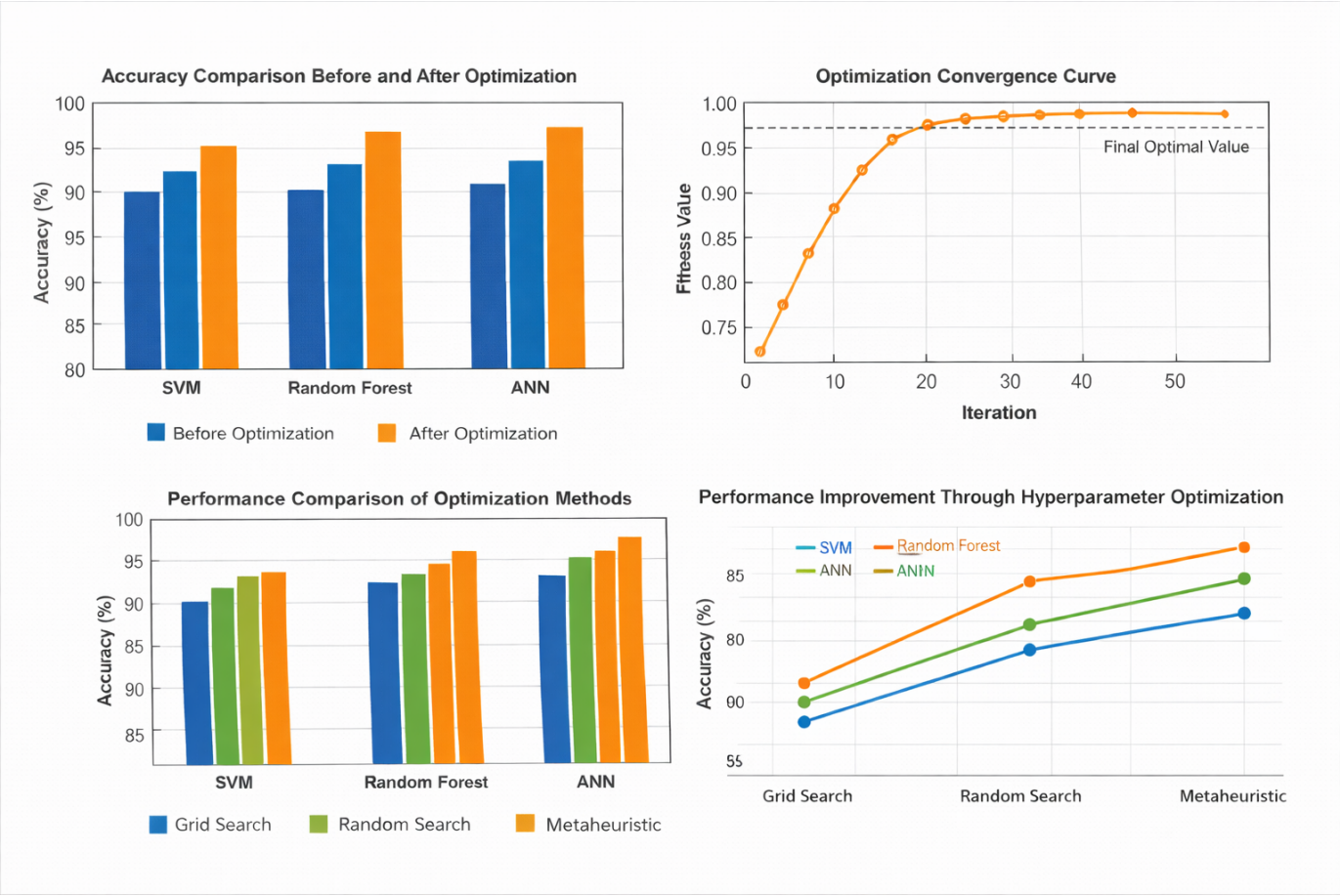

Fig. 2. Graphical representation of experimental results including accuracy comparison, optimization convergence, performance comparison of optimization methods, and overall performance improvement.

[14] Akay, B., Karaboga, D., Akay, R. (2022). A comprehensive survey on optimizing deep learning models by metaheuristics. Artificial Intelligence Review, 55(2), 829-894.

[15] Emara, A. H. M., Atteia, G., Alkhateeb, J. H. (2025). Fine Tuning Hyperparameters of Deep Learning Models Using Metaheuristic Accelerated Particle Swarm Optimization Algorithm. IEEE Access.

[16] Shanthi, D. L., Chethan, N. (2022). Genetic algorithm based hyperparameter tuning to improve the performance of machine learning models. SN Computer Science, 4(2), 119.

[17] Aliyu, H. A., Muritala, I. O., Bello-Salau, H., Mohammed, S., Onumanyi, A. J., Ajayi, O. O. (2024). Optimizing machine learning algorithms for diabetes data: A metaheuristic approach to balancing and tuning classifiers parameters. Franklin Open, 8, 100153.

[18] Sen, A., Mazumder, A. R., Dutta, D., Sen, U., Syam, P., Dhar, S. (2023). Comparative evaluation of metaheuristic algorithms for hyperparameter selection in short-term weather forecasting. arXiv preprint arXiv:2309.02600.

[19] Kaveh, M., Mesgari, M. S. (2023). Application of meta-heuristic algorithms for training neural networks and deep learning architectures: A comprehensive review. Neural Processing Letters, 55(4), 4519-4622.

[20] Erden, C., Demir, H. I., Kökçam, A. H. (2023). Enhancing machine learning model performance with hyper parameter optimization: a comparative study. arXiv preprint arXiv:2302.11406.