Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Knowledge-Based Systems 232 (2021) 107494

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

MFBCNNC: Momentum factor biogeography convolutional neural network for COVID-19 detection via chest X-ray images

Junding Sun [a,1], Xiang Li [a,1], Chaosheng Tang [a,*], Shui-Hua Wang [a,b,c,**], Yu-Dong Zhang [a,d,e,***]

[a] School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo, Henan 454000, PR China
[b] School of Architecture Building and Civil engineering, Loughborough University, Loughborough, LE11 3TU, UK
[c] School of Mathematics and Actuarial Science, University of Leicester, LE1 7RH, UK
[d] School of Informatics, University of Leicester, Leicester, LE1 7RH, UK
[e] Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia

ARTICLE INFO

Article history:
Received 2 November 2020
Received in revised form 16 August 2021
Accepted 11 September 2021
Available online 15 September 2021

Keywords:
Biogeography-based optimization
Convolutional neural network
COVID-19
Deep learning
Pneumonia

ABSTRACT

Aim: By October 6, 2020, Coronavirus disease 2019 (COVID-19) was diagnosed worldwide, reaching 3,355,7427 people and 1,037,862 deaths. Detection of COVID-19 and pneumonia by the chest X-ray images is of great significance to control the development of the epidemic situation. The current COVID-19 and pneumonia detection system may suffer from two shortcomings: the selection of hyperparameters in the models is not appropriate, and the generalization ability of the model is poor.
Method: To solve the above problems, our team proposed an improved intelligent global optimization algorithm, which is based on the biogeography-based optimization to automatically optimize the hyperparameters value of the models according to different detection objectives. In the optimization progress, after selecting the immigration of suitable index vector and the emigration of suitable index vector, we proposed adding a comparison operation to compare the value of them. According to the different numerical relationships between them, the corresponding operations are performed to improve the migration operation of biogeography-based optimization. The improved algorithm (momentum factor biogeography-based optimization) can better perform the automatic optimization operation. In addition, our team also proposed two frameworks: biogeography convolutional neural network and momentum factor biogeography convolutional neural network. And two methods for detection COVID-19 based on the proposed frameworks.
Results: Our method used three convolutional neural networks (LeNet-5, VGG-16, and ResNet-18) as the basic classification models for chest X-ray images detection of COVID-19, Normal, and Pneumonia. The accuracy of LeNet-5, VGG-16, and ResNet-18 is improved by 1.56%, 1.48%, and 0.73% after using biogeography-based optimization to optimize the hyperparameters of the models. The accuracy of LeNet-5, VGG-16, and ResNet-18 is improved by 2.87%, 6.31%, and 1.46% after using the momentum factor biogeography-based optimization to optimize the hyperparameters of the models.
Conclusion: Under the same experimental conditions, the performance of the momentum factor biogeography-based optimization is superior to the biogeography-based optimization in optimizing the hyperparameters of the convolutional neural networks. Experimental results show that the momentum factor biogeography-based optimization can improve the detection performance of the state-of-the-art approaches in terms of overall accuracy. In future research, we will continue to use and improve other global optimization algorithms to enhance the application ability of deep learning in medical pathological image detection.

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding authors.
** Correspondence to: MA208, Michael Atiyah Building, University of Leicester, University Road, Leicester, LE1 7RH, UK.
*** Corresponding author at: School of Informatics, University of Leicester, Leicester, LE1 7RH, UK.
E-mail addresses: tcs@hpu.edu.cn (C. Tang), shuihuawang@ieee.org (S.-H. Wang), yudongzhang@ieee.org (Y.-D. Zhang).

https://doi.org/10.1016/j.knosys.2021.107494
0950-7051/

1. Introduction

In 2019, novel coronavirus infection caused pneumonia in over 200 countries and regions worldwide. The novel coronavirus pneumonia (Coronavirus Disease 2019, COVID-19) was named by the world health organization (WHO) on February 11, 2020 [1]. COVID-19 is mainly caused by the novel coronavirus, and pneumonia (common pneumonia) is caused by a bacterial infection or atypical pathogens such as mycoplasma and chlamydia. Therefore, COVID-19 is more infectious than pneumonia, and can be transmitted through respiratory droplets or close contact. As of October 6, 2020, there were 3,355,7427 confirmed cases and 1,037,862 deaths worldwide [2].

Currently, the main detection methods for pneumonia and COVID-19 are chest X-ray imaging, chest computed tomography (CT) imaging, blood routine examinations [3], nucleic acid test [4], etc. For chest CT, the cost is high, there is a large amount of ionizing radiation to the human body, and it is difficult to detect small lesions with little or no density change or early lesions confined to the cell level. For routine blood examination, it is necessary to strictly prevent the collected blood samples from contacting the air. Meanwhile, it requires high timeliness of detection. The blood samples [5] taken should be tested within 30 min; otherwise, the samples should be stored in ice water for no more than 2 h. For nucleic acid testing, there is a risk that pharyngeal swab sampling will not be collected, and repeated sampling is needed if necessary. In patients with early infection, patients with low titer of laryngopharyngeal virus cannot be applied and are prone to false negative. In addition, sampling staff are easily exposed to the virus environment and have a greater risk of infection. Compared with the above-mentioned pneumonia detection methods, chest X-ray imaging [6] has the advantages of fast detection speed, clear imaging, low cost, permanent retention of samples, and easy review. The traditional chest X-ray images identification [7] mainly relies on the radiologist to perform manual examinations. This way not only consumes large amounts of manpower and time, but also is affected by many other factors, such as visual fatigue, psychological state, etc., which will increase the uncertainty of the identification results. Fig. 1 shows the chest X-ray images of COVID-19, Normal, and Pneumonia.

At present, with the rapid development of deep learning technology, scholars have begun to apply deep learning technology in the field of medical pathological image detection. Narin et al. [8] trained ResNet-50 on a small dataset, which consists of the chest X-ray images for Normal and COVID-19 (a total of 100 images). The experimental results showed that the accuracy of the model was 98%. Jin et al. [9] fused ResNet-50 structure and proposed to use UNet++ to detect COVID-19. The accuracy of the model was 97.4%. Zhang et al. [10] used the basic convolutional neural network (CNN) models to perform the classification detection of benign and malignant pulmonary nodules in the chest CT images, with an accuracy of 78%, a sensitivity of 80%, a specificity of 53%, and AUC of 71%. In the above literature, the accuracy of CNN models is lower than that of using the more complex network models. Although the detection accuracy of ResNet-50 and UNet++ is relatively high, they are only verified on the binary classification problem. The binary classification problem is relatively simple. It has practical significance only for the detection of COVID-19 and Normal by the chest X-ray images. It has less reference value for detection COVID-19, Normal, and Pneumonia by the chest X-ray images.

Jin et al. [9] used a variety of network models (FCN-8s, U-Net, V-Net, and 3D U-Net++) to test the image segmentation of medical lesions. In addition, they tested medical image classification using ResNet-50, Inception-v3, DPN-92, and attention ResNet-50. When detecting COVID-19, Wang et al. [11] used a randomly selected region of interest and a deep learning algorithm to extract features from the initial network. Song et al. [12] used ResNet-50 and Pyramid Networks to segment COVID-19 images and assisted doctors in detecting and locating the lesion area. In the field of image segmentation and image classification, the above literature involved many deep learning models. The hyperparameters in the models play an important role in the processes of model training and testing. A set of hyperparameters with appropriate values can improve the performance of the model, but the way of taking values is not described in detail in the above literature.

Ghoshal et al. [13] proposed a Bayesian convolutional neural network to estimate the uncertainty in the classification of COVID-19. Half of the data needed for the experiments came from Kaggle and the other images were collected from the COVID-19 patients by the author. The experimental results showed that the accuracy of VGG-16 can be increased by 7.2% by using Bayesian inference. Feng et al. [13] used VB-Net to segment the chest images and applied a random forest method based on the size of the infected area to the segmented image to classify the COVID-19 images. Zheng et al. [14] proposed a model using a 3D deep neural network to detecting CT images of COVID-19. Although many optimization methods are used in the models mentioned above, naive Bayes needs to satisfy the assumption that the distribution is independent. When the random forest is faced with the problems that more decision trees are needed, the time complexity and the space complexity are relatively large. If the samples have large noise, the random forest algorithm is prone to overfitting. Besides, the performance of the above models can only be better when dealing with the same datasets. When the category of the data changes, the accuracy of the models will be affected. Therefore, the above models have low robustness and poor generalization ability.

To solve the above problems and improve the accuracy of the CNNs on the dataset of the chest X-ray images detection for COVID-19, Normal, and Pneumonia. The momentum factor biogeography-based optimization is proposed to optimize the hyperparameters of three convolutional neural networks (LeNet-5, VGG-16, and ResNet-18). According to the global optimization characteristics of the momentum factor biogeography-based optimization, which can optimize the hyperparameters of the CNNs according to different categories of the input images. Therefore, it improves the robustness and generalization ability of models. In this paper, we proposed five novel **contributions** to improve the performances:

(i) A novel momentum factor biogeography-based optimization (MF-BBO) is proposed by optimizing the migration operation of biogeography-based.
(ii) Two novel frameworks – Biogeography convolutional neural network (BCNN) and Momentum factor biogeography convolutional neural network (MFBCNN) – are proposed.
(iii) We tested three configurations, viz., using LeNet-5, VGG-16, and ResNet-18 as backbones.
(iv) Two novel algorithms (BCNNC and MFBCNNC) are proposed for COVID-19. They are based on BCNN and MFBCNN tuned for the COVID-19 dataset.
(v) We find MFBCNNC gains superior results compared to state-of-the-art methods.

The structure of the rest is organized in the following way. Section 2 introduces the dataset of our experiments. Section 3 introduces the concepts, basic principles, and improved details of our proposed methods. Section 4 shows the experimental results and discussions for it. Finally, Section 5 concludes this paper.

[1] Those two authors contributed equally to this paper.

(a) COVID-19 chest X-ray image (b) Normal chest X-ray image (c) Pneumonia chest X-ray image

Fig. 1. Sample of chest X-ray images.

2. Dataset

To ensure the smooth running of the experiments and the validity of the experimental results. The dataset used in this paper is the chest X-ray images data from Kaggle. There are three categories of chest X-ray images in the dataset: COVID-19, Normal, and Pneumonia. Each category has 2,313 images, the total number of images is 6,939. We randomly selected 80% of each category of images as the training set. The remaining 20% images are as the test set. Considering the different sizes of each sample image, the sizes of all images are reset to 224×224 in the pre-processing images stage.

The samples of our dataset are shown in Fig. 2. The first row is 5 chest X-ray images of COVID-19. The second row is 5 chest X-ray images of Normal, and the last row is 5 chest X-ray images of Pneumonia.

It can be seen from Fig. 2 that the chest CT images of COVID-19 show clearly seen in the double lung ground glass shadow. The chest CT images of Normal show the symmetrical distribution of bilateral thorax, normal lung permeability, clear lung texture, no thickening, disorder, and abnormal lines. The chest CT images of Pneumonia show patchy lobar pulmonary parenchymal shadows with clear edges. We can distinguish the different categories of chest CT images by the human eyes using the features in the above mentioned. However, due to visual fatigue and other reasons, there may be some misdiagnosis by using human eyes.

3. Methodology

To ease the understanding of this paper, Table 13 shows all variables used in our study. Table 14 gives the abbreviation and their full names. Tables 13 and 14 are in the appendix at the end of the paper.

3.1. Convolutional neural network

To verify BBO and MF-BBO can effectively optimize the hyperparameters of the convolutional neural networks, we selected three convolutional neural networks (LeNet-5, VGG-16, and ResNet-18) for experiments. The network structure of these three models is gradually complex, and the number of network layers is gradually deepened. Therefore, the experimental results obtained are effective and convincing.

3.1.1. Lenet-5

LeNet-5 is a simple convolutional neural network, and its structure is shown in Fig. 3. LeNet-5 is mainly used for handwriting recognition [15]. All the convolution kernel size in LeNet-5 is 5×5; all the convolution kernel stride size is 1. All the size of the pooling kernel is 2×2, and all the stride size of the pooling kernel is 2. Here, the size of the convolution kernels and the stride size of the convolution kernels in the two convolutional layers in Fig. 3 are the optimization objectives of using BBO and MF-BBO in the following experiments.

As shown in Fig. 3, each convolutional layer in LeNet-5 is followed by a pooling layer. The formulas of convolution operation and pooling operation are shown as follows.

$$O_C = \left\lfloor \frac{X_C + 2L_C - V_C}{N_C} + 1 \right\rfloor \tag{1}$$

$$O_P = \left\lceil \frac{X_P + 2L_P - V_P}{N_P} + 1 \right\rceil \tag{2}$$

Here, O_C represents the output of the convolution operation. X_C represents the input of the convolution operation. L_C represents the size of padding. V_C represents the size of the convolution kernel, and N_C represents the stride size of the convolution kernel. O_P represents the output of the pooling operation. X_P represents the input of the pooling operation. V_P represents the size of the pooling kernel, and N_P represents the stride size of the pooling kernel.

3.1.2. VGG-16

Compared with LeNet-5, VGG-16 has a more complex structure [16]. This model participated in the 2014 ImageNet Image Classification and Positioning Challenge, and achieved excellent results: it ranked second in the classification tasks and first in the positioning tasks. The structure of VGG-16 is shown in Fig. 4. Here, the number of convolutional layers is thirteen, the convolution kernel size is 3×3, the convolution kernel stride size is 1, and the padding size is 1. There are five layers of pooling, the pooling kernel size is 2×2, the pooling kernel stride size is 2, and the padding size is 0 [17]. In the feature extraction part, for the first six layers, after every two consecutive convolutional layers, there is a pooling layer. For the remaining layers, after every three consecutive convolutional layers, there is a pooling layer. After that, there are three fully connected layers. In the experiments of this paper, we use BBO and MF-BBO to optimize the convolution kernel size and the convolution kernel stride size of the first four convolutional layers. As shown in Figs. 3 and 4, one of the characteristics of VGG-16 is the superimposed use of convolutional layers. Here, the convolution operation and the pooling operation still follow formulas (1) and (2).

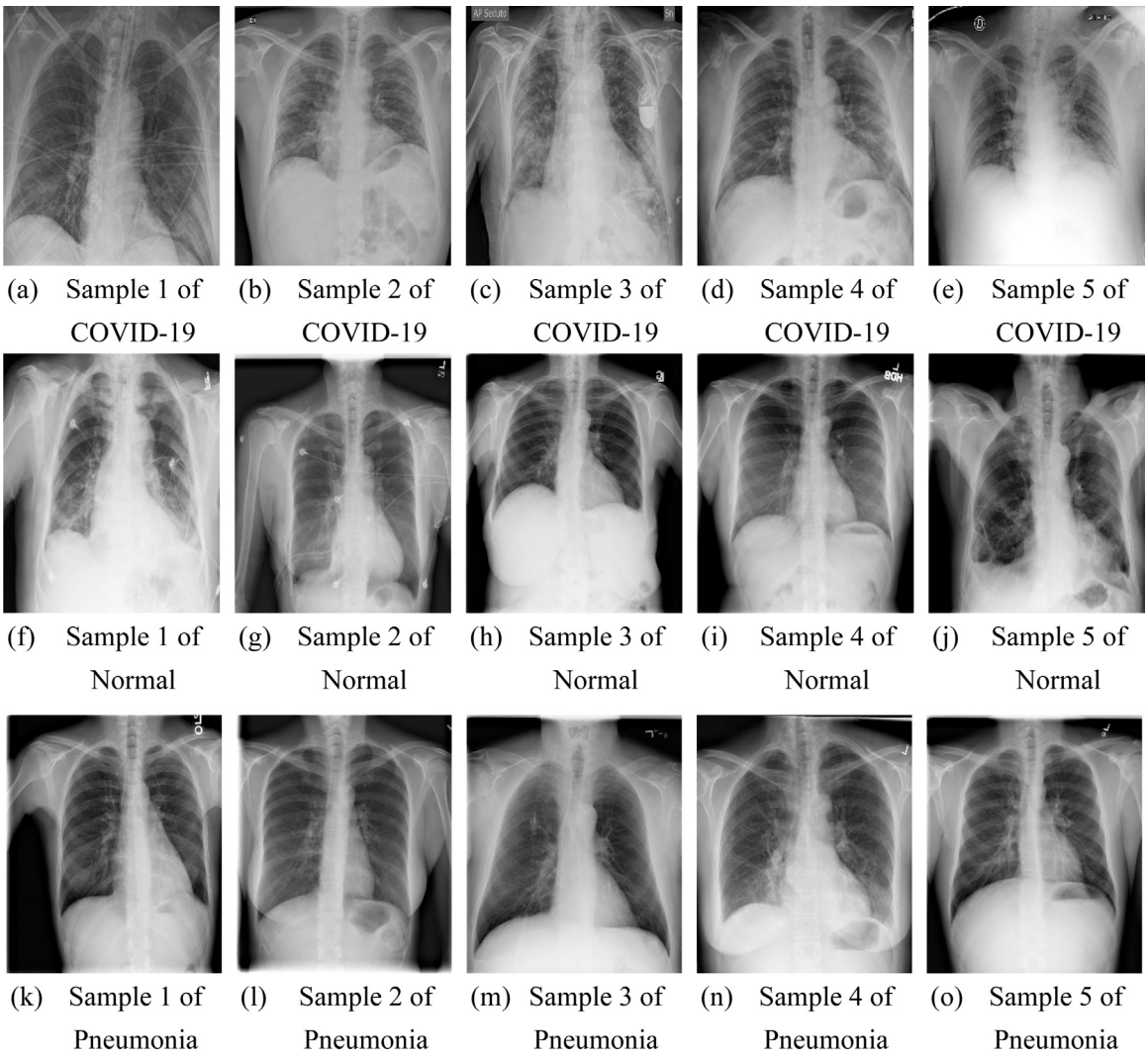


Fig. 2. Samples of chest X-ray images dataset.

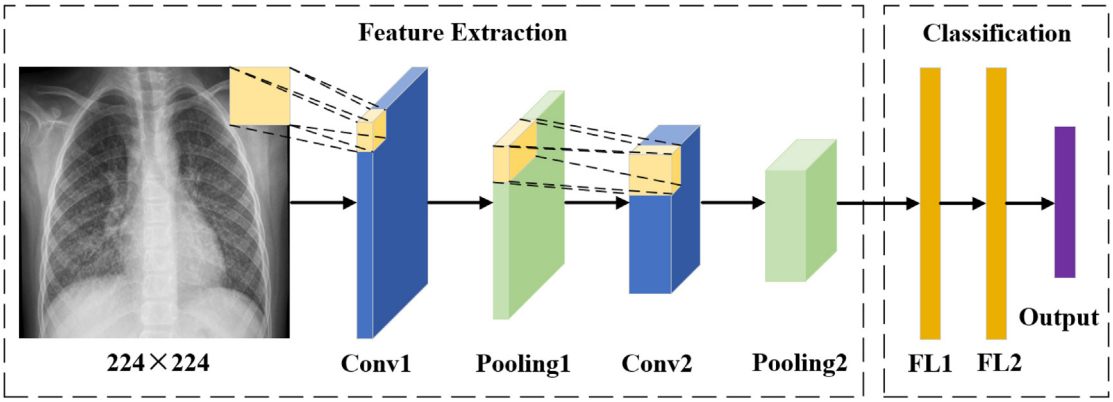


Fig. 3. Structure of LeNet-5.

3.1.3. ResNet-18

Compared with LeNet-5 and VGG-16, ResNet-18 introduces the residual block [18], as shown in the image inside the purple dotted line in Fig. 5. In ResNet-18, after the first pooling layer, there are 8 connect residual blocks, a total of 17 convolutional layers, 2 pooling layers, and a fully connected layer [19]. Here, in the first convolutional layer, the convolution kernel size is 7×7, the convolution kernel stride size is 2, and the padding size is 3. In the first pooling layer, the size of the pooling kernel is 3×3, the stride size of the pooling kernel is 2, and the padding size is 1. The size of the convolution kernel, the stride size of the convolution kernel, the size of the pooling kernel, and the stride size of the pooling kernel mentioned above are the optimized objectives of BBO and MF-BBO in the following experiments.

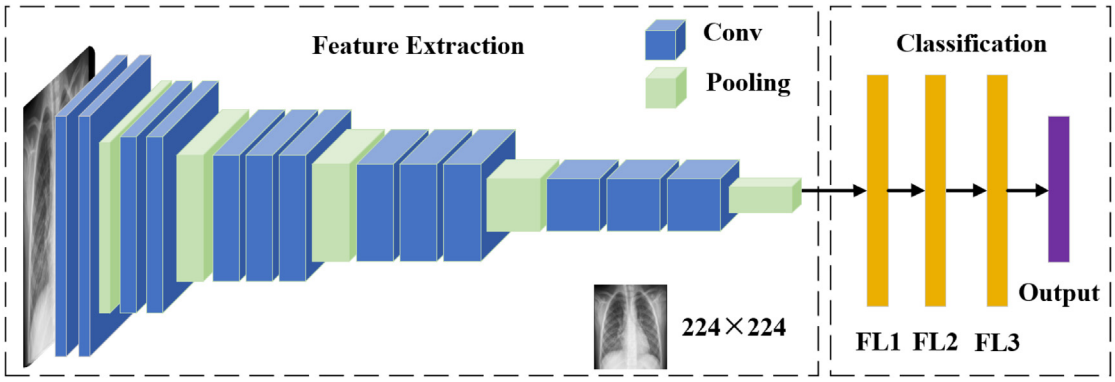


Fig. 4. Structure of VGG-16.

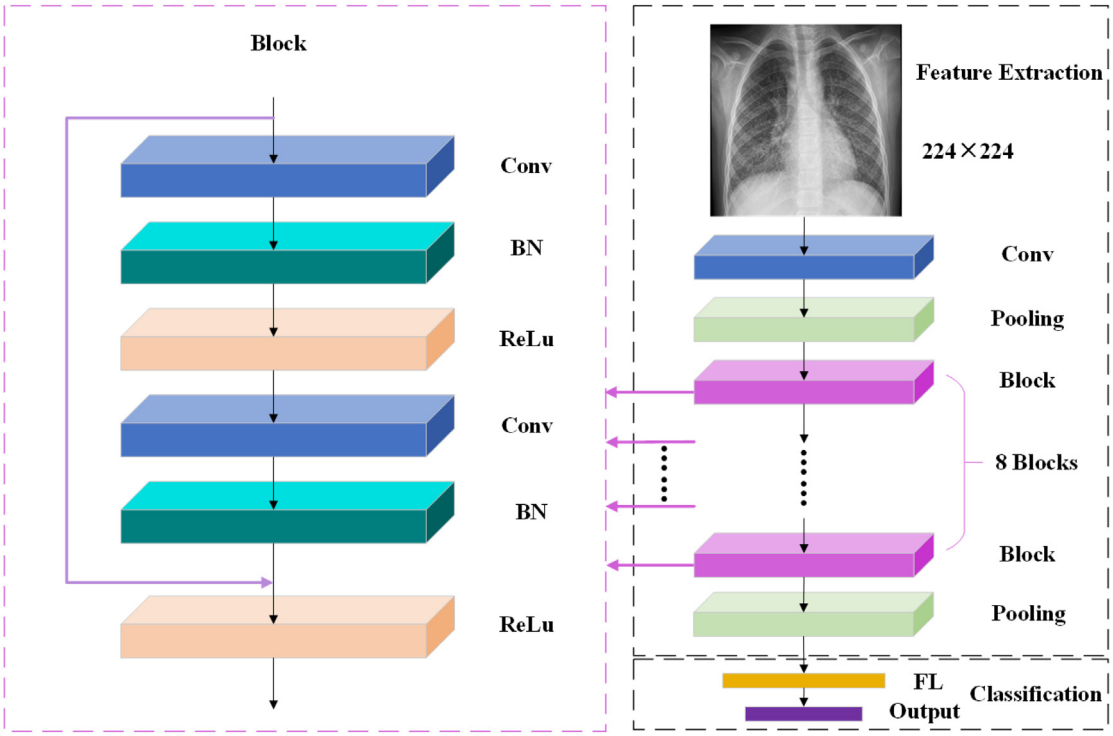


Fig. 5. Structure of ResNet-18.

3.2. Biogeography-based optimization

Biogeography-based optimization mainly comes from the theory of biogeography. The algorithm can promote the migration and mutation of species in different habitats by simulating and changing the conditions required for the survival of species in different habitats [20], so as to realize the information sharing among habitats, optimize and enhance the livability of the ecosystem. The habitats and migration paths of species are shown in Fig. 6. Here, the blue curve represents the migration paths of species between habitats. The remaining every single small icon represents a species habitat.

To describe the algorithm more accurately, this paper introduces the following term: habitat, which is used to describe the sites of species survival, reproduction, and mutation. Suitability index variables (SIV) are used to represent a certain factor affecting the number of species in a habitat, which can be regarded as the temperature, illumination time, rainfall, vegetation coverage, etc. [21]. Habitat suitability index (HSI) is used to indicate the suitability of habitats for species survival. SIV is a variable that needs to be considered when calculating HSI. At the same time, to better preserve the optimal solutions in the optimization process, BBO introduces elitism. When we use BBO to solve a problem, the ecosystem is regarded as the solution set of the problem, habitat is regarded as a solution to the problem, HSI is regarded as the applicability of a solution, and SIV is regarded as the independent variable of the problem. In the BBO optimization process, there are two important phases: migration and mutation. The following contents describe the migration operation and mutation operation, respectively.

3.2.1. Migration operation

Migration is an important operation of BBO performing optimization. The objective of the operation is SIV [22]. Migration operation consists of two operations: immigration and emigration. Considering BBO is used to optimize the hyperparameters of the convolutional neural networks, the linear migration model is helpful for the experiments [23]. Thus, the migration model of our experiment is shown as Fig. 7. Here, the immigration rate decreased with the increase of species number, and the emigration rate increased with the increase of species number [24].

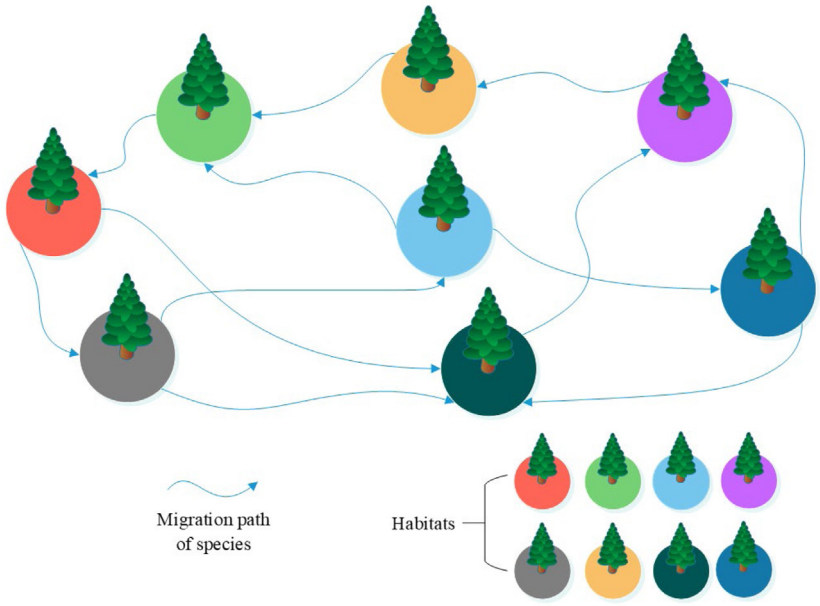


Fig. 6. BBO habitats and migration of species.

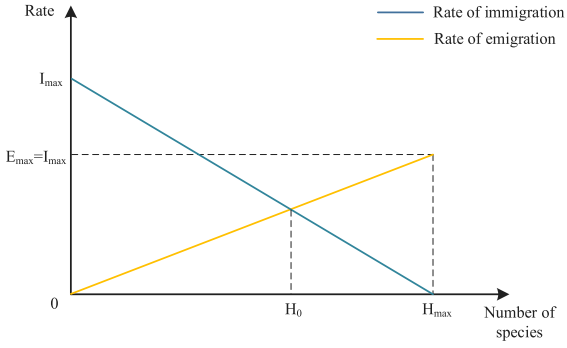


Fig. 7. Migration rate of species.

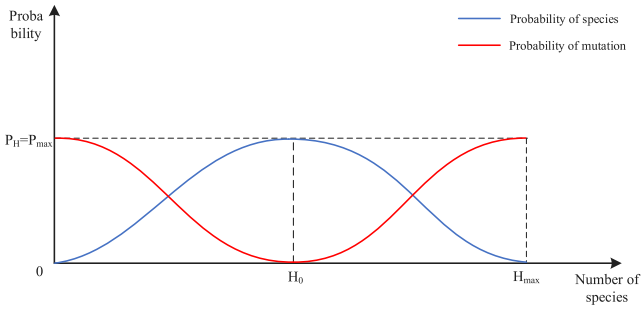


Fig. 8. Probability distribution of species and mutation.

When the species number of a habitat is 0, the immigration rate of the habitat is the highest, the emigration rate is 0, and the HSI value is the lowest. When the species number of a habitat reaches the maximum, the immigration rate of the habitat is 0, the emigration rate is the maximum, and the HSI value is the maximum [25]. Therefore, the following formulas can be obtained.

$$I_H = I_{max}(1 - \frac{H}{H_{max}}) \tag{3}$$

$$E_H = \frac{E_{max} \cdot H}{H_{max}} \tag{4}$$

Here, H represents the number of species. H_{max} represents the maximum number of species. I_H represents the immigration rate of the habitat that has H number species. I_{max} represents the maximum rate of immigration [26]. E_H represents the emigration rate of the habitat that has H number species. E_{max} represents the maximum rate of emigration.

The migration operation of BBO only select the immigration SIV and the emigration SIV, which is being relatively simple. Therefore, BBO can be improved on the migration phases. Section 3.3 elaborates on it.

3.2.2. Mutation operation

Mutation operation is another important operation of BBO. After migration operation, mutation operation randomly changes SIV in some habitats according to the mutation rate [27]. As the SIV value changes, the number of species in the habitat will be changed. Thus, the mutation operation improves the species number of the ecosystem [28], and enriches the diversity of candidate solutions. The formula for calculating the rate of mutation is as follows [29].

$$M_H = M_{max}(\frac{1 - P_H}{P_{max}}) \tag{5}$$

Here, M_H represents the rate of mutation with the number of species H. M_{max} represents the maximum rate of mutation. P_H represents the probability of the habitat with the number of species H [30]. P_{max} represents the maximum probability of species. The probability distribution of species and mutation as shown in Fig. 8.

According to formula (5) and Fig. 8, we can get the formula of P_H with a habitat that has the number of species H at time $(t+\Delta t)$ as shown follows.

$$P_H\,(t + \Delta t) = P_H\,(t)\,(1 - I_H - E_H) + P_{H-1}I_{H-1}\Delta t + P_{H+1}E_{H+1}\Delta t \tag{6}$$

By taking the derivative of Δt in (4), the following formula can be obtained.

$$\dddot{P}_H = \begin{cases} -P_H\left(I_H + E_H\right) + P_{H+1}E_{H+1}, H = 0 \\ P_{H-1}I_{H-1} - P_H\left(I_H + E_H\right) + P_{H+1}E_{H+1}, \quad 1 \le H \le H_{max} - 1 \\ P_{H-1}I_{H-1} - P_H\left(I_H + E_H\right), H = H_{max} \end{cases} \tag{7}$$

Here, $\dddot{P}_H$ represents the probability of species after the derivative. For simplicity, $\dddot{P}_H$ can be expressed as the multiplication of matrix A and P shown as below.

$$\dddot{P} = AP \tag{8}$$

$$A = \begin{bmatrix} -(I_0 + E_0) & E_1 & 0 & \cdots & 0 \\ I_0 & -(I_1 + E_1) & E_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & I_{max-2} & -(I_{max-1} + E_{max-1}) & E_{max} \\ 0 & \cdots & 0 & I_{max-1} & -(I_{max} + E_{max}) \end{bmatrix} \tag{9}$$

Fig. 9 shows the flow chart of BBO, and Table 1 shows the pseudocode of BBO. As can be seen from Fig. 9 and Table 1, BBO performs the migration operation after selecting the immigration SIV and emigration SIV (replace the immigration SIV with the emigration SIV). The experiments of this paper show that the migration operation not only limits the convergence speed of the algorithm, but also reduces the diversity of candidate solutions and increases the iteration times of searching for the optimal solution. To solve these problems, MF-BBO is proposed in Section 3.3. The migration operation of BBO is improved by introducing the migration momentum factor to enrich the diversity of candidate solutions, and it improves the convergence effect of BBO. It can be realized that within the same optimization time, MF-BBO can achieve better optimization effects than BBO.

3.3. Improvement I: Momentum factor biogeography-based optimization

The momentum factor biogeography-based optimization is the improvement of BBO, which is introduced from the following three aspects: (i) The backpropagation of convolutional neural network. (ii) The introduction of migration momentum operator and (iii) the migration operation of momentum factor biogeography-based optimization.

3.3.1. Backpropagation of feedforward neural network

The convolutional neural network is a feedforward neural network with a deep structure, including convolution computation. It is one of the representative algorithms of deep learning. In a feedforward neural network, each neuron has two attributes: weight and bias. The value of these two attributes directly affects the performance of the network model. Backpropagation (BP) using the loss value of the network in each iteration to automatically optimize the values of weight and bias, which plays an important role in model training. Therefore, the following formulas can be obtained.

$$z^{(l)} = W^{(l)} \cdot a^{(l-1)} + b^{(l)} \tag{10}$$

$$a^{(l)} = f_l(z^{(l)}) \tag{11}$$

Here, $z^{(l)}$ represents the output of layer l without activation function. $W^{(l)}$ represents the weight of layer l. $a^{(l-1)}$ represents the output value of layer $l-1$. $b^{(l)}$ represents the bias of layer l. $a^{(l)}$ represents the output of layer l. f_l represents the activation function of layer l. Therefore, a feedforward neural network can be regarded as a composite function, and the input x of the neural network can be regarded as $a^{(0)}$. The calculation formula of a feedforward neural network output is as follows.

$$x = a^{(0)} \to z^{(1)} \to a^{(1)} \to z^{(2)} \to \cdots \to a^{(l-1)} \to z^{(l)} \to a^{(l)} \tag{12}$$

Feedforward neural network uses backpropagation to optimize the parameters in the network. Assuming that input is (x, y), the loss function obtained after calculation of the feedforward neural network is $\mathcal{L}(y, \hat{y})$. To optimize the parameters in the feedforward neural network, the following formulas need to be calculated.

$$\frac{\partial \mathcal{L}(y,\hat{y})}{\partial W_{ij}^{(l)}} = \left(\frac{\partial z^{(l)}}{\partial W_{ij}^{(l)}}\right)^T \cdot \frac{\partial \mathcal{L}(y,\hat{y})}{\partial z^{(l)}} \tag{13}$$

$$\frac{\partial \mathcal{L}(y,\hat{y})}{\partial b^{(l)}} = \left(\frac{\partial z^{(l)}}{\partial b^{(l)}}\right)^T \cdot \frac{\partial \mathcal{L}(y,\hat{y})}{\partial z^{(l)}} \tag{14}$$

According to formulas (13) and (14), we still need to calculate $\frac{\partial z^{(l)}}{\partial W_{ij}^{(l)}}$, $\frac{\partial z^{(l)}}{\partial b^{(l)}}$ and $\frac{\partial \mathcal{L}(y,\hat{y})}{\partial z^{(l)}}$. The calculation methods are as follows.

$$\begin{aligned} \frac{\partial z^{(l)}}{\partial W_{ij}^{(l)}} &= \frac{\partial\left(W^{(l)} \cdot a^{(l-1)} + b^{(l)}\right)}{\partial W_{ij}^{(l)}} \\ &= \begin{bmatrix} \frac{\partial\left(W_{1:}^{(l)} a^{(l-1)} + b^{(l)}\right)}{\partial W_{ij}^{(l)}} \\ \vdots \\ \frac{\partial\left(W_{i:}^{(l)} a^{(l-1)} + b^{(l)}\right)}{\partial W_{ij}^{(l)}} \\ \vdots \\ \frac{\partial\left(W_{n^{(l)}:}^{(l)} a^{(l-1)} + b^{(l)}\right)}{\partial W_{ij}^{(l)}} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ a_j^{(l-1)} \\ \vdots \\ 0 \end{bmatrix} \end{aligned} \tag{15}$$

$$\frac{\partial z^{(l)}}{\partial b^{(l)}} = \frac{\partial\left(W^{(l-1)} \cdot a^{(l-1)} + b^{(l-1)}\right)}{\partial W^{(l)}} = Q_{n^{(l)}} \tag{16}$$

$$\begin{aligned} \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l)}} &= \frac{\partial a^{(l)}}{\partial z^{(l)}} \cdot \frac{\partial z^{(l+1)}}{\partial a^{(l)}} \cdot \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l+1)}} \\ &= f_l'\left(z^{(l)}\right) \odot \left(\left(W^{(l+1)}\right)^T \cdot \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l+1)}}\right) \end{aligned} \tag{17}$$

Here, $Q_{n^{(l)}}$ represents the identity matrix of the l-layer neurons. From formulas (15), (16), and (17), the following formulas can be obtained.

$$\frac{\partial \mathcal{L}(y,\hat{y})}{\partial W^{(l)}} = f_l'\left(z^{(l)}\right) \odot \left(\left(W^{(l+1)}\right)^T \cdot \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l+1)}}\right) \cdot \left(a^{(l-1)}\right)^T \tag{18}$$

$$\frac{\partial \mathcal{L}(y,\hat{y})}{\partial b^{(l)}} = f_l'\left(z^{(l)}\right) \odot \left(\left(W^{(l+1)}\right)^T \cdot \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l+1)}}\right) \tag{19}$$

Therefore, the execution sequence of BP in feedforward neural network is as follows: first (10) and (11); then (17); finally (18) and (19), so as to complete the updating the parameters in each layer of the neural network.

3.3.2. Backpropagation of convolutional neural network

The convolutional neural network has three important structures: the convolutional layer, the pooling layer, and the fully connected layer. Therefore, when the convolutional neural network performs BP, the corresponding solution methods are used for different structures of the network. For the convolutional layer, when ω^l of the convolutional layer is known, the following formulas should be followed when deriving W, b of the

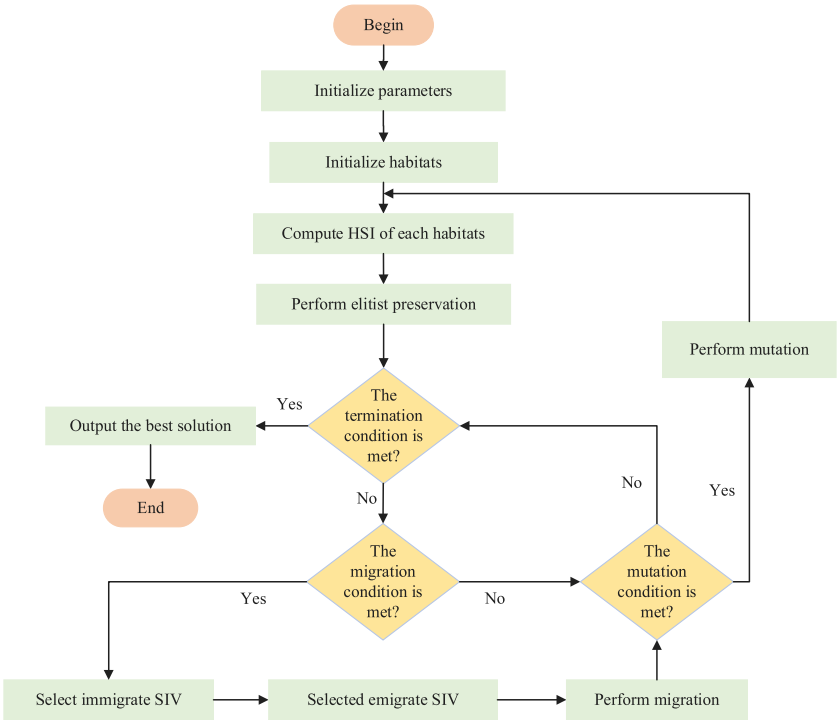


Fig. 9. Flow chart of BBO.

convolutional layer.

$$\omega^{(l)} = \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l)}} = \frac{\partial a^{(l)}}{\partial z^{(l)}} \cdot \frac{\partial z^{(l+1)}}{\partial a^{(l)}} \cdot \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l+1)}}$$
$$= f_l'\left(z^{(l)}\right) \odot \left(\left(W^{(l+1)}\right)^T \cdot \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l+1)}}\right) \tag{20}$$

$$\frac{\partial \mathcal{L}(y,\hat{y})}{\partial W^{(l)}} = \omega^{(l)} \cdot a^{(l-1)}$$
$$= f_l'\left(z^{(l)}\right) \odot \left(\left(W^{(l+1)}\right)^T \cdot \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l+1)}}\right) \cdot a^{(l-1)} \tag{21}$$

$$\frac{\partial \mathcal{L}(y,\hat{y})}{\partial b^{(l)}} = \sum_{i,j}\left(\omega^{(l)}\right)_{i,j}$$
$$= \sum_{i,j}\left(f_l'\left(z^{(l)}\right) \odot \left(\left(W^{(l+1)}\right)^T \cdot \frac{\partial \mathcal{L}\left(y,\hat{y}\right)}{\partial z^{(l+1)}}\right)\right)_{i,j} \tag{22}$$

Here, ω^l represents the loss function in the layer l. For the convolutional layer, refer to formulas (10), (17), and (20). When ω^l of the convolutional layer has been accumulated, the following formula should be followed when deriving the previous layer ω^{l-1}.

$$\omega^{(l-1)} = \left(\frac{\partial z^{(l)}}{\partial z^{(l-1)}}\right)^T \cdot \omega^{(l)} = \omega^{(l)} \cdot rot180\left(W^{(l)}\right) \odot f_{l-1}'\left(z^{(l-1)}\right) \tag{23}$$

When BP is performed in the pooling layer [31], the sizes of all matrices in ω^l will be restored to the size before pooling. This process is usually called upsampling. Therefore, when the ω^l of the pooling layer is known, the following formula should be followed when deriving the ω^{l-1} of the previous layer.

$$\omega^{(l-1)} = \left(\frac{\partial a^{(l-1)}}{\partial z^{(l-1)}}\right)^T \frac{\partial \mathcal{L}(y,\hat{y})}{\partial a^{(l-1)}} = h\left(\omega^{(l)}\right) \odot f_{l-1}'\left(z^{(l-1)}\right) \tag{24}$$

Here, h represents the upsampling function. For the BP of the fully connected layer, the calculation methods are the same as that of the feedforward neural network [32], and the formulas (10) to (19) in Section 3.3.1 can be referred to.

All the BP formulas in Sections 3.3.1 and 3.3.2 can be used to update the parameters of each layer in the convolutional neural network. Through the analysis of the above formulas, the reasonable value of each hyperparameter in the neural network is the main reason for better detection performance. When BBO is used to optimize the hyperparameters value of a convolutional neural network, the value of each SIV is the value of a corresponding hyperparameter in the neural network. Considering the migration of BBO is to directly replace the immigration SIV with the emigration SIV, and the emigration SIV is selected from the excellent habitat. For the hyperparameters of the convolutional neural network, all the hyperparameters value are a combination. The performance of the model is the evaluation criterion for the suitability of the hyperparameters value. However, changing only the value of one hyperparameter may not improve the performance of the model. It may cause the mismatch between the hyperparameters with the model and decline the accuracy of the model.

To solve the above problems, this paper proposes an improved migration method and introduces the migration momentum factor in Section 3.3.3. By setting the value of the migration momentum factor, the variable migration degree of migration operation in BBO can be reasonably adjusted to improve the optimization performance of BBO.

3.3.3. Migration momentum factor

The migration momentum factor is a variable introduced in the MF-BBO proposed in this paper. The main function of the migration momentum factor is to standardize the value of the migration SIV in the migration operation. The following formula is followed for the SIV migration.

$$SIV_{in} \leftarrow SIV_{out} \tag{25}$$

Here, the value of the immigration SIV is replaced by the value of the emigration SIV. Since the value of the emigration SIV is

Table 1
Pseudocode of BBO.

```
Initial parameters: the number of habitats, the maximum number of species, the rate of immigration
(I), the rate of emigration (E), the rate of mutation (M), elitist parameter
for each habitat
    initialize each SIV
end
for each habitat
    HSI = ObjectiveFunction (SIV [i, :])
    HSI_sorted = numpy. sort (HSI)
    preserve the optimal habitat
end
for each habitat
    for each SIV
        if Random. random () < I [habitat]
            randomnum = Random. random () * sum (E)
            select = E [1]
            selectindex = 0
            while (randomnum > select) and (selectindex < (habitats - 1))
                selectindex = selectindex + 1
                select = select + E [selectindex]
            end
            island [habitat, SIV] = habitat [selectindex, SIV]
        else
            island [habitat, SIV] = habitat [habitat, SIV]
    end
end
for each habitat
    for each SIV
        if M > Random. random ()
            island [habitat, parnum] = the lower limit of the parameter value range + (the upper
            limit of the parameter value range –the lower limit of the parameter value range) *
            Random. random()
        end
    end
end
while the maximum iterations number is not reached, or the optimal habitat does not meet the
conditions of the BBO termination
    execute BBO from the beginning again
end
Stop the iterations of BBO, output the optimized results
```

taken from the excellent habitat, it has a certain improvement effects on the immigration habitat. However, it can be seen from Section 3.3.2 that a hyperparameter with excellent value may not improve the performance of the model. The value of hyperparameters in each network layer of CNN could have a direct impact on the information processing effect of relevant network layer. The output of this network layer is the input of the next network layer. According to this information processing mode, the combination of hyperparameters value between different network layers could affect the performance of each network layer, which in turn affects the performance of the entire network model.

Therefore, the migration operation in BBO is not applicable for optimizing the hyperparameters value of convolutional neural network. And we need a more reasonable optimization method for the hyperparameters value of CNN. It could fully consider the correlation between the hyperparameters values of adjacent network layers and the importance of information processing performance of the network layer.

To solve this problem, the migration momentum factor is proposed by our team. By introducing the migration momentum factor in the migration operation of BBO, the algorithm does not directly exchange values after determining the immigration SIV and emigration SIV. Instead, a calculation operation is added to ensure the correlation between the optimized hyperparameters and the values of adjacent hyperparameters. The specific application steps are as follows.

When BBO selected the immigration SIV and the emigration SIV in the migration operation, it first performed the numerical comparison operation between the immigration SIV and the emigration SIV, then performed the value calculation operation. To complete the migration operation, the result of the previous step calculate is assigned to the immigration SIV by BBO.

The calculation of SIV by the migration momentum factor should follow the following formulas.

$$\dddot{SIV}_{out} = \begin{cases} \left\lceil \frac{SIV_{in}+SIV_{out}}{F} \right\rceil, SIV_{in} < SIV_{out} \\ \left\lfloor \frac{SIV_{in}+SIV_{out}}{F} \right\rfloor, SIV_{in} > SIV_{out}, \end{cases} 1 < F < 5 \quad (26)$$

$$SIV_{in} \leftarrow \dddot{SIV}_{out} \quad (27)$$

Here, F represents the migration momentum factor, $1 < F < 5$. $\dddot{SIV}_{out}$ represents the SIV of emigration calculated by the migration momentum factor. Since MF-BBO is applied to optimization the hyperparameters of CNN, the value of SIV corresponds to the value of the hyperparameters. Therefore, according to the formula (26), the value of F could directly affect the value of $\dddot{SIV}_{out}$. Considering the limitation of the input image size, in the hyperparameter optimization process of CNN, we default the candidate value of SIV to be less than 10. Therefore, F should be greater than 1. Otherwise, if $F < 0$ the value of $\dddot{SIV}_{out}$ will be negative, which does not meet the requirements of hyperparameters in CNN. If $0 < F < 1$, the value of $\dddot{SIV}_{out}$ will be amplified by tens or hundreds of times, which also does not meet the requirements of hyperparameters in CNN. In addition, F should be less than 5. Otherwise, the value of the hyperparameters tends to zero, which causes MF-BBO ineffective. Fig. 10 shows this more vividly.

In particular, the value of F in the formula (26) is set in accordance with the specific experimental conditions herein. When the method is applied to optimizing the hyperparameters value of other CNN models, the value range of F and the best value of F can be set according to the specific experimental conditions.

According to the formulas (26) and (27), if $SIV_{in} < SIV_{out}$, the $\dddot{SIV}_{out}$ is the integer value of the sum of SIV_{in} plus SIV_{out} divided by F. If $SIV_{in} > SIV_{out}$, the $\dddot{SIV}_{out}$ is the round down value of the sum of SIV_{in} plus SIV_{out} divided by F. Through the different values of F, it can control the value of $\dddot{SIV}_{out}$. That is, the value of F decide the value of $\dddot{SIV}_{out}$ close to or far from SIV_{out}, , as shown in formula (28). It is proved through experiments that when the value of F is 2, MF-BBO has the best optimization effects on the convolutional neural networks of this paper. Table 3 and Fig. 10 show the comparison of BBO and MF-BBO convergence effects with different values of F.

$$\dddot{SIV}_{out} = \begin{cases} \left\lceil \frac{SIV_{in}+SIV_{out}}{2} \right\rceil, SIV_{in} < SIV_{out} \\ \left\lfloor \frac{SIV_{in}+SIV_{out}}{2} \right\rfloor, SIV_{in} > SIV_{out} \end{cases} \quad (28)$$

On the premise of the same initialization method, the same objective function, and the same iteration times (iteration times = 500). It can be seen from Table 3 and Fig. 10 that for MF-BBO, when $0 < F < 1$, the convergence effect before 145 iterations is better than that of $F = 1$, and the convergence effect after 145 iterations is similar to that of $F = 1$, which is not as good as BBO. When $F = 1$, although MF-BBO is convergent, the convergence effect is poor, and the local convergence values are unstable. When $1 < F < 2$, the convergence effect of MF-BBO is better than that of BBO in the first 15 iterations, and which is not as good as BBO in the subsequent iterations. When $F = 2$, the convergence effect of MF-BBO is better than BBO. After the first iteration, the convergence value of MF-BBO is almost one-third less than that of BBO. When the number of iterations reaches 199, the MF-BBO ($F = 2$) almost completes the convergence operation. The numerical changes of adjacent iterations after that tend to be stable. When the number of iterations reaches 433, BBO almost completes the convergence operation. The numerical changes of adjacent iterations after that tend to be stable. When $2 < F < 5$, MF-BBO falls into the local optimal value after the first 17 iterations, and the convergence ability is lost in the subsequent iteration. Therefore, MF-BBO has the best performance of convergence when $F = 2$, and we set the value of F is 2 in the following experiments of this paper.

This section proposes an improved method for BBO. The BBO containing the migration momentum factor is called the momentum factor biogeography-based optimization (MF-BBO) by our team.

3.3.4. Migration operation of MF-BBO

Section 3.3.3 describes the function and the calculation methods of the migration momentum factor. This section describes how to apply the improved migration operation to MF-BBO. As shown in Fig. 9 and Table 1, the specific operations of BBO in the migration phase are as follows. The algorithm determines the immigration SIV and emigration SIV, fills the value of the emigration SIV into the location of the immigration SIV to complete the migration operation. The MF-BBO adds a numerical comparison and a calculation operation after determining the immigration SIV and the emigration SIV in the migration operation. The flowchart and the pseudocode for MF-BBO are shown below.

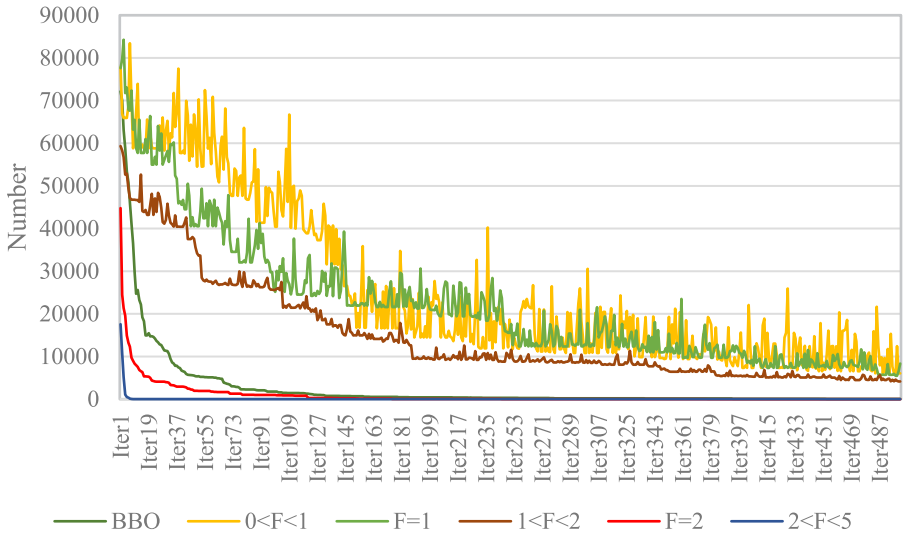
Note that our MF-BBO is different from Reference [33]. The differences are the following points: (i) In the migration phase of the algorithm, Reference [33] needs to find out the result of each pre-migration and its corresponding gradient, randomly select ten gradients to calculate their average gradient, and then uses the average gradient for subsequent calculation. In contrast, our method does not need to perform the above complex operations, so it is more concise and efficient than Reference [33]. (ii) According to Section 3, in the combination of hyperparameters in convolutional neural network, each hyperparameter is highly correlated with other hyperparameters. Compared with the linear migration model, the cosine migration model used in Reference [33] is more applicable for species migration in nature, but it is not applicable for the optimization of hyperparameters in the convolutional neural networks.

It can be seen from Fig. 11 and Table 2 that MF-BBO obtains the SIV_{in} after the calculation of migration momentum factor, and then performs the migration operation. Although MF-BBO adds an operation in the migration phase of BBO, through the analysis of the algorithm formulas and pseudocodes, it can be seen that the operation introduced by MF-BBO is a basic operation. The execution number of this program is a constant (no loop logic, only simple calculation, and numerical judgment). Therefore, the time complexity of the operation is O(1). It has no effect on the time complexity of algorithms.

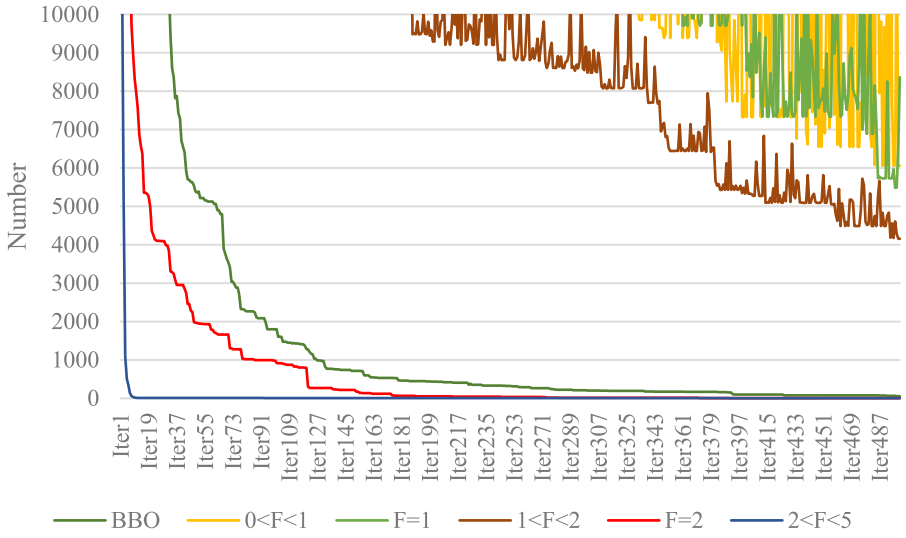
It can be seen from Fig. 10 that the optimization effect of MF-BBO is obvious and efficient compared with BBO. According to Table 3, the execution time difference between BBO and MF-BBO is only 0.5 s. Considering the more powerful optimization ability and convergence effect, our team thinks that the execution time difference of 0.5 s is acceptable.

3.4. Improvement II: Two proposed frameworks – BCNN and MF-BCNN

Compared with other optimization algorithms, BBO has fewer parameters, better performance of global optimization, easier to implement. It is good at solving problems of high-dimensional and multi-objective optimization. Therefore, two novel frameworks – Biogeography convolutional neural network (BCNN) and Momentum factor biogeography convolutional neural network (MFBCNN) – are proposed by our team. Here, BBO is proposed to optimize the hyperparameters of BCNN, and MF-BBO is proposed to optimize the hyperparameters of MFBCNN. MF-BBO improves



(a) The convergence effect of different algorithms



(b) The local graph of different algorithms convergence effect

Fig. 10. Comparison of convergence effect between BBO and MF-BBO with different values of F.

the migration operation of BBO. Fig. 12 shows the corresponding relationship between the parameters in BBO, MF-BBO, and CNN.

Fig. 13 shows the framework of our method. First, get the chest X-ray image from the subjects, and input the images into three CNN models (LeNet-5, VGG-16, and ResNet-18). Then BBO and MF-BBO start to optimize the hyperparameters value of CNN. When the output of CNN meets the end condition of optimization algorithm, it stops the optimization and output the optimized result; otherwise, BBO and MF-BBO continue to perform optimization.

3.5. Improvement III: Two proposed algorithms – BCNNC and MFBCNNC

This section introduces two algorithms proposed by our team: BCNNC – BCNN for COVID-19 and MFBCNNC – MFBCNN for COVID-19. In this paper, we define BCNNC-I to represent the LeNet-5 optimized by BBO for COVID-19, BCNNC-II to represent the VGG-16 optimized by BBO for COVID-19, BCNNC-III to represent the Resent-18 optimized by BBO for COVID-19, MFBCNNC-I to represent the LeNet-5 optimized by MF-BBO for COVID-19, MFBCNNC-II to represent the VGG-16 optimized by MF-BBO for COVID-19, and MFBCNNC-III to represent the Resent-18 optimized by MF-BBO for COVID-19. Fig. 14, Fig. 15, and Fig. 16 are schematic diagrams for optimizing hyperparameters of convolutional neural networks using BBO and MF-BBO. Here, Conv represents the convolutional layer. Pooling represents the pooling layer. BN represents batch normalization. ReLu represents the ReLu activation function.

As can be seen from Fig. 14, we use BBO and MF-BBO to optimize all convolution kernel sizes and convolution kernel stride sizes of LeNet-5.

As can be seen from Fig. 15, we use BBO and MF-BBO to optimize the convolution kernel sizes and the convolution kernel stride sizes in the first four convolutional layers of VGG-16.

As can be seen from Fig. 16, we use BBO and MF-BBO to optimize the convolution kernel size and the convolution kernel stride size in the first convolutional layer, the pooling kernel size,

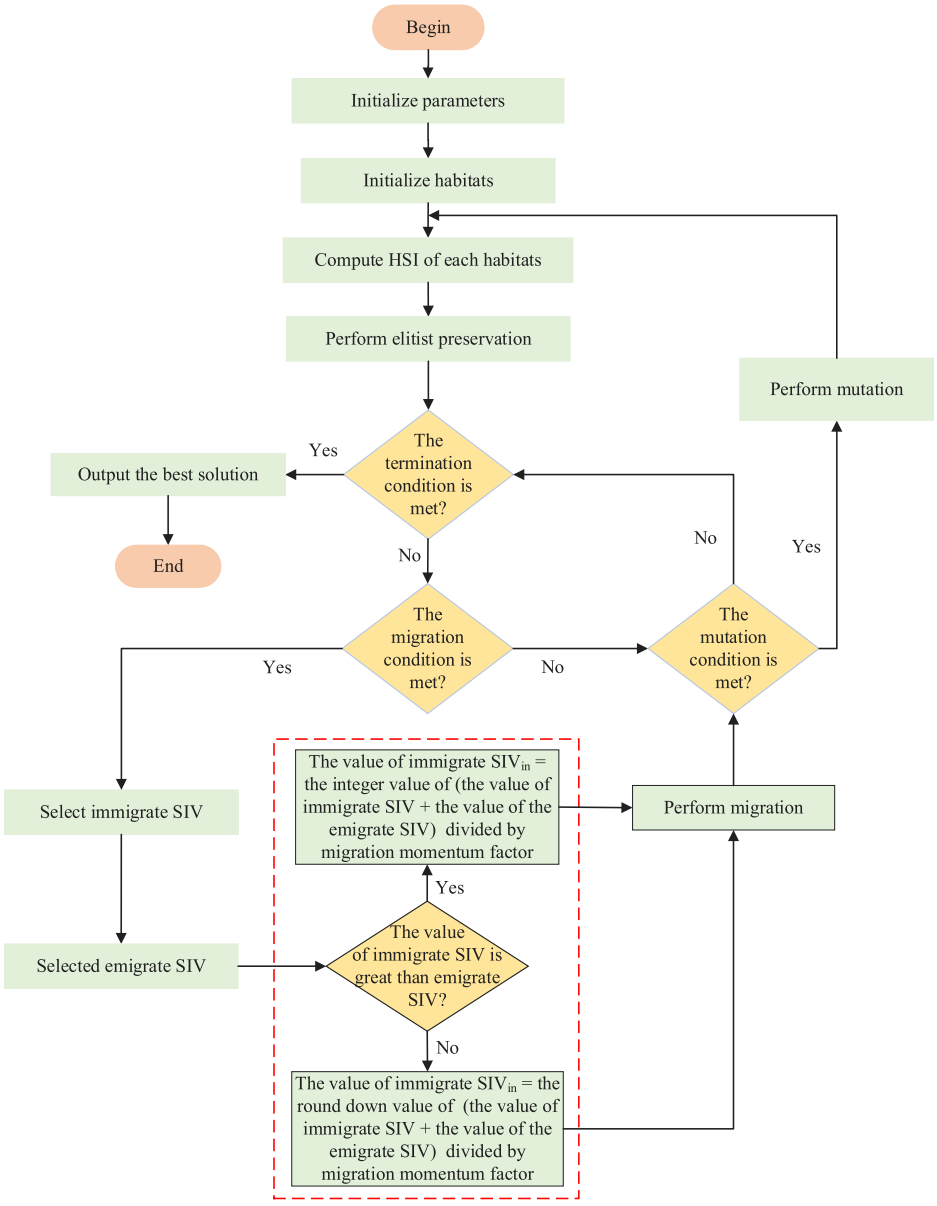


Fig. 11. Flow chart of MF-BBO.

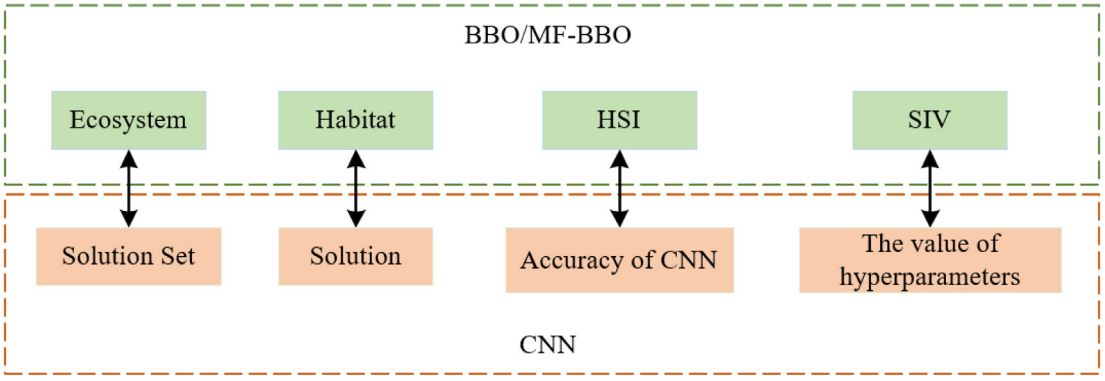


Fig. 12. The corresponding relationship between BBO, MF-BBO and CNN.

and the pooling kernel stride size in the first pooling layer of ResNet-18.

When BBO and MF-BBO perform hyperparameters optimization for the convolutional neural networks, the specific steps are as follows.

Table 2
Pseudocode of MF-BBO.

```
Initial parameters: the number of habitats, the maximum number of species, the rate of immigration
(I), the rate of emigration (E), the rate of mutation (M), elitist parameter, the migration momentum
factor (F)
for each habitat
    initialize each SIV
end
for each habitat
    HSI = ObjectiveFunction (SIV [i, :])
    HSI_sorted = numpy. sort (HSI)
    preserve the optimal habitat
end
for each habitat
    for each SIV
        if Random. random () < I [habitat]
            randomnum = Random. random () * sum (E)
            select = E [1]
            selectindex = 0
            while (randomnum > select) and (selectindex < (habitats - 1))
                selectindex = selectindex + 1
                select = select + E [selectindex]
            end
            if pos [0, SIV] > pos [selectindex, SIV]:
                move = math. ceil ((pos [0, SIV] + pos [selectindex, SIV]) / F)
            else
                move = math. floor ((pos [0, SIV] + pos [selectindex, SIV]) / F)
            pos [selectindex, SIV] = move
            island [habitat, SIV] = habitat [selectindex, SIV]
        else
            island [habitat, SIV] = habitat [habitat, SIV]
    end
end
for each habitat
    for each SIV
        if M > Random. random ()
            island [habitat, parnum] = the lower limit of the parameter value range + (the upper
            limit of the parameter value range –the lower limit of the parameter value range) *
            Random. random()
        end
    end
end
while the maximum iterations number is not reached, or the optimal habitat does not meet the
conditions of the BBO termination
    execute BBO from the beginning again
end
Stop the iterations of BBO, output the optimized results
```

Initialization operation: iteration times, number of habitats, dimension and values range of SIV, immigration rate, emigration rate, mutation rate, elitism parameters. The HSI and species number of each habitat are calculated by the objective function. Sort the HSIs in descending order. The habitat with the largest

Table 3
Execution time of BBO and MF-BBO.

Optimizer	Number of iterations	Start time	End time	Execution time
BBO	500	2020/10/08/17:24:46	2020/10/08/17:24:50	3.99268818
MF-BBO	500	2020/10/08/17:25:45	2020/10/08/17:25:49	4.41598773

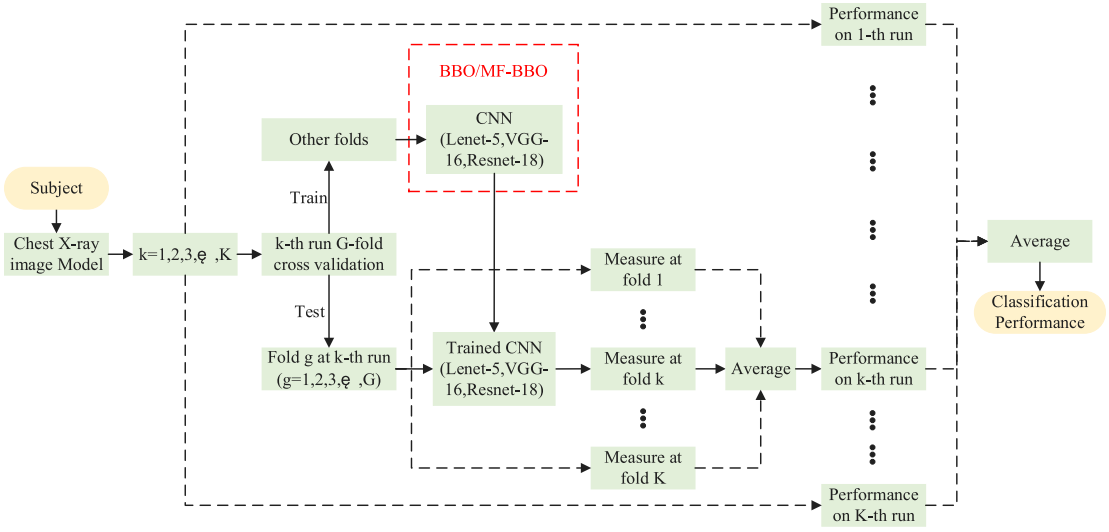


Fig. 13. Block diagram of our method.

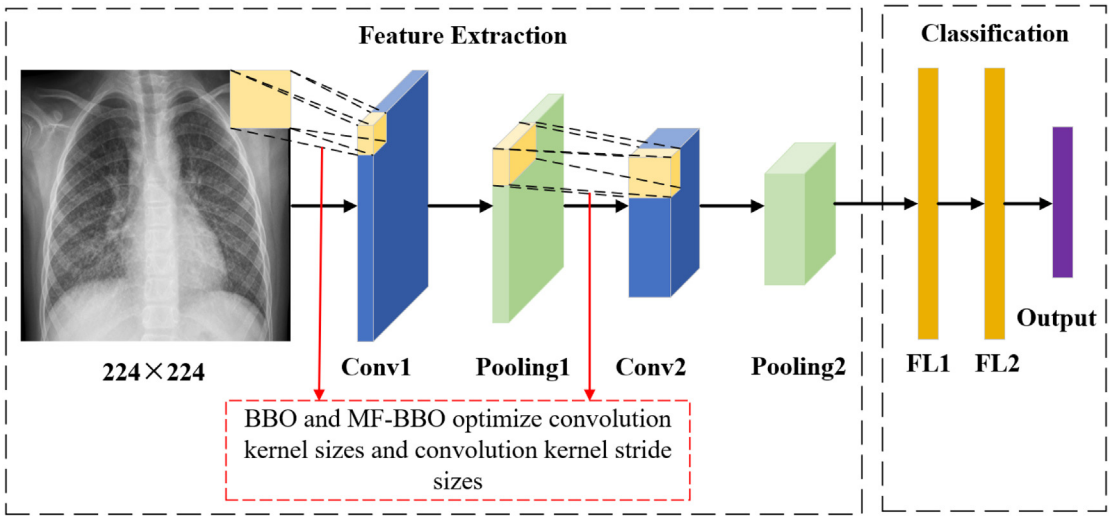


Fig. 14. BBO and MF-BBO optimize the hyperparameters of LeNet-5.

number of species is numbered as 1, and $u(c)_{max} = U(C)_{min}$. Here, c represents the index of habitats. C represents the number of habitats. u represents the value of HSI. U represents the index of ordinary numbers. In this case, $U(C)_{min}$ corresponds to the habitat with the largest number of species, the highest HSI value, and the optimal solution in each iteration process. Finally, elitism preserves the optimal solution. The algorithm begins to perform migration and mutation operations.

Migration operation (BBO): the random number R is generated by the random function, which is compared with the immigration rate I of each habitat. When $R(SIV) < I(SIV)$, select the corresponding SIV as the immigration SIV; otherwise, the algorithm does not perform the immigration operation. Then, the random number S is generated by the random function, which is compared with the emigration rate E of each habitat. When $S(SIV) < E(SIV)$, select the corresponding SIV as the emigration SIV; otherwise, the algorithm does not perform the emigration operation. After selecting the immigration SIV and the emigration SIV, move the emigration SIV to the location of the immigration SIV to complete the migration operation.

Migration operation (MF-BBO): the random number R is generated by the random function, which is compared with the immigration rate I of each habitat. When $R(SIV) < I(SIV)$, select the corresponding SIV as the immigration SIV; otherwise, the algorithm does not perform the immigration operation. Then, the random number S is generated by the random function, which is compared with the emigration rate E of each habitat. When $S(SIV) < E(SIV)$, select the corresponding SIV as the emigration SIV. Otherwise, the algorithm does not perform the emigration operation. After selecting the immigration SIV and the emigration SIV, the algorithm performs the formula (28) to obtain the

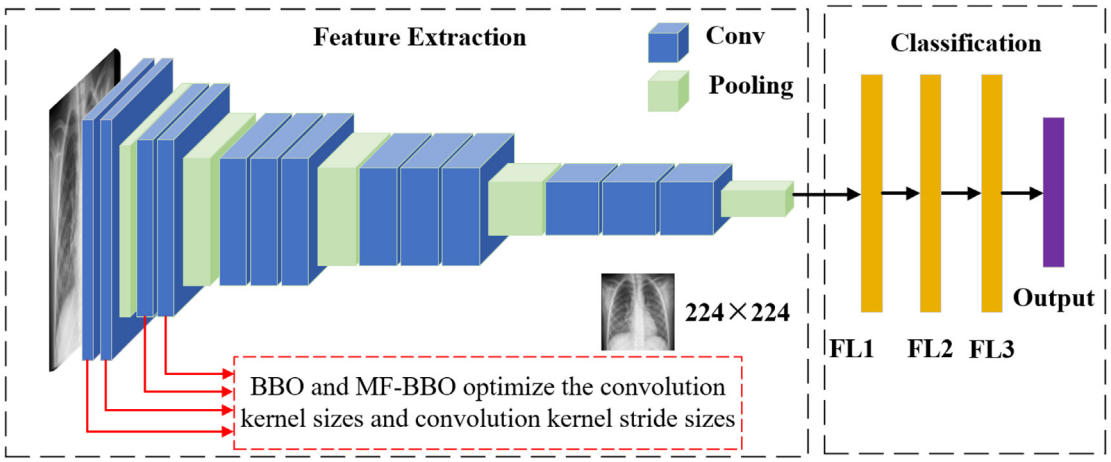


Fig. 15. BBO and MF-BBO optimize the hyperparameters of VGG-16.

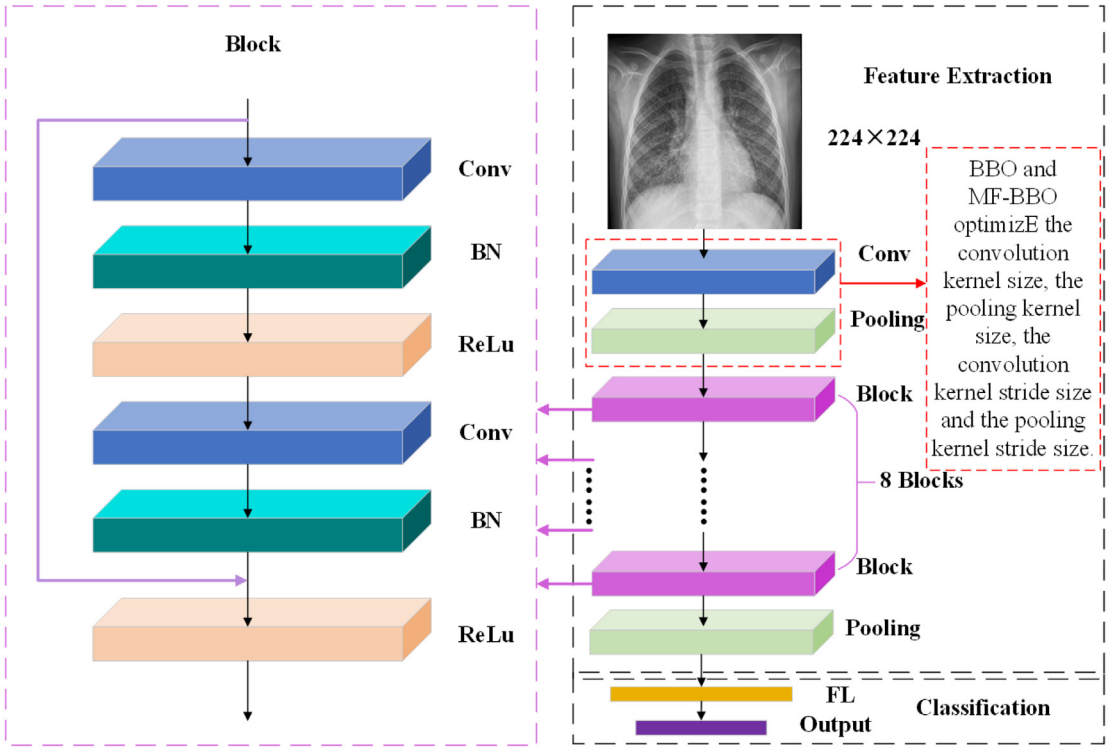


Fig. 16. BBO and MF-BBO optimize the hyperparameters of ResNet-18.

emigration $\overset{\cdots}{SIV}$. Move the emigration $\overset{\cdots}{SIV}$ to the location of the immigration SIV to complete the migration operation.

Mutation operation: the random number T is generated by the random function, which is compared with the variation rate M of each habitat. When $T(SIV) < M(SIV)$, the algorithm performs the mutation operation to the corresponding SIV; otherwise, the algorithm does not perform the mutation operation.

Elitism operation: the algorithm recalculates and sorts HSIs according to their numerical value after the mutation operation. All the corresponding habitats at the bottom of the list are replaced with those preserved by elitism in the initialization operation.

Termination condition judgment: the repeat habitats are removed to obtain the ecosystem. The HSI is recalculated and sorted by the algorithm. Determine whether the optimal solution $U(C)_{min}$ satisfies the termination condition. If it satisfies the termination condition, the iteration is terminated, and the result is output. If it does not satisfy the termination condition, the algorithm continues to perform the iteration.

3.6. Measure

To prevent overfitting, we introduce 10-fold cross-validation [34] in our experiments. The specific implementation method is as follows: The dataset in this paper is randomly divided into ten subsets and numbered from 1 to 10. Select subset No.1 as a test set; the other subsets are as a training set. All these subsets are as the dataset D1. Select subset No.2 as a test set; the other subsets are as a training set. All these subsets are as the dataset D2. Repeat the above operation in turn until subset No.10 is selected as a test set; the other subsets are as a test set. All these subsets

Table 4
Confusion matrix.

Confusion Matrix		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

are as the dataset D10. Finally, experiments are carried out on D1, D2, …, D10, respectively.

The experimental results are summation and averaged to evaluate the optimization performances of the algorithms on the hyperparameters of CNNs through accuracy. To describe the experimental results more accurately, we perform ten times 10-fold cross-validation, and we introduce the confusion matrix as shown in Table 4. Here, TP represents both the true value and the predicted value are positive. FN represents the true value is positive, but the predicted value is negative. FP represents the true value is negative, but the predicted value is positive. TN represents both the true value and the predicted value are negative.

An ideal 10-fold cross-validation confusion matrix should be as shown in formula (29). An ideal 10-fold cross-validation confusion matrix for 10 runs should be as shown in formula (30). K represents the number of repetitions (k loops from 1 to K). G represents the number of folds (g loops from 1 to G).

$$\delta\,(K=1, G=10)=\begin{bmatrix} 463 & 0 & 0 \\ 0 & 463 & 0 \\ 0 & 0 & 463 \end{bmatrix} \quad g=1,2,\ldots,G \tag{29}$$

$$\delta\,(K=10, G=10)=\begin{bmatrix} 4630 & 0 & 0 \\ 0 & 4630 & 0 \\ 0 & 0 & 4630 \end{bmatrix} \quad g=1,2,\ldots,G \tag{30}$$

Besides, we also introduce defined four metrics: Overall accuracy (OA), Precision, Sensitive (Recall), and Specificity. Their formulas are as follows.

$$\mathrm{OA}=\frac{TP+TN}{TP+TN+FP+FN} \tag{31}$$

$$\mathrm{Precision}=\frac{TP}{TP+FP} \tag{32}$$

$$\mathrm{Sensitive}=\mathrm{Recall}=\frac{TP}{TP+FN} \tag{33}$$

$$\mathrm{Specificity}=\frac{TN}{TN+FP} \tag{34}$$

$$\mathrm{F}_1=\frac{2{*}Precision{*}Sensitivity}{Precision+Sensitivity} \tag{35}$$

4. Experiment results and discussions

In this section, the experimental results are compared and discussed to prove that the MF-BBO, BCNN, MFBCNN, BCNNC, and MFBCNNC proposed in this paper are practical and effective in practical application.

4.1. Confusion matrix of three methods

4.1.1. Confusion matrix without BBO method

Table 5 shows the confusion matrix of 10-fold cross-validation for chest X-ray images with three convolutional neural networks using default values of hyperparameters. Here, COV represents the chest X-ray images of COVID-19. Nor represents the chest X-ray images of Normal. Pne represents the chest X-ray images of Pneumonia. Table 6 shows the four confusion matrix metrics of three convolutional neural networks.

According to the above contents, the accuracy of ResNet-18 is the highest, followed by LeNet-5, and VGG-16. All three convolutional neural networks have the highest precision for Pneumonia, the highest sensitivity to COVID-19. At the same time, the specificity of Pneumonia is the highest of three convolutional neural networks. The remaining detailed data is shown in Table 6.

4.1.2. Confusion matrix with BBO method

Table 7 shows the confusion matrix of three convolutional neural networks optimized by BBO to perform 10-fold cross-validation on the chest X-ray images. Here, COV represents the chest X-ray images of COVID-19. Nor represents the chest X-ray images of Normal. Pne represents the chest X-ray images of Pneumonia. Table 8 shows the four confusion matrix metrics of three convolutional neural networks with BBO.

According to the above contents, the accuracy of the BCNNC-III is the highest, followed by the BCNNC-I, and the BCNNC-II. All three BCNNC models have the highest precision for Pneumonia. The BCNNC-I has the highest sensitivity to COVID-19. The BCNNC-II and the BCNNC-III both have the highest sensitivity to Normal. At the same time, the specificity of Pneumonia is the highest of three BCNNC models. The remaining detailed data is shown in Table 8.

4.1.3. Confusion matrix with MF-BBO method

Table 9 shows the confusion matrix of three convolutional neural networks optimized by MF-BBO to perform 10-fold cross-validation on the chest X-ray images. Here, COV represents the chest X-ray images of COVID-19. Nor represents the chest X-ray image of Normal. Pne represents the chest X-ray images of Pneumonia. Table 10 shows the four confusion matrix metrics of three convolutional neural networks with MF-BBO.

According to the above contents, the accuracy of the MFBCNNC-II is the highest, followed by the MFBCNNC-I, and the MFBCNNC-III. The MFBCNNC-I and the MFBCNNC-II both have the highest precision for Pneumonia. The MFBCNNC-III has the highest precision for Normal. The MFBCNNC-I and the MFBCNNC-III both have the highest sensitivity to COVID-19. The MFBCNNC-II has the highest sensitivity to Normal. At the same time, the MFBCNNC-I and the MFBCNNC-II both have the highest specificity to Pneumonia. The MFBCNNC-III has the highest specificity to Normal. The remaining detailed data is shown in Table 10.

4.2. Statistical results

In this section, we list the average overall accuracy (OA) of the nine methods run 10 times. The specific data is shown in Table 11. Fig. 17 shows the average accuracy of the nine methods.

Table 11 shows the statistical results of the nine methods. The average OA of the LeNet-5 (BBO) is 1.56% higher than that of LeNet-5. The average OA of the LeNet-5 (MF-BBO) is 2.87% higher than that of the LeNet-5, which is 1.56% higher than that of the LeNet-5 (BBO). The average OA of the VGG-16 (BBO) is 1.48% higher than that of VGG-16. The average OA of the VGG-16 (MF-BBO) is 6.31% higher than that of the VGG-16. The average OA of the VGG-16 (MF-BBO) is 4.83% higher than that of the VGG-16 (BBO). The average OA of the ResNet-18 (BBO) is 0.73% higher than that of ResNet-18. The average OA of ResNet-18 (MF-BBO) is 1.46% higher than that of the ResNet-18, which is 0.73% higher than that of the ResNet-18 (BBO). Therefore, BBO and MF-BBO are effective for the hyperparameters optimization of CNNs. And the optimization effect of MF-BBO is better than that of BBO.

It can be seen from Fig. 17 that BBO has the best optimization effect on LeNet-5, and the OA improved by 1.56%. MF-BBO has

Table 5
Confusion matrix of three methods.

Confusion Matrix		LeNet-5 Predicted Class			VGG-16 Predicted Class			ResNet-18 Predicted Class		
		COV	Nor	Pne	COV	Nor	Pne	COV	Nor	Pne
Actual Class	COV	4317	262	51	4213	181	236	4310	158	162
	Nor	342	4247	41	644	3816	170	268	4275	87
	Pne	104	799	3727	95	351	4184	74	528	4028

Table 6
Four confusion matrix metrics of three methods.

	OA	Precision			Sensitive			Specificity		
		COV	Nor	Pne	COV	Nor	Pne	COV	Nor	Pne
LeNet-5	88.49%	90.64%	80.01%	97.59%	93.24%	91.73%	80.50%	95.18%	88.54%	99.01%
VGG-16	87.93%	85.08%	87.76%	91.15%	90.99%	82.42%	90.37%	92.02%	94.25%	95.62%
ResNet-18	90.81%	92.65%	86.17%	94.18%	93.09%	92.33%	87.00%	96.31%	92.59%	97.31%

Table 7
Confusion matrix of three methods (BBO).

Confusion Matrix		BCNNC-I Predicted Class			BCNNC-II Predicted Class			BCNNC-III Predicted Class		
		COV	Nor	Pne	COV	Nor	Pne	COV	Nor	Pne
Actual Class	COV	4332	191	107	4191	328	111	4292	241	97
	Nor	411	4167	52	285	4282	63	156	4343	131
	Pne	89	544	3997	108	573	3949	117	417	4096

Table 8
Four confusion matrix metrics of three methods (BBO).

	OA	Precision			Sensitive			Specificity		
		COV	Nor	Pne	COV	Nor	Pne	COV	Nor	Pne
BCNNC-I	89.96%	89.65%	85.01%	96.17%	93.56%	90.00%	86.33%	94.60%	92.06%	98.28%
BCNNC-II	89.43%	91.43%	82.62%	95.78%	90.52%	92.48%	85.29%	95.76%	90.27%	98.12%
BCNNC-III	91.66%	94.02%	86.84%	94.73%	92.70%	93.80%	88.47%	97.05%	92.89%	97.54%

Table 9
Confusion matrix of three methods (MF-BBO).

Confusion Matrix		MFBCNNC-I Predicted Class			MFBCNNC-II Predicted Class			MFBCNNC-III Predicted Class		
		COV	Nor	Pne	COV	Nor	Pne	COV	Nor	Pne
Actual Class	COV	4355	153	122	4415	78	137	4343	156	131
	Nor	308	4061	261	132	4439	59	246	4046	338
	Pne	93	276	4261	113	265	4252	144	201	4285

Table 10
Four confusion matrix metrics of three methods (MF-BBO).

	OA	Precision			Sensitive			Specificity		
		COV	Nor	Pne	COV	Nor	Pne	COV	Nor	Pne
MFBCNNC-I	91.27%	91.57%	90.45%	91.75%	94.06%	87.71%	92.03%	95.67%	95.37%	95.86%
MFBCNNC-II	94.36%	94.74%	92.83%	95.59%	95.36%	95.87%	91.84%	97.35%	96.30%	97.88%
MFBCNNC-III	91.25%	91.76%	91.89%	90.13%	93.80%	87.39%	92.55%	95.79%	96.14%	94.94%

Table 11
Average OA with nine methods.

Run	LeNet-5	LeNet-5 (BBO)	LeNet-5 (MF-BBO)	VGG-16	VGG-16 (BBO)	VGG-16 (MF-BBO)	ResNet-18	ResNet-18 (BBO)	ResNet-18 (MF-BBO)
	OA	OA	OA	OA	OA	OA	OA	OA	OA
1	88.15	90.10	91.13	89.34	89.09	94.40	91.91	92.37	91.63
2	88.49	88.80	89.80	87.93	88.24	94.41	91.56	90.80	91.25
3	88.62	89.31	89.91	90.71	91.05	94.71	90.71	91.63	92.30
4	88.36	90.92	92.29	85.79	89.36	95.01	90.76	91.79	92.40
5	88.73	89.62	91.92	89.44	89.43	94.05	89.41	92.08	92.04
6	88.42	89.29	90.62	88.99	89.11	94.48	91.06	89.87	92.65
7	88.25	90.31	91.31	87.07	87.82	93.79	91.29	91.40	92.56
8	88.76	91.04	91.04	88.72	90.54	95.55	90.81	91.71	92.79
9	88.08	90.30	90.30	85.41	90.53	93.61	90.45	90.80	92.59
10	88.23	89.96	91.96	87.22	90.27	93.71	90.12	92.90	92.48
Average	88.41 ± 0.24	89.97 ± 0.72	91.28 ± 0.87	88.06 ± 1.69	89.54 ± 1.05	94.37 ± 0.61	90.81 ± 0.72	91.54 ± 0.87	92.27 ± 0.49

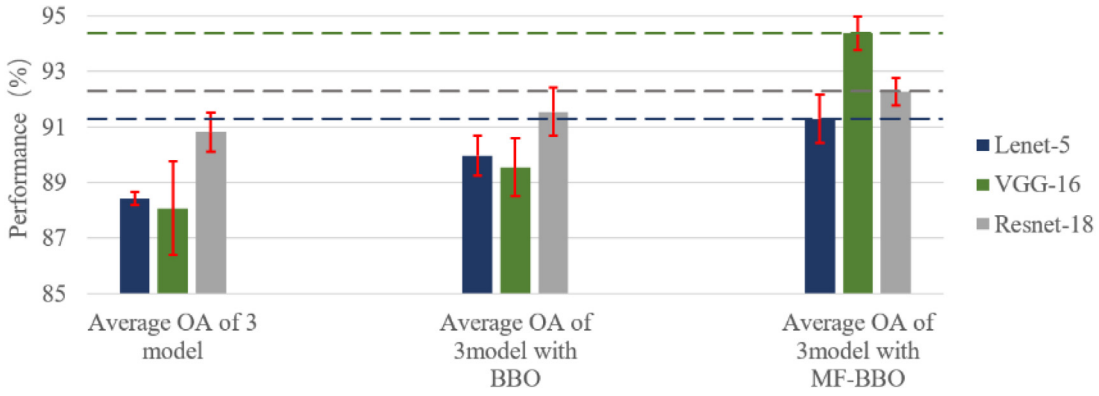


Fig. 17. Average OA of 3 models with 3 methods.

the best optimization effect on VGG-16, and the OA improved by 6.31%. Our team believes that the main reasons for this result are as follows: (i) Two optimization algorithms optimize all the convolution kernel sizes and convolution kernel stride sizes in LeNet-5, which improves the feature extraction ability of the model on input images. However, the LeNet-5 structure only has five layers, and its structure has been relatively simple. Therefore, the detection of multi-classification problems has structural deficiencies, which cannot improve the OA of the model to a greater extent. (ii) The sizes of the first two convolution kernels and the convolution kernels stride size in the VGG-16 are optimized, which improves the feature extraction ability of the model on the input images. On the other hand, the structure of VGG-16 is more reasonable than LeNet-5 in the multi-classification problems. Therefore, VGG-16 has a good classification detection ability. Applying MF-BBO to VGG-16 can improve the OA of the model. (iii) The two algorithms optimize the hyperparameters of the first convolutional layer and the first pooling layer in ResNet-18, which improved the feature extraction ability of the model on the input images. The other network layers of the model are residual structures, and it is relatively deeper than each layer of the other models. Therefore, the optimization effect of BBO and MF-BBO on the VGG-16 is better than that on ResNet-18.

4.3. Comparison to state-of-the-art approaches

In this section, we select two kinds of state-of-the-art methods to compare our methods. Table 12 shows the OA of all methods. Here, the OA of ResNet-18 [35] is 90.81 ± 0.72%. The OA of MFBCNNC-III (Ours) is 92.27 ± 0.49%. The OA of VGG-16 [36] is 88.06 ± 1.69%. The OA of MFBCNNC-II (Ours) is 94.37 ± 0.61%.

Fig. 18 is the OA graph. Here, the orange dotted line represents the OA horizontal line between ResNet-18 and the MFBCNNC-III (Ours); The blue dotted line is the OA horizontal line of VGG-16 and the MFBCNNC-II (Ours). It can be seen that the OA of the two models optimized by MF-BBO increases to different degrees. The optimization effect of VGG-16 model is more obvious.

Table 12
Comparison with state-of-the-art methods.

Approach	OA
ResNet-18	90.81 ± 0.72
MFBCNNC-III (Ours)	92.27 ± 0.49
VGG-16	88.06 ± 1.69
MFBCNNC-II (Ours)	94.37 ± 0.61

5. Conclusions

Based on BBO, this paper presents two frameworks — BCNN and MFBCNN. And two methods – BCNNC and MFBCNNC – based on the proposed frameworks. The classification accuracy of the chest X-ray images of COVID-19, Normal and Pneumonia in the model is improved by optimizing the size of convolution kernels and the stride size of convolution kernels, the size of pooling kernels, and the stride size of pooling kernels in the convolutional neural networks (LeNet-5, VGG-16, and ResNet-18). Among MFBCNNC, the overall accuracy of LeNet-5 is improved by 2.87%, and that of VGG-16 by 6.31%. The overall accuracy of ResNet-18 is improved by 1.46%. Using MF-BBO to optimize the hyperparameters of the convolutional neural networks can improve the overall accuracy of the model. It should be noted that, according to the different categories of input images, the values of the optimized hyperparameters of the same convolutional neural network are also different. This proves that the MFBCNN has a strong generalization ability.

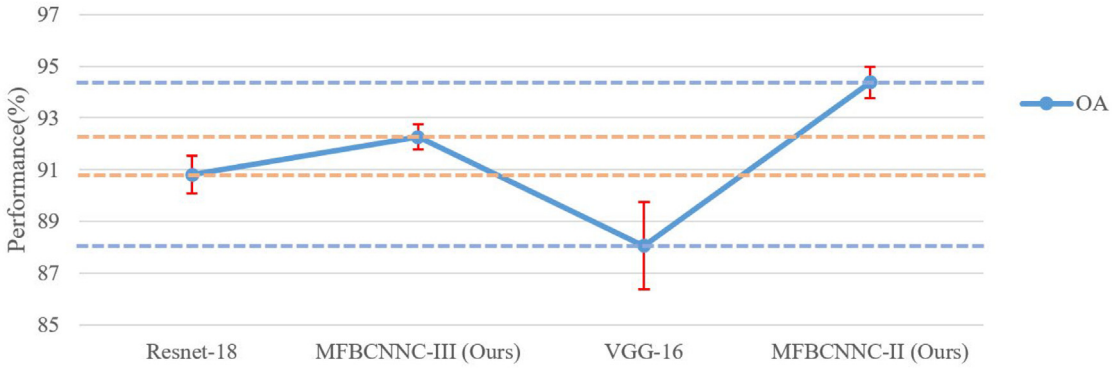


Fig. 18. The OA error graph of four methods.

Table 13
Variable definition table.

Variable name	Variable meaning
A	The matrix.
$a^{(l)}$	The output value of layer l.
$a^{(l-1)}$	The output value of layer $l-1$.
$b^{(l)}$	the bias of layer l.
C	Number of habitats.
c	Index of habitats.
D	Index of cross validation dataset.
E_{max}	Maximum rate of emigration.
F	Migration momentum factor.
f_l	The activation function of layer l.
G	Number of folds for cross validation.
g	Index of fold used as test set.
H	Index of the species number.
H_{max}	Maximum number of species.
h	The upsampling function of convolutional neural network.
I_{max}	Maximum rate of immigration.
i	The row i of the matrix.
j	The column j of the matrix.
K	Total number of runs (each run carries out a G-fold cross validation).
k	Run index (each run carries out a G-fold cross validation).
L_C	The padding size of convolution operation.
L_P	The padding size of pooling operation.
M	Rate of mutation.
M_{max}	Maximum rate of mutation.
N_C	The stride size of convolution kernel.
N_P	The stride size of pooling kernel.
$n^{(l)}$	The number of neurons in the layer l.
O_C	The output of convolution operation.
O_P	The output of pooling operation.
P_H	Probability of habitat has the number of H species.
P_{max}	Maximum probability of species.
$\dot{P}_H$	The probability of species after the derivative.
$Q_{n^{(l)}}$	The identity matrix of the l-layer neurons.
R	Rate of immigration made by random function.
S	Rate of emigration made by random function.
SIV_{in}	The SIV of immigration.
SIV_{out}	The SIV of emigration.
$\ddot{SIV}_{out}$	The SIV of emigration calculated by migration momentum factor.
T	Rate of mutation made by random function.
t	A time in the BBO process.
Δt	A very short time difference.
U	Index of ordinal numbers.
u	Value of HSI.
V_C	The size of convolution kernel.
V_P	The size of pooling kernel.
$W^{(l)}$	the output weight of layer l.
X_C	The input of convolution operation.
X_P	The input of pooling operation.
x	The input of the neural network.
y	The true label of neural network input.
$\hat{y}$	The prediction label of neural network.
$z^{(l)}$	The output of layer l without activation function.
$\mathcal{L}$	Loss function of feedforward neural network.
δ	The confusion matrix.
$\omega^{(l)}$	The partial derivative of the loss function in the l layer.

Table 14
Abbreviation table.

Abbreviation	Full definition
BBO	Biogeography-Based Optimization
BCNN	Biogeography Convolutional Neural Network
BCNNC	Biogeography Convolutional Neural Network for COVID-19
BP	Back Propagation
COV	The chest X-ray image of COVID-19
CNN	Convolutional Neural Networks
CT	Computed Tomography
FN	False Negative
FP	False Positive
HSI	Habitat Suitability Index
MF-BBO	Momentum Factor Biogeography-Based Optimization
MFBCNN	Momentum Factor Biogeography Convolutional Neural Network
MFBCNNC	Momentum Factor Biogeography Convolutional Neural Network for COVID-19
Nor	The chest X-ray image of Normal
OA	The overall accuracy
SIV	Suitability Index Variable
TN	True Negative
TP	True Positive
Pne	The chest X-ray image of Pneumonia

In future research, we will continue to focus on the hyperparameters value optimization of the deep learning models. To expand the application of deep learning technology in the field of medical imaging, our main research directions are as follows. (i) Attempts to optimize more hyperparameters of the deep learning models, and (ii) try to use more optimization algorithms. (iii) Try to improve the convergence ability and the optimization effect of the optimization algorithms.

CRediT authorship contribution statement

Junding Sun: Software, Validation, Investigation, Resources, Supervision, Project administration, Funding acquisition. **Xiang Li:** Conceptualization, Methodology, Software, Data curation, Writing – original draft. **Chaosheng Tang:** Software, Formal analysis, Investigation, Resources, Visualization, Supervision. **Shui-Hua Wang:** Methodology, Validation, Formal analysis, Resources, Writing – review & editing, Funding acquisition. **Yu-Dong Zhang:** Conceptualization, Methodology, Formal analysis, Investigation, Resources, Writing – original draft, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

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

Acknowledgments

Royal Society International Exchanges Cost Share Award, UK (RP202G0230), Medical Research Council Confidence in Concept Award, UK (MC_PC_17171), Hope Foundation for Cancer Research, UK (RM60G0680), British Heart Foundation Accelerator Award, UK (AA/18/3/34220), Sino-UK Industrial Fund, UK (RP202G0289), Global Challenges Research Fund (GCRF), UK (P202PF11), Fundamental Research Funds for the Central Universities (CDLS-2020-03), Key Science and Technology Program of Henan Province, China (212102310084), Provincial Key Laboratory for Computer Information Processing Technology, Soochow University (KJS2048).

Appendix

See Tables 13 and 14.

References

[1] F. Shi, et al., Review of Artificial Intelligence Techniques in Imaging Data Acquisition, Segmentation and Diagnosis for COVID-19. IEEE Reviews in Biomedical Engineering. PP(99): 1-1.
[2] Coronavirus disease (COVID-19) Weekly Epidemiological Update and Weekly Operational Update. Available from: https://www.who.int/emergencies/diseases/novel-coronavirus-2019/situation-reports.
[3] J. Viera-Artiles, et al., 3D-printable headlight face shield adapter, personal protective equipment in the COVID-19 era., Am. J. of Otolaryngol. 41 (5) (2020) 2: Article ID. 102576.
[4] K. Pang, et al., Systematic application of COVID-19 nucleic acid tests in general surgery departments in China: An update of current status with nationwide survey data, Int. J. Surg. 82 (2020) 100–102.
[5] T. Prazuck, et al., *Evaluation of performance of two SARS-CoV-2 rapid igm-igg combined antibody tests on capillary whole blood samples from the fingerti*, Plos One 15 (9) (2020) 11, Article ID. e0237694.
[6] R. Jain, et al., Deep learning based detection and analysis of COVID-19 on chest X-ray images, Appl. Intell. (2020) 11, http://dx.doi.org/10.1007/s10489-020-01902-1.
[7] M. Abdel-Basset, et al., Hsma_Woa: A hybrid novel slime mould algorithm with whale optimization algorithm for tackling the image segmentation problem of chest X-ray images, Appl. Soft Comput. 95 (2020) 19: Article ID. 106642.
[8] A. Narin, et al., Automatic detection of coronavirus disease (COVID-19) using X-ray images and deep convolutional neural networks, 2020, arXiv e-prints: arXiv:2003.10849.
[9] S. Jin, et al., Ai-assisted CT imaging analysis for COVID-19 screening: Building and deploying a medical AI system in four weeks, 2020.
[10] S. Zhang, et al., An investigation of CNN models for differentiating malignant from benign lesions using small pathologically proven datasets, Comput Med Imaging Graph 77 (2019) 101645.
[11] S. Wang, et al., A deep learning algorithm using CT images to screen for corona virus disease (COVID-19), 2020, medRxiv, arxiv:2020.02.14.20023028.
[12] Y. Song, et al., Deep learning enables accurate diagnosis of novel coronavirus (COVID-19) with CT images, 2020, medRxiv, arxiv:2020.02.23.20026930.
[13] B. Ghoshal, et al., Estimating uncertainty and interpretability in deep learning for coronavirus (COVID-19) detection, 2020.
[14] C. Zheng, et al., Deep learning-based detection for COVID-19 from chest CT using weak label, 2020.
[15] A. Cuesta-Infante, et al., Pedestrian detection with lenet-like convolutional networks, Neural Comput. Appl. 32 (17) (2020) 13175–13181.
[16] M. Rezaee, et al., Using a VGG-16 network for individual tree species detection with an object-based approach, in: 2018 10th Iapr Workshop on Pattern Recognition in Remote Sensing, IEEE, Beijing, Peoples R China, 2018.
[17] T. Kaur, et al., Automated brain image classification based on VGG-16 and transfer learning, in: International Conference on Information Technology (ICIT), India, IEEE, Bhubaneswar, India, 2019, pp. 94–98.
[18] K. He, et al., Deep Residual Learning for Image Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp: 770-778.

[19] B. Alotaibi, et al., A hybrid deep ResNet and inception model for hyperspectral image classification, Pfg-J. Photogrammetr. Remote Sensing Geoinfo. Sci. 14 (2020) http://dx.doi.org/10.1007/s41064-020-00124-x.
[20] A. Abuelrub, et al., Hybrid energy system design using greedy particle swarm and biogeography-based optimisation, IET Renew. Power Gener. 14 (10) (2020) 1657–1667.
[21] I. Mariyappan, et al., An Efficient Implementation of Divergence State Estimation with Biogeography-Based Optimization (DSEBBO) Framework in FPGA-Based Multiprocessor System. Arabian Journal for Science and Engineering, 12, doi: http://dx.doi.org/10.1007/s13369-020-04634-z.
[22] M.M. Kumar, et al., A computational algorithm based on biogeography-based optimization method for computing power system security constrains with multi FACTS devices, Comput. Intell. 19 (2020) http://dx.doi.org/10.1111/coin.12282.
[23] Gupta, et al., Biogeography-based meta-heuristic optimization for resource allocation in cloud for E-health services, J. Intell. Fuzzy Systems 38 (5) (2020) 5987–5997.
[24] C. Xia, et al., Delineating early warning zones in rapidly growing metropolitan areas by integrating a multiscale urban growth model with biogeography-based optimization, Land Use Policy 90 (16: Article ID. 104332) (2020).
[25] M.R. Shirani, et al., *BMDA: applying biogeography-based optimization algorithm and mexican hat wavelet to improve dragonfly algorithm*, Soft Comput. 24 (21) (2020) 15979–16004.
[26] E.G. Zahran, et al., *A self learned invasive weed-mixed biogeography based optimization algorithm for RFID network planning*, Wirel. Netw. 26 (6) (2020) 4109–4127.
[27] A. Reihanian, et al., Nbbo: A new variant of biogeography-based optimization with a novel framework and a two-phase migration operator, Inf. Sci. 504 (2019) 178–201.
[28] A. Salehi, et al., *KATZ Centrality with biogeography-based optimization for influence maximization problem*, J. Comb. Optim. 40 (1) (2020) 205–226.
[29] I. Marouani, et al., An improved biogeography-based optimization for economic/environmental dispatch, Iioab J. 10 (5) (2019) 24–33.
[30] J.H. Xiao, et al., Game theory-based multi-task scheduling in cloud manufacturing using an extended biogeography-based optimization algorithm, Concurr. Eng., Res. Appl. 27 (4) (2019) 314–330.
[31] M.K. Kim, et al., Impact of correlation of plug load data, occupancy rates and local weather conditions on electricity consumption in a building using four back-propagation neural network models, Sustainable Cities Soc. 62 (10: Article ID. 102321) (2020).
[32] G. Altamirano-Guerrero, et al., Intelligent design in continuous galvanizing process for advanced ultra-high-strength dual-phase steels using back-propagation artificial neural networks and MOAMP-squirrels search algorithm, Int. J. Adv. Manuf. Technol. 110 (9–10) (2020) 2619–2630.
[33] X. Zhao, et al., A novel biogeography-based optimization algorithm with momentum migration and taxonomic mutation, Adv. Swarm Intell. 12145 (2020) 83–93.
[34] F. Verneau, et al., Cross-validation of the entomophagy attitude questionnaire (EAQ): A study in China on eaters and non-eaters, Food Qual. Pref. 87 (2021) 7: Article ID. 104029.
[35] T. Rahman, et al., Transfer learning with deep convolutional neural network (CNN) for pneumonia detection using chest X-ray, Appl. Sci. 10 (2020) 3233.
[36] U. Shah, et al., An efficient method to predict pneumonia from chest X-rays using deep learning approach, 2020.