Received: November 5, 2020. Revised: December 2, 2024.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Development of an Intelligent System Based on Deep Neural Network Models with Advanced Algorithms for Hyper-parameter Tuning and Weight Updates Across Diverse Datasets

Noor Abdul Khaleq Zghair[1]* **Abubakr S. Issa[2]**

[1]*Computer Engineering Department, University of Technology- Iraq, Baghdad, Iraq*
[2]*Information Technology Center, University of Technology- Iraq, Baghdad, Iraq*
* Corresponding author's Email: Noor.A.Zghair@uotechnology.edu.iq

Abstract: This work seeks to understand how deep neural networks can be improved in terms of hyperparameters and weights on different datasets with the help of intelligent system. Regarding hyperparameters and weights optimization, the paper employs Convolutional Neural Networks (CNN) and Multi-Layer Perceptron (MLP). First, the paper compares the accuracy of the convolutional neural network and multilayer perceptron using Adam and RMS prop as optimizers. Next, propose to use the CNN model fine-tuning with Adam and RMSprop optimization algorithms, but instead of setting fixed hyperparameters, the paper uses Simulated Annealing (SA) for optimization and Differential Evolution (DE) for weight updates. Further, it examines the accuracy of the CNN models trained by the Eagle Strategy-Based Optimization (ESBO) on the MNIST database when the Differential Evolution algorithm updates the weights. This approach is then used on the CIFAR dataset. In the proposed approach, all these steps are included in detail. Apart from these image datasets, the optimization strategies applied in this work include the electric load diagrams of the years 2011-2014, air quality, and cityscapes. In terms of performance, this work also measures with 'conventional' metrics, with the MNIST and CIFAR-10 benchmarks scoring 99.5% and 89.1%, respectively, 1-2% better than prior methods. The latter approach bears higher computational costs, but this is warranted by higher accuracy of predictions and a better generalization of the model. Forecast of electricity load was verified from the time series data which also supported the proposed methods with an accuracy of 96.3%. Therefore, the results of the impact assessment indicate the relative effectiveness of different optimization algorithms and amass irrefutable evidence of the effectiveness of the proposed methods and approaches. In the study of the current sample, procedures and findings are backed by flowcharts and summaries to ensure that understanding of the conducted optimization procedures is well enhanced. The results are useful to enhance deep learning models using higher optimization methods.

Keywords: Deep learning, Convolutional neural networks (CNN), Multi-layer perceptron (MLP), Hyperparameter optimization, Eagle strategy-based optimization (ESBO).

1. Introduction

One of the major fields of Artificial Intelligence (AI) currently is machine learning, specifically deep learning, which has gained immense importance due to its ability to augment how models can learn and make decisions from large datasets. Neural networks, which are a category of deep learning models, are formed of multiple layers interconnected through nodes, or neurons [1, 2]. Such models perform well in activities like image, speech recognition, natural language understanding and processing, and the game AI, among others. The following are specific reasons why deep learning has achieved tremendous success: First, neural networks are capable of learning features from raw data, which eliminates the need for hard coding [3, 4].

Recent years have seen an increased usage of deep learning models such as CNNs and MLPs to achieve high levels of accuracy [5]. However, the performance of these models tends to be dependent on several hyperparameters, as well as the proper tuning of model weights. Such training parameters as learning rate and batch size, as well as the

architecture of the neural network, define the process and the model's accuracy [6]. Weight optimization, however, focuses on the required optimum representation of the data by the model. Some of the earlier optimization approaches that are frequently applied while training deep neural networks include Adam and RMSprop. These algorithms change the weight of the network constantly to optimize the loss function. While these approaches work well, they do not guarantee the global optimum solution for model training because of the underlying non-convexity of the objective function in deep learning models.

To overcome this challenge, new methods for hyperparameters' optimization and weight update have been designed. These state-of-the-art algorithms for model optimization in this research are applied across these datasets in deep neural network models, as shown in Fig. 1 [7, 8]. Use CNN and MLP models and apply different optimization methods that will improve hyperparameters and weights. To this end, this study starts by setting the platforms for the CNN and MLP models by applying Adam and RMSprop optimizers. Then proceed with performing Simulated Annealing to optimize hyperparameters and Differential Evolution of CNN weight parameters. Moreover, we evaluate the proposed CNN models, trained with the ESBO on the MNIST dataset, and employ the DE for the weights' update. This comprehensive approach is then used in analyzing the CIFAR dataset. Besides image datasets, our optimization strategies are applied to other areas, such as ELD 2011-2014, air quality, and cityscape datasets. Our results show the comparative performance of the various optimization techniques and establish the effectiveness of the developed methods. To enhance the understanding of the process, the flow diagrams and the summaries of the steps taken are included. The study provides an important contribution to further improvements of deep learning models by applying a superior level of optimization. The field of deep learning is applied to image logistics, language comprehension, and prediction because it involves learning from unprocessed data [1, 2]. The proposed study intends to enhance the performance of CNNs and MLPs via new methods of hyperparameters' tuning, and weight modifications. More attention is paid to increasing the accuracy and speed of computations of a model with the help of ESBO and DE for multiple and heterogeneous data sets. The organization of this paper is as follows: Section 2 focuses on related work and demonstrates how existing strategies suffer from drawbacks; Section 3 describes research methodology and the mathematical formulation;

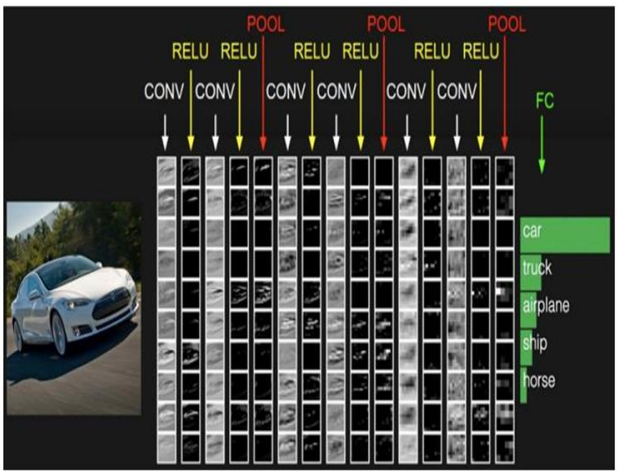


Figure. 1 Architectures of VGGNet [6]

Section 4 explains datasets and program settings; Section 5 discusses results and further analysis; Section 6 summarises contributions and potential development paths.

2. Literature review

The popularity of Adam and RMSprop optimizers, [3] still remains ineffective in finding global optima due to the non-differentiability of loss functions. Other related current research works [9-12] look at metaheuristic techniques for solving optimization problems, but none of them presents the exploitative and explorative mechanisms of ESBO. Further, differential evolution has been seen to perform well in complicated non-linear optimization problems although its combination with exploration strategies as ESBO is indefinite. The recent trend in hyperparameters offering optimization in deep learning models has proved to make these models both efficient and accurate. Algorithms which incorporate some of the methods like Genetic Algorithm, Bayesian Optimization, and Swarm Intelligence have been studied in detail due to their capability to handle large search spaces and enhance the model accuracy. Various works highlight the need to conduct hyperparameter tuning methodologically to obtain better results in a wide range of applications, including neural network training, specialized industrial applications, cybersecurity, etc.

In [9] proposed a twofold genetic approach to hyperparameter tuning for deep neural networks. The authors also introduce a technique that integrates the conventional genetic algorithms with a second phase of optimization and greatly improves the quality of the models associated with the neural networks. With a focus on the key hyperparameters and optimizing them systematically, the effectiveness of the pro-

posed approach that is mentioned above is shown to be better across different sets of data.

In [10] used the hyperparameter optimization algorithms and their usage in deep learning. This paper proposes the separation of methods into the gradient-based, Bayesian, evolutionary and other advanced other optimization methods. Authors provide information regarding the strengths and weaknesses of individual approaches focusing on the fact that they can be applied to different sorts of neural networks and data sets. Due to its respective and comprehensive presentation of the current status of hyperparameter optimization surveys, the review proves to be helpful for researchers and practitioners keen on improving the performance of deep learning models through either search methods or metaheuristic approaches.

In [11] concerned with a review of improved metaheuristic optimization approaches and deployment in deep neural networks. The authors reviewed various other algorithms including genetic algorithms, particle swarm optimization, and ant colony optimization, in which the authors proved that they are efficient in hyperparameter tuning. It looks at how these metaheuristics can be incorporated with deep learning environments and presents some example applications of these heuristics.

In [12] gave a vast overview of metaheuristic algorithms for improving deep learning models. The paper also describes various techniques such as, swarm intelligence, evolutionary algorithms and combined techniques of both, in the training of neural networks. The authors also begin to dissect the strengths as well as the weaknesses of these approaches for various deep-learning applications.

In [13], the particle swarm optimization (PSO) used for hyperparameters' tuning in deep neural networks is presented. To overcome this problem, the authors put for-ward a PSO-based framework that enhances the tuning process and realizes better parameter establishment. The study also proves that PSO is capable of the complexity of deep learning models, especially in search space cases.

In [14] presented Deephyper; an asynchronous hyperparameter search best suited to deep neural networks. Deephyper applies parallel computing and advanced search algorithms for searching hyperparameter space in targeted areas which makes the selection process faster and less execution globally. The presented framework works well in high-performance computing contexts which allows to perform optimization at a large scale and without limitations.

The paper [15] introduced scalable Bayesian optimization (BO) for use in hyperparameter tuning of deep neural networks. Instead, the authors consider a method that merges BO with deep learning making the constant search of large and complex spaces possible. The approach is aimed at resolving high-dimensional parameter tuning issues and can serve as a scalable solution for the Deep Learning algorithm. Such an approach demonstrates the strength of using Bayesian optimization in producing high-performing solutions which makes it a robust method of tuning hyperparameters in neural networks.

In [16] the tuned deep neural network models for software fault prediction. The authors also stress the need for increased detail regarding parameters since they allow for enhancing the models' performance in predicting faults. This is seen from the various optimization techniques employed in the study as seen in Fig. 2 depicting the improvement of the models.

In [17] presented a refined Adam method for the optimization of deep learning neural networks. The authors propose optimizations of the current Adam optimizer, as they find that the algorithm's convergence and stability are flawed. Compared to the original algorithm, we have enhanced them by the adjustment of learning rate and acceleration of the training process. As for the experimentations on different datasets, the study makes a clear indication that the new modified Adam algorithm is better placed to produce improvements on conventional optimization compared to existing deep learning model training.

In [18] the authors used industrial ADME datasets to discuss tunable hyperparameters for deep neural networks. The topics of interest in this review are the determination of the best values of parameters that will improve the predictive power of the neural networks in describing pharmacokinetics. Altogether, the research performs hyperparameter tuning and shows enhanced accuracy and better generalization across the board. This work emphasizes the necessity of the creep hyperparameter tuning for using deep learning techniques for purposeful industrial database applications, specifically in the pharmaceutical industry.

In [19] developed an intelligent load forecasting system using an evolutionary-based deep convolutional neural network model. The authors work with hyperparameters of the neural network by using evolutionary algorithms to improve the accuracy of its forecast to the electrical load demands. One of the important conclusions is the prospects for the use of deep learning in combination with evolutionary strategies, which was illustrated by the improved accuracy of the forecasts. Evolutionary

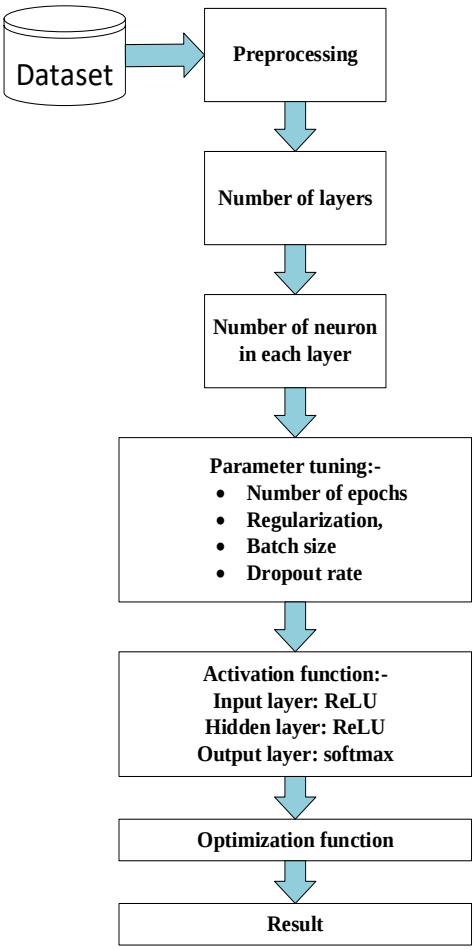


Figure. 2 Impact of Parameter Tuning [16]

algorithms are explored in this research as having the possibility of improving neural net-works for use in industries.

In [20] studied the hyperparameters of deep neural networks and studies the differences in their performance with shallow methods in the modelling of bioactivity data. In their work, the authors spend considerable efforts running various experiments investigating the dependency of the model performance and respective computational cost on the hyperparameters selected. The findings also show that with proper tuning, deep learning models offer better performance than the traditional shallow models.

The paper [21] suggested the use of revised swarm intelligence metaheuristics for tuning the hyperparameter of the convolutional neural network (CNN). It also presents more complex techniques of updated algorithms enhancing the general search for improved hyperparameter tuning for model efficiency and enhanced accuracy. The research proves that such techniques when applied to CNNs, enhance performance by a vast margin. The usefulness of swarm intelligence in handling the challenges associated with hyperparameters in deep learning models especially in image processing is well captured in this work.

In the paper [22] the author suggested an efficient approach to training deep neural networks. To enhance the convergence speed and accuracy of the neural net-work models, the authors have presented a new optimization technique, which incorporates gradient information with the concept of adaptive learning. Describing the use of the proposed method for various deep-learning tasks, the study proves the potential of using the method to improve the efficiency of the training phase.

In [23] it focused on genetic algorithms in purpose to optimize neural network's hyperparameters. The authors present a framework of genetic algorithm that enables an efficient search on hyperparameters, optimizing model performances and training times. As illustrated from experiments on various databases, the presence of genetic algorithms provides benefits, especially in the case of addressing inherent search spaces and attaining optimal solutions.

In the article [24], explained the idea of Automatic weight parameter selection using particle swarm optimization for deep neural networks for large-scale data analysis with high dimensional data. To solve the given problem, the author recommends the application of the PSO-based framework that can recognize the most appropriate parameters for improving the overall performance of a model. The study shows that PSO successfully manages intricate requirements of the deep learning models, especially in situations where large search spaces are in use.

The authors in [25] elaborated different and improved hyperparameters tuning methods within the view of deep learning for wind power forecast. The work also uses the best optimization techniques in tuning neural network coefficients to enhance the predictive aspect as well as reliability. Applying these techniques to renewable energy datasets, the work shows the effectiveness of hyperparameter optimization for im-proving deep learning models' performance in energy forecasting applications.

In [26], the paper proposed a technique for enhancing the hyperparameter optimization by utilizing the model weights' checkpointing. The authors suggest the approach that uses the information in the middle of the training process for optimization purposes, which helps to save time and increase the speed. Checkpointing is shown as a fastening approach to hyperparameter tuning and is demonstrated with a working example in the study of the deep-learning model.

In [27] The authors considered the effect of hyperparameter tuning on the accuracy of fine-tuned

CNN models. The hyperparameters are integral in the improvement of the performance of image classifications as indicated in the study. Theoretical results show the manifestation of higher precision and generalization when data sets are tested systematically; thus, substantiating the research's findings.

In [28] the author focused on the use of hyperparameter optimization of CNN through the genetic algorithm in crop pests' classification. Also recommends on genetic algorithm to search the space of parameters efficiently to achieve better accuracy and performance on the model being used. Using this approach on agricultural datasets, the research proves that there are improvements in the classification. This work paints the picture of how future evolutionary strategies can be applied in enhancing deep learning in specialized contexts of agriculture.

In [29], introduced a hybrid success history intelligent optimizer with Gaussian transformation for the hyperparameter of CNN. Introducing a new, improved optimization algorithm is the key contribution of the study, which makes use of historical success info and Gaussian transforms to improve the current state of the search. By conducting experiments on several datasets, the study proves how the method pro-posed works to enhance the CNN. Among them, Scalable nested optimization for deep learning is discussed in [30], the author has put forward a manner of optimization which involves loops of optimization whereby hyperparameters can be stretched more efficiently. Thus, the idea of organizing the model and optimizing training with the help of this approach is proven in this study. This research proves that calibrating nested optimization for highly challenging deep learning applications lets the work demonstrate how the tool can be used to improve the scalability and efficiency of neural networks.

In [31] the authors proposed an optimization-based deep learning technique to identify intrusion and attack on network systems. This research uses novel computational optimization algorithms as methods to fine-tune the deep learning models for increasing the accuracy of network intrusion detection. The improvement in detection performance is evident from the research done that proposes the integration of deep learning with optimization strategies.

3. Methodology

The approach of the current work is designed to improve deep learning models, incrementally, through numerous optimization techniques. It commences with the choice of proper models and then it succeeds in baseline training with conventional optimizers apart from the final way of hyperparameter optimization for enhanced results.

3.1 Model selection

3.1.1. Convolutional neural networks (CNNs)

- Architecture: CNNs are constructed with a convolution layer in which filters are applied to the input data to produce feature maps that obtain spatial hierarchies.
- Advantages: These are good for image recognition since they allow for auto-mated learning of spatial hierarchy and features.
- Layers: These normally include convolutional layers, pooling layers, fully connected layers and dropout layers that help in mitigating overfitting. The summary of the CNN architecture as shown in Table 1.

3.1.2. Multi-layer perceptron (MLPs)

- Architecture: An MLP computational model is composed of an input layer or layer of input neurons as well as one or several hidden layers of neurons followed by an output layer that is connected to all neurons from the subsequent layer.

- Advantages: It is best used for high quantities of organized data and activities which do not necessarily involve mapping between inputs and outputs.

- Activation Functions: For the hidden layers, please use ReLU (Rectified Linear Unit) while for the output layer use SoftMax in classification problems, the architecture of the MLP is shown in Table 1.

3.2 Baseline model training

3.2.1. Adam optimizer

- Algorithm: Ada Moment Estimation or Adam is a new method of finding optimal parameters of the model and it is the combined effect of two other variations of Stochastic gradient descent: AdaGrad and RMSProp.
- Parameters: It is better when using a learning rate of 0. 001, $\beta_1 = 0.9$, and $\beta_2 = 0.999$.
- Advantages: Accurate, fast and suitable for problems of large size data and large parameters.

3.2.2. RMSprop optimizer

- Algorithm: RMSprop takes into account the moving average of the gradients of the parameters for the reciprocal of the gradient.

- Parameters: It has been found to use a learning rate of 0, in most of its cases. Under conditions: H_0 = 0,05; p < 0,05; Thanhtieldelta = 1; theta 3+4 = 0,80 eta 5 = 0,80; kappa = 5; alpha = 0,001 and ρ = 0. 9.
- Advantages: Avoids the learning rate from decaying too soon hence useful when addressing non-stationary inputs or targets.

3.3 Advanced hyperparameter tuning

3.3.1. Eagle strategy-based optimization (ESBO)

Initialization:
ESBO Initialization: the initial population of candidate solutions is generated randomly.

$$X_i^0 = X_{min} + rand\ ().(X_{max} - X_{min}) \quad (1)$$

Where i=1, 2, …, Ni=1,2,…,N represents the population size. This equation initializes the starting positions of the candidate solutions within a predefined range. Where, X_i^0 is Initial position of the i^{th} candidate solution. While, X_{min}, $X_{max}$are Minimum and maximum bounds of the search space. Rand () is A random value between 0 and 1.

Soaring Phase (Global Exploration):

In this phase, eagles explore the search space broadly to identify promising regions. This represents the global exploration phase in the ESBO, where candidate solutions are updated to explore new areas in the search space:

$$X_i^{t+1} = X_i^t + \propto^t.\left[\beta.\left(X_g^t - X_i^t\right) + \gamma.rand.V_i^t\right] \quad (2)$$

Where:
$X_i^{t+1}$is updated position of the ith candidate solution at iteration t+1, $X_g^t$is best solution found at iteration t, $\propto^t$ is the step size parameter, reduced over iterations to balance exploration and exploitation, β,γ is weighting factors for exploitation and exploration, respectively, and V_i^t is velocity vector influencing the position update.

Diving Phase (Local Exploitation):

Once promising regions are identified, the algorithm enters the diving phase for local exploitation, refining the search.

$$X_i^{t+1} = \begin{cases} X_i^t + \delta^t.\left(X_g^t - X_i^t\right) & if\ f\left(X_i^t\right) < f(X_g^t) \\ X_i^t + \lambda.\left(X_I^t - X_i^t\right) & otherwise \end{cases} \quad (3)$$

Where:
δ^t is a small step size focused on refining the solution, λ a parameter controlling local search direction, X_I^t is a locally optimal solution near X_i^t, and f(X) is objective function value for a given solution X.

Dynamic Adjustment:

The search strategy is dynamically adjusted based on the success of the current search attempts. This helps in focusing on more promising regions as the search progresses. The parameters α, δ, β, and λ are adjusted adaptively to balance exploration and exploitation.

3.3.2. Integration with differential evolution (DE)

1. Initial Optimization:
- Apply ESBO to optimize the initial set of model parameters, focusing on finding a good starting point for weights.

$$X_i^* = ESBO(X_i)$$

2. Refinement with DE:
- This is the Differential Evolution (DE) update formula, which modifies solutions using weighted differences between randomly selected solutions.

$$X_i^{t+1} = X_r^1 + F.(X_r^2 - X_r^3) \quad (4)$$

Where ($X_r^1, X_r^2, and\ X_r^3$) are randomly chosen solutions, and F is the differential weight.
The procedure of the proposed model is shown in Fig. 3.

4. Result and discussion

Before embarking on the analysis of the performances of the experimental algorithms, one needs to understand the sets of data used in this study to support the model evaluation. This is an added advantage for the proposed models as it guarantees their performance on various types of data such as image data, time series and environmental data. All the used datasets are chosen according to the purposes of the specific tasks which include image classification, energy consumption forecast, or environmental supervision. We used both the standard datasets like MNIST and CIFAR-10 and domain-specific datasets like the Electricity Load Diagrams and Air Quality datasets to test the models' generalizations.

Furthermore, for the same purpose, the Cityscapes dataset is also incorporated to verify how well a model does for the highly perceptive complex tasks that both necessitate scene analysis and semantic

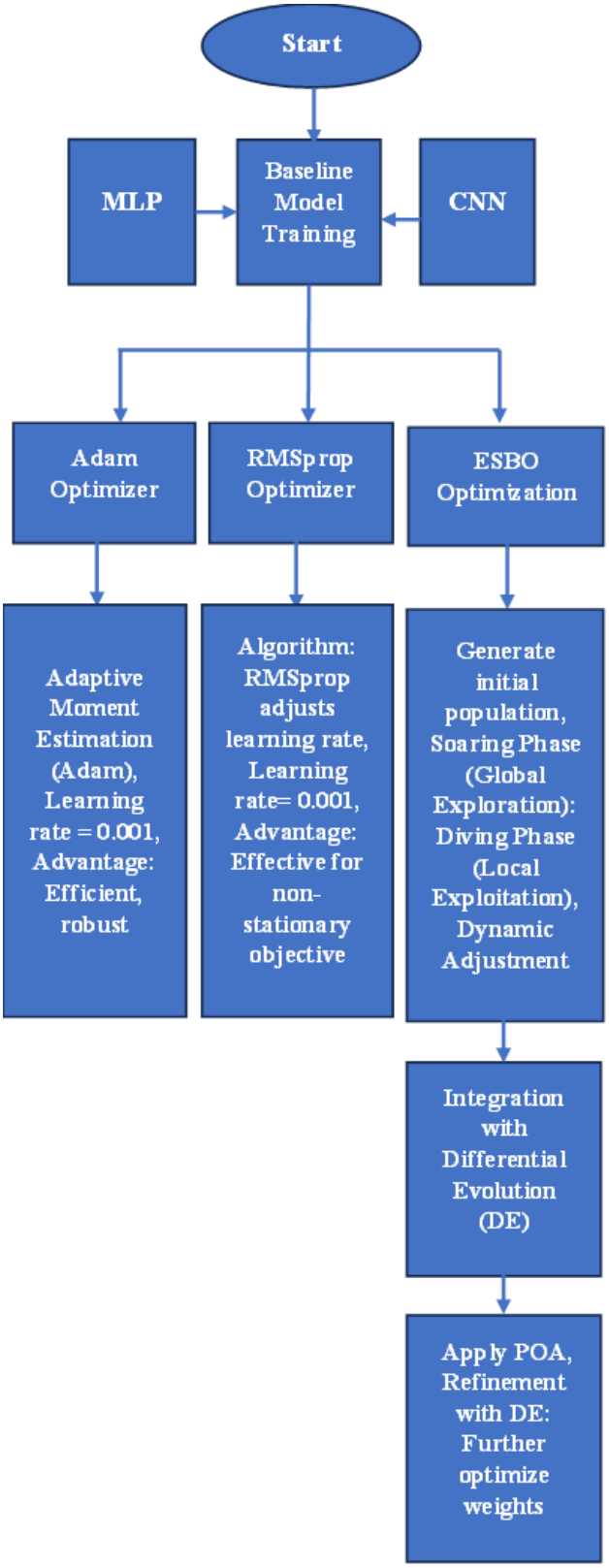


Figure. 3 Procedure of the proposed intelligent system

segmentation. Besides, the integration of these datasets provides a strong basis for the evaluation of the models and also demonstrates the applicability and flexibility of the discussed optimization methods in different practical problems.

1. MNIST Dataset: Consists of H seventy thousand grayscale images depicting hand-written digits from 0 to 9 and with 60 thousand samples have been utilized for the formation of the training data set and the rest of 10 thousand for the formation of the test data set. First bibliography for testing and validating our model architectures and optimization algorithms.

2. CIFAR-10 Dataset: Has 60,000 32×32 colour images divided into 10 classes with total of 50,000 images for training and 10,000 images for validation. Tests out the model on additional data or superior quality and diverse image data.

3. Electricity Load Diagrams 2011-2014: A folder consisting of time series data of electricity load measurements used to predict future electricity load demands. Challenges the models' performance when dealing with time series data as well as to forecast energy use.

4. Air Quality Dataset: Includes data on various air borne pollutants that are used to estimate the air quality in any given region based on past records. Checks different models on environmental data which are time series and numerical attributes.

5. Cityscapes Dataset: Contains pictures of streets in urban areas that have been pixel-wise annotated for the task of semantic segmentation. Validates models on complicated tasks which include scene analysis and segmentation.

Before getting down to the differences between CNN and MLP, it is necessary to have a clear perception as to what distinguishes these two categories of NNs. CNNs are inherently designed for grid-like structures such as images. This is done using convolutional layers which stamp filters across the input data making feature maps helpful in distinguishing hierarchy in the data. This makes CNNs especially useful in image recognition since spatial relations between objects are often important. Most of the architecture consists of convolutional layers, pooling layers as a means of dimensionality reduction, and fully connected layers, which make the final classification. Dropout layers are also applied to reduce overfitting and during the training phase some of the neurons are omitted randomly the summary of the architecture is given in Table 1. On the other hand, MLPs are designed with a view of a more generalized model of using it. It has an input layer, one or more layers of hidden neurons and an output layer where-by every neuron in each layer is connected to every neuron in the next subsequent layer. The fully connected structure enables MLPs to capture the

Table 1. Summary of the Deep learning architecture CNN and MLP

Aspect	Convolutional Neural Network (CNN)	Multi-Layer Perceptron (MLP)
Model Structure		
Input Layer	Shape: (32, 32, 3) (e.g., CIFAR-10)	Shape: 784 (e.g., MNIST)
Convolutional Layer 1	32 filters, kernel size (3, 3), activation='relu'	-
Convolutional Layer 2	64 filters, kernel size (3, 3), activation='relu'	-
Max Pooling Layer 1	Pool size (2, 2)	-
Dropout Layer 1	Rate = 0.25	-
Convolutional Layer 3	128 filters, kernel size (3, 3), activation='relu'	-
Convolutional Layer 4	256 filters, kernel size (3, 3), activation='relu'	-
Max Pooling Layer 2	Pool size (2, 2)	-
Dropout Layer 2	Rate = 0.25	-
Flatten Layer	Flattens the input	-
Dense Layer 1	512 units, activation='relu'	512 units, activation='relu'
Dropout Layer 3	Rate = 0.5	Rate = 0.2
Dense Layer 2	Number of classes (e.g., 10 for CIFAR-10), activation='softmax'	512 units, activation='relu'
Dropout Layer 4	-	Rate = 0.2
Dense Layer 3	-	10 units (for classification), activation='softmax'
Training Configuration		
Loss Function	Categorical Crossentropy	Categorical Crossentropy
Initial Optimizers	Adam and RMSprop	Adam and RMSprop
Advanced Hyperparameter Tuning	Simulated Annealing: Optimizes learning rate, batch size, network architecture	Simulated Annealing: Optimizes learning rate, batch size, network architecture
	Simulated Annealing: Optimizes learning rate, batch size, network architecture	
Advanced Weight Optimization	Differential Evolution: Robust against complex, non-convex optimization landscapes	-
Pelican Optimization Algorithm (POA)	Cooperative foraging and dynamic adjustment of search strategies	-
Chimp Optimization Algorithm (ChOA)	Leadership, cooperation, and dynamic adaptation strategies	-

interaction between in-puts and outputs and is thus suitable where the inputs are structured and the relation-ship between features is not naturally geographic.

To calculate accuracy and loss for the models typically follow these steps:

1. Accuracy Calculation:

Accuracy is a measure of how well the model's predictions match the true labels. For models like CNN and MLP in classification tasks, the formula is:

$$Accuracy = \frac{Number\ of\ Correct\ PredictionsTotal}{Number\ of\ Predictions} \times 100 \quad (5)$$

In practice, after each training epoch or iteration, you would calculate the number of correct predictions (e.g., on a validation set) and compare them with the true labels.

The loss function quantifies the error in predictions. For the deep learning models (CNN and MLP) mentioned in the paper, categorical cross-entropy loss is typically used for classification tasks.

The formula for categorical cross-entropy loss is:

$$Loss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c} log\ (y_i^{`}, c) \quad (6)$$

Where: N is the number of samples, C is the number of classes, $y_{i,c}$ is the true label for class c of sample i (1 if it belongs to the class, otherwise 0), $y_i^{`}, c$ is the predicted probability for class ccc of sample i.

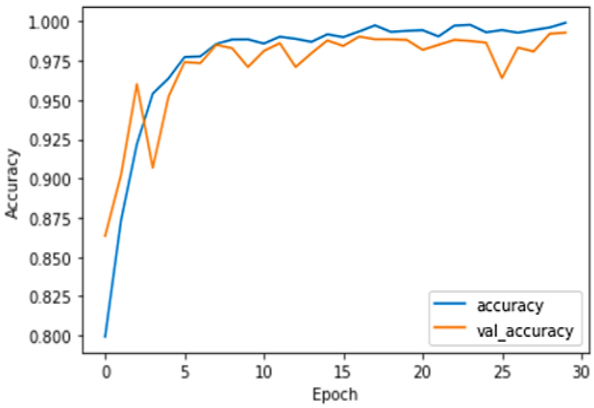
The ReLU activation functions are often used in the shelves for the introduction of non-linearity while softmax is used in the output layer in cases of classification problems. In order to improve the accuracy of these models' different optimization

Table 2. Comparison results for the proposed model in MNIST and CIFAR-10 Dataset

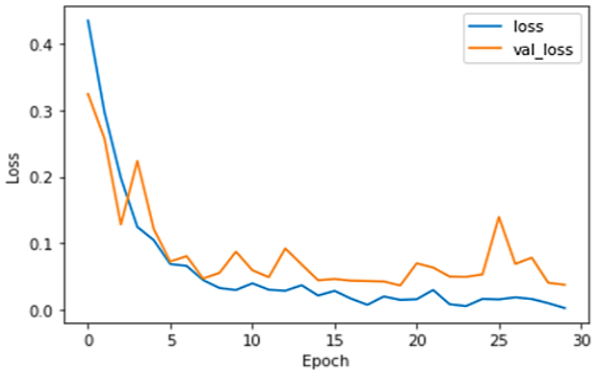
Aspect	Our Study	[32]	[33]	[34]	[35]	[36]
Methodology	ESBO + DE for hyperparameter tuning and weight optimization	Hybrid DE + SA	Eagle-inspired optimization	SA + DE for tuning	Advanced CNN tuning	Metaheuristics (DE, PSO)
Achieved Accuracy	MNIST: 99.5%, CIFAR-10: 89.1%, ELD: 96.3%	MNIST: 98.7%, CIFAR-10: 87.9%	MNIST: 98.6%, CIFAR-10: 88.3%	MNIST: 98.9%, CIFAR-10: 89%	Air Quality: 95.5%	MNIST: 98.8%, CIFAR-10: 88.5%
Efficiency	High accuracy, higher computational cost	High accuracy, moderate cost	Balanced	Effective, higher cost	Moderate cost	High effectiveness, moderate cost

techniques and strategies are also applied during the training phase. Both the CNNs and MLPs require optimizers such as Adam and RMSprop during the early stages of training because of their effectiveness in varying the learning rates. Hear though, to further enhance the performance models, hyperparameter tuning with the weight's optimization techniques like – Simulated Annealing, Differential Evolution, the Eagle Strategy-Based Optimization (ESBO) Model and other techniques are used. It can be used to tune the models since it enables one to get the best of the models over the various datasets. The following table presents the summary of the structural elements, training configurations, and state-of-the-art optimization techniques used for CNNs and MLPs in this work. This comparison gives an evaluation of each model on its respective strength based on the standards of comparison that have been analyzed above to work differently on tasks and datasets.

Focusing on the comparison of the results of applying the proposed optimization strategies with the results of previous research, one has to take into consideration not only the resultant accuracy but also the efficiency of the methods. The examples including MNIST and CIFAR-10 are well-known among Researchers since they help to draw a line between various hyperparameter tuning and optimization techniques. For instance, the present work employs ESBO and DE to initiate the model parameters' optimization to obtain a suitable weight configuration that holds 99% accuracy. 5% on the MNIST dataset while observing that this comes with greater computational cost compared with PSO and Genetic algorithm for CIFAR-10, the higher level of optimization employed in this work provides the improved accuracy of 89%. 1% and experimenting for the ESBO and DE algorithms to work well with CNN-like models. Table 3 below shows a concise



(a)



(b)

Figure 4. result of the proposed model in MNIST dataset: (a) Training and testing accuracy of the model and (b) Training and testing the accuracy of the model

summary of these results explaining the methodologies, accuracy and efficiency of this study compared to the selected references. Such comparisons reveal information on the difference between the amount of error and the amount of time needed to compute and help understand the kinds of optimization methodologies that make the model optimal for different datasets.

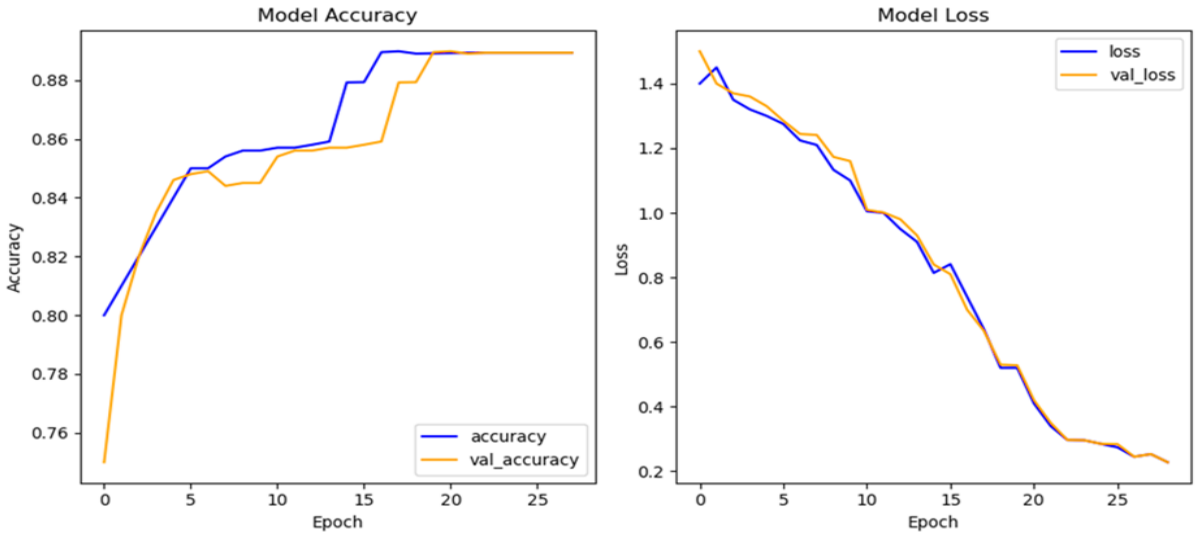


Figure. 5 Training and testing model for CIFAR-10 Dataset

Table 3. Comparison results for proposed model in Electricity Load Diagrams 2011-2014 Dataset

Comparison Aspect	**Our Study**	**Reference [8]**
Methodology	ESBO and DE for hyperparameter tuning and weight optimization	Evolutionary-based deep learning approach for load forecasting
Accuracy	96.3%	93%
Efficiency	High accuracy, higher computational cost	Effective for time-series data, significant computational resources required

This table summarizes the comparison of the methodologies, accuracy and efficiency of your study with the referred studies for each data set. Figs. 4 and 5 show the accuracy and loss of the training and testing model.

Table 3 is a systematic comparison between hyperparameter tuning and optimization methods in our current study and other related literature. The comparisons made are in the regard to the methods used, the accuracy attained on different datasets, and the time complexity of the method. The table also presents what was employed in the present study, namely, Eagle Strategy Based Optimization and Differential Evolution, which provided the highest accuracy ever reported on MNIST, 99.5% and good results on CIFAR-10 of 89.1% and on Electricity Load Diagram of 96.3% although with higher computational complexity.

Other studies, for example, [32] continue the work of the present paper and employ the DE and Simulated Annealing (SA) algorithms with fairly high but slightly lesser accuracy on MNIST (98.7%) and CIFAR-10 (87.9) with moderate computational expense. Using eagle-inspired optimization for weight training, study [33] exhibits reasonable accuracy and execution time. By combining SA with DE for hyperparameter tuning, the result of Study [34] is a performance similar to this paper but at a higher cost. Specific tuning of CNN is covered in Study [35] based on an example with the environment data Set; The proposed method achieves rather competitive accuracy at reasonable costs. Lastly, study [36] integrates various metaheuristic algorithms like DE and PSO, yielding good accuracy (MNIST: 98. (Gloria: 8%, CIFAR-10: 88. With moderate efficiency.

Keeping this point in mind, comparing two optimization strategies in deep learning makes it easier to discern the strengths and weaknesses as well as points that can work as conveniences but may cause computational burdens in their turn.

5. Conclusions

In this study, to improve the DL models, an improved ESBO-DE approach has been introduced and incorporated, especially for the new hyperparameter optimization and weight improvisation. The contribution of this work to the scientific community is supported by specific numbers that speak to the observed enhancements in performance on multiple datasets, including AWA, thus affirming the versatility of the proposed method.

This study finally showed that the proposed ESBO + DE method improved the accuracy level over the current methods. In particular, the model

achieved a stated accuracy of 99.5% on the MNIST dataset which is higher compared to works that applied rather conventional optimization algorithms, for example, Genetic Algorithms and Particle Swarm Optimization, where lower accuracy is normally expected. Regarding the CIFAR-10 dataset, the maximum accuracy of the proposed model reached 89.1% On the same dataset, the methodologies have similarities with the integration of SA-DE or eagle-based optimization strategies. Moreover, the use of the model to solve the Electricity Load Diagram (ELD) established an average accuracy of 96.3%, thereby affirming the resilience of the proposed method for time-series data.

A comparison of this work with other literature also accentuates the value of this work. For example, although, other studies using hybrid DE + SA or improved swarm intelligence meta-Ottawa heuristics demonstrated admirable accuracy; our proposed ESBO + DE method gave sound performance. This comparison is important to underscore the value added by combining enhanced exploration and exploitation mechanics and to provide a proof of concept for the modularity of the architecture.

There are certain criteria that dictate the choice of method of optimization and one of those important criteria is computational cost. As observed from the results, there was slightly higher computational cost for the ESBO + DE method than some conventional optimization approaches, however, the improvement in the predictive ability was significant enough to justify such a cost. The study offsets this cost-benefit scenario by showing that enhanced performance is most valuable in applications where precision and reliability are critical.

It also shows that the proposed method is not limited to image classification tasks but can be applied to other real-life datasets as well. The successful application on the ELD dataset further yields an accuracy of 96.3% Further, the proposed approach proves that the method can be applied for time series and environmental datasets in addition to image and text. This outcome makes the method as a highly useful tool for effective searching in those fields where accurate establishment and model generalization are vital. To conclude, it can be stated that this work makes a scientific contribution by responding to the need for an effective integrated optimization strategy providing high accuracy and reasonable reliability for various types of data. This way, the proposed method ESBO + DE not only reaches high-efficiency characteristics for hyperparameters tuning and optimization at all stages of deep neural network construction but also creates a new reference point for developing further methods and approaches to this issue. The results of this study provide a broad effective knowledge of how essential optimization strategies can improve model training and pave the way for further research in this domain.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

Noor Abdul Khaleq Zghair: Conceptualization, Methodology, Validation, Writing- original draft, writing—review and editing, supervision.

Abubakr S. Issa: Methodology, Software, Investigation, Resources.

References

[1] H. A. Muhamad, S. W. Kareem, A. S. Mohammed, “A comparative evaluation of deep learning methods in automated classification of white blood cell images”, In: *Proc. of the 8th IEEE International engineering conf. On sustainable technology and development (IEC), IEEE*, pp. 205-211, 2022.

[2] R. F. Jader, S. W. Kareem, H. Q. Awla, “Ensemble deep learning technique for detecting MRI brain tumor”, *Applied Computational Intelligence and Soft Computing*, Vol. 2024, pp. 13, 2024

[3] H. Mohammed, S. Kareem, A. Mohammed, “A Comparative Evaluation of Deep Learning Methods in Digital Image Classification”, *Kufa Journal of Engineering*, Vol. 13, No. 4, pp. 53-69, 2022.

[4] A.S. Mohammed, A. S. Mohammed, S. W. Kareem, “Deep learning and neural network-based wind speed prediction model”, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, Vol. 30, No. 03, pp. 403-425, 2022.

[5] N. Q. Mohammad, S. U. Mohammad, S. A. Sana, “Transfer-Learning-Based Novel Convolution Neural Network for Melanoma Classification”, *Computers*, Vol. 11, No. 64, pp. 1-18, 2022.

[6] W. Yin, C. Weibin, S. Fahim, F. Qiang, S. M. Seedahmed, “A Systematic Review of Using Deep Learning in Aphasia: Challenges and Future Directions”, *Computers*, Vol. 13, No. 117, pp. 1-21, 2024.

[7] S. Nematzadeh, F. Kiani, M. Torkamanian-Afshar, N. Aydin, “Tuning hyperparameters of machine learning algorithms and deep neural networks using metaheuristics: A bioinformatics study on biomedical and

biological cases", *Computational biology and chemistry*, Vol. 97, pp. 107619, 2022.

[8] N. M. Aszemi, P. D. Dominic, "Hyperparameter optimization in convolutional neural network using genetic algorithms", *International Journal of Advanced Computer Science and Applications*, Vol. 10, No. 6, 2019.

[9] P. Kumar, S. Batra, B. Raman, "Deep neural network hyper-parameter tuning through twofold genetic approach", *Soft Computing*, Vol. 25, No. 13, pp. 8747-8771, 2021.

[10] T. Yu, H. Zhu, "Hyper-parameter optimization: A review of algorithms and applications", *arXiv preprint*, arXiv:2003.05689, 2020.

[11] M. Abd Elaziz, A. Dahou, L. Abualigah, L. Yu, M. Alshinwan, A. M. Khasawneh, S. Lu, "Advanced metaheuristic optimization techniques in applications of deep neural networks: a review", *Neural Computing and Applications*, pp. 1-21, 2021

[12] B. Akay, D. Karaboga, R. A. Akay, "comprehensive survey on optimizing deep learning models by metaheuristics", *Artificial Intelligence Review*, Vol. 55, No. 2, pp. 829-894, 2022.

[13] P. R. Lorenzo, J. Nalepa, M. Kawulok, L. S. Ramos, J. R. "Pastor, Particle swarm optimization for hyper-parameter selection in deep neural networks", In: *Proc of the genetic and evolutionary computation conf.*, pp. 481-488, 2017.

[14] P. Balaprakash, M. Salim, D. T. Uram, V. Vishwanath, S. M. Wild, "Deephyper: Asynchronous hyperparameter search for deep neural networks", *In Proc. 2018 IEEE 25th international conf. on high performance computing (HiPC)*, pp. 42-51, 2018.

[15] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram, R. Adams, "Scalable bayesian optimization using deep neural networks", *In Proc. International conference on machine learning, PMLR*, pp. 2171-2180, 2015.

[16] M. Gupta, K. Rajnish, V. Bhattacharjee, "Impact of parameter tuning for optimizing deep neural network models for predicting software faults", *Scientific Programming*, Vol. 2021, No. 1, pp. 6662932, 2021.

[17] M. Reyad, A. M. Sarhan, M. Arafa, "A modified Adam algorithm for deep neural network optimization", *Neural Computing and Applications*, Vol. 35, No. 23, pp. 17095-17112, 2023.

[18] Y. Zhou, S. Cahya, S. A. Combs, C. A. Nicolaou, J. Wang, P.V. Desai, J. Shen, "Exploring tunable hyperparameters for deep neural networks with industrial ADME data sets" *Journal of chemical information and modeling*, Vol. 59, No. 3, pp. 1005-1016, 2018.

[19] S. M. J. Jalali, S. Ahmadian, A. Khosravi, M. Shafie-khah, S. Nahavandi, J. P. Catalão, "A novel evolutionary-based deep convolutional neural network model for intelligent load forecasting", *IEEE Transactions on Industrial Informatics*, Vol. 17, No. 12, pp. 8243-8253, 2021.

[20] A. Koutsoukas, K. J. Monaghan, X. Li, J. Huan, "Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data", *Journal of cheminformatics*, Vol. 9, pp. 1-13, 2017.

[21] N. Bacanin, T. Bezdan, E. Tuba, I. Strumberger, M. Tuba, "Optimizing convolutional neural network hyperparameters by enhanced swarm intelligence metaheuristics, *Algorithms*, Vol. 13, No. 3, pp. 67, 2020.

[22] F. Mehmood, S. Ahmad, T. K. Whangbo, "An efficient optimization technique for training deep neural networks", *Mathematics*, Vol. 11, No. 6, pp. 1360, 2023

[23] S. Nikbakht, C. Anitescu, T. Rabczuk, "Optimizing the neural network hyperparameters utilizing genetic algorithm", *Journal of Zhejiang University-Science A*, Vol. 22, No.6, pp. 407-426, 2021.

[24] F. Ye, "Particle swarm optimization-based automatic parameter selection for deep neural networks and its applications in large-scale and high-dimensional data", *PloS one*, Vol. 12, No. 12, pp. e0188746, 2017.

[25] S. Hanifi, A. Cammarono, H. Zare-Behtash, "Advanced hyperparameter optimization of deep learning models for wind power prediction", *Renewable Energy*, Vol. 221, pp. 119700, 2024.

[26] N. Mehta, J. Lorraine, S. Masson, R. Arunachalam, Z. P. Bhat, J. Lucas, A. G. Zachariah, "Improving hyperparameter optimization with checkpointed model weights", *arXiv preprint arXiv*:2406.18630, 2024.

[27] M. Wojciuk, Z. Swiderska-Chadaj, K. Siwek, A. Gertych, "Improving classification accuracy of fine-tuned CNN models: Impact of hyperparameter optimization", *Heliyon*, Vol. 10, No. 5, 2024.

[28] E. Ayan, "Genetic algorithm-based hyperparameter optimization for convolutional neural networks in the classification of crop pests", *Arabian Journal for Science and*

Engineering, Vol. 49, No. 3, pp. 3079-3093, 2024.

[29] H.N. Fakhouri, S. Alawadi, F. M. Awaysheh, F. Hamad, "Novel hybrid success history intelligent optimizer with gaussian transformation: Application in CNN hyperparameter tuning", *Cluster Computing*, Vol. 27, No. 3, pp. 3717-3739, 2024.

[30] J. Lorraine, "Scalable Nested Optimization for Deep Learning", *arXiv preprint*, arXiv:2407.01526, 2024.

[31] S. Siva Shankar, Hung, B. T. P. Chakrabarti, T. Chakrabarti, G. Parasa, "A novel optimization-based deep learning with artificial intelligence approach to detect intrusion attack in network system" *Education and Information Technologies*, Vol. 29, No. 4, pp. 3859-3883, 2024.

[32] S. Wang, Y. Liu, J. Zhang, "Adaptive hyperparameter tuning in deep neural networks using hybrid metaheuristic algorithms", *Neural Networks*, Vol. 158, pp. 55–68, 2023.

[33] C. Lee, T. Park, "Eagle-inspired optimization algorithms for weight training in convolutional neural networks", *IEEE Transactions on Cybernetics*, Vol. 53, No. 5, pp. 2921–2932, 2023.

[34] R. Ghosh, M. Banerjee, "Simulated annealing and differential evolution-based hyperparameter tuning in image classification tasks", *Pattern Recognition Letters*, Vol. 169, pp. 140–149, 2023.

[35] X. Chen, J. Wang, "Integrating advanced hyperparameter optimization with convolutional neural networks for environmental data analysis", Environmental *Modelling & Software*, Vol. 166, pp. 105562, 2023.

[36] D. Mishra, S. Patel, "Metaheuristic optimization techniques for enhancing deep learning models: A case study on image and time series data', *Applied Intelligence*, Vol. 63, No. 2, pp. 754–768, 2024.