Metaheuristics-Based Hyperparameter Tuning for Convolutional Neural Networks

Tong Van Luyen and Nguyen Van Cuong(✉)

Faculty of Electronic Engineering, Hanoi University of Industry, Hanoi 100000, Vietnam
cuongnv@haui.edu.vn

Abstract. Convolutional neural networks have made remarkable strides in the field of deep learning, achieving outstanding successes. However, to ensure the efficiency and high performance of these networks, it is crucial to optimize their hyperparameters. This paper presents a novel approach that focuses on optimizing hyperparameters for convolutional neural networks. The proposed approach leverages the binary bat algorithm, which is recognized as one of the most efficient algorithms among nature-inspired metaheuristic algorithms. By utilizing this approach, a set of optimal hyperparameters can be obtained, enabling the construction of convolutional neural network models that exhibit superior performance for specific applications. To demonstrate the effectiveness of this approach, the study employs it to determine hyperparameters such as the learning rate of optimizers and the number of filters in each convolutional layer. The objective is to build optimal models for the task of handwritten Chinese character classification. The empirical results obtained demonstrate the remarkable capabilities of the proposed approach. The models generated through this method exhibit higher performance in terms of classification accuracy and convergence ability when compared to the LeNet-5 model, as well as models based on Hyperband, Random Search, and Bayesian Optimization.

Keywords: Hyperparameter optimization · Convolutional neural networks · Binary bat algorithm · Metaheuristic algorithms · Handwritten Chinese character classification

1 Introduction

The convolutional neural network (CNN) has emerged as a prominent network within the realm of deep learning. Its significant advancements across diverse domains such as computer vision and natural language processing have captured substantial attention from both academic and business communities. Remarkable achievements, including facial recognition, autonomous vehicles, automated supermarkets, intelligent medical therapies, and pattern classification, have shattered previous limits and expanded the realm of what was once considered unattainable [1].

The convolution layer, the pooling layer, and the fully connected (FC) layer are the fundamental components of CNNs. It can effectively complete a variety of visual tasks

N. Thi Dieu Linh et al. (Eds.): ADHOCNETS 2023, LNICST 558, pp. 41–54, 2024.
https://doi.org/10.1007/978-3-031-55993-8_4

by properly stacking these layers in a deep network. The performance of a CNN is heavily dependent on various factors, such as the configuration of convolutional layers, the number of filters utilized, filter sizes, stride values, dropout probabilities, batch sizes, and the number of training epochs. The regularization technique and activation function selection both have a big impact on how well the network performs [2, 3]. Although it necessitates a thorough grasp of the used CNNs and their hyperparameter value settings, manual testing is a conventional way of identifying the proper hyperparameters to obtain high-performance CNNs. However, manual tuning becomes impractical for many problems due to several factors such as a large number of hyperparameters, complex models, time-consuming evaluations, and non-linear interactions among hyperparameters. Consequently, these factors have prompted extensive research into automatic optimization techniques for hyperparameters, commonly referred to as hyperparameter optimization (HPO). HPO's primary goal is to automate the process of hyperparameter tuning and enable users to successfully deploy CNN models with the best hyperparameters to real-world issues [2–5].

Numerous metaheuristic optimization techniques have been developed in recent years to address challenging computing issues. Particle swarm optimization, genetic algorithms, the whale optimization algorithm, grey wolf optimizer, and the bat algorithm are a few of them [6–9]. When compared to traditional optimization techniques, these algorithms are preferred by academics because of their flexibility and improved capacity to handle a variety of issues. Furthermore, it has been established that no metaheuristic algorithm can operate broadly enough to address every optimization issue. In other words, while some issues can be solved with the current algorithms, not all of them can. One of these new metaheuristic optimization algorithms among the aforementioned algorithms is the bat algorithm. This algorithm employs artificial bats as search agents to conduct the optimization process, simulating the natural pulse loudness and emission rate observed in real bats. The algorithm takes inspiration from the echolocation behavior exhibited by bats. It has been demonstrated that this algorithm may deliver results that are comparable to those of other algorithms, such as particle swarm optimization [6, 10, 11]. The binary bat algorithm (BBA) was introduced to address problems characterized by discrete binary search spaces, such as dimensionality reduction, feature selection, and signal processing. This distinguishes it from the original version of the bat algorithm, which was designed for solving problems with continuous real search spaces [6, 12–15].

To find the best hyperparameters, it is essential to use suitable optimization techniques. Due to the fact that many HPO problems fall under the category of NP-hard problems or are non-convex or non-differentiable optimization problems, conventional optimization techniques may not be effective for solving HPO problems [2, 4, 16, 17]. BBA is one of the most effective metaheuristic-based approaches, and it can solve the above problems. Therefore, this paper utilizes BBA to propose an HPO approach for CNN models. Particularly for large datasets or complex models with many hyperparameters, this approach aids deep learning developers in reducing the amount of effort spent tuning the hyperparameters. Moreover, the proposed approach improves the performance of CNN models. The efficiency of the proposal will be verified by optimizing hyperparameters for CNN models which are used to classify handwritten Chinese characters.

The rest of this paper is structured as follows. The next section proposes an HPO approach to optimize hyperparameters for CNN models. Section 3 evaluates the performance of the proposed approach via several scenarios before concluding the paper in Section 4.

2 Proposed Approach

The primary function of deep learning is the solution to optimization problems. An optimization method is employed to initialize and optimize the weight parameters of a deep learning model, aiming to minimize the objective function or maximize the accuracy until a minimum value or maximum value, respectively, is approached [4]. Similar to this, hyperparameter optimization approaches aim to improve the architecture of a deep learning model by identifying the best combinations of hyperparameters. This section presents a description of the fundamental concepts behind the hyperparameter optimization problem and outlines the proposed algorithm designed for CNN models.

2.1 Hyperparameter Optimization Problem

Deep learning model designers can find the best hyperparameters for their models by efficiently scanning the hyperparameters space utilizing optimization techniques. The exploration of hyperparameter combinations encompasses four key components: an estimator (a regressor or classifier) coupled with its objective function, a search space (also known as the configuration space), a search or optimization technique, and an evaluation function utilized to evaluate the performance of different hyperparameter configurations [4].

The domain of a hyperparameter can be classified into different types, including categorical (e.g., optimizer type), binary (e.g., early stopping), discrete (e.g., number of clusters), or continuous (e.g., learning rate). In actual applications, continuous and discrete hyperparameters' domains are typically constrained [4]. On the other side, conditionality can occasionally be contained in the hyperparameter configuration space. A conditional hyperparameter, also known as a hyperparameter that changes dependent on the value of another hyperparameter, may need to be used or tuned [18].

Generally, the objective of a hyperparameter optimization problem is to attain [4]:

$$\mathbf{h}* = \arg\min_{\mathbf{h}\in\mathbf{H}} f(\mathbf{h}), \tag{1}$$

where the objective function to be minimized is denoted as $f(\mathbf{h})$, $\mathbf{h}$ represents a hyperparameter vector. The optimum value of $f(\mathbf{h})$ is achieved with the hyperparameter vector $\mathbf{h}*$. The search space $\mathbf{H}$ encompasses all possible values that the hyperparameter vector can take. The objective of hyperparameter optimization is to fine-tune the hyperparameters within given resource constraints, aiming to achieve optimal or near-optimal model performance. Various evaluation metrics such as accuracy, root mean square error, F1-score, and false alarm rate can be utilized to assess the model's performance [4].

For CNN models, the search space $\mathbf{H}$ includes hyperparameters, including the number of filters, filter sizes within convolutional layers, the number of nodes in FC layers,

activation functions, optimizers, and learning rate. Considering that CNN models necessitate the optimization of m distinct hyperparameters, with each hyperparameter having n_i choices within the categorical and discrete domain in the i - th search space H_i for $i = 1, 2, \ldots, m$. Hence, the search space can be expressed as:

$$\mathbf{H} = \begin{matrix} H_{1,1} & H_{1,2} & \ldots & H_{1,n_1} \\ H_{2,1} & H_{2,2} & \ldots & H_{2,n_2} \\ \ldots & \ldots & \ldots & \ldots \\ H_{m,1} & H_{m,2} & \ldots & H_{m,n_m} \end{matrix}. \tag{2}$$

The hyperparameter vector $\mathbf{h}* = [h_1, h_2, \ldots, h_m]^T$ comprises of m optimized hyperparameters. To obtain $\mathbf{h}*$, the index vector $\mathbf{k} = [k_1, k_2, \ldots, k_m]^T$, which consists of m values mapping to search space $\mathbf{H}$, needs to be optimized. Each value in $\mathbf{k}$ should be less than or equal to the corresponding number of choices in the respective search space H_i. For instance, if the first row of the search space H_1 has n_1 choices, then k_1 should be less than or equal to n_1, and the first optimized hyperparameter h_1 is H_{1,k_1}. Consequently, effective optimization algorithms should be employed to address HPO problems, enabling the determination of the index vector and subsequent identification of optimized hyperparameters.

2.2 Proposed Algorithm

The proposed algorithm is derived from the basic BBA (refer to [6] for details) for the purpose of determining the optimal hyperparameter vector $\mathbf{h}*$. Nevertheless, alternative metaheuristics can also be employed in conjunction with the proposed algorithm, instead of utilizing BBA exclusively. The flowchart illustrating the algorithm is presented in Fig. 1, and its description is provided below:

Initialization (a red border block):
Initially, it is essential to determine the type of learning (supervised or unsupervised) and identify the datasets to be used. Subsequently, the search space $\mathbf{H}$ needs to be defined, encompassing hyperparameters such as the number of filters, filter sizes in convolutional layers, and the range of choices or upper and lower limits for each hyperparameter. Additionally, considerations should be made regarding the inclusion of early stopping as a hyperparameter. Given that the optimization problem aims to minimize the objective function, the selection of this function relies on performance metrics evaluated on test datasets. The objective function can be formulated as:

$$\text{Accuracy:} f = \frac{1}{accuracy_{test}}, \tag{3}$$

$$\text{Loss:} f = loss_{test}. \tag{4}$$

Next, the population size and the number of iterations are initialized, and the dimension (d) of a single solution (bat's position) in BBA is calculated as follows:

$$d = \sum_{i=1}^{m} \lceil \log_2 n_i \rceil. \tag{5}$$

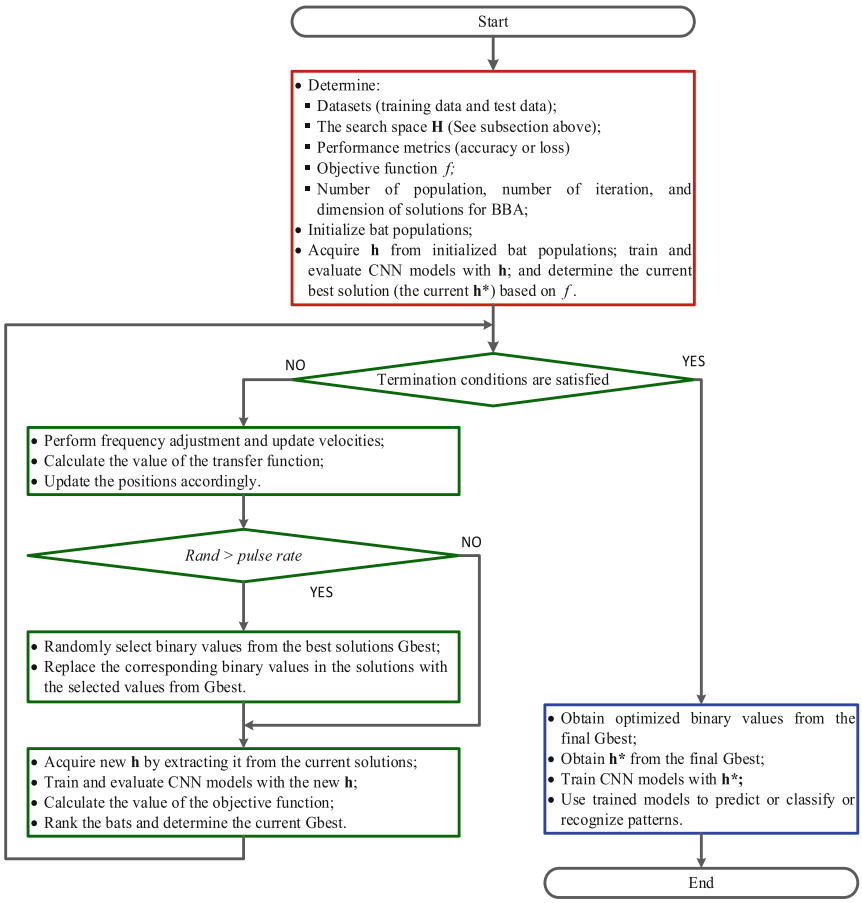

Fig. 1. The flowchart of the proposed algorithm.

The rounding up operation to the nearest number is denoted by $\lceil \bullet \rceil$. The binary bat populations are randomly initialized, and each bat's position $\mathbf{s}$ is represented as a binary number vector, which needs to be converted into a decimal number vector denoted as $\mathbf{k}$. For each element k_i in $\mathbf{k}$ for $i = 1, 2, \ldots, m$, k_i is formulated as:

$$k_i = \left\lceil \frac{n_i}{2^{\lceil \log_2 n_i \rceil} - 1} \text{int}(\mathbf{s}) \right\rceil, \tag{6}$$

where $\lceil \bullet \rceil$ represents the rounding to the nearest number while int $(\bullet)$ signifies converting to an integer. Subsequently, the hyperparameter vector $\mathbf{h}$ is acquired by mapping $\mathbf{k}$ into the search space $\mathbf{H}$. Using this hyperparameter vector, CNN models are constructed, trained, and tested to evaluate their performance based on the objective function described in (3) or (4). Through this process, the current best hyperparameter vector can be identified by considering the performance metrics.

Determining the optimal hyperparameters (green border blocks):
The search operation of BBA is implemented. For the p - th bat with $p = 1, 2, \ldots, numPop$, the velocity V_p^{iter} and the frequency Q_p at the $iter$ - th iteration are updated as follows:

$$Q_p = Q_{\min} + (Q_{\max} - Q_{\min})rand, \tag{7}$$

$$V_p^{iter} = V_p^{iter-1} + (\mathbf{S}_p^{iter-1} - G_{best})Q_p. \tag{8}$$

Here, $Q_{\min}$ and $Q_{\max}$ represent the minimum and maximum frequency, respectively. G_{best} denotes the current best solutions, and $rand$ signifies random values drawn from a uniform distribution between 0 and 1. In order to update the positions of bats or enforce their movement within a binary space, velocity values are mapped to binary values using a V-shaped transfer function. This transfer function, employed to update the position of the p - th, is described as follows:

$$F_{transfer}\left(V_p^{iter}\right) = \left|\frac{2}{\pi}\arctan\left(\frac{2}{\pi}V_p^{iter}\right)\right|, \tag{9}$$

$$\mathbf{S}_p^{iter} = \begin{cases} (\mathbf{S}_p^{iter-1})^{-1} & \text{if } rand < F_{transfer}(V_p^{iter}) \\ \mathbf{S}_p^{iter-1} & \text{if } rand \geq F_{transfer}(V_p^{iter}) \end{cases}, \tag{10}$$

$$\mathbf{S}_p^{iter}(j) = G_{best}(j) \text{ if } rand > pulse\ rate, \tag{11}$$

where $(\bullet)^{-1}$ denotes binary numbers' complements. If $pulserate$ is lower than $rand$, the binary numbers in $\mathbf{s}$ are modified by replacing them with randomly selected binary values from G_{best}, where $pulserate$ indicates bats' pulse emission rates. This adjustment directs the local solution, $\mathbf{s}$, towards the current best solution G_{best}, using (11) with $j = 1, 2, \ldots, d$. During the process of obtaining a new hyperparameter vector from the current solution, $\mathbf{h}$ can be derived using the same methodology as explained earlier. The optimization process continues until the termination conditions are met. Based on experimental evaluations, this process finishes after executing 20 iterations.

Constructing and testing CNN models with optimized hyperparameters (a blue border block):
The best hyperparameter vector $\mathbf{h}*$ is derived from the optimal solution represented by binary numbers. Subsequently, the optimized CNN models are constructed and trained using $\mathbf{h}*$. Eventually, the trained model is utilized for various tasks such as prediction, classification, or pattern recognition.

3 Performance Evaluation

This section presents a comprehensive evaluation of the effectiveness of our proposed approach. To begin, we define the datasets, relevant parameters, and the search space $\mathbf{H}$ utilized in our experiments, ensuring a clear and well-defined foundation for our evaluations. Next, we showcase the convergence capability of the proposed approach,

emphasizing its ability to efficiently converge to optimal solutions. Through illustrative demonstrations and convergence plots, we provide evidence of its effectiveness in finding superior hyperparameters for convolutional neural network models. Moreover, we conduct a thorough comparative analysis to assess how our proposed approach fares against other state-of-the-art methods, including Hyperband, Random Search (RS), and Bayesian Optimization (BO). By quantitatively comparing their performances on various metrics, such as classification accuracy and convergence speed, we aim to identify the strengths and weaknesses of each method.

3.1 Datasets and Parameter Setup

This study uses the CASIA offline database which includes plain gray-scale images of isolated handwritten Chinese characters [19]. Specifically, a subset of the HWDB1.1 dataset is used, and it includes 20 Chinese characters written by about 300 writers (with minor differences for some categories). The test set contains about 50 randomly selected images per category, while the remaining images (approximately 240) form the training set.

The performance of metaheuristic algorithms is closely influenced by two factors: the population size and the maximum number of iterations [20, 21]. Through experimental analysis, it has been determined that a population size of 20 and a maximum number of iterations of 20 yield favorable results. As for the remaining parameters, they are set based on recommendations provided in [6]: $pulserate = 0.5$; $Q_{\min} = 0$; $Q_{\max} = 2$. The termination condition for the algorithm is reached when all iterations have been executed. The simulation results in all scenarios represent the average values obtained from 20 independent runs.

This study validates the proposed approach by applying it to optimize the hyperparameters of the baseline model, namely the LeNet-5 model. The LeNet-5 model is composed of three convolutional layers and two fully connected layers, including the output layer, as illustrated in Fig. 2.

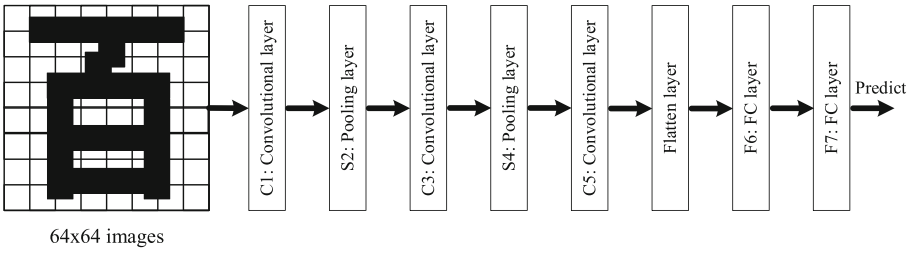

Fig. 2. The LeNet-5 model for handwritten Chinese character classification problems.

The search space $\mathbf{H}$, as defined in (12), encompasses various hyperparameters. It includes the number of filters in the three convolutional layers (C1-C3) represented by the first three rows, the number of hidden neurons in the first fully connected layer (F6) denoted by the fourth row, different optimizers listed in the fifth row, and the learning

rate mentioned in the last row. Each hyperparameter has four possible choices, resulting in a solution dimension (d) of 12, as calculated using (5). The performance metric used in this study is accuracy, so the objective function is chosen as (3).

$$\mathbf{H} = \begin{matrix} 32 & 64 & 96 & 128 \\ 64 & 96 & 128 & 192 \\ 96 & 128 & 192 & 256 \\ 256 & 512 & 1024 & 1536 \\ \text{Adamax} & \text{SGD} & \text{RMSprop} & \text{Adam} \\ 0.0001 & 0.01 & 0.001 & 0.005 \end{matrix}. \tag{12}$$

Table 1 displays the hyperparameters utilized in both the proposed approach-based model and the LeNet-5 model. The input of CNN models is 64 x 64 gray-scale images, and the output is the prediction of Chinese characters written in input images. Besides, L2 regularization penalties, dropout layers, and scheduler are used to prevent overfitting.

Table 1. Hyperparameters of CNN models.

	LeNet-5 model	Proposed approach-based model
Number of filters in C1	6	Optimization
Number of filters in C2	16	Optimization
Number of filters in C3	120	Optimization
Number of neurons in F6	84	Optimization
Optimizer	SGD	Optimization
Learning rate	0.01	Optimization
Filter size	5	3
Activation function	Tanh	ReLu
	The Softmax function is followed by F7	
Number of neurons in F7	20	
Loss	Categorical cross-entropy	
Pooling layer	Max pooling	
Stride and Padding	1	
Regularizer	L2 with the regularization factor of 0.0005	
Dropout (precede F6 and F7)	0.5	
Scheduler	Halve the learning rate every 3 epochs	
Batch size	100	

3.2 Convergence Ability

We evaluate both the convergence capability and the accuracy of the optimized CNN models on test datasets in this subsection. Figure 3 and Fig. 4 illustrate the values of the objective function and classification accuracy on test datasets over 20 iterations. The results are the average values of 20 independent experiments. It can be seen that the proposed approach converges rapidly within the first 3 iterations, and the accuracy has reached greater than 95.65%. From the 4th iteration onwards, the accuracy insignificantly increases, about 95.8%. Figure 5 demonstrates some Chinese handwritten characters which are classified by a CNN model with optimized hyperparameters. According to the results, the lowest classification accuracy is the character 依, with 99.47% while almost characters are classified with 100% accuracy. The hyperparameters optimized by the proposed approach are summarized in Table 2.

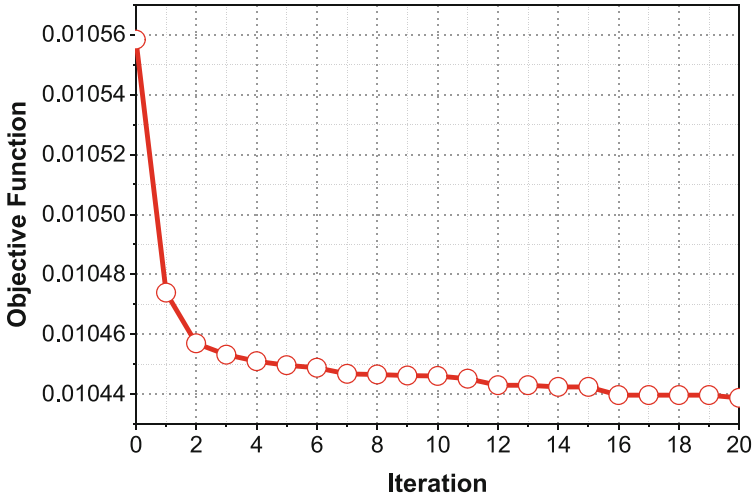

Fig. 3. The objective function over 20 iterations.

3.3 Comparative Analysis

This subsection compares the proposed approach to Hyperband, RS, and BO which are popular and efficient approaches for HPO [3, 4]. Besides, the LeNet-5 model is used as the reference model in this comparison. Figure 6 and Fig. 7 compare the accuracy of optimized CNN models that utilize different approaches. The results indicate that the proposed approach-based optimized model had higher accuracy than other approaches, and the model utilizing the proposed approach demonstrates a significant convergence trend starting from the seventh epoch onwards. At this epoch, the models based on the proposed approach, Hyperband, RS, BO, and the LeNet-5 model achieved accuracies of **95.45%**, 94.66%, 94.97%, 89.69%, and 59.68%, respectively.

The accuracies of trained models at the twentieth epoch and optimized hyperparameters are summarized in Table 2 and Table 3, respectively. The proposed approach

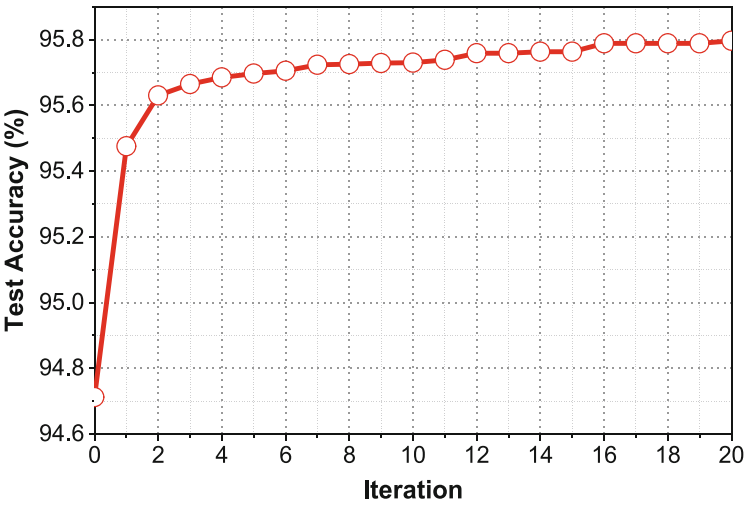

Fig. 4. The classification accuracy of the optimized CNN model.

Fig. 5. The classification accuracy for Chinese handwritten characters.

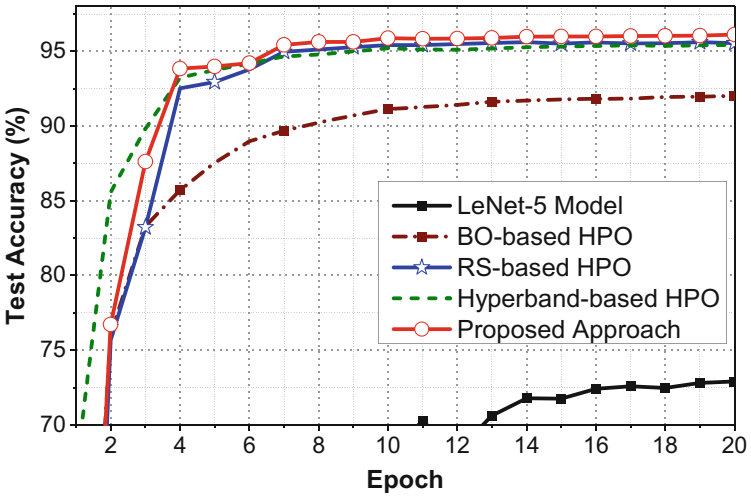

Fig. 6. The comparative graph of the accuracy.

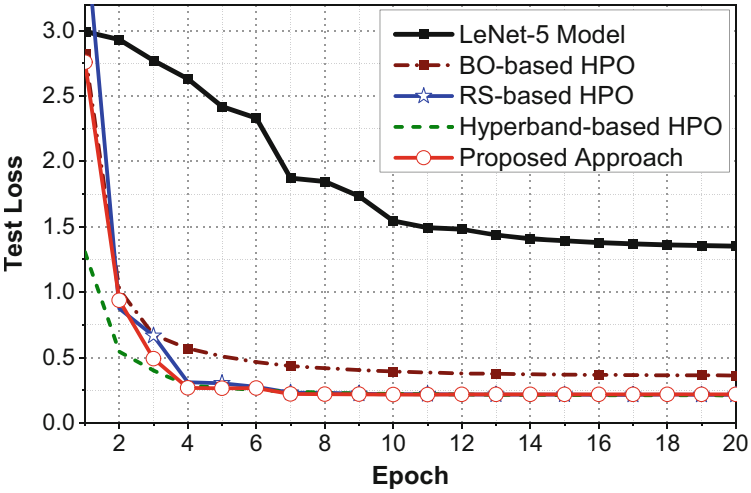

Fig. 7. The comparative graph of the loss.

outperformed all other methods with an impressive accuracy of 96.12%. This indicates that the proposed approach effectively fine-tuned the hyperparameters, leading to a highly accurate CNN model. The hyperparameter settings for the proposed approach involved significantly higher numbers of filters in C1 (128), C2 (64), C3 (192), and neurons in F6 (1024) compared to the LeNet-5 model. This increase in model complexity allows the proposed approach to extract more sophisticated features and achieve superior accuracy. Hyperband and RS also demonstrated competitive performances with accuracies of 95.43% and 95.54%, respectively. Their hyperparameter configurations revealed a balance between model complexity and accuracy, with a moderate number of

filters and neurons. Both approaches utilized RMSprop as the optimizer and a learning rate of 0.001, which contributed to their strong performances. BO, on the other hand, achieved an accuracy of 92.02%, showing slightly lower performance compared to the other approaches. Its hyperparameter settings were more conservative, with a relatively lower number of filters and neurons compared to the proposed approach, Hyperband, and RS. BO employed the Adam optimizer with a smaller learning rate of 0.0001. In contrast, the LeNet-5 model obtained the lowest accuracy of 72.91%. This is expected as the LeNet-5 model is a basic architecture and lacks the fine-tuned hyperparameters of the other approaches. Its hyperparameter configuration consisted of relatively fewer filters and neurons, and it employed the SGD optimizer with a learning rate of 0.01.

Table 2. The classification accuracy of different approaches.

Approach	Accuracy (%)
Proposed approach	96.12
Hyperband	95.43
RS	95.54
BO	92.02
LeNet-5 model	72.91

Table 3. Optimized hyperparameters of different approaches.

Hyperparameter	LeNet-5 model	Proposed approach	Hyperband	RS	BO
Number of filters in C1	6	128	32	64	128
Number of filters in C2	16	64	192	96	64
Number of filters in C3	120	192	256	256	256
Number of neurons in F6	84	1024	512	512	1536
Optimizer	SGD	RMSprop	Adam	RMSprop	Adam
Learning rate	0.01	0.001	0.001	0.001	0.0001

Overall, the results reveal that the choice and tuning of hyperparameters significantly impact the accuracy of CNN models. The proposed approach stands out as the most effective method, showcasing the importance of comprehensive hyperparameter optimization in achieving state-of-the-art performance. Hyperband and RS demonstrate competitive performances, highlighting their potential in finding good hyperparameter settings with fewer evaluations. Meanwhile, BO, despite being less accurate in this study, remains a valuable approach due to its probabilistic nature and potential for exploration in high-dimensional spaces. The findings provide valuable insights into the selection

and tuning of hyperparameters in CNN models, aiding researchers and practitioners in making informed decisions for model optimization in real-world applications.

4 Conclusion

This paper has proposed a novel approach for finding optimal hyperparameters in CNN models. The efficacy of this approach has been demonstrated by optimizing hyperparameters for a CNN model used in handwritten Chinese character classification. The results clearly indicate that the proposed approach achieves higher classification accuracy compared to other popular and efficient methods named Hyperband, RS, and BO, when tested under similar experimental conditions. These findings have highlighted the potential of our proposed approach as a promising solution for hyperparameter optimization problems. For future work, it would be beneficial to explore the application of this approach to more complex CNN models or even consider its applicability to other types of neural networks, such as recurrent neural networks. Additionally, experimenting with diverse datasets could provide valuable insights into the versatility and robustness of the proposed approach across different problem domains. By extending the evaluation to a wider range of scenarios, we can better understand its capabilities and limitations, which will contribute to its broader adoption and potential improvements in the field of hyperparameter optimization.

References

1. Li, Z., Liu, F., Yang, W., Peng, S., Zhou, J.: A survey of convolutional neural networks: analysis, applications, and prospects. IEEE Trans. Neural Netw. Learn. Syst. Inst. Electr. Electron. Eng. pp. 1–21 (2021)
2. Bacanin, N., Bezdan, T., Tuba, E., Strumberger, I., Tuba, M.: Optimizing convolutional neural network hyperparameters by enhanced swarm intelligence metaheuristics. Algorithms **13**(3), 67 (2020)
3. Akay, B., Karaboga, D., Akay, R.: A comprehensive survey on optimizing deep learning models by metaheuristics. Artif. Intell. Rev. **55**(2), 829–894 (2021)
4. Yang, L., Shami, A.: On hyperparameter optimization of machine learning algorithms: theory and practice. Neurocomputing **415**, 295–316 (2020)
5. Thuc, K.X., et al.: A metaheuristics-based hyperparameter optimization approach to beamforming design. IEEE Access **11**, 52250–52259 (2023)
6. Mirjalili, S., Mirjalili, S.M., Yang, X.-S.: Binary bat algorithm. Neural Comput. Appl. **25**(3), 663–681 (2014)
7. Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., Cosar, A.: A survey on new generation metaheuristic algorithms. Comput. Ind. Eng. **137**, 106040 (2019)
8. Yoo, Y.: Hyperparameter optimization of deep neural network using univariate dynamic encoding algorithm for searches. Knowl.-Based Syst. **178**, 74–83 (2019)
9. Wang, Y., Zhang, H., Zhang, G.: CPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks. Swarm Evol. Comput. **49**, 114–123 (2019)
10. Luyen, T.V., Cuong, N.V.: An effective beamformer for interference suppression without knowing the direction. Int. J. Electr. Comput. Eng. **13**(1), 601–610 (2023)

11. Kha, H.M., et al.: An efficient beamformer for interference suppression using rectangular antenna arrays. J. Commun. **18**(2), 116–122 (2023)
12. Luyen, T.V., Cuong, N.V., Duy, L.: An effective beamformer for interference mitigation. In: Anh, N.L., Koh, SJ., Nguyen, T.D.L., Lloret, J., Nguyen, T.T. (eds.) Intelligent Systems and Networks. LNCS, vol. 471, pp. 630–639. Springer, Singapore (2022). https://doi.org/10.1007/978-981-19-3394-3_73
13. Liu, F., Yan, X., Lu, Y.: Feature selection for image steganalysis using binary bat algorithm. IEEE Access **8**, 4244–4249 (2019)
14. Ghanem, W.A.H.M., et al.: Cyber intrusion detection system based on a multiobjective binary bat algorithm for feature selection and enhanced bat algorithm for parameter optimization in neural networks. IEEE Access **10**, 76318–76339 (2022)
15. Luyen, T.V., et al.: Null-steering beamformers for suppressing unknown direction interferences in sidelobes. J. Commun. **17**(8), 600–607 (2022)
16. Nakisa, B., Rastgoo, M.N., Rakotonirainy, A., Maire, F., Chandran, V.: Long short term memory hyperparameter optimization for a neural network based emotion recognition framework. IEEE Access **6**, 49325–49338 (2018)
17. Sabar, N.R., Turky, A., Song, A., Sattar, A.: An evolutionary hyper-heuristic to optimise deep belief networks for image reconstruction. Appl. Soft Comput. **97**, 105510 (2020)
18. Luo, G.: A review of automatic selection methods for machine learning algorithms and hyper-parameter values. Netw. Model. Anal. Health Inform. Bioinform. **5**, 1–16 (2016)
19. Liu, C.L., Yin, F., Wang, D.H., Wang, Q.F.: Online and offline handwritten Chinese character recognition: benchmarking on new databases. Pattern Recogn. **46**(1), 155–162 (2013)
20. Li, Q., Liu, S.Y., Yang, X.S.: Influence of initialization on the performance of metaheuristic optimizers. Applied Soft Comput. **91**, 106193 (2020)
21. Kha, H.M., et al.: A null synthesis technique-based beamformer for uniform rectangular arrays. In: 2022 International Conference on Advanced Technologies for Communications (ATC), pp. 13–17 (2022)