Computers in Biology and Medicine 139 (2021) 104994

Contents lists available at ScienceDirect

Computers in Biology and Medicine

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

A novel deep neuroevolution-based image classification method to diagnose coronavirus disease (COVID-19)

Sajad Ahmadian [a], Seyed Mohammad Jafar Jalali [b,*], Syed Mohammed Shamsul Islam [c], Abbas Khosravi [b], Ebrahim Fazli [d], Saeid Nahavandi [b]

[a] *Faculty of Information Technology, Kermanshah University of Technology, Kermanshah, Iran*
[b] *Institute for Intelligent Systems Research and Innovation, (IISRI), Deakin University, Geelong, Australia*
[c] *School of Science, Edith Cowan University, Joondalup, WA, Australia*
[d] *Department of Computer, Zanjan Branch, Islamic Azad University, Zanjan, Iran*

ARTICLE INFO

Keywords:
COVID-19 diagnosis
Evolutionary computation
Improved salp swarm algorithm
Convolutional neural network

ABSTRACT

COVID-19 has had a detrimental impact on normal activities, public safety, and the global financial system. To identify the presence of this disease within communities and to commence the management of infected patients early, positive cases should be diagnosed as quickly as possible. New results from X-ray imaging indicate that images provide key information about COVID-19. Advanced deep-learning (DL) models can be applied to X-ray radiological images to accurately diagnose this disease and to mitigate the effects of a shortage of skilled medical personnel in rural areas. However, the performance of DL models strongly depends on the methodology used to design their architectures. Therefore, deep neuroevolution (DNE) techniques are introduced to automatically design DL architectures accurately. In this paper, a new paradigm is proposed for the automated diagnosis of COVID-19 from chest X-ray images using a novel two-stage improved DNE Algorithm. The proposed DNE framework is evaluated on a real-world dataset and the results demonstrate that it provides the highest classification performance in terms of different evaluation metrics.

1. Introduction

COVID-19 was reported for the first time in the Chinese city of Wuhan in December 2019 [1,2]. COVID-19's natural composition is a positive-oriented, one-stream RNA shape, and because of its mutating characteristic, it is difficult to eradicate [3]. Globally, healthcare providers are conducting extensive work to create an appropriate cure for this epidemic. COVID-19 has caused thousands of mortalities in countries around the globe [4]. It is marked by a sore throat, headaches, nausea, throat irritation and coughing, and can cause the mortality of individuals with weakened immunologic systems [5,6]. COVID-19 typically spreads through physical interaction from human to human. Healthy individuals can become infected from people carrying COVID-19 through breathing communication, touch contact, or mucosal touch [7].

In recent years, medical research has widely benefited from artificial intelligence (AI) methodologies [8–14]. In various applications, such as medical informatics [15,16,16], energy forecasting [17–25], uncertainty quantification [26–28], probabilistic forecasting [29–31], and recommendation models [32–45], AI has been employed with powerful algorithms. Deep-learning approaches have been highlighted to be used with the diagnosis of COVID-19 in chest X-rays after appearing the epidemic [46–50]. Several scientific studies have been conducted to analyze chest X-rays taken from COVID-19 patients. In a previous study [51], three separate deep-learning models (a perfectly tuned model, a model without fine tuning, and a model trained from scratch) were deployed to categorize COVID-19 X-ray images. The authors sorted the dataset into many categories such as age and sex using the ResNet model. They also used the MLP neural network as a classification method. Yadav et al. [52] developed an Algorithm for classifying X-ray images using support vector machine (SVM) as a classification tool and VGG-16 as a deep-learning network. In their study, the dataset is divided into three classes: regular, bacterial, and viral COVID-19. They used an image enhancement method for each image in the dataset to increase the contrast and brightness. The best result obtained using their proposed classification algorithm was 96.6%. Sarker et al. [53] used DenseNet for COVID-19 classification. They used transfer-learning approaches to train the deep-learning system to alleviate the problems affected by the

[*] Corresponding author.
E-mail address: mohammadjj.it@gmail.com (S.M.J. Jalali).

https://doi.org/10.1016/j.compbiomed.2021.104994
Received 12 June 2021; Received in revised form 24 October 2021; Accepted 25 October 2021
Available online 1 November 2021

gradient descent. They developed a web interface that takes X-ray images and creates a bounding box around the affected areas.

A novel technique based on deep-learning was developed by Shan et al. [54] to automatically classify the infected lung regions. Their technique was assessed on a dataset containing images from 300 people infected with COVID-19. For the binary classification of COVID-19, Elasnaoui and Chawki [55] used convolutional neural network (CNN) models. They used pre-trained models such as Inception ResNet V2, VGG16, VGG19, Inception V3, Xception, MobileNet V2 and ResNet50. Li et al. [56] have developed a neural network for the detection of COVID-19 infection (COVNet) in order to extract valuable features from chest-CT images. Over 4357 chest-CT images of 3322 patients were used to train the COVNet. The final achieved accuracy of COVNet was 95%. In Ref. [57], Okeke Stephen et al. proposed a CNN model, which was developed by extracting characteristics from chest X-ray images to evaluate the presence of COVID-19. In Ref. [58], Hu et al. presented a weakly-supervised CNN framework and obtained an accuracy and the AUC of 96.2% and 97% respectively. Abbas et al. [59] employed a CNN application named Decompose, Transfer, and Compose (DeTraC) to classify COVID-19 through chest X-ray images. They achieved 95.12% accuracy, 97.91% sensitivity, and 91.87% specificity. Narin et al. [60] used two deeper frameworks for classifying COVID-19 and normal cases based on a publicly available dataset. In a hybrid model named ADOPT [61], a spotted hyena optimizer, a decision tree model, and deep CNN models are used for identifying COVID-19 pneumonia from X-ray images. In comparison with eleven different CNN algorithms, ADOPT demonstrated better outcomes in relation to different evaluation metrics such as accuracy, recall, precision, F1-score, and specificity. In another study conducted by Dhiman et al. [62], a model named DON was proposed using emperor penguin optimizer for optimization of the CNN parameters to detect infected patients using X-ray COVID-19 images efficiently. Extensive evaluation findings demonstrate that DON outperforms competitor approaches.

By reviewing the literature, it can be inferred that most of the research carried out by deep-learning models uses pre-trained deep-learning architectures to diagnose COVID-19 in X-ray radiological images. The main problem of transfer-learning architectures is the rise of the overfitting issue when a small number of labeled training data are available, and the pre-trained deep learning models (such as MobileNet and ResNet) have millions of parameters. Other research works use network architectures that are obtained via the trial-and-error process in deep-network hyperparameters, but this is a very time-consuming procedure. It is worth noting that the efficiency of deep neural networks is significantly dependent on the values of their hyperparameters. Deep neuroevolution (DNE) is known as a very practical and useful method in designing network architecture using evolutionary algorithms that both optimally select the best hyperparameter values and minimize the problem of network overfitting.

In this paper, we propose a novel evolutionary Algorithm named the boosted salp swarm algorithm (BSSA) that enhances the original version of SSA through incorporating two strong optimization operators: opposition-based learning (OBL) and chaotic maps. It should be noted that the efficiency of evolutionary algorithms is highly dependent on performing the exploration and exploitation phases in an effective way. Therefore, in this paper, we aim to propose powerful mechanisms to improve the efficiency of the exploration and exploitation phases. By applying the two optimization operators, both the diversity of the search process and the exploration and exploitation of the SSA method are enhanced. Moreover, we apply the proposed BSSA algorithm to optimize the key hyperparameters of CNN models. This leads to an improvement in the efficiency of CNN models in classification problems. In other words, in contrast to previous deep neural-network-based COVID-19 diagnosis models, the proposed method automatically obtains the optimal values of CNN's hyperparameters through the optimization process of the proposed BSSA algorithm. The optimized CNN model is then used to diagnose COVID-19 cases based on X-ray images with a high accuracy. To improve the effectiveness of the CNN model in classifying the input images accurately, the support vector machine approach is applied to the outputs of the CNN model. The experimental results demonstrate that the proposed method can significantly outperform other classification models in regard to COVID-19 diagnosis. The main contributions of this work are:

- A DNE framework is proposed based on a deep CNN model to classify the patients as COVID-19 infected or normal (i.e. non-COVID-19) by investigating their chest X-ray images.
- A novel evolutionary Algorithm is proposed by improving the original version of the salp swarm algorithm through incorporating two effective optimization operators including opposition-based learning and chaotic maps to make an acceptable balance between the exploration and exploitation phases in the search space. This leads to a reduction in the probability of falling into local optima solutions.
- The proposed evolutionary Algorithm is used to obtain the optimal values of the hyperparameters of the CNN model, leading to an improvement in the performance of the proposed image classification method.
- The support vector machine approach is used in the proposed image classification method to generate more accurate results by applying this approach to the outputs obtained by the CNN model.
- Comprehensive experiments are carried out to examine the feasibility of applying the DNE Algorithm developed in this research to COVID-19 chest X-ray images. Since the proposed algorithm performs with a high accuracy, it can help physicians to diagnose COVID-19.

The rest of this paper is organized as follows: Section 2 describes the proposed DNE framework for X-ray images. The experimental procedures are presented in Section 3. The experimental results are given and discussed in Section 4. Finally, concluding remarks are set out in Section 5.

2. Methodology

In this section, a novel image classification method is proposed based on deep convolutional neural networks in order to diagnose COVID-19. The aim of the proposed method is to apply deep CNN to classify chest X-ray images in order to distinguish between COVID-19 and non-COVID-19 cases. To this end, an efficient evolutionary Algorithm called the boosted salp swarm algorithm (BSSA) is proposed to obtain the optimal values of the hyperparameters of the deep CNN model. The main advantage of the proposed evolutionary algorithm is to automatically configure the hyperparameters of the deep CNN model. Moreover, the support vector machine approach is applied to the outputs obtained by the CNN model to generate more accurate results leading to an improvement in the performance of the proposed image classification model. In the subsections below, an overview of the CNN, the BSSA, and the support vector machine is presented. The proposed model is then discussed in detail.

2.1. An overview of CNN

In contrast to task-specific techniques, deep learning belongs to a part of the wider range of machine-learning algorithms associated with data interpretation. The learning procedure can be supervised, semi-supervised, or unsupervised. The convolutional neural network (CNN) is among the most significant architectural modeling techniques for deep learning [63].

The CNN is a multilayered network with the convolutional and pooling layers as the core network architecture [64]. In order to achieve a variety of feature maps, the convolutional layer uses several convolution kernels for function extraction on the input layer, while the pooling layer can primarily reduce the feature map dimension via a pooling mechanism, prevent overfitting, and minimize the number of parameters as much as needed to maintain reasonable accuracy [65,66].

In CNNs, input data are considered in the input layer. There are a variety of two-dimensional (2D) filters with weighted parameters in the convolution layer [67]. These filters are connected to input layer and generate an output in the form of feature maps. All of the filters have the same weighted parameters. Assume the input of the convolutional layer is determined by X with $R^{A\times B}$ dimension where the size of the input data is denoted by A and B. The output can be given as follows for the convolutional layer:

$$C_n = f\left(\sum\nolimits_{c=1}^{N} X_c^{l-1} * W_m^l + B_m^l\right), \tag{1}$$

where C_n is the nth convolutional layer output, n indicates the output number which equals to the number of the filters, the channel number is represented by N and the convolutional operator is denoted by $*$. X_c is the cth channel of input data for the previous layer ($l-1$), W_m^l represents the mth filter weight for the current layer (l), the filter's width and height are represented by W and H respectively, B_m^l indicates the mth bias, and f is one of the commonly known activation functions including Rectified Linear Unit (ReLU), sigmoid function, and hyperbolic tangent (tanh). ReLU function is the widely used activation function and delineated as follows:

$$f(x) = \max(0, x) \tag{2}$$

The pooling procedure helps to minimize the computational complexity of the feature maps following the last convolutional layer, avoid overfitting appropriately, and decrease training parameters, ensuring that the model's training time is decreased. The generally known and utilized pooling method is max-pooling, responsible for traversing feature maps with a flexible step size and to replace the traversed regions by the optimum values. Finally, new feature maps can be extracted after the traversing procedure. This is how CNNs generally work.

2.2. Support vector machine

A support vector machine (SVM) is a powerful linear supervised learning Algorithm used for solving linear and also non-linear regression and classification problems. This technique uses a hyper-plane line to distinguish features in the data classes throughout the classification phase. Put simply, this technique utilizes data as input and produces an output with a line separating the available classes in the dataset. The desirable line implies a generalized machine line for all classes to be the most common separator.

Softmax is a mathematical operation that transforms a vector of integers into a vector of probabilities, with the probability within each value related to the vector's proportional scaling. The Softmax function has been commonly used as an activation function in a neural network model. The network is set up to output values, one for each class in the classification task, and the Softmax function is employed to normalize the outputs, transforming weighted summation values towards probabilities that add up to one. Each integer in the Softmax function's output is regarded as the probability of belonging to each class. We substitute the Softmax classifier (the top layer of the CNN model) with the SVM linear dropout classifier. The motivation behind this is that the utilization of the dropout SVM can provide a better performance than Softmax for the deep CNN, and also reduce the overfitting problem [68,69]. Another advantage of SVM compared to Softmax is that SVM aims to find the maximum margin between data points of different classes. This helps to improve the accuracy of the classification model. The mathematical operation functions of the SVM Algorithm are provided in Eqs. (3)–(5). In these equations, x and y describe the coordinate points of hyper-plane features, M denotes the margin width parameter, and parameter b represents the values for bias.

$$u = \overline{M}.\overline{x} - b \tag{3}$$

$$\frac{1}{2} = \|\overline{M}\|^2 \tag{4}$$

$$y_i(\overline{M}.\overline{x} - b) \geq 1, \forall i \tag{5}$$

2.3. Improved evolutionary algorithm: boosted salp swarm algorithm (BSSA)

The inspiration for the salp swarm Algorithm (SSA) is based on the foraging and navigation behavior of ocean organisms called salps. This evolutionary swarm intelligence algorithm was introduced by Mirjalili et al. [70] in 2017. Salps commonly float together in the form of a salp chain while foraging and navigating in seas and oceans. Similar to other swarm intelligence models, SSA is a population-based method commencing with the random initialization of a predefined number of search agents. In the swarm of salps, there are two components: a leader, and followers. The following is the mathematical model representing the salp chain. The position of the leader is updated using Eq. (6).

$$x_j^1 = \begin{cases} F_j + c_1\left(\left(ub_j - lb_j\right) c_2 + lb_j\right) & c_3 \geq 0.5 \\ F_j - c_1\left(\left(ub_j - lb_j\right) c_2 + lb_j\right) & c_3 < 0.5 \end{cases}, \tag{6}$$

where F_j is the position of the source food, and ub_j and lb_j represent the upper and lower bound of the jth dimension, respectively. Three numbers, c_1, c_2 and c_3 are computed randomly into the [0, 1] interval. c_1 is an important vector which is slowly reduced and measured over the iteration periods, where L and l are the maximum number of iterations and the current iteration, respectively. Therefore, the value of parameter c_1 depends on the values of parameters l and L. The value of parameter l changes from 1 to L with step 1. Accordingly, the value of parameter c_1 will change based on the following equation:

$$c_1 = 2e^{-\left(\frac{4l}{L}\right)^2} \tag{7}$$

The position of the follower salps is updated using Eq. (8):

$$x_j^i = \frac{1}{2}\left(x_j^i + x_j^{i-1}\right), \tag{8}$$

where x_j^i is the ith follower's position on the jth dimension and $i \geq 2$.

The SSA has demonstrated its strong capabilities in solving various optimization problems [71,71–73]. However, we intend to further improve the performance of this evolutionary technique in order to achieve the best possible solutions. To do this, we apply two novel modifications named opposition-based learning (OBL) and three chaotic maps to the SSA, which leads to the higher exploration and exploitation of the SSA as well as higher convergence speed. In the following, we discuss the details of these two modifications.

- *Opposition-based learning (OBL):*

OBL is an optimization strategy used to boost the diversity of the initialized population solutions. The search procedure in SSA starts with the initial solutions using a set of randomly generated numbers. This leads to a slow convergence rate [73]. Thus, OBL is introduced to avoid such issues.

Assume $M = \{x_1, x_2, x_3, \ldots, x_J\}$, which $x_i \in [a_i, b_i]$ is considered as the ith dimension of a search agent. Let a and b represent the bounds of variables and $i = 1, 2, \ldots, J$. The opposite position of each salp (x_i) is defined by $x_i^{opposite}$ as follows:

$$x_i^{opposite} = a_i + b_i - x_i \tag{9}$$

- *Chaotic maps:*

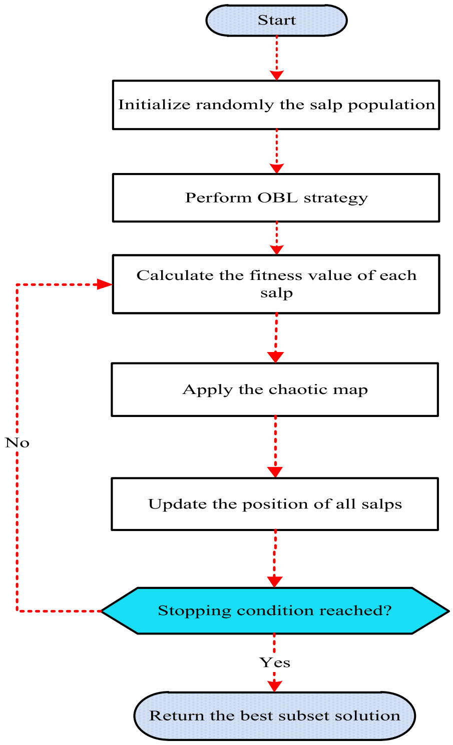

Start
Initialize randomly the salp population
Perform OBL strategy
Calculate the fitness value of each salp
Apply the chaotic map
Update the position of all salps
Stopping condition reached?
No
Yes
Return the best subset solution

Fig. 1. The flowchart of the proposed BSSA Algorithm.

As Eq. (6) shows, the key parameter c_2 controls the update of the positions of the salps. Chaotic maps are among the powerful evolutionary operators that are used to enhance the tuning of randomness parameters in metaheuristic algorithms. They have a similar functionality to randomness, with better dynamic capabilities. Among different chaotic maps, the piecewise chaotic map as represented in Eq. (11) shows its effectiveness in guaranteeing the diversity of the generated solutions [74,75]. Thus, we use a piecewise map to tune the c_2 parameter as shown in Eq. (12), where the value of b^t denotes the obtained value from the chaotic map at tth iteration.

$$c_2 = b \tag{10}$$

$$b = \begin{cases} \frac{b}{l}, & 0 \leq b < l \\ \frac{b - l}{0.5 - l}, & l \leq b < 0.5 \\ \frac{1 - l - b}{0.5 - l}, & 0.5 \leq b < 1 - l \\ \frac{1 - b}{l}, & 1 - l \leq b < 1 \end{cases} \tag{11}$$

$$y_i^t = \begin{cases} F_i + c_1((ub_i - lb_i)b^t + lb_i), & c_3 \geq 0 \\ F_i - c_1((ub_i - lb_i)b^t + lb_i), & c_3 < 0 \end{cases} \tag{12}$$

where, $l = 0.4$.

The whole procedure of our novel boosted SSA (BSSA) is illustrated in Fig. 1.

2.4. Proposed image classification approach

In this section, we present our proposed hybrid deep neuroevolution method, named boosted SSA-CNN (BSSA-CNN). This method is used to optimize the CNN hyperparameters that helps boosting the accuracy of COVID-19 diagnosis. It is worth noting that in the proposed method, the weights of the fully connected layers in the CNN model are updated and fine-tuned using the gradient descent algorithms.The proposed BSSA Algorithm is applied in order to obtain optimal values for the CNN's hyperparameters, and as a consequence, the optimal weights for the CNN's connections are obtained through the gradient descent algorithms. In any evolutionary optimization problem, two important issues - the representation of solutions and the calculation of fitness function - should be taken into consideration. In the proposed method, we address nine important hyperparameters: kernel size, number of filters, number of epochs, batch size, max-pooling size, dropout rate, learning rate, momentum rate, and number of convolutional layers to be optimized by the BSSA model. Thus, in BSSA, all solutions can be described as a vector with nine dimensions, each corresponding to one of the nine CNN hyperparameters. Dropout rate, learning rate, and momentum rate are the hyperparameters with continuous values, and their optimal values can be obtained by BSSA directly. Kernel size, number of filters, number of epochs, batch size, max-pooling size, and number of convolutional layers are other hyperparameters with discrete values. Since BSSA is continuously investigating the solution space, we need to transform the obtained optimum values for such hyperparameters into their respective discrete values. An efficient model is used to enable each real value to be transformed into an integer value. To this end, the continuous value of each hyperparameter is transferred to $D = [K_1, K_2, \ldots, K_n]$ as a discrete search space. The following equations are used to formulate the discretization model:

$$\alpha = 1 + n \times R \tag{13}$$

$$\beta = min(\lfloor \alpha \rfloor, n) \tag{14}$$

where R is a real value in the range of [0, 1] to explore in the continuous search space, α is used as a mapping from R to $[1, n + 1]$, and β is used as a mapping from α to $[1, 2, 3, \ldots, n]$. Then, the integer value corresponding to the continuous dimension of each solution can be calculated as follows:

$$X_{ij} = K_\beta \tag{15}$$

The proposed BSSA-CNN Algorithm randomly initializes the population of n solutions by Eq. (6). Each solution is labeled with a nine-dimensional X_{ij}, $i = 1, \ldots, n$ and $j = 1, \ldots, 9$ vector where j is one of the nine CNN hyperparameters for each dimension. New solutions can be obtained by continuously upgrading current solutions with Eq. (8) following the generation of the initial population. We use OBL technique at initialization phase of SSA to improve its population diversity in the search space using Eq. (9). Then, we tune the c_2 key parameter with the piecewise chaotic map strategy to incorporate a balance between the exploration and exploitation phases. The entire process continues to repeat until the end criterion is satisfied and the best achieved solution is perceived as the final outcome. The values generated in this solution can be employed as the optimum CNN hyperparameter values.

We need to determine a fitness function for evaluating the effectiveness of all the obtained solutions. In order to achieve this, the data from the input COVID-19 image dataset are split up into two separate training and test sets. The training set is considered to optimize the hyperparameters of the CNNs using BSSA, whereas the test set is required to examine the effectiveness of the final COVID-19 diagnosis model. It can be noted that the CNN model is configured using the values of hyperparameters achieved from each solution in the BSSA. As such, the fitness function can be perceived for the performance of the configured CNN model for COVID-19 diagnosis. To this end, the images in the training set are used as the inputs of the CNN model in the training phase. The accuracy of the CNN classifier in the classification of input images is used as the fitness value of the solutions in the optimization process. The accuracy metric refers to the number of correctly classified images over the total number of input images. Therefore, the value of the accuracy metric can be calculated as follows:

$$Accuracy = \frac{\#\ \text{number of correctly classified images}}{\#\ \text{total number of input images}} \tag{16}$$

Obviously, a solution with a higher accuracy value has a higher fitness value, and vice versa. Therefore, the aim of the proposed method is to obtain a solution with the highest accuracy value (i.e., the highest fitness value) containing the optimal values of CNN hyperparameters. This leads to obtaining a CNN model with maximum performance in regard to the classification of the images in the test set. After determining the optimal values of the CNN hyperparameters using BSSA, the configured CNN model is used to classify the images in the test set. It should be noted that the SVM model is applied to the outputs obtained by the flattening layer of the CNN model to determine the final outputs of the proposed classification method. Algorithm 1 represents the overall steps of the proposed BSSA-CNN method. In Fig. 2, the overall procedure of COVID-19 diagnosis using the proposed framework is illustrated.

Algorithm: 1. Pseudo-code of the proposed DNE COVID-19 diagnosis model (BSSA-CNN)

```
Algorithm 1 Pseudo-code of the proposed DNE COVID-19 diagnosis model (BSSA-CNN)
Input: $P$ (Salps population size), $u_b$ (Upper bound), $l_b$ (Lower bound) and $Max_{ite}$ (maximum
    number of iterations).
Output: Diagnosed COVID-19 or normal images.
Begin algorithm:
Split images of dataset into training $Tr$ and test $Te$ sets;
Generate initial population of salps $X_i$ ($i$=1,2,..., $P$);
Set $t$=1;
while ($t < Max_{ite}$) do
    Adjust a CNN model depending on the values of the solution hyperparameters $X_i$;
    Calculate the fitness of each $X_i$ using Eq.(16) as the accuracy of CNN algorithm extracted
    from $Tr$;
    Set $Best_S$= the best obtained salp search agent;
    for each salp $X_i$ do
        Update the parameter $c_2$ using Eqs.(10);
        if $X_i \sim$ leader salp then
            Update the leader salp position using Eqs.(12) and (7);
        else if $X_i \sim$ followers then
            Update the leader salp position using Eq.(8);
        end if
        Repair and check the salp swarm if exceeded the search boundaries;
        Perform the OBL strategy to generate the $N_{op}$ opposite solutions using Eq.(8);
        Merge the obtained $N_{op}$ opposite solutions and $P_{op}$ solutions from the sorted swarm to
    construct the population size $P$;
        Update the $Best_S$ ;
    end for
    Set $t$=$t$+1;
end while
Adjust the CNN model based on the hyperparameters obtained by the best salp $Best_S$;
Classify the images in the test set $Te$ using the best obtained CNN model;
Return the classified images as the output;
End algorithm
```

3. Implementation details

3.1. Collected COVID-19 dataset

In order to examine the effectiveness of our proposed deep neuro-evolution method, we use a well-known COVID-19 dataset containing chest X-ray images. This dataset was gathered from a Mendeley

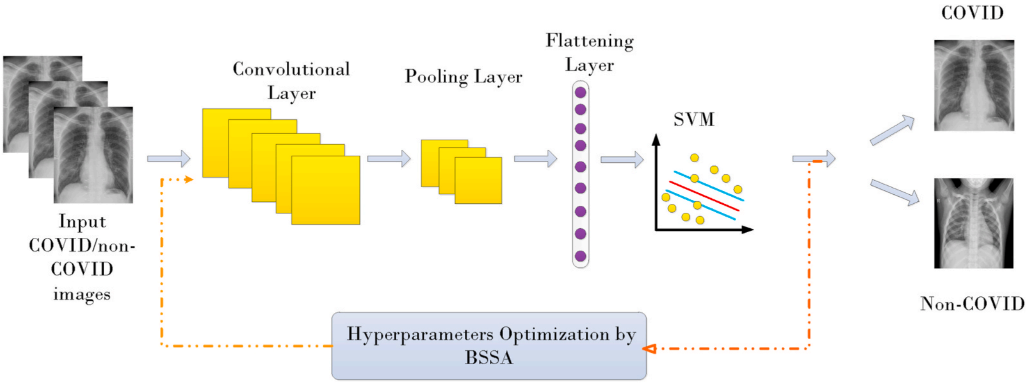


Fig. 2. Overall procedure of COVID-19 diagnosis using the proposed BSSA-CNN model.

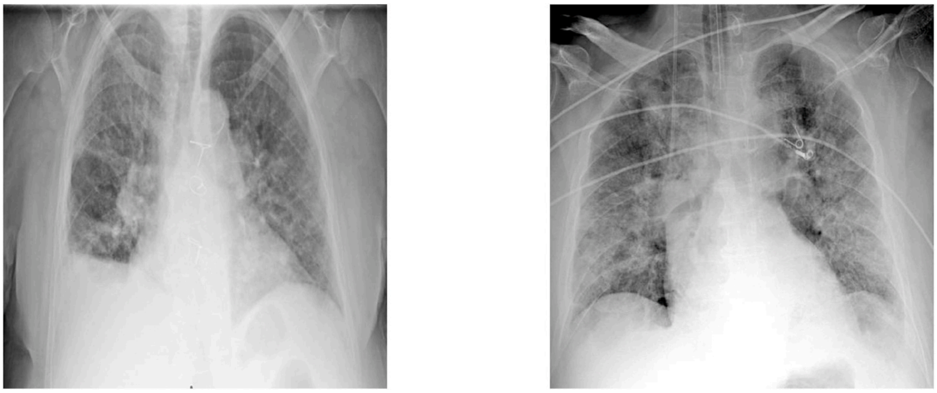


Fig. 3. Two samples of available images in the dataset related to the patients with COVID-19.

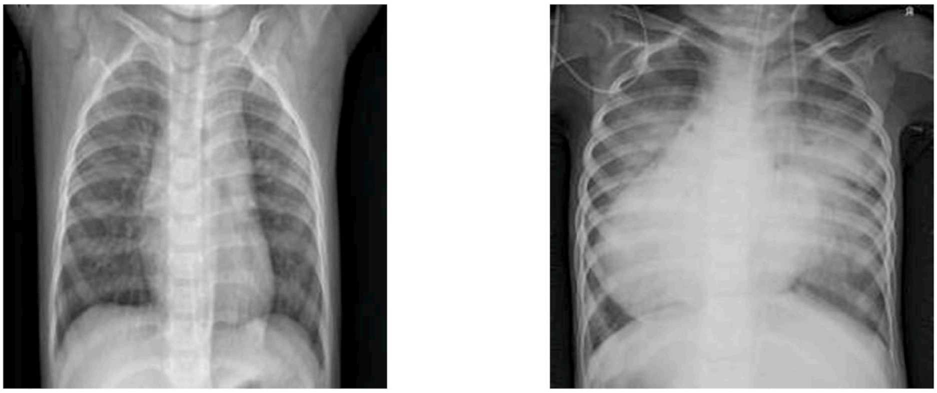


Fig. 4. Two samples of available images in the dataset related to normal persons.

repository containing augmented images for both COVID-19 and non-COVID-19 individuals.[1] This is a balanced dataset with 912 positive (COVID-19) and 912 normal (non-COVID-19) cases. The samples of confirmed and non-confirmed COVID-19 images from the provided dataset can be seen in Figs. 3 and 4. As this dataset contains the X-ray images for two groups of cases (COVID-19 and non-COVID-19), the classification problem related to this dataset is a binary classification.

For the pre-processing stage, we transformed all of the images into the same size 224 * 224 pixels because the data collection was not consistent, and the X-ray images were of varying sizes. This was

Table 1
List of CNN hyperparameter symbols and their values.

Symbol	Value
K_s	[1, 30]
N_f	[1, 500]
N_e	[1, 400]
B_s	[10, 20, …, 200]
MP_s	[1, 20]
D_r	[0.2, 0.25, …, 0.65]
L_r	[0.001, 0.006,.., 0.1]
M_r	[0.05, 0.1,.., 0.95]
N_c	[1, 2, …, 20]

[1] https://data.mendeley.com/datasets/2fxz4px6d8/4.

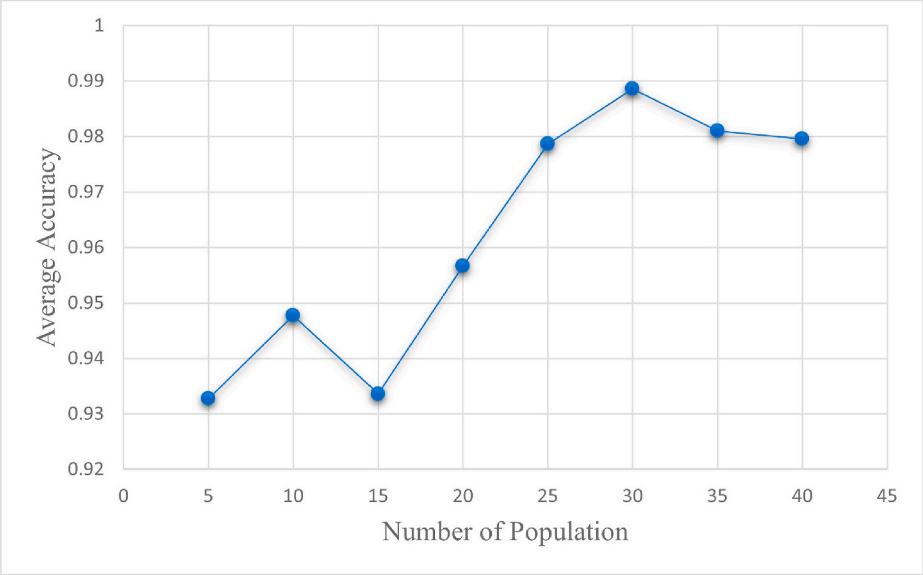


Fig. 5. The influence of different initialized population sizes on the performance of our proposed model.

accomplished using RGB reordering, and the final input images for use in the proposed method are sized 224*224*3. The X-ray images were also rotated horizontally and vertically to considerably improve the dataset's diversity.

3.2. *Implementation and parameter settings*

We evaluate the proposed evolutionary optimization Algorithm for CNN structures using the COVID-19 dataset. To this end, 80% of the dataset is devoted to training and the remaining 20% is used for test samples. The TensorFlow library with Python version 3.6.0 is used to train all deep CNN models. We also use one 1080 Ti GTI 16 GB on a Linux machine with Ubuntu version of 16.04 operating system. The training phase of CNN is conducted with the ReLU as the activation function, Adam as the optimizer and accuracy as the cost function. Training samples were required for every epoch. When the training loss is smaller than the loss of validation, the training procedure is stopped. Otherwise, a maximum number of training iterations is permitted. This procedure is needed to avoid overfitting and to decrease the training time.

In the experiments, to have a fair comparison, the population size of BSSA and other EA benchmarks is set to 30 search agents, and the number of iterations and the number of runs are set to 20 and 10, respectively. It is worth mentioning that the parameters of SVM (e.g., kernel method parameters, the penalty parameter, etc.) in the proposed method are set based on a greedy search process to find their optimal values. We also optimize nine hyperparameters: kernel size (K_s), number of filters (N_f), number of epochs (N_e), batch size (B_s), max-pooling size (MP_s), dropout rate (D_r), learning rate (L_r), momentum rate (M_r), and number of convolutional layers (N_c) by the BSSA Algorithm for evolving the CNN structures. These are tabulated in Table 1. The input parameters of the comparison models are set according to the optimal values reported in their corresponding papers. A trial-and-error approach is used based on a greedy search process to obtain the best values of the parameters of these models which can be fitted with the problem addressed in this paper.

3.3. *Performance evaluation metrics*

We employ five different metrics namely, the accuracy, precision, recall, F-measure, and area under the curve (AUC) to assess the efficiency of the proposed classification model and other compared benchmarks. These metrics are widely used in medical area and are formulated as follows [76,77]:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}} \tag{17}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{18}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{19}$$

$$\text{AUC} = \int_0^1 \frac{\text{TP}}{P}\, \text{d}\, \frac{\text{FP}}{N} = \frac{1}{P.N}\int_0^1 \text{TP d FP} \tag{20}$$

$$\text{F}_1 = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} \tag{21}$$

where TP, FP, TN and FN given in Eqs (17) – (21) refer to True Positive, False Positive, True Negative and False Negative, respectively. Given a test dataset and a deep learning classification Algorithm, TP represents the proportion of positive (i.e., COVID-19) samples that are correctly labeled as COVID-19 by the classification algorithm; FP represents the proportion of negative (i.e., non-COVID-19) samples that are mislabeled as positive; TN is the proportion of negative samples that are correctly labeled as normal and FN is the proportion of positive samples that are mislabeled as negative by the classification algorithm.

4. Experimental results and discussion

In this section, the performance evaluation of the proposed BSSA-CNN method and the other evolutionary-based and state-of-the-art algorithms are discussed.

4.1. *Sensitivity analysis of population size on the performance of the proposed method*

To examine the effect of different population sizes on the overall performance of our proposed Algorithm, we apply the sensitivity analysis procedure on a grid of different population sizes in 10 independent runs. The results are shown in Fig. 5. As can be observed from this figure, an initial population value of 30 results in the best performance with an average accuracy of 0.988539 for our proposed algorithm. To this end,

Table 2
Quantitative performance results of evaluation metrics for evolutionary algorithms over the binary COVID-19 dataset. The best results among different DNEs are shown by bold black fonts.

Metric		GA-CNN	PSO–CNN	DE-CNN	GWO-CNN	MFO-CNN	SSA-CNN	OSSA-CNN	CSSA-CNN	BSSA-Softmax	BSSA-CEL	DON	ADOPT	Proposed
ACC	AVG	0.917442	0.931232	0.942693	0.919771	0.928367	0.936963	0.974212	0.977077	0.975479	0.969248	0.973228	0.978555	**0.988539**
	STD	0.008957	0.008912	0.009358	0.007384	0.006933	0.007019	0.007566	0.007044	0.006446	0.006653	0.007182	0.00692	**0.006189**
	Best	0.937685	0.943488	0.948802	0.932559	0.938876	0.945667	0.978081	0.983667	0.982541	0.977113	0.981822	0.988703	**0.990218**
	Worst	0.894537	0.903663	0.916766	0.895053	0.913443	0.908897	0.965445	0.968003	0.966743	0.961199	0.969822	0.971634	**0.983337**
Precision	AVG	0.948718	0.961538	0.920455	0.884615	0.955414	0.929412	0.981818	0.981928	0.980302	0.970118	0.980213	0.981228	**0.988095**
	STD	0.018232	0.017033	0.017163	0.009408	0.018145	0.029241	0.005666	0.003367	0.003812	0.003135	0.004119	0.003291	**0.002934**
	Best	0.958434	0.966768	0.930221	0.902443	0.961113	0.938986	0.984554	0.983227	0.982105	0.976142	0.982716	0.98312	**0.989778**
	Worst	0.911679	0.923572	0.901088	0.879044	0.913008	0.909077	0.963114	0.968922	0.967688	0.961457	0.970005	0.973649	**0.984971**
Recall	AVG	0.880952	0.892857	0.964286	0.958333	0.892857	0.940476	0.964286	0.970238	0.968704	0.962842	0.963433	0.968099	**0.988095**
	STD	0.028157	0.037647	0.036811	0.017883	0.019884	0.027636	0.016889	0.013776	0.012288	0.011026	0.013433	0.010199	**0.010117**
	Best	0.897411	0.913444	0.974521	0.959222	0.920557	0.959088	0.974407	0.976885	0.975762	0.970231	0.975062	0.971388	**0.991309**
	Worst	0.858999	0.867621	0.943222	0.916588	0.858804	0.912223	0.961282	0.961434	0.96019	0.954067	0.959989	0.965622	**0.985677**
F-measure	AVG	0.913582	0.925926	0.94186	0.919778	0.923077	0.934911	0.972973	0.976048	0.974712	0.968389	0.975451	0.978044	**0.988095**
	STD	0.012807	0.013191	0.011576	0.011867	0.011989	0.009428	0.010339	0.009676	0.008416	0.007154	0.009545	0.016522	**0.006526**
	Best	0.922748	0.934389	0.955619	0.925561	0.925655	0.948888	0.975668	0.980033	0.978891	0.972268	0.980488	0.984731	**0.993044**
	Worst	0.891833	0.883533	0.921319	0.888346	0.883044	0.923346	0.958988	0.965543	0.964342	0.958019	0.967812	0.969596	**0.979663**
AUC	AVG	0.918377	0.929854	0.943469	0.921156	0.927092	0.937089	0.973856	0.976832	0.975346	0.969883	0.975557	0.978522	**0.988523**
	STD	0.008993	0.011899	0.010256	0.009545	0.009943	0.006575	0.009918	0.008774	0.007352	0.006429	0.008151	0.006333	**0.006332**
	Best	0.932845	0.936767	0.963133	0.928461	0.935449	0.954478	0.978888	0.982359	0.981239	0.974616	0.981911	0.979477	**0.989446**
	Worst	0.909657	0.901532	0.908748	0.900355	0.891001	0.920004	0.964544	0.970669	0.969455	0.962693	0.971331	0.970036	**0.981088**

we choose an initial population size of 30 in our experiment.

4.2. *Performance comparison based on different evolutionary algorithms (EAs)*

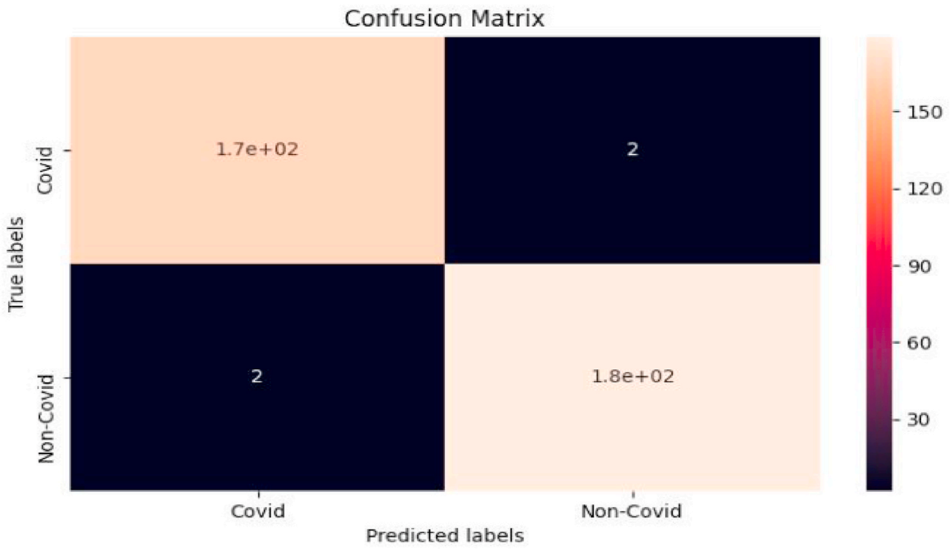
We compare the performance of different EAs in optimizing the hyperparameters of the CNN model. To this end, GA (genetic Algorithm) [78], PSO (particle swarm optimization) [79], DE (differential evolution) [80], GWO (grey wolf optimizer) [81], MFO (moth-flame optimization) [82], SSA (salp swarm algorithm) [83], OSSA (OBL version of SSA), and CSSA (chaotic version of SSA) are considered in comparison with the proposed method. Two powerful evolutionary-based deep--learning models for COVID-19 detection, ADOPT [61], and DON [62], are also utilized as the competitive algorithms to show the strength of our proposed model. It should be noted that in the proposed method, the SVM model is used instead of the Softmax model to obtain better results. Experiments are conducted in order to compare the performance of the SVM model with two other models including Softmax and cross entropy loss. The models related to the Softmax and cross entropy loss are named in the experiments as BSSA-Softmax and BSSA-CEL, respectively. It is worth noting that we use the BSSA algorithm to optimize the hyperparameters of the CNN model in the proposed method. Table 2 reports the results of experiments based on different evaluation metrics: accuracy, precision, recall, F-measure, and AUC. In order to better represent the results, the values of the evaluation metrics are shown up to six digits after the decimal point. This leads to a better comparison of the results of different algorithms. Moreover, in this table, we report the average (AVG), the standard deviation (STD), the best result (Best), and the worst result (Worst) obtained by each algorithm for all evaluation metrics. These results reveal that the proposed method significantly outperforms other methods in terms of all evaluation metrics. Therefore, it can be concluded that the BSSA algorithm has a better search ability in comparison to other EA models. In other words, the BSSA algorithm can efficiently optimize the hyperparameters of the CNN model by exploring the search space to find optimal values. This leads to an improvement in the performance of the proposed classification method in regard to classifying COVID/non-COVID images.

Fig. 6 shows the confusion matrix of the best and the second-best classification models based on their EAs. It should be noted that the confusion matrix can be used to calculate evaluation metrics. This matrix consists of four factors: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). In this paper, positive refers to the COVID-19 cases and negative refers to the non-COVID-19 cases. As can be concluded from Eqs.(17)-(21), the higher values of TP and TN indicate the higher performance of the classifier models. Therefore, Fig. 6 shows that the proposed method and the CSSA-CNN model are the best and the second-best performers, respectively.

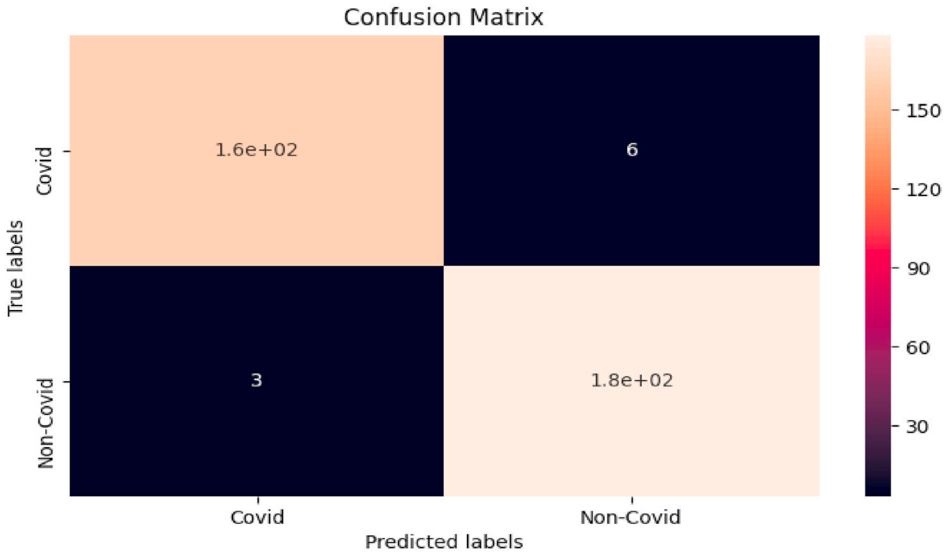
The results of experiments are shown in Fig. 7 as the box plots of different DNEs based on the accuracy metric. As we can see from these results, the proposed method is the best performer among all compared models, obtaining the best average accuracy value. Fig. 8 shows the convergence curves of different DNEs based on the accuracy metric. This analysis can help to compare the exploration ability of EAs in finding the optimal solution at high speed. As can be seen from Fig. 8, the proposed method achieves a better accuracy value when it is converged. The OSSA-CNN model is the second-best performer as it obtains a lower accuracy value than the proposed method. Therefore, it can be concluded that the proposed method can effectively explore the search space to find the optimal values of the CNN's hyperparameters leading to improved outcomes for the image classification problem.

4.3. *Performance comparison with other image classification approaches*

In this section, experiments are conducted to compare the proposed method with other image classification approaches. To this end, several image classification approaches (MobileNet, DenseNet121,



(a) Proposed model



(b) CSSA-CNN

Fig. 6. Confusion matrices of the best (proposed model) and the second-best (CSSA-CNN) algorithms based on the accuracy metric.

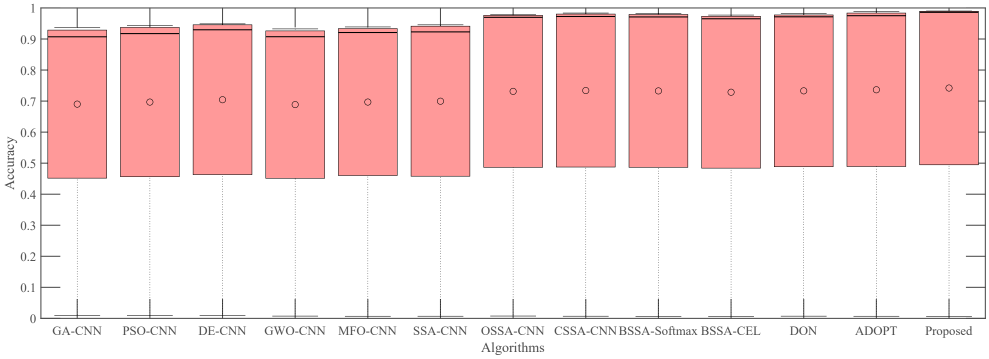


Fig. 7. Box plots of different DNEs for the accuracy metric.

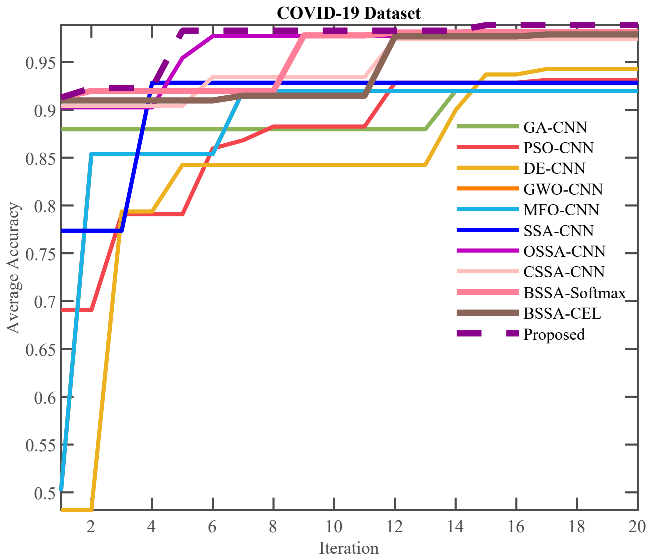


Fig. 8. Convergence curves of different DNEs based on the accuracy metric.

InceptionV3, ResNet50, VGGNet16, ResNet50V2 and ResNet152V2) are utilized to evaluate the performance of the proposed method. Two recent COVID-19 diagnosis research approaches (nCOVnet [84] and CTnet-10 [85]) are also considered as the competitive benchmark algorithms in order to compare with our proposed model. Experimental results based on different evaluation metrics are reported in Table 3 in terms of the average, best, and worst obtained results for the different approaches. The standard deviations of the results are also reported in this table. These results reveal that the proposed method significantly outperforms other models in terms of all evaluation metrics. The proposed method achieves an accuracy of 0.988 while the accuracy of the ResNet152V2 model, the second-best performer, is 0.908. Therefore, the proposed method improves classification accuracy by about 8% in comparison to the second-best model. Moreover, the best and worst results obtained by the proposed method are better than other models in terms of different evaluation metrics. The standard deviation of the results is known as a negative factor. For example, the lower the standard deviation, the better the performance of the model being evaluated. As the results show, the proposed method has a lower standard deviation for all evaluation metrics than the other classification models.

Fig. 9 shows the confusion matrix of the best and the second-best classification approaches. These results help to show how well the classifiers classify the input images in regard to both COVID-19 and non-COVID-19 cases. Obviously, the performance of the classification models depends on the number of COVID-19 and non-COVID-19 images that are truly classified. It can be seen from these results that the proposed method has a better confusion matrix than the ResNet152V2 model, which is the second-best classification model, because the number of COVID-19 and non-COVID-19 cases that are truly classified by the proposed method is higher than the ResNet152V2 model. The box plots of different classification models based on the accuracy metric are shown in Fig. 10. As these results show, the proposed method is the best performer among other models according to the average accuracy. Moreover, the standard deviation of the accuracy metric for the proposed method is lower than the other classification models. Therefore, the proposed method is the best performer according to the standard deviation of the accuracy metric.

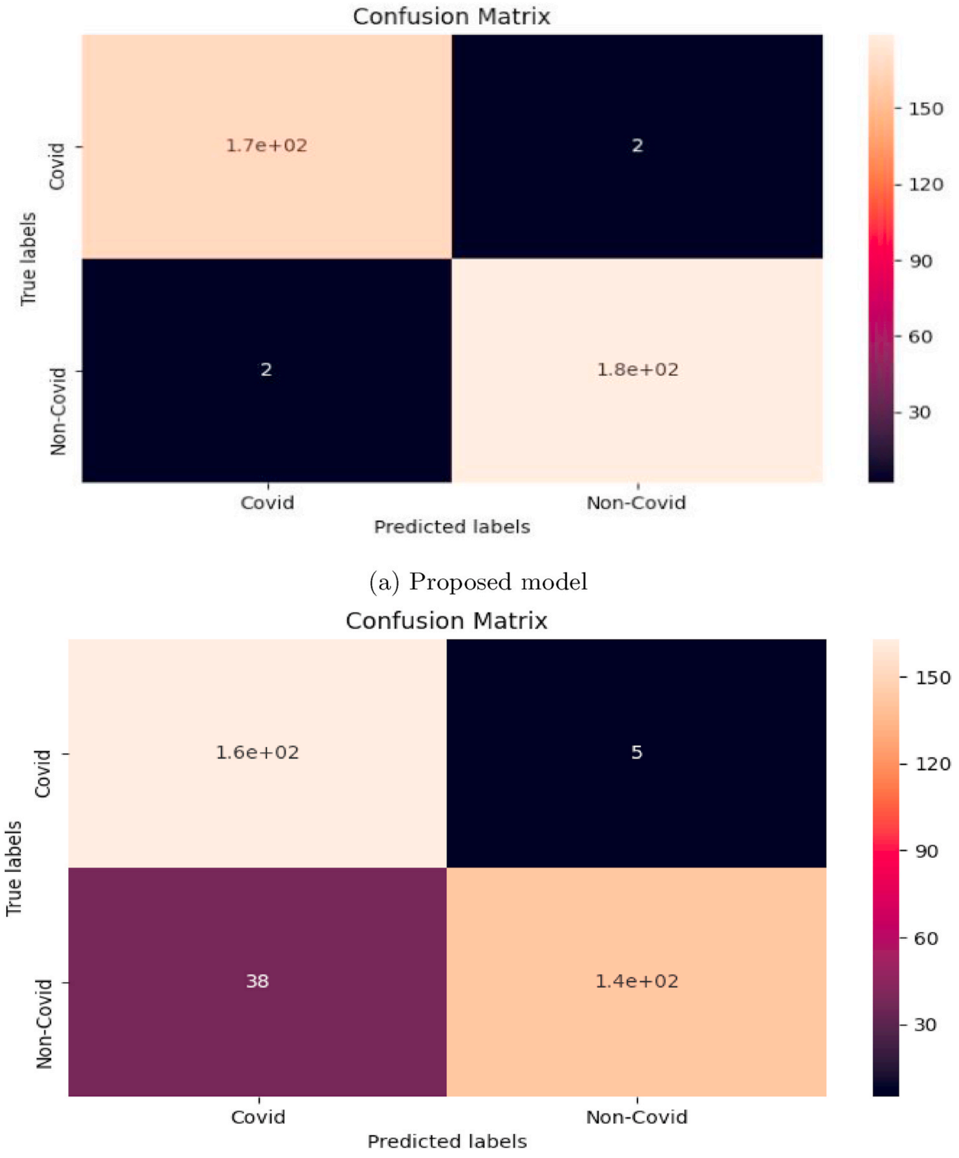
In order to demonstrate the significance of our proposed model in comparison to other EA-based and pre-trained deep-learning models, we perform the Wilcoxon paired signed-rank test on the COVID-19 dataset with confidence intervals of 95%. As can be observed from Tables 4 and 5, there is a significant difference between our model and the other 21 competitors. Also, it should be mentioned that the closest competitor – the ADOPT model, with a value of 4.495E-4 – is one of the EA-based deep-learning models.

4.4. Run-time analysis of different algorithms

In this section, we analyze the time spent on the execution of the proposed Algorithm and other competitive algorithms for the binary COVID-19 dataset. Table 6 demonstrates the time consumed by each algorithm in seconds. It should be noted that we report the run-time values in terms of optimization time, training time, and test time. As some of the compared algorithms do not undergo the optimization process, their optimization times are not reported. As can be seen from this table, the optimization time of our proposed algorithm is less than other methods that have used optimization models. Furthermore, in terms of training time and test time, our proposed algorithm takes less time than all the benchmark algorithms. This demonstrates the

Table 3
Quantitative performance results of evaluation metrics for state-of-the-art algorithms for the binary COVID-19 dataset. The best results among different classifiers are shown by bold black fonts.

Metric		MobileNet	DesnseNet121	InceptionV3	ResNet50	VGG16	ResNet50V2	ResNet152V2	nCOVnet	CTnet-10	Proposed
ACC	AVG	0.888252	0.882521	0.902579	0.836676	0.885387	0.785111	0.908309	0.896731	0.886574	**0.988539**
	STD	0.008435	0.009437	0.007891	0.040947	0.018467	0.047282	0.056214	0.043103	0.032791	**0.006189**
	Best	0.901883	0.903922	0.913226	0.893533	0.905344	0.826711	0.948979	0.936001	0.924471	**0.990218**
	Worst	0.865554	0.869047	0.879011	0.803216	0.827056	0.748022	0.853432	0.842854	0.832498	**0.983337**
Precision	AVG	0.852459	0.950355	0.848958	0.930233	0.890244	0.854962	0.914634	0.904323	0.894084	**0.988095**
	STD	0.046995	0.009899	0.042101	0.030749	0.033152	0.042653	0.110804	0.100705	0.090687	**0.002934**
	Best	0.883007	0.965652	0.858922	0.954767	0.915038	0.884578	0.937876	0.927388	0.917053	**0.989778**
	Worst	0.815561	0.937689	0.803574	0.874521	0.854538	0.821366	0.794022	0.782444	0.770778	**0.984971**
Recall	AVG	0.928571	0.797619	0.970238	0.714286	0.869048	0.666667	0.892857	0.881279	0.869121	**0.988095**
	STD	0.009231	0.073547	0.000213	0.086395	0.027655	0.069103	0.090967	0.078957	0.066918	**0.010117**
	Best	0.928895	0.814423	0.978845	0.786454	0.891044	0.736754	0.952612	0.943496	0.933685	**0.991309**
	Worst	0.897554	0.730567	0.966328	0.659113	0.837529	0.608932	0.811042	0.797931	0.785092	**0.985677**
F-measure	AVG	0.888889	0.867314	0.905556	0.808081	0.879518	0.749164	0.903614	0.892503	0.880941	**0.988095**
	STD	0.015655	0.027781	0.008944	0.053299	0.037424	0.061454	0.046513	0.037597	0.028482	**0.006526**
	Best	0.900767	0.893563	0.922155	0.846568	0.899045	0.795423	0.939082	0.929966	0.920728	**0.993044**
	Worst	0.841911	0.789022	0.899294	0.748989	0.845652	0.681921	0.857939	0.847628	0.837406	**0.979663**
AUC	AVG	0.889711	0.879473	0.905009	0.832281	0.884836	0.780847	0.907755	0.896839	0.885949	**0.988523**
	STD	0.010494	0.011373	0.009879	0.042439	0.035194	0.048984	0.053612	0.043531	0.031866	**0.006332**
	Best	0.910898	0.893315	0.921883	0.868494	0.908935	0.834326	0.943089	0.925978	0.908786	**0.989446**
	Worst	0.875497	0.869454	0.883533	0.773545	0.867435	0.738782	0.854025	0.841809	0.828476	**0.981088**



(a) Proposed model

(b) ResNet152V2

Fig. 9. Confusion matrices of the best (proposed model) and the second-best (ResNet152V2) classification algorithms based on the accuracy metric.

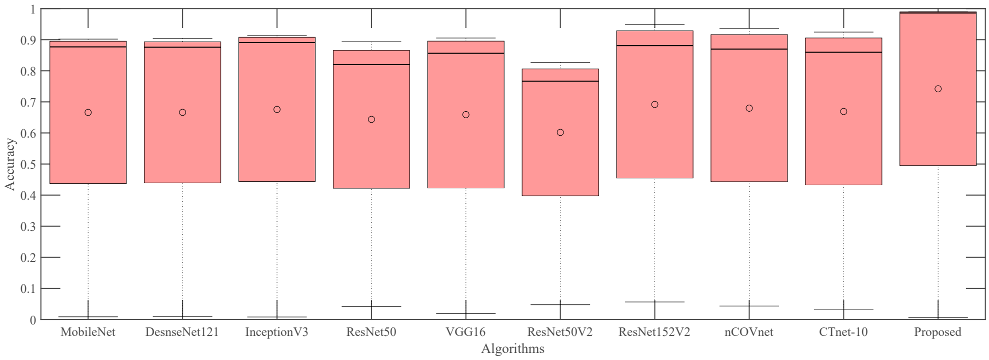


Fig. 10. Box plots of different classifiers based on the accuracy metric.

Table 4
Performance comparison of our proposed model against other EA-based deep-learning models based on the Wilcoxon paired signed-rank test with confidence intervals of 95%.

GA-CNN	PSO–CNN	DE-CNN	GWO-CNN	MFO-CNN	SSA-CNN	OSSA-CNN	CSSA-CNN	BSSA-Softmax	BSSA-CEL	DON	ADOPT
9.646E-6	9.725E-5	7.545E-5	9.121E-6	4.412E-5	8.333E-5	4.509E-4	4.661E-4	5.408E-4	2.394E-5	5.661E-4	4.495E-4

Table 5
Performance comparison of our proposed model against other deep-learning models based on the Wilcoxon paired signed-rank test with confidence intervals of 95%.

MobileNet	DesnseNet121	InceptionV3	ResNet50	VGG16	ResNet50V2	ResNet152V2	nCOVnet	CTnet-10
7.369E-6	7.825E-6	6.131E-6	7.375E-7	7.694E-6	5.661E-8	5.656E-6	6.779E-6	7.551E-6

Table 6
The run-time consumed in seconds for all algorithms over the COVID-19 dataset.

Model	Optimization time	Training time	Test time
MobileNet	–	641	239
DesnseNet121	–	624	231
InceptionV3	–	583	198
ResNet50	–	612	213
VGG16	–	567	192
ResNet50V2	–	539	195
ResNet152V2	–	542	183
nCOVnet	–	531	166
CTnet-10	–	512	171
GA-CNN	2846	497	169
PSO–CNN	2689	451	142
DE-CNN	2710	482	157
GWO-CNN	2635	423	162
MFO-CNN	2647	411	153
SSA-CNN	2479	394	149
OSSA-CNN	2422	361	132
CSSA-CNN	2461	389	128
BSSA-Softmax	2453	376	117
BSSA-CEL	2411	348	106
Proposed	**2378**	**311**	**94**

effectiveness of this algorithm in the calculations related to the classification of X-ray COVID-19 images in the benchmark dataset. It should be noted that the optimization and training phases of the proposed method can be performed once in offline mode. Then, the trained model can be applied frequently to diagnose COVID-19 cases in online mode. Therefore, the test time is an important criterion for real-time applications which need to be executed in a hard-deadline scenario. As the proposed method needs the shortest test time among the compared classification models, it can be concluded that the proposed method is more applicable than other models for real-time applications.

5. Conclusion

This paper has proposed a method based on a deep neuroevolution model for classifying COVID-19 cases from X-ray images. To this end, the CNN model is employed as a classifier model to diagnose COVID-19 cases based on chest X-ray images. To enhance the performance of the CNN model in the classification of the input images, an evolutionary Algorithm named BSSA is used to obtain the optimal values of the hyperparameters of the CNN model. Experimental results on a real-world dataset show that the proposed image classification model performs efficiently and accurately in classifying COVID-19 cases. It is worth noting that the diagnosis of COVID-19 cases is a critical issue in controlling the spread of COVID-19 disease. Therefore, the proposed model can be employed in many real applications related to COVID-19. Moreover, the proposed method is a general image classification model which can be applied to any image classification problem. For instance, skin cancer classification is a real application that can be addressed by the proposed method in future work. Although it is shown that the proposed method has significant advantages in accurately diagnosing COVID-19 cases, this method has also some limitations and drawbacks. In particular, the proposed method only uses one deep neural network model in the classification process. Considering a set of deep neural networks in the classification process to make an ensemble classifier may increase the effectiveness of the proposed method. Moreover, the performance of the proposed method strongly depends on the values of the hyperparameters of the CNN model, which are determined by the proposed evolutionary algorithm. Therefore, if the proposed evolutionary algorithm fails to determine the optimal values of these hyperparameters, it will lead to a decline in the effectiveness of the proposed method. Therefore, other evolutionary algorithms should also be tested for a specific image classification problem to compare their performance with the proposed evolutionary algorithm. In future work, other deep neural network models and evolutionary algorithms can be used to improve the performance of the proposed image classification model. Moreover, advanced data pre-processing models can be applied on the input images to improve the quality of these images leading to enhanced classifier performance.

Declaration of competing interest

The authors do not have any conflict of interest for our manuscript.

References

[1] P. Mehta, D.F. McAuley, M. Brown, E. Sanchez, R.S. Tattersall, J.J. Manson, H.A. S. Collaboration, et al., Covid-19: consider cytokine storm syndromes and immunosuppression, Lancet (London, England) 395 (2020) 1033–1034.

[2] S. Kumar, V. Maheshwari, J. Prabhu, M. Prasanna, P. Jayalakshmi, P. Suganya, B. A. Malar, R. Jothikumar, Social economic impact of covid-19 outbreak in India, Int. J. Pervasive Comput. Commun. (2020) 1–12.

[3] M. Toğaçar, B. Ergen, Z. Cömert, Covid-19 detection using deep learning models to exploit social mimic optimization and structured chest x-ray images using fuzzy color and stacking approaches, Comput. Biol. Med. (2020) 103805.

[4] P. Walker, C. Whittaker, O. Watson, M. Baguelin, K. Ainslie, S. Bhatia, S. Bhatt, A. Boonyasiri, O. Boyd, L. Cattarino, et al., reportReport 12: the Global Impact of Covid-19 and Strategies for Mitigation and Suppression.

[5] H.A. Rothan, S.N. Byrareddy, The epidemiology and pathogenesis of coronavirus disease (covid-19) outbreak, J. Autoimmun. (2020) 102433.

[6] L. Pan, M. Mu, P. Yang, Y. Sun, R. Wang, J. Yan, P. Li, B. Hu, J. Wang, C. Hu, et al., Clinical characteristics of covid-19 patients with digestive symptoms in hubei, China: a descriptive, cross-sectional, multicenter study, Am. J. Gastroenterol. 115 (2020) 766–773.

[7] J. Wong, Q.Y. Goh, Z. Tan, S.A. Lie, Y.C. Tay, S.Y. Ng, C.R. Soh, Preparing for a covid-19 pandemic: a review of operating room outbreak response measures in a large tertiary hospital in Singapore, Canad. J. Anesthesia/J. canadien d'anesthésie (2020) 1–14.

[8] S.M.J. Jalali, M. Karimi, A. Khosravi, S. Nahavandi, An efficient neuroevolution approach for heart disease detection, in: 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), IEEE, 2019, pp. 3771–3776.

[9] S.M.J. Jalali, S. Ahmadian, P.M. Kebria, A. Khosravi, C.P. Lim, S. Nahavandi, Evolving artificial neural networks using butterfly optimization Algorithm for data classification, in: International Conference on Neural Information Processing, Springer, 2019, pp. 596–607.

[10] S.M.J. Jalali, A. Khosravi, R. Alizadehsani, S.M. Salaken, P.M. Kebria, R. Puri, S. Nahavandi, in: Parsimonious Evolutionary-Based Model Development for Detecting Artery Disease, ICIT, 2019, pp. 800–805.

[11] S. Ahmadian, A.R. Khanteymoori, Training back propagation neural networks using asexual reproduction optimization, in: 2015 7th Conference on Information and Knowledge Technology (IKT), IEEE, 2015, pp. 1–6.

[12] S.J. Mousavirad, G. Schaefer, S.M.J. Jalali, I. Korovin, A benchmark of recent population-based metaheuristic algorithms for multi-layer neural network training, in: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, 2020, pp. 1402–1408.

[13] S.J. Mousavirad, S.M.J. Jalali, S. Ahmadian, A. Khosravi, G. Schaefer, S. Nahavandi, Neural network training using a biogeography-based learning strategy, in: International Conference on Neural Information Processing, Springer, 2020, pp. 147–155.

[14] Z. Ebrahimi, M. Loni, M. Daneshtalab, A. Gharehbaghi, A review on deep learning methods for ecg arrhythmia classification, Expert Syst. Appl. X 7 (2020) 100033.

[15] X. Chen, L. Li, A. Sharma, G. Dhiman, S. Vimal, The application of convolutional neural network model in diagnosis and nursing of mr imaging in alzheimer's disease, Interdiscipl. Sci. Comput. Life Sci. (2021) 1–11.

[16] P.D. Singh, R. Kaur, K.D. Singh, G. Dhiman, A novel ensemble-based classifier for detecting the covid-19 disease for infected patients, Inf. Syst. Front (2021) 1–17.

[17] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M.P. Reyes, M.-L. Shyu, S.-C. Chen, S. Iyengar, A survey on deep learning: algorithms, techniques, and applications, ACM Comput. Surv. (CSUR) 51 (5) (2018) 1–36.

[18] S.M.J. Jalali, S. Ahmadian, A. Khosravi, S. Mirjalili, M.R. Mahmoudi, S. Nahavandi, Neuroevolution-based autonomous robot navigation: a comparative study, Cognit. Syst. Res. 62 (2020) 35–43.

[19] M. Khodayar, M. E. Khodayar, S. M. J. Jalali, Deep learning for pattern recognition of photovoltaic energy generation, Electr. J. 34 (1) 106882.

[20] S.M.J. Jalali, S. Ahmadian, M. Khodayar, A. Khosravi, V. Ghasemi, M. Shafie-khah, S. Nahavandi, J.P. Catalão, Towards novel deep neuroevolution models: chaotic levy grasshopper optimization for short-term wind speed forecasting, Eng. Comput. (2021) 1–25.

[21] S.M.J. Jalali, S. Ahmadian, A. Khosravi, M. Shafie-khah, S. Nahavandi, J. P. Catalão, A novel evolutionary-based deep convolutional neural network model for intelligent load forecasting, IEEE Trans. Ind. Inform. 17 (2021) 8243–8253.

[22] L. Ru, B. Zhang, J. Duan, G. Ru, A. Sharma, G. Dhiman, G.S. Gaba, E.S. Jaha, M. Masud, A detailed research on human health monitoring system based on internet of things, Wireless Commun. Mobile Comput. (2021) 1–9, 2021.

[23] S.M.J. Jalali, S. Ahmadian, A. Kavousi-Fard, A. Khosravi, S. Nahavandi, Automated deep cnn-lstm architecture design for solar irradiance forecasting, IEEE Trans. Syst. Man Cybern. Syst. (2021) 1–12.

[24] M. Saffari, M. Khodayar, S.M.J. Jalali, M. Shafie-khah, J.P. Catalão, Deep convolutional graph rough variational auto-encoder for short-term photovoltaic power forecasting, in: 2021 International Conference on Smart Energy Systems and Technologies (SEST), IEEE, 2021, pp. 1–6.

[25] S.M.J. Jalali, M. Khodayar, S. Ahmadian, M. Shafie-khah, A. Khosravi, S.M. S. Islam, S. Nahavandi, J.P. Catalão, A new ensemble reinforcement learning strategy for solar irradiance forecasting using deep optimized convolutional neural network models, in: 2021 International Conference on Smart Energy Systems and Technologies (SEST), IEEE, 2021, pp. 1–6.

[26] H.D. Kabir, A. Khosravi, A. Kavousi-Fard, S. Nahavandi, D. Srinivasan, Optimal uncertainty-guided neural network training, Appl. Soft Comput. 99 (2021) 106878.

[27] H. D. Kabir, A. Khosravi, S. Nahavandi, D. Srinivasan, Neural network training for uncertainty quantification over time-range, IEEE Transactions on Emerging Topics in Computational Intelligence.

[28] H.D. Kabir, A. Khosravi, S.K. Mondal, M. Rahman, S. Nahavandi, R. Buyya, Uncertainty-aware decisions in cloud computing: foundations and future directions, ACM Comput. Surv. (CSUR) 54 (4) (2021) 1–30.

[29] M. Mir, H.D. Kabir, F. Nasirzadeh, A. Khosravi, Neural network-based interval forecasting of construction material prices, J. Build. Eng. 39 (2021) 102288.

[30] F. Nasirzadeh, H. D. Kabir, M. Akbari, A. Khosravi, S. Nahavandi, D. G. Carmichael, Ann-based Prediction Intervals to Forecast Labour Productivity, Engineering, Construction and Architectural Management.

[31] H.D. Kabir, A. Khosravi, D. Nahavandi, S. Nahavandi, Uncertainty quantification neural network from similarity and sensitivity, in: 2020 International Joint Conference on Neural Networks (IJCNN), IEEE, 2020, pp. 1–8.

[32] F. Tahmasebi, M. Meghdadi, S. Ahmadian, K. Valiallahi, A hybrid recommendation system based on profile expansion technique to alleviate cold start problem, Multimed. Tool. Appl. 80 (2) (2021) 2339–2354.

[33] S. Ahmadian, P. Moradi, F. Akhlaghian, An improved model of trust-aware recommender systems using reliability measurements, in: 2014 6th Conference on Information and Knowledge Technology (IKT), IEEE, 2014, pp. 98–103.

[34] P. Moradi, F. Rezaimehr, S. Ahmadian, M. Jalili, A trust-aware recommender algorithm based on users overlapping community structure, in: 2016 Sixteenth International Conference on Advances in ICT for Emerging Regions (ICTer), IEEE, 2016, pp. 162–167.

[35] S. Ahmadian, M. Meghdadi, M. Afsharchi, Incorporating reliable virtual ratings into social recommendation systems, Appl. Intell. 48 (11) (2018) 4448–4469.

[36] H.A. Rahmani, M. Aliannejadi, S. Ahmadian, M. Baratchi, M. Afsharchi, F. Crestani, Lglmf: local geographical based logistic matrix factorization model for poi recommendation, in: Asia Information Retrieval Symposium, Springer, 2019, pp. 66–78.

[37] S. Ahmadian, N. Joorabloo, M. Jalili, Y. Ren, M. Meghdadi, M. Afsharchi, A social recommender system based on reliable implicit relationships, Knowl. Base Syst. 192 (2020) 105371.

[38] P. Moradi, S. Ahmadian, A reliability-based recommendation method to improve trust-aware recommender systems, Expert Syst. Appl. 42 (2015) 7386–7398.

[39] P. Moradi, S. Ahmadian, F. Akhlaghian, An effective trust-based recommendation method using a novel graph clustering algorithm, Phys. Stat. Mech. Appl. 436 (2015) 462–481.

[40] M. Jalili, S. Ahmadian, M. Izadi, P. Moradi, M. Salehi, Evaluating collaborative filtering recommender algorithms: a survey, IEEE Access 6 (2018) 74003–74024.

[41] F. Rezaeimehr, P. Moradi, S. Ahmadian, N. Nasih Qader, M. Jalili, Tcars: time-and community-aware recommendation system, Future Generat. Comput. Syst. 78 (2018) 419–429.

[42] S. Ahmadian, M. Meghdadi, M. Afsharchi, A social recommendation method based on an adaptive neighbor selection mechanism, Inf. Process. Manag. 54 (2018) 707–725.

[43] S. Ahmadian, M. Afsharchi, M. Meghdadi, A novel approach based on multi-view reliability measures to alleviate data sparsity in recommender systems, Multimed. Tool. Appl. 78 (2019) 17763–17798.

[44] S. Ahmadian, M. Afsharchi, M. Meghdadi, An effective social recommendation method based on user reputation model and rating profile enhancement, J. Inf. Sci. 45 (2019) 607–642.

[45] S. Ahmadian, N. Joorabloo, M. Jalili, M. Meghdadi, M. Afsharchi, Y. Ren, A temporal clustering approach for social recommender systems, in: 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2018, pp. 1139–1144.

[46] A. Jaiswal, N. Gianchandani, D. Singh, V. Kumar, M. Kaur, Classification of the covid-19 infected patients using densenet201 based deep transfer learning, J. Biomol. Struct. Dyn. (2020) 1–8.

[47] I.D. Apostolopoulos, T.A. Mpesiana, Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks, Phys. Eng. Sci. Med. (2020) 635–640.

[48] G.K. Sodhi, S. Kaur, G.S. Gaba, L. Kansal, A. Sharma, G. Dhiman, Covid-19: role of robotics, artificial intelligence, and machine learning during pandemic, Curr. Med. Imag. (2021) 1–12.

[49] S. Gomathi, R. Kohli, M. Soni, G. Dhiman, R. Nair, Pattern analysis: predicting covid-19 pandemic in India using automl, World J. Eng. (2020) 1–11.

[50] N. Yuvaraj, K. Srihari, S. Chandragandhi, R.A. Raja, G. Dhiman, A. Kaur, Analysis of protein-ligand interactions of sars-cov-2 against selective drug using deep neural networks, Big Data Min. Anal. 4 (2) (2021) 76–83.

[51] I.M. Baltruschat, H. Nickisch, M. Grass, T. Knopp, A. Saalbach, Comparison of deep learning approaches for multi-label chest x-ray classification, Sci. Rep. 9 (1) (2019) 1–10.

[52] S.S. Yadav, S.M. Jadhav, Deep convolutional neural network based medical image classification for disease diagnosis, J. Big Data 6 (1) (2019) 1–18.

[53] L. Sarker, M.M. Islam, T. Hannan, Z. Ahmed, Covid-densenet: a deep learning architecture to detect covid-19 from chest radiology images, Math. Comput. Sci. (2020) 1–13.

[54] F. Shan, Y. Gao, J. Wang, W. Shi, N. Shi, M. Han, Z. Xue, Y. Shi, Lung infection quantification of covid-19 in ct images with deep learning, Comput. Vision Pattern Recogn. (2020) 1–19.

[55] K. Elasnaoui, Y. Chawki, Using x-ray images and deep learning for automated detection of coronavirus disease, J. Biomol. Struct. Dyn. (2020) 1–22 (just-accepted).

[56] L. Li, L. Qin, Z. Xu, Y. Yin, X. Wang, B. Kong, J. Bai, Y. Lu, Z. Fang, Q. Song, et al., Artificial intelligence distinguishes covid-19 from community acquired pneumonia on chest ct, Radiology (2020) 200905.

[57] O. Stephen, M. Sain, U.J. Maduh, D.-U. Jeong, An efficient deep learning approach to pneumonia classification in healthcare, J. Healthcare Eng. (2019) 1–7, 2019.

[58] S. Hu, Y. Gao, Z. Niu, Y. Jiang, L. Li, X. Xiao, M. Wang, E.F. Fang, W. Menpes-Smith, J. Xia, et al., Weakly supervised deep learning for covid-19 infection detection and classification from ct images, IEEE Access 8 (2020) 118869–118883.

[59] A. Abbas, M.M. Abdelsamea, M.M. Gaber, Classification of covid-19 in chest x-ray images using detrac deep convolutional neural network, Appl. Intell. 51 (2) (2021) 854–864.

[60] A. Narin, C. Kaya, Z. Pamuk, Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks, Pattern Anal. Appl. 24 (2020) 1207–1220.

[61] G. Dhiman, V. Chang, K. Kant Singh, A. Shankar, Adopt: automatic deep learning and optimization-based approach for detection of novel coronavirus covid-19 disease using x-ray images, J. Biomol. Struct. Dynam. (2021) 1–13.

[62] G. Dhiman, V.V. Kumar, A. Kaur, A. Sharma, Don: deep learning and optimization-based framework for detection of novel coronavirus disease using x-ray images, Interdiscipl. Sci. Comput. Life Sci. (2021) 1–13.

[63] J. Cao, X. Zhang, C. Zhang, J. Feng, Improved convolutional neural network combined with rough set theory for data aggregation algorithm, J. Ambient Intell. Humanized Comput. 11 (2) (2020) 647–654.

[64] U.R. Acharya, H. Fujita, O.S. Lih, M. Adam, J.H. Tan, C.K. Chua, Automated detection of coronary artery disease using different durations of ecg segments with convolutional neural network, Knowl. Base Syst. 132 (2017) 62–71.

[65] D. Han, Q. Liu, W. Fan, A new image classification method using cnn transfer learning and web data augmentation, Expert Syst. Appl. 95 (2018) 43–56.

[66] D.-X. Xue, R. Zhang, H. Feng, Y.-L. Wang, Cnn-svm for microvascular morphological type recognition with data augmentation, J. Med. Biol. Eng. 36 (6) (2016) 755–764.

[67] K. Zeng, S. Ding, W. Jia, Single image super-resolution using a polymorphic parallel cnn, Appl. Intell. 49 (1) (2019) 292–300.

[68] F.J. Huang, Y. LeCun, Large-scale learning with svm and convolutional for generic object categorization, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 1, IEEE, 2006, pp. 284–291.

[69] A. Michele, V. Colin, D.D. Santika, Mobilenet convolutional neural networks and support vector machines for palmprint recognition, Procedia Comput. Sci. 157 (2019) 110–117.

[70] S. Mirjalili, A.H. Gandomi, S.Z. Mirjalili, S. Saremi, H. Faris, S.M. Mirjalili, Salp swarm algorithm: a bio-inspired optimizer for engineering design problems, Adv. Eng. Software 114 (2017) 163–191.

[71] G.I. Sayed, G. Khoriba, M.H. Haggag, A novel chaotic salp swarm algorithm for global optimization and feature selection, Appl. Intell. 48 (10) (2018) 3462–3481.

[72] H. Faris, M.M. Mafarja, A.A. Heidari, I. Aljarah, A.-Z. Ala'M, S. Mirjalili, H. Fujita, An efficient binary salp swarm algorithm with crossover scheme for feature selection problems, Knowl. Base Syst. 154 (2018) 43–67.

[73] R. Abbassi, A. Abbassi, A.A. Heidari, S. Mirjalili, An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models, Energy Convers. Manag. 179 (2019) 362–372.

[74] A.A. Ewees, M. Abd El Aziz, A.E. Hassanien, Chaotic multi-verse optimizer-based feature selection, Neural Comput. Appl. 31 (4) (2019) 991–1006.

[75] Y. Luo, R. Zhou, J. Liu, Y. Cao, X. Ding, A parallel image encryption algorithm based on the piecewise linear chaotic map and hyper-chaotic map, Nonlinear Dynam. 93 (3) (2018) 1165–1181.

[76] A.A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, An efficient hybrid multilayer perceptron neural network with grasshopper optimization, Soft Comput. 23 (17) (2019) 7941–7958.

[77] S.M.J. Jalali, M. Ahmadian, S. Ahmadian, A. Khosravi, M. Alazab, S. Nahavandi, An oppositional-cauchy based gsk evolutionary algorithm with a novel deep ensemble reinforcement learning strategy for covid-19 diagnosis, Appl. Soft Comput. 111 (2021) 107675.

[78] S.M.J. Jalali, P.M. Kebria, A. Khosravi, K. Saleh, D. Nahavandi, S. Nahavandi, Optimal autonomous driving through deep imitation learning and neuroevolution, in: 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), IEEE, 2019, pp. 1215–1220.

[79] T.Y. Tan, L. Zhang, C.P. Lim, Intelligent skin cancer diagnosis using improved particle swarm optimization and deep learning models, Appl. Soft Comput. 84 (2019) 105725.

[80] W. Deng, H. Liu, J. Xu, H. Zhao, Y. Song, An improved quantum-inspired differential evolution algorithm for deep belief network, IEEE Trans. Instrument. Meas. (2020) 7319–7327.

[81] P. Hu, J.-S. Pan, S.-C. Chu, Improved Binary Grey Wolf Optimizer and its Application for Feature Selection, Knowledge-Based Systems, 2020, p. 105746.

[82] C. Li, S. Li, Y. Liu, A least squares support vector machine model optimized by moth-flame optimization algorithm for annual power load forecasting, Appl. Intell. 45 (4) (2016) 1166–1178.

[83] B. Yang, L. Zhong, X. Zhang, H. Shu, T. Yu, H. Li, L. Jiang, L. Sun, Novel bio-inspired memetic salp swarm algorithm and application to mppt for pv systems considering partial shading condition, J. Clean. Prod. 215 (2019) 1203–1222.

[84] H. Panwar, P. Gupta, M.K. Siddiqui, R. Morales-Menendez, V. Singh, Application of deep learning for fast detection of covid-19 in x-rays using ncovnet, Chaos, Solit. Fractals 138 (2020) 109944.

[85] V. Shah, R. Keniya, A. Shridharani, M. Punjabi, J. Shah, N. Mehendale, Diagnosis of covid-19 using ct scan images and deep learning techniques, Emerg. Radiol. (2021) 1–9.