# Artificial Neural Network Hyperparameters Optimization: A Survey

https://doi.org/10.3991/ijoe.v18i15.34399

Zahraa Saddi Kadhim(✉), Hasanen S. Abdullah, Khalil Ibrahim Ghathwan
Department of Computer Science, University of Technology, Baghdad, Iraq
cs.20.44@grad.uotechnology.edu.iq

**Abstract—**Machine-learning (ML) methods often utilized in applications like computer vision, recommendation systems, natural language processing (NLP), as well as user behavior analytics. Neural Networks (NNs) are one of the most essential ways to ML; the most challenging element of designing a NN is determining which hyperparameters to employ to generate the optimal model, in which hyperparameter optimization improves NN performance. This study includes a brief explanation regarding a few types of NN as well as some methods for hyperparameter optimization, as well as previous work results in enhancing ANN performance using optimization methods that aid researchers and data analysts in developing better ML models via identifying the appropriate hyperparameter configurations.

**Keywords—**hyperparameter optimization, artificial neural network, deep learning, machine learning

## 1 Introduction

NNs are algorithms utilized in ML [1]. The ability of a researcher to efficiently create and train an Artificial Neural Network (ANN) based on their skill set; is most likely a combination of domain knowledge from previous studies and experience gained through continuously trying and failing to construct ANNs. ANNs with multi-hiding layers are utilized in NNs, which are a subset of deep learning (DL). Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) are instances of various NN implementations with minor structural differences [2], [3]. The DNN's hyperparameters optimization approaches could be used for defining the models' structure, which is a complicated and time-consuming procedure that includes picking the best algorithm and creating the best model architecture, despite the output targets and input characteristics [4].

Tuning hyperparameters is a critical part of developing a successful ML model, particularly for deep neural networks (DNN) and tree-based ML models [5].

The most extensively utilized hyperparameter selection techniques for ML algorithms are Random Search (RS) and Grid Search (GS) [6], [7], Furthermore, in optimization problems, excellent optimization procedures are typically necessary for minimizing or maximizing objective functions. Algorithms for optimizing weights, learning rules,

network design, activation function, neurons, and bias commonly employed. Another method of improving and optimizing the ANN is to use an optimizer for replacing the NN's basic algorithms with optimization algorithms, and replacing backpropagation with any optimization approaches to overcome particular concerns [3], For example, practical swarm optimization (PSO), Bayesian optimization algorithm (BOA), and genetic algorithm (GA) are a few optimization hyperparameters techniques. We briefly explore the different types of NNs in this study, as well as the most frequent optimization approaches. Furthermore, we address the essential hyperparameters of standard ML models that must be modified, as well as a few past works on hyperparameter optimization methods that can be used for improving the performance of specific applications as well as solving certain challenges. Finally, the drawbacks and benefits of different hyperparameter selection approaches indicated, as well as the types of hyperparameters utilized to address each one of the problems with the dataset employed. The rest of the study structured in the following way: the second section introduces a new taxonomy for NN creation and optimization activities. The third section begins with a review of earlier work on a few of the most relevant ANN hyperparameter optimization and applications. The fourth section provides a discussion and analysis of the work. Lastly, section five provides a conclusion of this work.

## 2 Materials and methods

In this section, we give a new taxonomy in Figure 1, depending on which several NN structures are presented, selected, and the superiority of employing different optimization methods to search for appropriate ANN parameters is demonstrated.

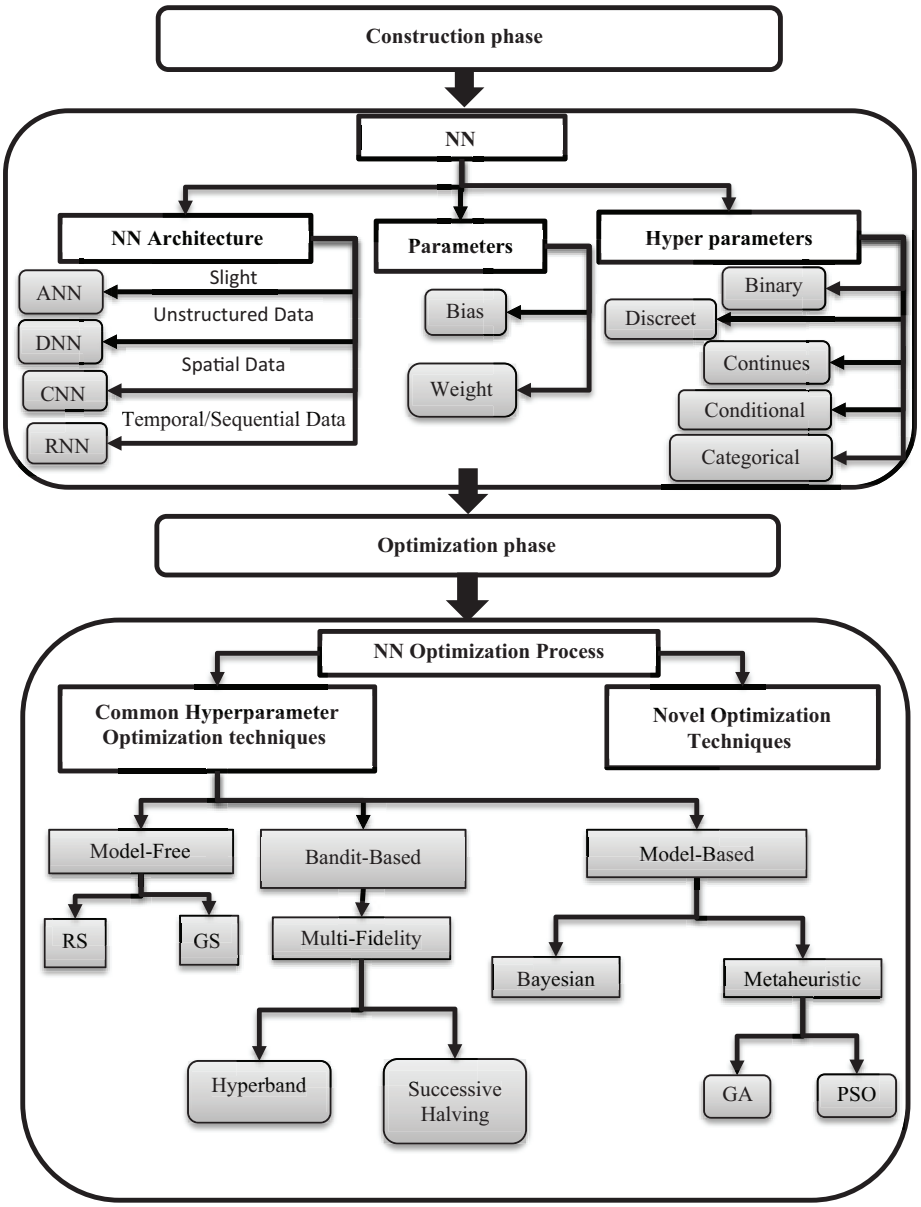


**Fig. 1.** The new taxonomy of hyperparameters optimization processes

## 2.1 Neural networks

Few algorithms identify patterns and are loosely modeled after the human brain. They use a sort of machine perception to understanding sensory data, categorizing or clustering raw data. In addition, the patterns they identify numerically encoded in vectors, where all real-world data should be converted, including sound, images, text, and

time series. The ability to handle with nonlinear functions and learn weights that help map any input to the output for any data known as ANN. An ANN's training can be defined as a continuous optimization process which entails mapping input into output in order to get the optimum set of biases and weights in the shortest time possible [8]. The activation functions give their ANN non-linear qualities, which might help the net learn any complex relationship between output and input data, which is referred to as a universal approximation [9]. RNN, DNN, and CNN are examples of NN implementations with minor differences [10], [11].

**DNN.** DL is a method that uses a hierarchy of concepts in a field to help a computer learn from experience [12]. They use NN topologies to connect various processing layers [13]. This method has been used in a variety of fields, including speech recognition and visual object, medicine, and genomics [14], [15]. DNNs divided into two types: feed-forward and recurrent. CNN are a type of feed-forward DNN that is similar to RNNs.

**CNN.** Convolutional Neural Networks (CNN) are a particular type of Neural Networks (NN) that replicate how the visual system processes information. In general, CNN is a type of feed-forward ANN that uses the backpropagation approach to automatically and adaptively learn complicated hierarchies of data and patterns [16]. Convolutional layers are hidden layers in CNN. In addition, CNN features non-convolutional layers. The convolutional layers draw through the input weight and convert the neurons' input on the activation function, which is the essential notion of CNN structure [17], [18]. CNN is good for audio and image [10], [19].

**RNN.** RNN architecture is a type of DNN that differs from ANN in that the looping requirement on the hidden layer is reversed, resulting in RNN [18]. It is excellent for text and a, time-series data, and audio data, and it can save computing time since the gradient is computed only at the last step and vanishes in every neuron in RNN [20], [21].

### 2.2 Hyperparameters optimization problem

This approach demanded a priori DNN architecture definition. Adjusting the DNN's various hyperparameters is required [22]. In ML models, there are two sorts of parameters: model parameters, which could be initialized as well as updated via data learning processing (the weights of neurons in NNs), and model parameters. The others, known as hyperparameters, cannot be predicted directly from data learning and should be defined before training an ML model since they constitute the ML model's architecture [23]. Hyperparameters are parameters which are utilized to create an ML model or specifying the loss function minimization procedure [24]. The practice of tuning hyperparameters still considered a "black art." Those hyperparameters might regulate model complexity (for instance, the number of layers and nodes in a DNN) or describe the learning technique (for instance, step sizes, learning rate, initialization conditions, and momentum decay parameters) [25]. A hyper-domain parameters could be discrete (number of clusters), continuous (learning rate), categorical (e.g., type of optimizer), or binary (whether to use early stopping or not). In actual applications, the domains of discrete and continuous hyperparameters are frequently bounded [26], [27].

While depending on the value regarding another hyperparameter, conditional hyperparameters might need to be employed or modified [28]. The primary goal of hyperparameter optimization methods is to automate the hyperparameter tuning process and enable users to efficiently apply ML models to real-world problems [29]. Because the DNN's performance is heavily dependent on hyperparameter modification, the model's quality due to the training process, or its capacity to generalize to new data when employed in the wild [30], Hyperparameter tuning therefore plays a significant role in the DNN's success. Finding the appropriate DNN model hyperparameter combination that performs best when scaled on a validation set is necessary [31]. For classifying, analyzing, or upgrading present systems or data, an optimization algorithm is a critical tool for choosing the optimum answer from a set of all viable options. Good optimization techniques are frequently required for minimizing or maximizing objective functions in optimization problems [32]. Weights, learning rules, network design, activation function, neurons, and bias are all commonly optimized using optimization algorithms. Another method for improving and optimizing the ANN is to use an optimizer for replacing the NN's original algorithms with optimization algorithms, and replacing backpropagation with any optimization approaches to overcome particular issues. Yet, instead of back-propagation, an optimization algorithm, such as the Liebenberg Marquardt NN with any optimization approaches for rapid or accurate NN training, can be used.[33]–[35].

### 2.3 Hyperparameters optimization methods

1. **Decision-theoretic:** Those approaches work by first creating a hyperparameter search space, after that detecting hyperparameter combinations inside it, and then picking the best-performing hyperparameter combination:
   - **Grid Search** (GS) Because the hyperparameter values employed by the method are usually unrelated to one another, (GS) can be defined as a decision-theoretic method which requires exhaustively searching for a specified domain of hyperparameter values [26], [36]. It was acknowledged as one of the most widely-used approaches for exploring hyperparameter configuration space [37], [38], as well as exhaustive search or a brute-force technique which assesses all hyperparameter combinations supplied to configurations' grid [39]. GS calculates the cartesian product regarding a finite collection of values given via the user [7].
   - **Random Search** (RS) can be defined as a decision-theoretic approach that, given limited resources and execution time, chooses hyper-parameter combinations in search space at random [40]. This method might be utilized in discrete instances, yet it might also be employed in continuous and mixed spaces. RS may outperform grid search in the case when just some hyperparameters affect the performance of the ML algorithm [36], yet it is not commonly adaptive, however it might be utilized in hybrid ways to drastically enhance performance [41].
2. **Bayesian Optimization (BO):** it could be defined as one of the probabilistic optimization methods which seeks to reduce a global objective black-box function [37]. The models determined the next value of the hyper-parameter depending upon previous results regarding the values of the tested hyperparameter, avoiding various

unnecessary assessments; therefore, BO identify optimum hyperparameter combination in fewer iterations compared to RS and GS [42]. Variable conditionality could be preserved in BO. In addition, BO includes two fundamental components: an acquisition function and a surrogate model, and may therefore be utilized for optimizing conditional hyper-parameters [43], [44]. The surrogate could be defined as one of the probabilistic models, and the posterior estimate regarding the expensive function is generated with the use of Bayes' rule. Through optimizing a selected acquisition function, the next most promising point was discovered. The acquisition function achieves a balance between exploring points in uncharted territory and exploiting points in areas where a track record has been established [25]. The objective purpose of the surrogate model is to fit all of the currently observable points into it. The acquisition function regulates the use of various points through balancing the trade-off between exploitation and exploration after getting the predictive distribution of the probabilistic surrogate model. Exploration entails sampling examples in previously uncharted locations, whereas exploitation entails sampling in the most promising locations in which the global optimum is most likely to take place. The GP [45], random forest (RF) [46], and the tree parzen estimator (TPE) [26].

3. **Multi-Fidelity Optimization Techniques:** these are standard methods for dealing with resource and time constraints. People might utilize a subset of the features or a subset of the original dataset for saving time [47]. The term "multi-fidelity" comes from the combination of high-fidelity and low-fidelity evaluations for practical applications [48]. A small subset of low-fidelity evaluations assessed at low costs, yet with poor generalization performance. A large subset of high-fidelity evaluations had greater generalization performance, yet at high costs compared to low-fidelity evaluations. Badly performing configurations were removed after every one of the rounds of the hyperparameter evaluation on created sub-sets in multi-fidelity optimization methods, and just well performing hyperparameter configurations were reviewed overall training dataset. Multi-fidelity optimization techniques, such as bandit-based algorithms, have shown success in handling DL optimization problem [29]. The successive halving [49] and Hyperband [50] are two popular bandit-based methods.
   - **Successive Halving (ASHA):** this is an approach based on the multi-armed bandit algorithm. The ASHA algorithm can be defined as an asynchronous algorithm of combining random search with principled early stopping [51], Sample a collection of hyperparameter configurations at random, evaluate them, and discard the ones with the lowest scores. Repeat until only one-configuration remains.
   - **Hyperband:** this is a prominent bandit-based optimization approach that compared to RS. It creates small versions of the data sets and gives each hyperparameter combination the same budget. For saving resources and time, Hyperband eliminates poorly performing hyper-parameter combinations with each iteration [52].
4. **Metaheuristic Algorithms:** it is a collection of methods that could be used to solve any optimization problem [53]. Because of its benefits of quick speed, minimal parameters, and straightforward implementation, it has grown to be one of the most well-known classical algorithms in the field of evolutionary computation [54].

The two most common metaheuristic algorithms utilized for hyperparameters optimization issues are genetic algorithm (GA) and particle swarm optimization (PSO) [50], [55], [56].

- **GA:** Through the application of a series of mutation, crossover, and fitness assessments to several chromosomes, the GA addresses optimization problems. This method starts with a population with many chromosomes [57], [58], after that simulates natural selection to see which species could adapt to changes in their environment and reproduce and carry on to the next generation [59], where each one is representing the problem's optimal solution, as determined via an objective function [35].
- **PSO:** One of the most prominent algorithms of the evolutionary optimization is the PSO [60]. The PSO is based on particle position and velocity [61]. One aspect of the PSO is that it does not employ a selection procedure; all members of the population (particles) survive from the beginning to the end of a trial. In addition, their interactions lead to a constant enhancement in the quality of their interactions, which is quantified as the fitness value [62]. PSO is used to create an ANN technique for each neuron that improves synaptic mass, architecture, transfer function [63], [64].

## 3 Literature survey

This section we display some of the prior relevant works on ANN hyperparameters optimization and applications that use hyperparameters technique.

### 3.1 Applications used hyperparameters optimization algorithms

In this part, various previous work that relied on optimization algorithms in their applications are reviews, in order for us to find the most effective algorithms and the most used hyperparameters.

1. Maytham S. Ahmed, et al. [65] Proposed using a hybrid lightning search algorithm (LSA)-based NN in order to forecast the best ON/OFF status for household appliances. They created an ANN with five inputs, 2 hidden layers with sigmoid functions as activation functions, and four outputs. The feed-forward NN and the Levenberg–Marquardt training technique used for the training of ANN.
2. Tian Zhang, et al. [66] Proposed a unique method for plasmatic waveguide-coupled with cavities structure (PWCCS) spectrum prediction, inverse design, parameter fitting, and performance optimization, to construct the network architecture and set hyper-parameters for ANNs.
3. Wenzhe Shi, et al. [67] Depending on a DNN model and the PSO as a hyperparameter optimizer, the authors suggested an efficient digital modulation recognition technique. This suggested approach employs signal preprocessing and an improved DNN model to detect multiple modulation signals in wireless communications.

4. J. F. Torres , et al. [68] Suggested a random search to adjust the technique's many hyper-parameters, followed by a moving averages-based approach to smooth predictions that have been given by the variety of the models for every prediction horizon value.
5. Seunghyup Shin, et al. [4] Developed DNN model by using the worldwide harmonized light vehicles test procedure (WLTP) of diesel engines, adjust its hyper-parameters with the Bayesian optimization approach, and employ hidden-node determination logic for predicting engine-out NOx emissions.
6. Vessela Krasteva, et al. [69] recommended fine-tuning hyper-parameters (HPs) of deep CNN for recognizing non-shockable (NSh) and shockable (Sh) rhythms, as well as confirming the optimal HP settings for long and short analysis durations (2s–10s).
7. Taehywon Kim, et al. [70] Suggested a global optimum rank selection technique based upon Bayesian optimization, which is a global optimization approach based on ML. The suggested approach generates a global optimal rank providing reasonable trade-off between computation complexity and accuracy deterioration through combining a basic objective function with a correct optimization scheme.
8. Guoyin Zhang,et al. [71] Introduced an enhanced adaptive dynamic particle swarm optimization (ADPSO) method, which is based on the PSO algorithm and can dynamically change the program's settings to update particle positions, ensuring that particles find the global best solution.
9. Xueli Xiao, et al. [72] Recommended using a GA with varied duration to boost the performance of a CNN through modifying its hyperparameters to tackle the towing concerns. Overfitting, along with the computing and time expenses, are all factors to consider. Through changing one hyperparameter value as well as applying random mutation to the population segment with the lowest fitness value, the population segment with the lowest fitness value can be improved.
10. H Harafani, et al. [73] Suggested the use of a genetic algorithm for optimizing hyperparameters of NN for predicting liver disease , rather than manually optimizing hyperparameters of NN. They focused on two hyperparameters, momentum coefficient and learning rate, for improving estimation results, and the RMSE was utilized as a result assessor.
11. Razvan Andonie, et al. [74] Used the Weighted Random Search (WRS) technique and compared it to a number of cutting-edge hyperparameter optimization methods. Respecting Convolutional Neural Network (CNN) hyperparameter optimization.
12. Maher G. M. Abdolraso, et al. [75] Introduced a PSO augmentation for ANNs in a virtual power plant (VPP) system, for managing renewable energy resources (RESs).
13. Matteo Miani, et al. [76] Used a Bayesian optimization algorithm for optimizing hyperparameters for prediction of Marshall test results, stiffness modulus, and air voids data regarding various bituminous mixtures for road pavements, with the use of an ML approach based on (ANNs).
14. Mohammad Masum, et al. [37] Proposed a new intrusion detection framework for networks, by optimizing the hyperparameters regarding the DNN architecture with a Bayesian optimization approach. The suggested framework was after that

evaluated and put to comparison with the approach of random search hyperparameter optimization.

15. Puneet Kumar, et al. [77] Developed a new GA-based technique for quickly identifying the optimum hyper-parameter combinations for DNN training, as well as recommending an additional optimization step. Furthermore, the ideal values of all hyperparameters discovered using this method.
16. Jussi Kalliola, et al. [78] Suggested an ANN optimization model for real estate price prediction. To handle the nonlinear problem of real estate price prediction without under- and over-fitting problems, a multilayer perceptron (MLP) NN model utilized, along with fine-tuning hyperparameters in Helsinki, Finland.
17. Mansi Gupta, et al. [79] Through focusing on components of training that effect classifier performance, they produced a DNN model for software defect prediction. In the case when the train's accuracy is inadequate, they recommend adding extra hidden layers and epochs, as well as the DNN model with dropout.
18. Andrea Menapace, et al. [80] Suggested a grid search algorithm to tune ANN hyperparameters for drinking water demand forecasting and implemented it on 4 ANN architectures: Long Short-Term Memory (LSTM), Feed Forward Neural Network (FFNN), Gated Recurrent Unit (GRU), and Simple Recurrent Neural Network (SRNN) with 4 times: 1 hour, 6 hours, 24 hours, and 168 hours.
19. Sebastian Blume, et al. [81] Compared various hyperparameter optimization approaches for the creation of an ANN-based roll angle estimator. Hyperband and Random Search, two random-based approaches, and Genetic Algorithm and Bayesian Optimization, two knowledge-based approaches, compared. The aim of this research is to create ANN-based software sensors.
20. Parampreet Kaur, et al. [82] Proposed a stacked ensemble model employing DNN, a DL model, GBM, and DRF, a distributed form of the RF technique , for the prediction of the breast cancer survival. The Artificial bee colony (ABC) approach employed on the dataset for feature optimization, and parallel Bayes optimization utilized in order to discover the appropriate HPs for ML models.
21. Warut Pannakkong, et al. [83] Applied (RSM) to fine-tune the hyperparameters of three machine learning algorithms: (SVM), (ANN), and (DBN). The goal was to show that RSM is more efficient than grid search in keeping ML algorithm performance while decreasing the number of the runs that needed in order to reach appropriate hyper-parameter values.

### 3.2 Another technique

Some of Optimization algorithms and their variants are not efficient at solving specific issues. Furthermore, while some optimization approaches are effective, they can still improve to increase efficiency. In addition, for the purpose of developing computational intelligence or heuristic optimization, new Nature-inspired optimization methodologies must be regularly developed because speeding up the convergence of an algorithm remains a challenging task. [64], [84]–[86], there are other optimization methods that are used to select the suitable hyperparameters for ANN models researchers in [87] suggest utilizing variance Matrix Adaptation Evolution Strategy (CMA-ES),

which is well-known for its cutting-edge efficiency in derivative-free optimization, while in [88] adapted a simpler coordinate-search and Nelder-Mead methods for the optimization of the hyper-parameters. In [25] the researchers applied RBFs as error surrogates and use an integer algorithm called (HORD) for hyper-parameter optimization that is both deterministic and efficient. Also, in an article [89] they introduced a new technique that optimizes numerous hyperparameters by combining multiscale and multilevel evolutionary optimization (MSMLEO) with GPEI. In [90] suggested using a Microcanonical optimization algorithm for hyperparameter optimization and architecture selection for CNNs. It is interesting to note that the results of all of these papers were compared using the Bayesian optimization algorithm, in [90] and [88] they use simulated annealing and random search algorithm for comparison. In [91], [92] they Compare their results with the use of novel methods ( univariate dynamic encoding algorithm and the using an improved Gene Expression Programming) to enhance hyperparameters sequentially by GA and PSO algorithms. All results were significant improvement by choosing the right hyperparameters.

## 4 Analysis and discussion

Neural networks have wide spread and common uses and the results extracted from the neural networks are satisfactory. However, optimization algorithms, if used in combination with the network, will significantly increase the accuracy because of selecting the best hyperparameters. In this part, the optimization process is analysis and discuses in two axes:

### 4.1 Optimization algorithms

In order for us to clarify the strengths and weaknesses of the optimization algorithms, we summarize Table 1 that views previously work that using optimization algorithms that lead to improving different models, taking into consideration the limitations that countered.

**Table 1.** Summary of hyperparameters optimizations methods and concept

| Ref. Year | Type Model | NN Type | Optimization Algorithm | Comparison to Evaluate | Results | Drawbacks |
|---|---|---|---|---|---|---|
| [93] 2011 | BpNN RBFNN | ANN | GA | • RBF1<br>• RBF2 | • Reduce training error.<br>• Find better network structure.<br>• Reduce the constructing network structure time.<br>• Improve performance.<br>• Stronger classification. | • The performance of GA should enhanced, and the approach should determine the network's maximum scale. |
| [25] 2016 | RBF HORD | CNN | Bayesian optimization | • GP<br>• TPE | • Find a near optimal solution with fewer expensive function evaluations | • Expensive covariance computation |
| [94] 2017 | DNN | CNN | PSO | • Compared the result before and after hyperparameters optimization | • Image recognition accuracy improved by 0.7% – 5.7%. | • Heavy computational load when applying the evolutionary algorithm. |
| [55] 2017 | DNN | CNN, DNN | PSO | • GS<br>• RS | • Less running time<br>• Higher accuracy | • Overfitting is shown |
| [30] 2017 | Master-slave | DNN | Parallel PSO | Compared parallel PSO in small search space with parallel PSO on large search space. | • Decreases the time of the execution in the small search spaces, yet still very beneficial in the larger spaces. | • Eexpensive evaluation of the objective function |
| [36] 2019 | NAS | CNN | GS,RS,GA | Compared result by proposed algorithms performances | • Random search faster than grid search.<br>• GS or RS could be a suitable option if the model and search space are not too huge.<br>• If a model has a lot of layers and a lot of search space, the GA method might be the best option. | • RS is limited to search space distributions.<br>• Grid Search is too slow to choose the best model. |

*(Continued)*

**Table 1.** Summary of hyperparameters optimizations methods and concept *(Continued)*

| Ref. Year | Type Model | NN Type | Optimization Algorithm | Comparison to Evaluate | Results | Drawbacks |
|---|---|---|---|---|---|---|
| [95] 2019 | DNN | CNN | GA | • RS<br>• GS | • GA has superior in optimization with LS in optimizing both network training and structures. | • The small CNNs on the CPU indicated that rates of learning of 0.010 and higher do not produce satisfactory results on models. |
| [72] 2020 | DNN | CNN | Variable length GA | • RS<br>• Classical GA<br>• Large scale evaluation | • The obtained Accuracy variable length GA is 88.92%<br>• RS is 58.66%,<br>• Classical GA is 80.75%<br>• Large Scale evaluation is 51.90% | • Practical to traverse every single possibility to find the best model because of long evaluation time. |
| [74] 2020 | DNN | CNN | WRS | • RS<br>• NM<br>• PSO<br>• SS<br>• BO<br>• TPE | • WRS approach finds the global optimum more quickly than the RS method<br>• WRS has shown to be convergent. | • Computationally expensive |
| [77] 2021 | DNN | CNN, RNN | GA | • GS | • In the first optimization face of finding optimal parameters, CNN are faster than RNN.<br>• In the second face (CNN) is slower than (RNN) in identifying a suitable sub-set of training data for the near-optimum efficiency of prediction. | • The fitness measure for GA is somewhat weak. |

Based on what previously reviewed in section (2.3) and Table 1, we propose Table 2 for comparison and analysis between optimization algorithms advantage and drawbacks.

**Table 2.** Advantages and disadvantages of hyperparameters optimization algorithms

| Technique | Advantage | Disadvantage |
|---|---|---|
| Grid search (GS) [7], [36], [39], [55] | • Considered a brute-force technique or exhaustive search that assesses all hyperparameter combinations given to the grid of setups.<br>• Could be simply parallelized and implemented. | • It is not possible to further exploit the high-performing regions on its own.<br>• It is ineffective for high-dimensionality hyperparameter configuration space.<br>• A boundary must be defined before using a GS.<br>• Due to the constantly increasing cost, GS could lose parallelism. |
| Random search (RS) [6], [7], [55], [96] | • Because each one of the evaluations is independent, it is simple to parallelize and allocate resources.<br>• Enhances the efficiency of the system through decreasing the likelihood of wasting a lot of time on small poor-performing area.<br>• It can explore larger search space compared with GS.<br>• It can identify global optima or near global optima in the case where provided with enough budget. | • Previously well-performing regions are not exploited. |
| Bayesian algorithm [42], [55], [86], [97] | • utilizes the previous evaluation records for the determination of the next evaluation, which is why, don't waste time on the evaluation of the poorly-performing search space areas.<br>• Can typically detect near optimum hyperparameter combinations within few iterations. | • Difficulty in parallelize.<br>• Inference time increases cubically in a number of the cases, because it necessitates inverting a dense covariance matrix. |
| Multi-fidelity algorithms [98] | • Makes it possible to carry out the optimization with numerous design variables and responses utilizing the computationally expensive analyses through the virtual elimination of the costs of the gradient computations.<br>• Reduces any need for the correlation of low and high fidelity analyses results throughout the optimization. | • The designer must create 2 models of various fidelity with similar design responses and variables.<br>• It could be the designer's responsibility to compute finite-difference gradients with the use of low-fidelity analyses codes, in the case where there are not any explicit gradient computations in low-fidelity analyses codes.<br>• The optimizer will potentially bring design into optimum region, instead of pinpointing a precise location of an optimum. |
| PSO [84] | • Rapid convergence.<br>• The ability to solve complicated problems in another domain of the application. | • could be trapped in the local minima easily.<br>• unsuitable control parameter selection results in poor solution.<br>• It has vulnerability to getting stuck in the local minima and incorrectly choosing the control parameters, which could lead to bad solutions |

*(Continued)*

**Table 2.** Advantages and disadvantages of hyperparameters optimization algorithms *(Continued)*

| Technique | Advantage | Disadvantage |
|---|---|---|
| GA [93], [99] | • It requires no derivative information.<br>• proper for large numbers of the variables.<br>• Ignore the gradient information related to the error functions; also learn the approximate optimal solution. | • No guarantees to find global minimum.<br>• Difficult to fine-tune all of the parameters, such as the rate of the mutation, parameters of the crossover, and so on, which is often done by just trial and error.<br>• Long time for the convergence. |

Based on the foregoing, we can summarize the following points:

- Grid search is optimal for spot-checking combinations that had performed well previously. Even though it takes longer to execute, random search is fantastic for discovery and finding hyperparameter combinations that you would not have anticipated intuitively. Those two techniques are effective in prediction and classification. The application in [37], [68], [74], [80], [81], [83] used GS and RS as hyperparameter selection methods. The GS explored all hyperparameters combinations in the search space, but it is expensive and not efficient if the search space has high dimensions, while RS from its name selected hyperparameters randomly and considered as the most efficient way of searching the hyperparameters configurations [5], [96], [100].
- Bayesian Optimization is an approach that uses Bayes theorem to direct the search in order to find the minimum or maximum of an objective function. In addition, it balance the exploration and the exploitation processes to detect the current most likely optimal regions and avoid missing better configurations in the unexplored areas [101]. And it is an approach that is most useful for objective functions that are complex, noisy, and/or expensive to evaluate like developing in [4], detection in [37], selection in [70], and designing in [82], addition can use BO as hyperparameter optimization for enhance the model accuracy.
- Using PSO to search for hyper-parameters has widely studied and tested as well in a variety of studies, with positive results in a variety of applications. For instance, CNN-based PSO reduced CNN weights in final network by optimizing the hyperparameter linearly [18]. PSO optimization also improves NNs through locating the best hyperparameters for network architecture design [67]. In addition for digital modulation recognition applications, a PSO-based deep NN has been utilized for optimizing the number of the hidden layer nodes [65], [67], [71].

### 4.2 Hyperparameters

The hyperparameters considered the main key in the optimization process carried out by optimization algorithms. In the Tables 3–4 we make a comparison between the previous works reviewed in Part (3.1) this perform contributes to determining the best hyperparameters results. Finding the best optimization method enables suitable tuning of the neural structures in the professional networks, such as the optimal number of the neurons, weights, hidden layers, self-shaping architecture, bias, and multi-stage objective functions, in comparative performance output regarding controllers of an ANN.

**Table 3.** Summary of applications used hyperparameters optimizations

| Ref. Year | Hyperparameter Optimization Utilizes | Name of Hyperparameter Used | Problems Encountered | Overcome the Problem |
|---|---|---|---|---|
| [65] 2016 | Scheduling home energy management. | • Number of nodes.<br>• Learning rate. | • The optimal learning rate and number of hidden layer nodes are unknown. | • They combined LSA optimization with ANN for enhancing the performance of ANN by choosing the best hyperparameter |
| [66] 2019 | Plasmatic waveguide systems | • The solvers for weights activation functions<br>• No. of neurons per layer.<br>• No. of layers | • The deviation harder to detect with the naked eye.<br>• The optimized and expected waveguide coupling distances are comparable in all of the techniques. | • They didn't overcome this problems |
| [67] 2019 | Digital modulation recognition | • Number of nodes<br>• Learning rate<br>• Iteration number<br>• Number of a particle swarm<br>• Cognitive coefficients | • The number of hidden layer nodes must be manually selected in the standard DNN because it is trapped in local minima. | • Use PSO-DNN |
| [68] 2019 | Power Consumption Forecasting | • Number of nodes<br>• Number of layers | • Random search work on discrete hyperparameters value. | • Proposed random search with continuous value give hyperparameters range of values. |
| [4] 2020 | $NO_X$ Predictions at transient conditions in the diesel engines | • No. of layers<br>• No. of nodes<br>• Learning rate decay<br>• Batch size<br>• Learning rate | • Computationally expensive time and costs. | • Replaced each one of the hidden layer's numbers of nodes by number of the nodes in 1st hidden layer. |
| [69] 2020 | Shockable and Non-Shockable rhythms detection. | • Learning rate.<br>• Number of convolution layer. | • Generalization performance and training stability is not good enough. | • Split the training data into batch. |
| [71] 2020 | Ship motion attitude prediction | • No. of nodes in hidden layer. | • Probability of falling into the local optima. | • Using ADPSO |
| [70] 2020 | Improve low-rank decomposition | • Acquisition function.<br>• Activation function.<br>• Learning rate. | • The trade-offs between the rate of the compression and accuracy degradation represented by objective function, which contains computational complexity and reconstruction losses. | • Controlled by rank selection, which returns 0 in the case where the input is less than the threshold, but returns the value of the input otherwise. |

*(Continued)*

**Table 3.** Summary of applications used hyperparameters optimizations *(Continued)*

| Ref. Year | Hyperparameter Optimization Utilizes | Name of Hyperparameter Used | Problems Encountered | Overcome the Problem |
|---|---|---|---|---|
| [72] 2020 | Improve the CNN performance | • Mutation rate<br>• No. of layers<br>• No. of nodes<br>• No. of phases<br>• Survival chance of the less fit individual activation function | • Overfitting if the number of layers is too big.<br>• Under fitting, if layer are too small. | • Fit number of layers manually |
| [73] 2020 | Liver disease estimation | • Learning rate<br>• momentum coefficient | • Local minimum | • Using genetic algorithm |
| [74] 2020 | Reducing time needed for complete training-testing cycle for CNN architecture. | • The number of output filters for each one of the convolutional layers.<br>• No. of the nodes for every one of fully connected layers.<br>• The number of convolutional layers.<br>• No. of the fully connected layers. | • Each data set has a different architecture, which requires changes for each individual case. | • They did not overcome this problem. |
| [75] 2021 | Microgrid scheduling and management | • rate of Learning<br>• No. of the nodes in hidden layer | • Occurs Some errors | • Utilizing Feed-forward NN and Levenberg-Marquardt approach for the minimization of the summation of square error functions. |
| [76] 2021 | Predictions of stiffness modulus, Marshall test results, and air voids data of a variety of the bituminous mixes for the pavements of the roads | • Number of layers<br>• Activation function<br>• Acquisition function<br>• Weight decay<br>• learning rate<br>• Number of nodes | • Overfitting | • Use L2 regularization |

| | | | | |
|---|---|---|---|---|
| [37] 2021 | Network intrusion detection | • Dropout rate<br>• No. of nodes<br>• No. of dense layers<br>• Learning rate<br>• Optimizer<br>• Activation function | • Exploding gradients problem in training phase.<br>• Overfitting. | • Applied min-max normalization approach<br>• Using dropout technique |
| [77] 2021 | Quickly select optimum hyperparameter values and training data sub-set for near optimum efficiency of model. | • Activation function<br>• Number of nodes<br>• Number of layers<br>• Loss function<br>• Regularization function<br>• Network optimizer | • Overfitting | • Early stopping |
| [78] 2021 | Prediction of the prices of the real estate in Helsinki | • Batch size<br>• Optimization algorithm<br>• Activation function<br>• Validation split<br>• Learning rate<br>• Dropout | • Vanishing gradients problem | • Use ReLU |
| [79] 2021 | Predicting software faults. | • Loos function<br>• Number of nodes<br>• Learning rate<br>• Number of layers<br>• Optimization algorithm<br>• Regularization | • Overfitting | • The L2 regularization method modifies the cost function to reduce overfitting.<br>• The dropout strategy reduces overfitting By changing the network itself. |
| [80] 2021 | Predicting drinking water demand. | • Number of nodes<br>• Number of layers | • Difficulty in choosing the best configuration<br>• The stochastic character of ANN lead to the misleading model selection with a traditional tuning search. | • Use median to select the more suitable neural network configuration.<br>• An interactive tuning procedure suggested, with the number of iterations determined by a balance of accuracy and computational intensity. |

*(Continued)*

**Table 3.** Summary of applications used hyperparameters optimizations *(Continued)*

| Ref. Year | Hyperparameter Optimization Utilizes | Name of Hyperparameter Used | Problems Encountered | Overcome the Problem |
|---|---|---|---|---|
| [81] 2021 | Designing Software Sensors | • Sequence Lookback Input<br>• Recurrent Layer Type<br>• Number of nodes<br>• Number of Layers<br>• L2-Regularization value<br>• Recurrent L2- Regularization value<br>• Dropout value<br>• Recurrent Dropout value<br>• Batch Size<br>• Learning Rate | • confuse to select many configurations (high n) for a short time of training or a few configurations (small n) for a longer time of the training.<br>• Because elite and crossover individuals don't produce new information, convergence occurs locally. | • The hyper-band algorithm has the goal of circumventing the problem through testing many potential values for number of the configurations n given fixed maximum. Several overalls.<br>• Applying mutation |
| [82] 2022 | Breast cancer disease prediction | • Hidden layer sizes<br>• Epochs<br>• Number of iteration<br>• Dropout ratio<br>• Learning rate<br>• Max depth<br>• Sample rate<br>• Number of trees<br>• Maximum tree depth | • The overfitting with the heterogeneous dataset. | • Stacked ensemble method and (DRF) used to overcome the overfitting problem. |
| [83] 2022 | Predict the quality of the raw material | • Learning rate<br>• No. of training cycles<br>• No. of pre-training cycles<br>• Convergence epsilon<br>• No. of hidden nodes<br>• Complexity constant<br>• Size of the Batch<br>• Pre-training learning rate | • In the case when the regression model only includes significant linear terms, the process of the tuning hasn't achieved desired hyperparameters. | • The steepest descent route which can enhance the performance of ML model, as determined by coefficients of the variables that represent the regression model's hyperparameters. |

**Table 4.** Applications used hyperparameters optimizations methods comparison

| Ref. Year | Network Type | Used Dataset | Techniques / Tools | Obtained Result |
|---|---|---|---|---|
| [65], 2016 | • Feed-forward neural network | • Dataset are collected from the simulation system | • A lightning search algorithm based PSO | • During the DR event, decrease peak-hour energy use by up to 9.7138 %. |
| [66], 2019 | ANN | Their own dataset | GA | • PWCCS inverted design with high precision<br>• Improves several crucial transmission spectrum performance parameters |
| [67], 2019 | DNN | Their own dataset | PSO | • If SNR is equal to 0 and 1 dB, the recognition rate using this approach improves by 9.40% and 8.80%, respectively, when compared to approaches which use traditional DNN and SVMs. |
| [68], 2019 | DNN | Electrical electricity consumption in Spain | RS | • The smoothing technique minimizes the forecasting error, while random search gives competitive accuracy outcomes with few models. |
| [4], 2020 | DNN | Logged data from ETK–ECU interface | Bayesian Optimization algorithm | • The accuracy level has been increased, as can be noticed by an $R^2$ of 0.9675. |
| [69], 2020 | CNN | Public Holter ECG OHCA | RS | • Increase the Sh and NSh rhythms' detection and can significantly reduce analysis time while adhering to resuscitation requirements for minimal hands-off pauses |
| [71], 2020 | BiLSTM NN | Collects the measured motion data of a ship. | ADPSO | • Compared to BiLSTM, LSTM, and PSO-BiLSTM NN models, the ADPSO-BiLSTM NN model might better fit the data. |
| [70], 2020 | CNN | • CIFAR-10<br>• CIFAR-100<br>• ImageNet | • Bayesian Optimization algorithm | • The proposed approach gives multi-rank with increased performance and compression in comparison to state-of-the-art rank selection method, VBMF. |
| [72], 2020 | CNN | • CIFAR-10 | • Variable-length GA | • The accuracy results obtained in the 30 (Hours) compared to methods (random search 58.66%, classical GA 80.75%, and large scale Evaluation 51.90%) is (88.92%). |

*(Continued)*

**Table 4.** Applications used hyperparameters optimizations methods comparison *(Continued)*

| Ref. Year | Network Type | Used Dataset | Techniques / Tools | Obtained Result |
|---|---|---|---|---|
| [73], 2020 | ANN | • Liver Disorder dataset from BUPA | • GA | • GA gives the smallest RMSE compared with other ML algorithms like K-NN, SVM, and NN. |
| [74], 2020 | CNN | • CIFAR-10 | • WRS<br>• RS | • The accuracy that obtained when using WRS is 0.85% and it is the best one compared with other methods are used. |
| [75], 2021 | ANN | • Their own data | • PSO | • ANN-PSO offers more exact decisions than the BPSO algorithm, indicating that Neural Net augmentation has reached the optimum level of energy scheduling. |
| [76], 2021 | ANN | • Variable dataset | • Bayesian Optimization algorithm | • ANN model that has been identified by Pearson coefficient of 0.868. |
| [37], 2021 | DNN | • NSL-KDD | • Bayesian optimization algorithm<br>• Gaussian Processes<br>• RS | • The BO-GP method performs better than the random search approach. For the KDDTest + and KDDTest-21 datasets, BO-GP had the maximum accuracy of 82.95% and 54.99%, respectively. |
| [77],2 021 | RNN CNN | • Stock market data<br>• MINIST | GA | • Find near optimal performance and speed up the optimization. |
| [78], 2021 | ANN | Information gleaned from a real estate price search in Helsinki on the internet. | • Neighborhood component analysis (NCA)<br>• Bayesian optimization algorithm<br>• Person correlation.<br>• Regression tree. | • According to the analysis, 0.05 enhances The R2 value, and the RME value is enhanced by 2.5 percent Reaching a mean error of 8.3% |
| [79], 2021 | DNN | 4 NASA system datasets (PC1, KC1, KC3, PC2) | • RF<br>• DT.<br>• Naïve byes for calculating the accuracy | • The accuracy generated by proposed DNN with dropout highest in comparison with other ML techniques. |

*(Continued)*

**Table 4.** Applications used hyperparameters optimizations methods comparison *(Continued)*

| Ref. Year | Network Type | Used Dataset | Techniques / Tools | Obtained Result |
|---|---|---|---|---|
| [80], 2021 | ANN | Synthetic real datasets | Grid search method | • The number of layers and nodes increasing as well as the ANN models change depending on forecasting horizontal.<br>• performance of LTSM and GRU is the best and performance of the FFNN and SRNN is good in the one-hour horizon then the accuracy will be decreasing. |
| [81], 2021 | ANN | Predefined standard driving maneuvers dataset | • Random Search<br>• Hyperband algorithm<br>• Bayesian Optimization<br>• Genetic Algorithm | • All methods proved a better solution, yet the GA results in promising solutions in this application |
| [82], 2022 | • DNN<br>• GBM<br>• DRF | • TCGA dataset<br>• RNA-seq dataset<br>• Metabolomics dataset<br>• METABRIC dataset | • Bayesian optimization with Gaussian Processes. | License module for prediction are:<br>• TCGA data-set produces 83.90% AUC.<br>• METABRIC data-set gives 87.30% AUC<br>• RNA-seq dataset it gives 80.10% AUC.<br>• Metabolomics data-set gives 91.10% AUC. |
| [83], 2022 | • DBN<br>• SVM<br>• ANN | • An industrial user in the food industry dataset. | • RSM<br>• Grid search | For ANN and DBN, GS hyperparameter settings are 80% reliable, while RSM settings are 90% and 100% reliable, respectively. |

Based on what described previously in Tables 3–4 adscription, analysis and dictation will be present in the subsections below according to the obtained strength and weakness points as follows:

One of the most significant hyperparameters is the number of the hidden layers and number of neurons learning rate; the first two are set and tuned depending on the data sets or problem complexity, while the last one determines step size at every one of the iterations, allowing the objective function to converge.

In [37], [72], [76], [77], [79], [82] overfitting problem is showed, in [37], [82] dropout rate is most important hyperparameter to prevent this issue, which the dropout and number of epochs enable model to reduce overfitting, while in [76], [79] L2 regularization is used to reduce the chance of model overfitting in this case the learning rate hyperparameter is affected . Lastly in [77] [72] they overcome the overfitting without relying on the hyperparameter in [77] use early stopping technique and in another one fit the neuron number manually. The second problem accurse is filling in local minimum the most active hyperparameter in this case is momentum coefficient just like problem that showed in [73].

In [4], [78], [81], [83] the batch size hyper parameter was improved since it represents the number of the samples that have been processed prior to updating the model and the number of the complete passes through the whole training dataset in cases of a large dataset. Furthermore, the learning algorithm's dynamics are influenced by an important hyperparameter. It is critical to investigate the dynamics of the model in order to get the most out of it.

## 5 Conclusions

ML has become the go-to method for tackling data-related problems, and it was integrated into a wide range of applications. Hyperparameters should be adjusted to fit individual datasets in order to apply ML models to reality. The adoption of an automated method for optimizing hyper-parameters has become crucial. Depending on past related work testing findings for enhancing the performance of ANN, we have focused on creating the NN by employing optimization techniques to discover the optimum ANN hyperparameters to achieve the best structure network in this work. Number of layers, learning rate, and number of neurons were determined to be the most commonly employed hyperparameters to improve accuracy. Also, we discovered that BO is more effective compared to RS and GS because it could detect the best combinations of the hyper-parameters through evaluating previously tested values, and executing a surrogate model is frequently less expensive compared to running the complete objective function. PSO can simple handle discrete problems, but because it is continuous, it should be adapted for handling discrete problems. In high-dimensional space, on the other hand, it is simple to fall into a local optimum, and the iterative process has a poor convergence rate. The genetic algorithm and the Bayesian algorithm were the most commonly utilized optimization algorithms, and they produced excellent results. Finally, the overfitting problem was the most frequently encountered problem across all articles. We anticipate that this work will contribute to a better knowledge of the present difficulties in HPO domain, paving the way for future research on hyperparameter optimization and machine learning applications.

## 6 References

[1] J. D. Pineda-Jaramillo, "A review of machine learning (ML) algorithms used for modeling travel mode choice," *DYNA*, vol. 86, no. 211, pp. 32–41, 2019, https://doi.org/10.15446/dyna.v86n211.79743

[2] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative study of CNN and RNN for natural language processing," 2017, [Online]. Available: http://arxiv.org/abs/1702.01923

[3] M. G. M. Abdolrasol *et al.*, "Artificial neural networks based optimization techniques: A review," *Electron.*, vol. 10, no. 21, 2021, https://doi.org/10.3390/electronics10212689

[4] S. Shin, Y. Lee, M. Kim, J. Park, S. Lee, and K. Min, "Deep neural network model with Bayesian hyperparameter optimization for prediction of NOx at transient conditions in a diesel engine," *Eng. Appl. Artif. Intell.*, vol. 94, p. 103761, 2020, https://doi.org/10.1016/j.engappai.2020.103761

[5] B. Panda, "A survey on application of population based algorithm on hyperparameter selection," 2020, https://doi.org/10.13140/RG.2.2.11820.21128
[6] C. Witt, "Worst-case and average-case approximations by simple randomized search heuristics," *Lect. Notes Comput. Sci.*, vol. 3404, pp. 44–56, 2005, https://doi.org/10.1007/978-3-540-31856-9_4
[7] F. Hutter, *Meta-learning*, vol. 498. 2019.
[8] Terrence L. Fine, "Feedforward neural network methodology," *Feed. Neural Netw. Methodol.*, 1999, https://doi.org/10.1007/b97705
[9] N. Ketkar, "Convolutional neural networks," *Deep Learn. with Python*, pp. 63–78, 2017, https://doi.org/10.1007/978-1-4842-2766-4_5
[10] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nat. 2015 5217553*, vol. 521, no. 7553, pp. 436–444, May 2015, https://doi.org/10.1038/nature14539
[11] A. Mosavi, S. Ardabili, and A. R. Várkonyi-Kóczy, "List of deep learning models," *Lect. Notes Networks Syst.*, vol. 101, pp. 202–214, 2020, https://doi.org/10.1007/978-3-030-36841-8_20
[12] I. Conference, *Engineering for Sustainable Future*. 2019.
[13] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006, https://doi.org/10.1162/neco.2006.18.7.1527
[14] P. Rani, Kavita, S. Verma, and G. N. Nguyen, "Mitigation of black hole and gray hole attack using swarm inspired algorithm with artificial neural network," *IEEE Access*, vol. 8, pp. 121755–121764, 2020, https://doi.org/10.1109/ACCESS.2020.3004692
[15] A. Milad, I. Adwan, S. A. Majeed, N. I. M. Yusoff, N. Al-Ansari, and Z. M. Yaseen, "Emerging technologies of deep learning models development for pavement temperature prediction," *IEEE Access*, vol. 9, pp. 23840–23849, 2021, https://doi.org/10.1109/ACCESS.2021.3056568
[16] M. E. Hoque and K. Kipli, "Deep learning in retinal image segmentation and feature extraction: A review," *Int. J. online Biomed. Eng.*, vol. 17, no. 14, pp. 103–118, 2021, https://doi.org/10.3991/ijoe.v17i14.24819
[17] S. R. Dubey, S. Chakraborty, S. K. Roy, S. Mukherjee, S. K. Singh, and B. B. Chaudhuri, "DiffGrad: An optimization method for convolutional neural networks," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 31, no. 11, pp. 4500–4511, 2020, https://doi.org/10.1109/TNNLS.2019.2955777
[18] K. G. Kim, "Book review: Deep learning," *Healthc. Inform. Res.*, vol. 22, no. 4, pp. 351–354, Oct. 2016, https://doi.org/10.4258/hir.2016.22.4.351
[19] L. Liu, W. Liu, and D. A. Cartes, "Particle swarm optimization-based parameter identification applied to permanent magnet synchronous motors," *Eng. Appl. Artif. Intell.*, vol. 21, no. 7, pp. 1092–1100, Oct. 2008, https://doi.org/10.1016/j.engappai.2007.10.002
[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, https://doi.org/10.1162/neco.1997.9.8.1735
[21] F. Yang, H. Moayedi, and A. Mosavi, "Predicting the degree of dissolved oxygen using three types of multi-layer perceptron-based artificial neural networks," *Sustain. 2021*, vol. 13, no. 17, p. 9898, Sep. 2021, https://doi.org/10.3390/su13179898
[22] M. Feurer and F. Hutter, "Hyperparameter optimization," pp. 3–33, 2019, https://doi.org/10.1007/978-3-030-05318-5_1
[23] M. Kuhn and K. Johnson, *Applied Predictive Modeling with Applications in R*, vol. 26. 2013, https://doi.org/10.1007/978-1-4614-6849-3
[24] G. I. Diaz, A. Fokoue-Nkoutche, G. Nannicini, and H. Samulowitz, "An effective algorithm for hyperparameter optimization of neural networks," *IBM J. Res. Dev.*, vol. 61, no. 4, pp. 1–20, 2017, https://doi.org/10.1147/JRD.2017.2709578

[25] I. Ilievski, T. Akhtar, J. Feng, and C. A. Shoemaker, "Efficient hyperparameter optimization of deep learning algorithms using deterministic RBF surrogates," *31st AAAI Conf. Artif. Intell. AAAI 2017*, pp. 822–829, 2017. https://doi.org/10.1609/aaai.v31i1.10647

[26] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Adv. Neural Inf. Process. Syst. 24–25th Annu. Conf. Neural Inf. Process. Syst. 2011, NIPS 2011*, pp. 1–9, 2011.

[27] B. Shahriari, A. Bouchard-Côté, and N. de Freitas, "Unbounded bayesian optimization via regularization," *Proc. 19th Int. Conf. Artif. Intell. Stat. AISTATS 2016*, pp. 1168–1176, 2016.

[28] G. Luo, "A review of automatic selection methods for machine learning algorithms and hyper-parameter values," *Netw. Model. Anal. Heal. Informatics Bioinforma.*, vol. 5, no. 1, pp. 1–16, 2016, https://doi.org/10.1007/s13721-016-0125-6

[29] R. Elshawi, M. Maher, and S. Sakr, "Automated machine learning: State-of-the-art and open challenges," 2019, [Online]. Available: http://arxiv.org/abs/1906.02287

[30] P. R. Lorenzo, J. Nalepa, L. S. Ramos, and J. R. Pastor, "Hyper-parameter selection in deep neural networks using parallel particle swarm optimization," *GECCO 2017 - Proc. Genet. Evol. Comput. Conf. Companion*, pp. 1864–1871, 2017, https://doi.org/10.1145/3067695.3084211

[31] R. Andonie and A. C. Florea, "Weighted random search for CNN hyperparameter optimization," *Int. J. Comput. Commun. Control*, vol. 15, no. 2, pp. 1–11, 2020, https://doi.org/10.15837/ijccc.2020.2.3868

[32] J. Jiang and J. A. Fan, "Simulator-based training of generative neural networks for the inverse design of metasurfaces," *Nanophotonics*, vol. 9, no. 5, pp. 1059–1069, 2019, https://doi.org/10.1515/nanoph-2019-0330

[33] L. Kouhalvandi, O. Ceylan, and S. Ozoguz, "Automated deep neural learning-based optimization for high performance high power amplifier designs," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 67, no. 12, pp. 4420–4433, Dec. 2020, https://doi.org/10.1109/TCSI.2020.3008947

[34] E. Uzlu, M. Kankal, A. Akpinar, and T. Dede, "Estimates of energy consumption in Turkey using neural networks with the teaching–learning-based optimization algorithm," *Energy*, vol. 75, pp. 295–303, Oct. 2014, https://doi.org/10.1016/j.energy.2014.07.078

[35] H. Wu, Y. Zhou, Q. Luo, and M. A. Basset, "Training feedforward neural networks using symbiotic organisms search algorithm," *Comput. Intell. Neurosci.*, vol. 2016, 2016, https://doi.org/10.1155/2016/9063065

[36] P. Liashchynskyi and P. Liashchynskyi, "Grid search, random search, genetic algorithm: A big comparison for NAS," no. 2017, pp. 1–11, 2019, [Online]. Available: http://arxiv.org/abs/1912.06059

[37] M. Masum *et al.*, "Bayesian hyperparameter optimization for deep neural network-based network intrusion detection," pp. 5413–5419, 2022, https://doi.org/10.1109/BigData52589.2021.9671576

[38] M. N. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Systematic ensemble model selection approach for educational data mining," *Knowledge-Based Syst.*, vol. 200, 2020, https://doi.org/10.1016/j.knosys.2020.105992

[39] M. N. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-split optimized bagging ensemble model selection for multi-class educational data mining," *Appl. Intell.*, vol. 50, no. 12, pp. 4506–4528, 2020, https://doi.org/10.1007/s10489-020-01776-3

[40] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012.

[41] J. Bergstra, J. B. Ca, and Y. B. Ca, "Random search for hyper-parameter optimization Yoshua Bengio," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012, Accessed: May 26, 2022. [Online]. Available: http://scikit-learn.sourceforge.net

[42] K. Eggensperger *et al.*, "Towards an empirical foundation for assessing Bayesian optimization of hyperparameters," *BayesOpt Work. @ NeurIPS*, pp. 1–5, 2013.
[43] K. Eggensperger, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Efficient benchmarking of hyperparameter optimizers via surrogates," *Proc. Natl. Conf. Artif. Intell.*, vol. 2, pp. 1114–1120, 2015. https://doi.org/10.1609/aaai.v29i1.9375
[44] M. Injadat, F. Salo, A. B. Nassif, A. Essex, and A. Shami, "Bayesian optimization with machine learning algorithms towards anomaly detection," *2018 IEEE Glob. Commun. Conf. GLOBECOM 2018 - Proc.*, 2018, https://doi.org/10.1109/GLOCOM.2018.8647714
[45] M. Seeger, "Gaussian processes for machine learning university of California at Berkeley," *Int. J. Neural Syst.*, vol. 14, pp. 69–109, 2004. https://doi.org/10.1142/S0129065704001899
[46] F. Hutter, H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration lecture notes in computer science," *Int. Conf. Learn. Intell. Optim.*, pp. 507–523, 2011, [Online]. Available: https://www.cs.ubc.ca/~hutter/papers/11-LION5-SMAC.pdf
[47] L. Bottou, "Large-scale machine learning with stochastic gradient descent," *Proc. COMPSTAT 2010 - 19th Int. Conf. Comput. Stat. Keynote, Invit. Contrib. Pap.*, pp. 177–186, 2010, https://doi.org/10.1007/978-3-7908-2604-3_16
[48] S. Zhang, J. Xu, E. Huang, and C. H. Chen, "A new optimal sampling rule for multi-fidelity optimization via ordinal transformation," *IEEE Int. Conf. Autom. Sci. Eng.*, vol. 2016, pp. 670–674, Nov. 2016, https://doi.org/10.1109/COASE.2016.7743467
[49] Z. Karnin, T. Koren, and O. Somekh, "Almost optimal exploration in multi-armed bandits," *30th Int. Conf. Mach. Learn. ICML 2013*, vol. 28, pp. 2275–2283, 2013.
[50] Q. Yao *et al.*, "Taking human out of learning applications: A survey on automated machine learning," pp. 1–20, 2018, [Online]. Available: http://arxiv.org/abs/1810.13306
[51] R. Schmucker, M. Donini, M. B. Zafar, D. Salinas, and C. Archambeau, "Multi-objective asynchronous successive halving," 2021, [Online]. Available: http://arxiv.org/abs/2106.12639
[52] Bharadwaj, K. B. Prakash, and G. R. Kanagachidambaresan, *Pattern Recognition and Machine Learning*. 2021. https://doi.org/10.1007/978-3-030-57077-4_11
[53] E. Talbi, "[PDF] Metaheuristics - From Design to Implementation | Semantic Scholar." https://www.semanticscholar.org/paper/Metaheuristics-From-Design-to-Implementation-Talbi/d7d0feaf0c3668182f7e724b0734846f66cc3a7d (accessed May 25, 2022).
[54] Q. Wan, M. J. Weng, and S. Liu, "Optimization of wireless sensor networks based on differential evolution algorithm," *Int. J. online Biomed. Eng.*, vol. 15, no. 1, pp. 183–195, 2019, https://doi.org/10.3991/ijoe.v15i01.9786
[55] P. R. Lorenzo, J. Nalepa, M. Kawulok, L. S. Ramos, and J. R. Pastor, "Particle swarm optimization for hyper-parameter selection in deep neural networks," *GECCO 2017 - Proc. 2017 Genet. Evol. Comput. Conf.*, pp. 481–488, 2017, https://doi.org/10.1145/3071178.3071208
[56] S. Lessmann, R. Stahlbock, and S. F. Crone, "Optimizing hyperparameters of support vector machines by genetic algorithms," *Proc. 2005 Int. Conf. Artif. Intell. ICAI'05*, vol. 1, pp. 74–80, 2005.
[57] U. Seiffert, "Multiple layer perceptron training using genetic algorithms," *Eur. Symp. Artif. Neural Networks*, pp. 159–164, 2001, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.20.6859&rep=rep1&type=pdf
[58] J. Shi, M. Habib, and H. Yan, "A review paper on different application of genetic algorithm for mobile ad-hoc network (MANET)," *Int. J. online Biomed. Eng.*, vol. 16, no. 5, pp. 119–139, 2020, https://doi.org/10.3991/ijoe.v16i05.13325

[59] Sehla Loussaief and Afef Abdelkrim, "Convolutional neural network hyper-parameters optimization based on genetic algorithms." https://thesai.org/Publications/ViewPaper?Volume=9&Issue=10&Code=ijacsa&SerialNo=31 (accessed May 25, 2022). https://doi.org/10.14569/IJACSA.2018.091031

[60] M. A. Hannan *et al.*, "Role of optimization algorithms based fuzzy controller in achieving induction motor performance enhancement," *Nat. Commun.*, vol. 11, no. 1, pp. 1–11, 2020, https://doi.org/10.1038/s41467-020-17623-5

[61] M. A. Hannan, J. A. Ali, A. Mohamed, and A. Hussain, "Optimization techniques to enhance the performance of induction motor drives: A review," *Renew. Sustain. Energy Rev.*, vol. 81, pp. 1611–1626, Jan. 2018, https://doi.org/10.1016/j.rser.2017.05.240

[62] James Kennedy, "(PDF) Particle swarm optimization | James Kennedy - Academia.edu." https://www.academia.edu/1446115/Particle_swarm_optimization (accessed May 26, 2022).

[63] K. Miao, Q. Feng, and W. Kuang, "electronics article," 2021, https://doi.org/10.3390/electronics10050597

[64] A. D. Rosli, N. S. Adenan, H. Hashim, N. E. Abdullah, S. Sulaiman, and R. Baharudin, "Application of particle swarm optimization algorithm for optimizing ANN model in recognizing ripeness of citrus," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 340, no. 1, 2018, https://doi.org/10.1088/1757-899X/340/1/012015

[65] M. S. Ahmed, A. Mohamed, R. Z. Homod, and H. Shareef, "Hybrid LSA-ANN based home energy management scheduling controller for residential demand response strategy," *Energies*, vol. 9, no. 9, 2016, https://doi.org/10.3390/en9090716

[66] T. Zhang *et al.*, "Efficient spectrum prediction and inverse design for plasmonic waveguide systems based on artificial neural networks," *Photonics Res.*, vol. 7, no. 3, p. 368, 2019, https://doi.org/10.1364/PRJ.7.000368

[67] W. Shi, D. Liu, X. Cheng, Y. Li, and Y. Zhao, "Particle swarm optimization-based deep neural network for digital modulation recognition," *IEEE Access*, vol. 7, pp. 104591–104600, 2019, https://doi.org/10.1109/ACCESS.2019.2932266

[68] J. F. Torres, D. Gutiérrez-Avilés, A. Troncoso, and F. Martínez-Álvarez, "Random hyperparameter search-based deep neural network for power consumption forecasting," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11506 LNCS, pp. 259–269, 2019, https://doi.org/10.1007/978-3-030-20521-8_22

[69] V. Krasteva, S. Ménétré, J. P. Didon, and I. Jekova, "Fully convolutional deep neural networks with optimized hyperparameters for detection of shockable and non-shockable rhythms," *Sensors (Switzerland)*, vol. 20, no. 10, 2020, https://doi.org/10.3390/s20102875

[70] T. Kim, J. Lee, and Y. Choe, "Bayesian optimization-based global optimal rank selection for compression of convolutional neural networks," *IEEE Access*, vol. 8, pp. 17605–17618, 2020, https://doi.org/10.1109/ACCESS.2020.2968357

[71] G. Zhang, F. Tan, and Y. Wu, "Ship motion attitude prediction based on an adaptive dynamic particle swarm optimization algorithm and bidirectional LSTM neural network," *IEEE Access*, vol. 8, pp. 90087–90098, 2020, https://doi.org/10.1109/ACCESS.2020.2993909

[72] X. Xiao, M. Yan, S. Basodi, C. Ji, and Y. Pan, "Efficient hyperparameter optimization in deep learning using a variable length genetic algorithm," Jun. 2020, [Online]. Available: http://arxiv.org/abs/2006.12703

[73] H. Harafani, I. Suryani, Ispandi, and N. Lutfiyana, "Neural network parameters optimization with genetic algorithm to improve liver disease estimation," *J. Phys. Conf. Ser.*, vol. 1641, no. 1, 2020, https://doi.org/10.1088/1742-6596/1641/1/012034

[74] R. Andonie and A. C. Florea, "Weighted random search for CNN hyperparameter optimization," *Int. J. Comput. Commun. Control*, vol. 15, no. 2, pp. 1–11, 2020, https://doi.org/10.15837/ijccc.2020.2.3868

[75] M. Abdolrasol, R. Mohamed, M. Hannan, A. Al-Shetwi, M. Mansor, and F. Blaabjerg, "Artificial neural network based particle swarm optimization for microgrid optimal energy scheduling," *IEEE Trans. Power Electron.*, vol. 36, no. 11, pp. 12151–12157, 2021, https://doi.org/10.1109/TPEL.2021.3074964

[76] M. Miani *et al.*, "Bituminous mixtures experimental data modeling using a hyperparameters-optimized machine learning approach," *Appl. Sci.*, vol. 11, no. 24, 2021, https://doi.org/10.3390/app112411710

[77] P. Kumar, S. Batra, and B. Raman, "Deep neural network hyper-parameter tuning through twofold genetic approach," *Soft Comput.*, vol. 25, no. 13, pp. 8747–8771, 2021, https://doi.org/10.1007/s00500-021-05770-w

[78] J. Kalliola, J. Kapočiūte-Dzikiene, and R. Damaševičius, "Neural network hyperparameter optimization for prediction of real estate prices in Helsinki," *PeerJ Comput. Sci.*, vol. 7, pp. 1–25, 2021, https://doi.org/10.7717/peerj-cs.444

[79] M. Gupta, K. Rajnish, and V. Bhattacharjee, "Impact of parameter tuning for optimizing deep neural network models for predicting software faults," *Sci. Program.*, vol. 2021, 2021, https://doi.org/10.1155/2021/6662932

[80] A. Menapace, A. Zanfei, and M. Righetti, "Tuning ann hyperparameters for forecasting drinking water demand," *Appl. Sci.*, vol. 11, no. 9, 2021, https://doi.org/10.3390/app11094290

[81] S. Blume, T. Benedens, and D. Schramm, "Hyperparameter optimization techniques for designing software sensors based on artificial neural networks," *Sensors*, vol. 21, no. 24, 2021, https://doi.org/10.3390/s21248435

[82] P. Kaur, A. Singh, and I. Chana, "BSense: A parallel Bayesian hyperparameter optimized Stacked ensemble model for breast cancer survival prediction," *J. Comput. Sci.*, vol. 60, p. 101570, 2022, https://doi.org/10.1016/j.jocs.2022.101570

[83] W. Pannakkong, K. Thiwa-Anont, K. Singthong, P. Parthanadee, and J. Buddhakulsomsiri, "Hyperparameter tuning of machine learning algorithms using response surface methodology: A case study of ANN, SVM, and DBN," *Math. Probl. Eng.*, vol. 2022, pp. 1–17, 2022, https://doi.org/10.1155/2022/8513719

[84] M. R. Sarker, R. Mohamed, M. H. M. Saad, and A. Mohamed, "DSPACE controller-based enhanced piezoelectric energy harvesting system using PI-lightning search algorithm," *IEEE Access*, vol. 7, pp. 3610–3626, 2019, https://doi.org/10.1109/ACCESS.2018.2888912

[85] P. Civicioglu, "Backtracking search optimization algorithm for numerical optimization problems," *Appl. Math. Comput.*, vol. 219, no. 15, pp. 8121–8144, Apr. 2013, https://doi.org/10.1016/j.amc.2013.02.017

[86] D. Chen, F. Zou, R. Lu, and S. Li, "Backtracking search optimization algorithm based on knowledge learning," *Inf. Sci. (Ny).*, vol. 473, pp. 202–226, Jan. 2019, https://doi.org/10.1016/j.ins.2018.09.039

[87] I. Loshchilov and F. Hutter, "CMA-ES for Hyperparameter optimization of deep neural networks," 2016, [Online]. Available: http://arxiv.org/abs/1604.07269

[88] Y. Ozaki, M. Yano, and M. Onishi, "Effective hyperparameter optimization using Nelder-Mead method in deep learning," *IPSJ Trans. Comput. Vis. Appl.*, vol. 9, 2017, https://doi.org/10.1186/s41074-017-0030-7

[89] H. Cui and J. Bai, "A new hyperparameters optimization method for convolutional neural networks," *Pattern Recognit. Lett.*, vol. 125, pp. 828–834, 2019, https://doi.org/10.1016/j.patrec.2019.02.009

[90] A. Gulcu and Z. Kus, "Hyper-parameter selection in convolutional neural networks using microcanonical optimization algorithm," *IEEE Access*, vol. 8, pp. 52528–52540, 2020, https://doi.org/10.1109/ACCESS.2020.2981141

[91] Y. J. Yoo, "Hyperparameter optimization of deep neural network using univariate dynamic encoding algorithm for searches," *Knowledge-Based Syst.*, vol. 178, pp. 74–83, 2019, https://doi.org/10.1016/j.knosys.2019.04.019

[92] Y. Peng, D. Gong, C. Deng, H. Li, H. Cai, and H. Zhang, "An automatic hyperparameter optimization DNN model for precipitation prediction," *Appl. Intell.*, vol. 52, no. 3, pp. 2703–2719, 2022, https://doi.org/10.1007/s10489-021-02507-y

[93] S. Ding, X. Xu, H. Zhu, J. Wang, and F. Jin, "Studies on optimization algorithms for some artificial neural networks based on genetic algorithm (GA)," *J. Comput.*, vol. 6, no. 5, pp. 939–946, 2011, https://doi.org/10.4304/jcp.6.5.939-946

[94] T. Yamasaki, T. Honma, and K. Aizawa, "Efficient optimization of convolutional neural networks using particle swarm optimization," *Proc. - 2017 IEEE 3rd Int. Conf. Multimed. Big Data, BigMM 2017*, pp. 70–73, 2017, https://doi.org/10.1109/BigMM.2017.69

[95] N. M. Aszemi and P. D. D. Dominic, "Hyperparameter optimization in convolutional neural network using genetic algorithms," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 6, pp. 269–278, 2019, https://doi.org/10.14569/IJACSA.2019.0100638

[96] X. He, K. Zhao, and X. Chu, "AutoML: A survey of the state-of-the-art," *Knowledge-Based Syst.*, vol. 212, no. Dl, 2021, https://doi.org/10.1016/j.knosys.2020.106622

[97] N. Decastro-García, Á. L. Muñoz Castañeda, D. Escudero García, and M. V. Carriegos, "Effect of the sampling of a dataset in the hyperparameter optimization phase over the efficiency of a machine learning algorithm," *Complexity*, vol. 2019, 2019, https://doi.org/10.1155/2019/6278908

[98] V. O. Balabanov and G. Venter, "Multi-fidelity optimization with high-fidelity analysis and low-fidelity gradients," *Collect. Tech. Pap. - 10th AIAA/ISSMO Multidiscip. Anal. Optim. Conf.*, vol. 3, pp. 1777–1788, 2004, https://doi.org/10.2514/6.2004-4459

[99] C. Li, "Biodiversity assessment based on artificial intelligence and neural network algorithms," *Microprocess. Microsyst.*, vol. 79, p. 103321, Nov. 2020, https://doi.org/10.1016/j.micpro.2020.103321

[100] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, 2020, https://doi.org/10.1016/j.neucom.2020.07.061

[101] E. Hazan, A. Klivans, and Y. Yuan, "Hyperparameter optimization: A spectral approach," *6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc.*, 2018.

## 7 Authors

**Zahraa Saddi Kadhim:** she obtained her B.Sc. in computer science from University of Technology, Iraq in 2015. Currently studying M.Sc.in computer science. she worked as a trainer for a non-governmental group teaching young people how to code robots in 2017–2019 her interested is mobile application, Artificial Intelligence, Machine Learning, Pattern Recognition, Data encryption and robotic (email: cs.20.44@grad.uotechnology.edu.iq).

**Hasanen S. Abdullah:** Assist Professor qualified to direct research at University of Technology, Iraq, and other Iraqi Institutions. He got his B.Sc., M.Sc. and Ph.D. in computer science from University of Technology, Iraq in 2000, 2004 and 2008 respectively. His area of interests is Artificial Intelligence, Machine Learning, Swarm Intelligence, Pattern Recognition, Data Mining & warehouse, and Business Intelligence (email: Hasanen.S.Abdullah@uotechnology.edu.iq).

**Khalil Ibrahim Ghathwan** PhD/M.Sc./HD (AI)/B.S./Dip Senior University of Technology – Iraq Computer sciences. His Skills Computer Science Education Computer Programming NET technology, research interest is broadly in Science in Computing (Artificial Intelligence (AI) /Machine Learning (ML)), Computer Network, Security and data mining algorithm. This includes Routing, optimization and swarm intelligence. Currently, he involved in several projects relating to artificial Intelligence and routing algorithm (email: Khalil.i.ghathwan@uotechnology.edu.iq).

Article submitted 2022-08-02. Resubmitted 2022-09-19. Final acceptance 2022-09-20. Final version published as submitted by the authors.