Computers, Materials & Continua
DOI: 10.32604/cmc.2023.031519
Article

Tech Science Press

A Deep Learning Approach for Detecting Covid-19 Using the Chest X-Ray Images

Fatemeh Sadeghi[1], Omid Rostami[2], Myung-Kyu Yi[3] and Seong Oun Hwang[3,*]

[1]Department of Industrial Engineering, Sharif University of Technology, Tehran, 14588-89694, Iran
[2]Department of Industrial Engineering, University of Houston, Houston, TX, 77204, USA
[3]Department of Computer Engineering, Gachon University, Seongnam, 13120, Korea
*Corresponding Author: Seong Oun Hwang. Email: sohwang@gachon.ac.kr
Received: 20 April 2022; Accepted: 24 June 2022

Abstract: Real-time detection of Covid-19 has definitely been the most widely-used world-wide classification problem since the start of the pandemic from 2020 until now. In the meantime, airspace opacities spreads related to lung have been of the most challenging problems in this area. A common approach to do on that score has been using chest X-ray images to better diagnose positive Covid-19 cases. Similar to most other classification problems, machine learning-based approaches have been the first/most-used candidates in this application. Many schemes based on machine/deep learning have been proposed in recent years though increasing the performance and accuracy of the system has still remained an open issue. In this paper, we develop a novel deep learning architecture to better classify the Covid-19 X-ray images. To do so, we first propose a novel multi-habitat migration artificial bee colony (MHMABC) algorithm to improve the exploitation/exploration of artificial bee colony (ABC) algorithm. After that, we optimally train the fully connected by using the proposed MHMABC algorithm to obtain better accuracy and convergence rate while reducing the execution cost. Our experiment results on Covid-19 X-ray image dataset show that the proposed deep architecture has a great performance in different important optimization parameters. Furthermore, it will be shown that the MHMABC algorithm outperforms the state-of-the-art algorithms by evaluating its performance using some well-known benchmark datasets.

Keywords: Chest X-ray image processing; evolutionary deep learning; covid-19

1 Introduction

The rapid and accurate detection of Covid-19 has been one of the most important/challenging problems for the scientists since the beginning of the pandemic in 2020. Among various solutions/tests, the Polymerase Chain Reaction (PCR) test has been the most-used method for detecting the Covid-19 as a world-wide approach. PCR test though has shown as an arduous test, requiring considerable time to get a result, and needing complicated procedure with the short supply kits. Contrarily, not only the X-ray images are highly attainable but also the scans are considerably low cost [1–6].

The exigency of building a real-time Covid-19 indicator that accurately detects the coronavirus is growing day-by-day. As analyzing the X-ray images are driven from the image processing topic, deep learning would be a promising candidate to do on that score. Therefore, our aim is to provide accurate X-ray images classifier based on deep convolutional neural network (DCNN) to efficiently detect the COVID-19 cases. According to the current literature, a few researches have been done in this specific area since the beginning of the pandemic [7–15]. Regardless of deep learning's extraordinary ability to solve a wide range of complex problems, optimally updating the weights and biases of the fully-connected neural network (training procedure) is one of the most challenging issues that creates so much potential to improve the performance of the classifier [16–21]. There are a few algorithms which have been frequently used to train the fully-connected in the deep architecture e.g., gradient descent (GD) [22,23], conjugate gradient (CG) [24–26], arbitrary Lagrangian–Eulerian (ALE) [27–29], Hessian-Free optimization (HFO) [30], and Krylov subspace descent (KSD) [31].

1.1 Related Works

It is known that GD-based algorithms for training purposes have relatively lower implementation cost and more rapid execution time [32]. However, these algorithms require various tuning parameters (which should be set manually) to improve the entire performance of the system. Furthermore, the GD-based training algorithms is sequential in essence, thus, parallelizing them during the hardware implementations, for example, on microcontrollers [33] or field programmable gate arrays (FPGAs) would be so challenging. In addition, GD-based algorithms are comparatively slow, regardless of their stable behavior when they are using to train a fully-connected [34]. Therefore, GD-based algorithms require a considerable computational capacity and large random access memory (RAM) resources [35]. A little after, HFO has been used to train the deep architectures especially for deep auto-encoders. It has been shown that this algorithm has a better performance in pre-training and optimizing of deep auto-encoders in comparison with the proposed algorithm by Hinton and Salakhutdinov. It is worth mentioning that, KSD has a lower implementation complexity comparing to HFO while having more robust performance. In addition, the further studies have shown that KSD has a higher classification accuracy and convergence rate than HFO. Nevertheless, KSD needs larger storage space than HFO [30].

Although meta-heuristic-based approaches have played an important role to find the best solutions for a wide range of optimization and complicated problems, the efforts regarding optimizing the deep architectures using the meta-heuristic algorithms are still on the very beginning of their path. The proposed scheme in [36] was the first effort that employed the genetic algorithm (GA) to train a DCNN. This effort had benefited the advantage of different operators in GA (e.g., selection, crossover and mutation) to train the deep architecture by considering the weights and biases of each network as a chromosome. Furthermore, in the re-mixture stage, only the weights and biases of the first and third convolution layers have been rectified. Rosa et al. [37] proposed a harmony search (HS)-based approach for optimizing the DCNN parameters. The authors in [38] have presented a scheme based on progressive unsupervised learning (PUL) for tuning the parameters of pre-trained DCNN. They have shown that their approach can be efficiently implemented and can also be used as an optimized baseline for unsupervised learning. This approach can add a new feature as a selection option between the clustering and fine-tuning phases when the results of the clustering phase is noisy.

The authors in [39] have proposed a GA-based automatic DCNN for optimal image classification. The automatic characteristic of the presented GA is its most important feature in which there is no need to know additional information about the trained DCNN. Though, the main weakness of this approach is the training algorithm turns slow when the deep structure is large. Because the

GA's chromosomes will be considerably large consequently. The authors in [40] have also used the advantage of social-spider optimization (SSO) algorithm to train a deep adaptive neuro fuzzy inference system (ANFIS) for predicting the biochar yields [41]; though, the proposed scheme has faced the ill-conditioning issue. More training algorithms have also been proposed in recent years to better the performance of deep architectures [42–45]. Though, the mentioned schemes lack in providing low computational cost and unreliability when the inputs of the deep architecture are high-dimension and a large set of images.

1.2 Paper Contributions

According to the mentioned drawbacks and weaknesses of the proposed schemes in the literature, we aim to develop a novel meta-heuristic-based approach to optimally training a DCNN to accurately detect the coronavirus by using the COVIDetectioNet dataset [46]. Furthermore, we will update the last layer in the deep architecture of a fully-connected neural network by replacing it with a new fully-connected neural network trained by a novel MHMABC algorithm. The contributions of this paper are as follows:

- We develop a meta-heuristic algorithm named MHMABC in which the migration operator of the biogeography-based optimization (BBO) is used to improve the exploitation ability of ABC algorithm for the first time.
- We evaluate the performance of MHMABC algorithm using some benchmark datasets [46].
- We optimally improve the performance of the DCNN by updating its optimization parameters using the proposed MHMABC algorithm.
- By using the proposed deep architecture to analyze the X-ray images, we propose an accurate scheme over PCR to detect the Covid-19 cases.

The reminder of this paper is organized as follows. Section 2 represents the proposed MHMABC algorithm. Section 3 describes the developed deep architecture trained by MHMABC algorithm. Section 4 evaluates the performance of MHMABC algorithm on some well-known benchmark datasets. In Section 5, we use the proposed deep architecture to analyze the chest X-ray images by using the COVIDetectioNet dataset [46], and finally, we draw a conclusion of this paper in Section 6.

2 The Proposed MHMABC Algorithm

Artificial bee colony (ABC) is a swarm-based meta-heuristic algorithm that first introduced by Karaboga [47] in 2005. ABC algorithm was inspired by the intelligent search behavior of honey bees. In ABC algorithm, a colony consists of three types of artificial bees:

- **Scout bees:** Solutions that are randomly generated to discover new spaces are called scout bees. Scout bees are responsible for exploring the search space.
- **Employed bees:** A number of scout bees with good fitness function become employed bees. Employed bees are responsible for advertising quality food sources.
- **Onlooker bees:** The onlooker bees are responsible for searching the neighborhood for employed bees. Onlooker bees receive information about food sources and search around these sources. The role of these bees is both exploitation and exploration of algorithm.

In ABC algorithm, scout bees discover a population of initial solution vectors randomly and repeatedly improve them by onlooker and employed bees (using neighbor search method to move towards better solutions while eliminating poor solutions). In general, ABC algorithm uses two main methods (neighbor search and random search) to get the optimal answer: Random search by scout and

onlooker bees and neighbor search by employed and onlooker bees. In ABC algorithm, each candidate answer indicates the position of food source, and the quality of the nectar is used as a fitness function. In this algorithm, first, all initial populations are explored by scout bees. Scout bees with best fitness functions are selected as the employed bees. Employed bees exploit the solution positions and then onlooker bees are created. The higher the quality of the employed bee, the more onlooker bees will be created around it. The onlooker bee selects new food position (based on the information of the employed bee) and exploit around these positions. Next, random scout bees are created to find new random food positions. ABC algorithm can be formulated as Eqs. (1)–(3) [46].

$$P_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{1}$$

$$V_{ij} = X_{ij} + \varphi_{ij}\left(X_{ij} - X_{kj}\right) \tag{2}$$

$$X_L^j = X_{min}^j + rand\ (0,\ 1)\left(X_{max}^j - X_{min}^j\right) \tag{3}$$

where, P_i = Probability of selecting employed bees by onlooker bees, fit_i = Fitness function of the i^{th} solution, V_{ij} = Onlooker bee, X_L^j = Scout bees, X_{min}^j = Low limit of search space, X_{max}^j = High limit of search space, SN = Number of employed bees, $I \in \{1, 2, \ldots, \text{SN}\}$, j = Dimension $\in \{1, 2, \ldots, \text{D}\}$, k = Onlooker bee number, φ_{ij} is the random number $\in [0, 1]$, and L = Scout bee number. The weaknesses of the standard ABC algorithm include the lack of exploitation, as well as becoming stuck in local minimums. In ABC algorithm, only neighborhood searches is done around the employed bees. The onlooker bees select new food positions and exploit around these positions. The main purpose of this operator is to improve the fitness function of solutions by neighborhood searches around the employed bee. If the onlooker bees, in addition to neighborhood searches around the employed bee, move towards the several best positions, this movement will cause a variety of good solutions and find other parts of the search space. Thus, a better local/global search is performed in the ABC algorithm, as well as creating variety in the solutions, the convergence rate of the algorithm will also increase. In this paper, a new multi-habitat migration is proposed to improve the exploitation and exploration of ABC algorithm. To develop proposed MHMABC algorithm, the migration operator of the BBO algorithm [48] is used to improve the exploitation of the ABC algorithm. Multi-habitat migration is also proposed to improve the exploration of the algorithm. The migration operation occurs when a guest habitat with an emigration rate sends its species to a host habitat with an immigration rate. In this paper the exponential-logarithmic migration model is used [15]. Emigration rate ($\mu_{k(j)}$) and immigration rate ($\lambda_{k(j)}$) are defined as functions of the number of their species as Eqs. (4) and (5):

$$\begin{cases} \mu_{k(j)} = E \times Ln\left(\dfrac{k(j)}{N} + 1\right) & k(j) \le \dfrac{N}{3} \\ \mu_{k(j)} = E \times exp\left(\dfrac{k(j)}{N} - 1\right) & k(j) > \dfrac{N}{3} \end{cases} \tag{4}$$

$$\begin{cases} \lambda_{k(j)} = I \times \exp\left(-\dfrac{k(j)}{N}\right) & k(j) \le \dfrac{N}{3} \\ \lambda_{k(j)} = I \times \text{Ln}\left(2 - \dfrac{k(j)}{N}\right) & k(j) > \dfrac{N}{3} \end{cases} \tag{5}$$

where, $k(j)$ represents the rank of the species (based on fitness function) living in the j^{th} habitat, N represents population size, E and I display the highest rates of emigration and immigration, respectively. In a migration model of a standard BBO algorithm, only two habitats are combined, so the habitat produced is not significantly different from the parent's habitats. Another disadvantage

of the migration model of the standard BBO algorithm is that only the space between two habitats is searched (which is a small part of the search space). In this paper, a new multi-habitat migration is proposed as a novel operator to improve the exploitation and exploration of ABC algorithm. Fig. 1 shows the multi-habitat migration and neighborhood search of the MHMABC algorithm. The value of mean square error (MSE) is the same as the fitness function, and the lower the MSE, the better the solution. Fig. 2 shows the flowchart of the MPMABC algorithm. The MPMABC algorithm combines local search (by onlooker and employed bees and migration model) with global search methods (by scouts and onlookers bees and multi-parent migration), and seeks to balance the exploration and exploitation of algorithm.

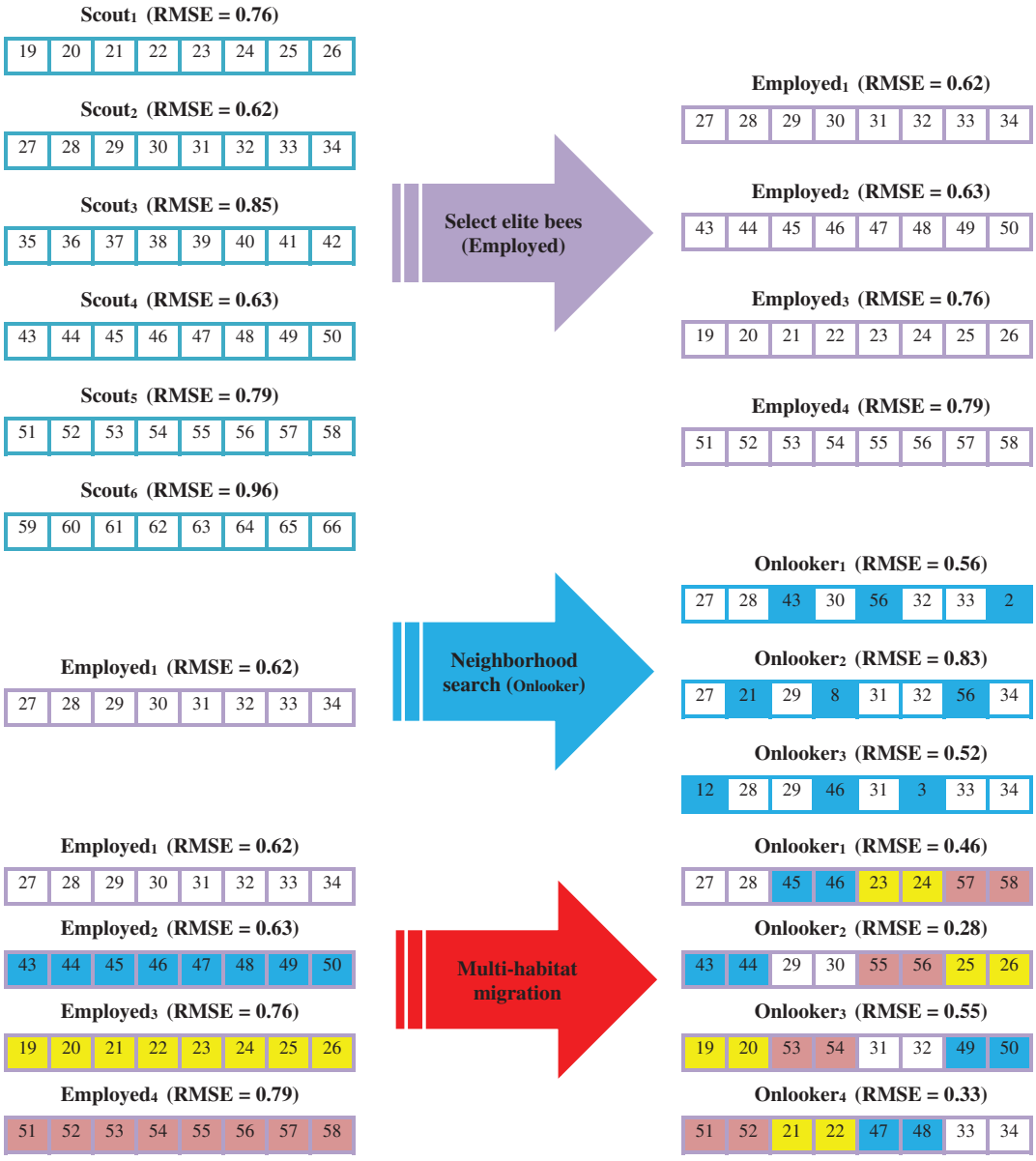


Figure 1: An example of neighborhood search and multi-habitat migration in MHMABC algorithm

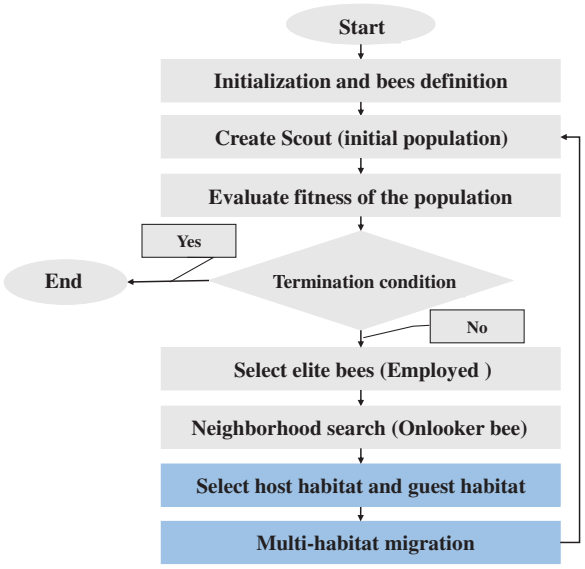


Figure 2: The flowchart of the MHMABC algorithm

3 Training Deep Architecture with MHMABC Algorithm

The main architecture of a DCNN is based on four layers: convolution, pooling, activation function, and fully connected layers. The role of the convolution layer is to put a convolution filter to the input data in order to extract the features by which a set of values will be multiplied with the input data. Next, a kernel passes over an image many times when the weights are multiplied with the input data. The kernel move from top to bottom and right to left, in order to cover all data before a mathematical operation be performed as a next step. The overall inputs will be summed and then generate a distinctive value per every kernel position. Rectified linear unit (ReLu) will act as a nonlinear activation function in the next step where the mapped data produced by the convolution operation will be passed through. ReLu finally replaces the negative values with zeros.

The pooling layer is responsible for reducing the size of the input data and hold the data which are the most worthy. As an example, consider a sample group of four pixels, the max pooling here means that the most valuable pixels which will be held. Pooling layer has an important role in deep learning by decreasing the computational cost balancing the total over fitting. After the convolution and pooling processes are done, the fully connected layer will act which mostly consists of multi-layer perceptron (MLP) neural network. One of the most important contributions of our work is to optimally update the parameters of the fully-connected in the proposed deep architecture using the MHMABC algorithm. To do on that score, we suppose the weights and biases of the MLP neural network as the optimization parameters for the MHMABC algorithm. MHMABC algorithm optimizes the weights and biases of the used DCNN to better classify the Covid-19 X-ray images.

We name the proposed approach MHMABC-DCNN. Fig. 3 shows the definition of a bee in MHMABC algorithm. The fitness function of algorithm (mean square error (MSE)) can be calculated as Eq. (6), where n is the total number of sample, y_i is the system output, and d_i shows the desire value.

$$Fitness\ function\ \ (MSE) = \frac{1}{n}\sum_{i=1}^{n}(y_i - d_i)^2 \tag{6}$$

Bees definition	Weights				Biases			
Scout	W_1	W_2	…	W_n	B_1	B_2	…	B_m

Figure 3: An overview of bees in MHMABC algorithm

4 MHMABC Algorithm Performance Evaluation

In this section, the performance of the proposed MHMABC algorithm on different benchmark functions (21 competitive benchmark functions) is evaluated. To evaluate the performance of MHMABC algorithm, three well-known and two novel algorithms called GA, ABC, BBO, capuchin search algorithm (CapSA) [49], and chimp optimization algorithm (ChOA) [50] are used. Tab. 1 shows the calibration parameters of all algorithms.

Table 1: Calibration parameters of different algorithms

Algorithm	Parameter	Value
MHMABC	Number of onlooker bees	120
	Number of employed bees	60
	The probability range for migrating into for each gene	[0, 1]
	Maximum emigration (I) and immigration (E) coefficient	1
	Number of scout bees (population size)	150
	Iteration	300
CapSA	Velocity control constants	1.00
	Inertia parameter	0.66
	Balance and elasticity factors	0.71, 9
	Population size	150
	Iteration	300
ChOA	A	[−1, 1]
	F	Linearly decreased from 2 to 0
	Population size	150
	Iteration	300
ABC	Number of onlooker bees	120
	Number of employed bees	60
	Number of scout bees (population size)	150
	Iteration	300
GA	Elitism percent	12%
	Mutation rate	0.13
	Crossover rate	0.94
	Population size	150
	Iteration	300

(Continued)

Table 1: Continued

Algorithm	Parameter	Value
BBO	The probability range for migrating into for each gene	[0, 1]
	Maximum emigration (I) and immigration (E) coefficient	1
	Elitism percent	8%
	Mutation rate	0.14
	Population size	150
	Iteration	300

4.1 Competitive Test Functions

In this section, 16 multimodal and 5 unimodal functions are used to evaluate the performance of the proposed algorithm. Tab. 2 shows the details of these benchmark functions (see [51] for more details). Furthermore, Tab. 3 shows the results of algorithms in this benchmark functions. All the algorithms have been implemented 30 times for each problem. Tab. 3 shows best value of the fitness function (Best), the standard deviation of 30 run (StdDev), and the average runtime of the algorithms (AvgTime). As presented in Tab. 3, the results of the MHMABC algorithm are better than the other algorithms. MHMABC in 21 (from 24 function) test functions (87.5%) has achieved the best value of the "Best". CapSA in 12 functions (50%), ChOA in 10 functions (41.66%), BBO in 6 functions (25%), ABC in 5 functions (20.83%), and GA in 3 functions (12.50%) have achieved the best value of "Best".

Table 2: The details of 21 test functions

No.	Name of the benchmark problem	Dimension	Type	Range
F1	Ackley's problem (ACK)	10	Multimodal	[−30, 30]
F2	Aluffi-Pentini's problem (AP)	2	Multimodal	[−10, 10]
F3	Becker and Lago problem (BL)	2	Multimodal	[−10, 10]
F4	Bohachevsky 1 problem (BF1)	2	Multimodal	[−50, 50]
F5	Bohachevsky 2 problem (BF2)	2	Multimodal	[−50, 50]
F6	Camel back-3 three hump problem (CB3)	2	Multimodal	[−5, 5]
F7	Camel back-6 six hump problem (CB6)	2	Multimodal	[−5, 5]
F8	Cosine mixture problem (CM)	2, 4	Unimodal	[−20, 20]
F9	Dekkers and aarts problem (DA)	2	Multimodal	[−20, 20]
F10	Exponential problem (EXP)	10	Unimodal	[−1, 1]
F11	Hosaki problem (HSK)	2	Multimodal	$0 \leq x1 \leq 5, 0 \leq x2 \leq 6$
F12	Levy and montalvo 2 problem (LM2)	5, 10	Multimodal	[−10, 10]
F13	Miele and cantrell problem (MCP)	4	Multimodal	[−1, 1]
F14	Modified rosenbrock problem (MRP)	2	Unimodal	[−5, 5]
F15	Multi-gaussian problem (MGP)	2	Multimodal	[−2, 2]
F16	Powell's quadratic problem (PWQ)	4	Multimodal	[−10, 10]
F17	Rastrigin problem (RG)	10	Multimodal	[−5.12, 5.12]

(Continued)

Table 2: Continued

No.	Name of the benchmark problem	Dimension	Type	Range
F18	Rosenbrock problem (RB)	10	Unimodal	[−30, 30]
F19	Salomon problem (SAL)	5, 10	Multimodal	[−100, 100]
F20	Schaffer 1 problem (SF1)	2	Multimodal	[−100, 100]
F21	Schwefel problem (SWF)	10	Unimodal	[−500, 500]

Table 3: The results of algorithms on test functions

Function	F (x∗)	Criteria	Meta-heuristics					
			GA	ABC	BBO	ChOA	CapSA	MHMABC
F1_(ACK)	0	Best	2.36E−3	2.36E−4	1.33E−5	3.18E−6	1.26E−7	0
		SD	0.0156	4.12E−2	7.09E−3	2.09E−3	3.90E−5	0
		AvgTime	63	49	52	56	47	35
F2_(AP)	= −0.35239	Best	0.35238	0.35238	0.35238	0.35239	0.35238	−0.35239
		SD	2.63E−4	2.23E−5	2.13E−5	1.35E−10	6.03E−9	0
		AvgTime	59	52	48	49	42	32
F3_(BL)	0	Best	1.70E−4	1.53E−9	4.75E−8	3.76E−10	0	0
		SD	2.01E−2	3.96E−6	4.09E−5	6.09E−8	0	0
		AvgTime	46	40	43	35	39	29
F4_(BF1)	0	Best	7.71E−7	1.98E−6	1.75E−5	3.16E−8	0	0
		SD	2.29E−5	3.23E−5	4.19E−4	1.98E−6	0	0
		AvgTime	56	43	49	42	39	23
F5_(BF2)	0	Best	2.38E−3	0	3.83E−8	0	2.79E−9	0
		SD	2.16E−2	0	5.35E−2	0	3.02E−8	0
		AvgTime	52	46	52	41	36	20
F6_(CB3)	0	Best	2.38E−5	1.86E−6	0	4.19E−10	0	0
		SD	6.29E−4	2.29E−4	1.86E−9	1.06E−7	0	0
		AvgTime	61	46	56	44	39	28
F7_(CB6)	= −1.03163	Best	−1.03162	−1.03162	−1.03162	−1.03163	−1.03162	−1.03163
		SD	1.09E−5	4.26E−4	6.88E−4	0	6.28E−07	3.27E−08
		AvgTime	59	41	46	39	42	22
F8_(CM)	0.2	Best	0.2	1.9999E−1	1.9998E−1	0.2	0.2	0.2
		SD	0	2.50E−7	2.16E−5	1.85E−5	7.19E−9	0
		AvgTime	40	39	36	33	29	18
	0.4	Best	3.9990E−1	3.9991E−1	3.9996E−1	0.4	3.9999E−1	0.4
		SD	2.19E−2	4.13E−3	5.09E−5	2.39E−8	8.09E−13	5.39E−12
		AvgTime	61	53	40	45	41	26
F9_(DA)	−24777	Best	−24776.510	−24776.510	−24776.510	−24776.510	−24776.520	−24776.510
		SD	1.08E−3	2.19E−5	1.09E−7	1.39E−8	2.96E−11	2.13E−12
		AvgTime	55	41	46	42	36	29
F10_(EXP)	1	Best	9.9979E−1	9.9996E−1	9.9995E−1	9.9996E−1	9.9982E−1	1
		SD	2.09E−2	8.09E−8	3.02E−7	4.36E−8	9.12E−5	0
		AvgTime	73	58	61	52	49	35
F11_(HSK)	−2.34580	Best	−2.34581	−2.34581	−2.34581	−2.34581	−2.34581	−2.34581

(Continued)

Table 3: Continued

Function	F (x∗)	Criteria	Meta-heuristics					
			GA	ABC	BBO	ChOA	CapSA	MHMABC
		SD	6.22E−4	1.38E−5	2.10E−4	2.96E−7	4.39E−6	2.59E−9
		AvgTime	49	40	44	35	39	23
F12_(LM2)	0	Best	1.13E−6	1.92E−7	0	0	5.26E−11	0
		SD	6.52E−4	4.96E−6	2.29E−10	0	2.26E−9	0
		AvgTime	73	69	62	52	59	36
	0	Best	1.73E−4	1.06E−8	1.86E−4	1.16E−7	3.73E−5	1.02E−6
		SD	2.03E−2	3.76E−6	2.09E−1	3.86E−6	6.06E−2	3.06E−2
		AvgTime	103	90	86	76	73	62
F13_(MCP)	0	Best	0	6.09E−8	2.29E−5	2.03E−8	0	0
		SD	0	4.73E−4	3.76E−4	1.76E−6	0	0
		AvgTime	45	48	42	35	39	26
F14_(MRP)	0	Best	2.80E−3	2.69E−7	0	4.19E−10	0	0
		SD	3.43E−2	1.13E−3	0	3.07E−9	0	0
		AvgTime	52	45	49	44	31	22
F15_(MGP)	1.29695	Best	1.29586	1.29694	1.29559	1.29695	1.29695	1.29695
		SD	6.71E−2	2.35E−4	1.26E−3	2.16E−6	3.76E−6	1.26E−9
		AvgTime	53	40	42	38	36	20
F16_(PWQ)	0	Best	3.19E−5	0	2.43E−6	2.76E−11	0	0
		SD	1.71E−2	0	1.19E−4	3.70E−7	0	0
		AvgTime	71	68	62	55	53	35
F17_(RG)	0	Best	0.85696	1.19E−2	2.79E−3	6.76E−8	0	1.86E−6
		SD	2.88E−1	3.76E−1	2.36E−2	2.36E−6	0	2.36E−4
		AvgTime	113	108	103	80	86	72
F18_(RB)	0	Best	4.0263	3.2185	6.4125	2.09E−3	2.43E−5	4.81E−8
		SD	6.2563	5.8463	4.8526	5.71E−1	1.82E−3	3.18E−5
		AvgTime	99	101	96	72	63	55
F19_(SAL)	0	Best	7.26E−9	0	0	2.75E−11	0	0
		SD	4.26E−7	0	0	1.25E−10	0	0
		AvgTime	53	43	48	39	41	29
	0	Best	5.19E−4	6.16E−6	1.86E−5	0	2.09E−11	0
		SD	4.06E−2	1.85E−3	2.26E−3	0	1.96E−10	0
		AvgTime	92	89	85	65	69	52
F20_(SF1)	0	Best	1.76E−7	8.16E−8	0	1.76E−8	1.12E−9	0
		SD	6.72E−6	1.69E−4	0	2.18E−5	2.39E−5	0
		AvgTime	46	38	40	31	26	18
F21_(SWF)	−4189.829	Best	−4136.318	−4165.186	−4163.269	−4189.680	−4185.841	−4189.580
		SD	2.1251	1.8412	0.9521	2.19E−6	5.06E−1	1.92E−4
		AvgTime	95	89	85	76	75	63

Also, the MHMABC algorithm is better than others in terms of AvgTime (Fig. 4) and SD. As shown in Fig. 4, the total "Average Time" of the MHMABC is less than other algorithms. The standard deviation (SD) values presented in Tab. 3 indicate that MHMABC has the highest rank of repeatability compared to other algorithms. According to the results, in most of the benchmark problems, the

MHMABC produce results with the least amount of SD. Thus, the MHMABC is a very stable search algorithm.

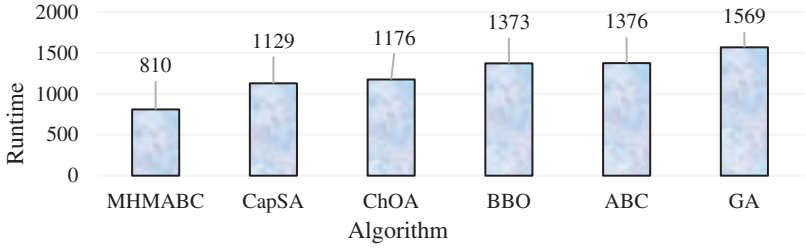


Figure 4: Total AvgTime of algorithms

4.2 Pressure Vessel System Benchmark

In this section, the performance of the proposed MHMABC and other algorithms on the pressure vessel system benchmark are investigated, which can be formulated as Eqs. (7)–(12).

$$f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{7}$$

Subjected to

$$g_1(X) = -x_1 + 0.0193x_3 \leq 0 \tag{8}$$

$$g_2(X) = -x_2 + 0.00954x_3 \leq 0 \tag{9}$$

$$g_3(X) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 129600 \leq 0 \tag{10}$$

$$g_4(X) = x_4 - 240 \leq 0 \tag{11}$$

$$0 \leq x_1,\ \ x_2 \leq 100,\ 10 \leq x_3,\ \ x_4 \leq 200 \tag{12}$$

where, x_1 = the shell thickness (T_s), x_2 = The head thickness (T_h), x_3 = The radius of the cylindrical shell (R), and x_4 = The length of the shell (L). Tab. 4 indicates the results of the different algorithms in pressure vessel system. The best value of the cost function obtained by the MHMABC algorithm is 5918.2586 $, which is better than the other algorithms. In addition, MHMABC performs better than other algorithms in the mean fitness and standard deviation. MHMABC is close to its highest accuracy in iterations = 50, While other algorithms do not have good accuracy. With increasing iteration, MHMABC has achieved high stability and high convergence speed. As shown in Tab. 4, the total "Average Time" of the MHMABC is less than other algorithms for the vessel system benchmark problem.

Table 4: The results of the algorithms for pressure vessel system

Algorithm	Best fitness ($)	Mean fitness ($)	Standard deviation	Iteration	AvgTime (s)
MPCGA	5918.2586	5942.5478	0.0008452	300	152
CapSA	6085.2365	6195.7892	0.0475632	300	186
ChOA	6098.1795	6309.4789	0.8563271	300	215
BBO	6165.1785	6821.1478	5.2563981	300	228

(Continued)

Table 4: Continued

Algorithm	Best fitness ($)	Mean fitness ($)	Standard deviation	Iteration	AvgTime (s)
ABC	6196.2869	7009.1453	9.2145896	300	274
GA	6345.2456	7179.9625	25.148563	300	293

5 Simulation Results for Covid-19 Detection

In this part, the performance of proposed evolutionary deep architecture called MHMABC-DCNN and the other are evaluated for COVID-19 X-ray images classification. For this comparison, the accuracy, sensitivity, and specificity analysis are used. These three criteria are obtained from the confusion matrix and can be calculated as Eqs. (13)–(15) [52].

$$Sensitivity = \frac{TP}{TP + FN} \tag{13}$$

$$Specificity = \frac{TN}{TN + FP} \tag{14}$$

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{15}$$

where TP = True positive, FN = False negative, TN = True negative, and P = False positive. In order to evaluate the performance of MHMABC-DCNN, we use the advantage of COVID X-ray-5 k [46] dataset, which consists of 2084 and 3100 training and test images, respectively. According to the radiologist advices, we only use the anterior–posterior Covid-19 X-ray images in this dataset since the lateral images have not been appreciate for detection goals. Furthermore, we have ignored using images which lacked minimum range of signs visibility of COVID-19. Therefore, we have only consider 184 images out of the 203 images, and removed 19 images whose signs of COVID-19 were not clear enough. Next, among the 184 remained images, we have choose 84 and 100 photos for the training and test sets, respectively. In order to increase the number of COVID-19 samples to 420, we have used the augmentation techniques. We have also used the benefit of ChexPert dataset [53] (as a supplementary dataset) since the number of non-COVID photos was not enough in the Covid-chestxray-dataset. The ChexPert dataset consists of 224,316 chest X-ray images obtained from 65,240 patients. Finally, we have considered 2000 non-COVID images for the training set and 3000 non-COVID images for the training set and test set. Fig. 5 depicts six stochastic sample images from the COVID-Xray-5k dataset, including two COVID-19 and four standard samples.

Tab. 5 shows the accuracy, specificity, and sensitivity of different deep learning architectures for COVID-19 X-ray images classification. According to Tab. 5, MHMABC-DCNN shows the best results in training and validation datasets. MHMABC-DCNN achieved 99.32% and 98.32% of accuracy in the test and train datasets, respectively. MHMABC-DCNN also achieved 99.63% and 98.82% of sensitivity in the test and train datasets, respectively. The results of evolutionary deep learning architectures in the test dataset show that deep learning architectures are well trained using meta-heuristic algorithms. Because accuracy, specificity, and sensitivity of different deep learning architectures in the test and train datasets are highly stable. As shown in Tab. 5, the "Average Time" of the MHMABC-DCNN is less than other architectures for COVID-19 X-ray images classification. Fig. 6 shows the radar plot of the architectures in both dataset. As can be seen, the accuracy, sensitivity, and specificity of the MHMABC-DCNN is better than other architectures.

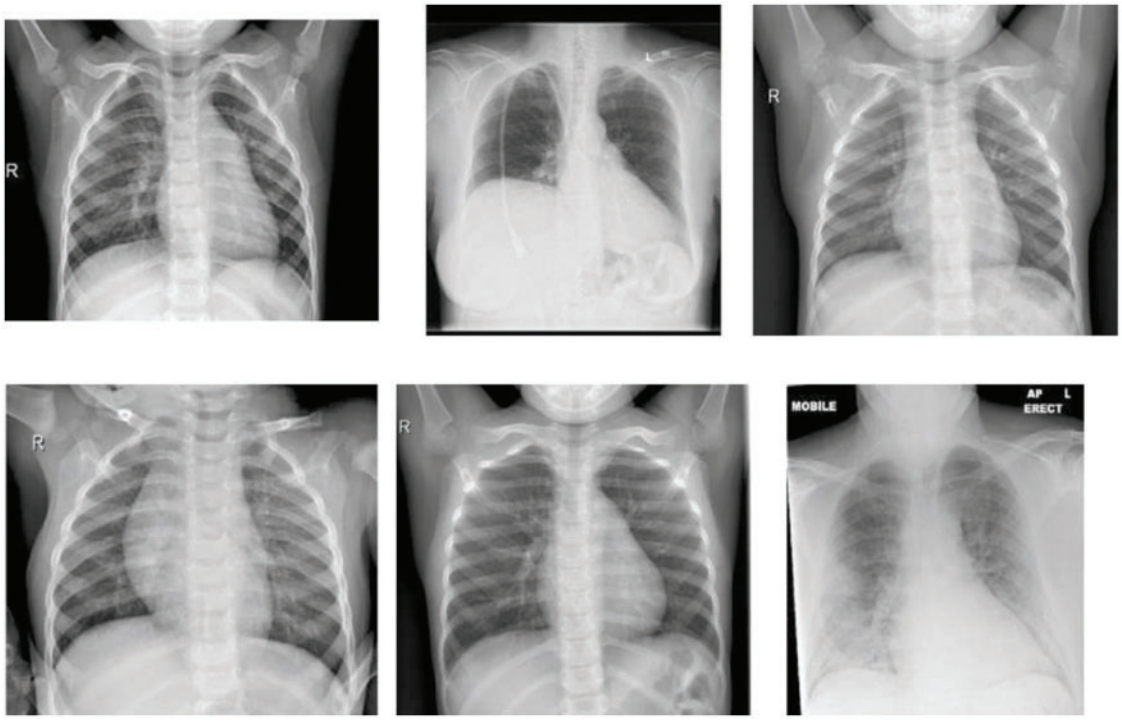
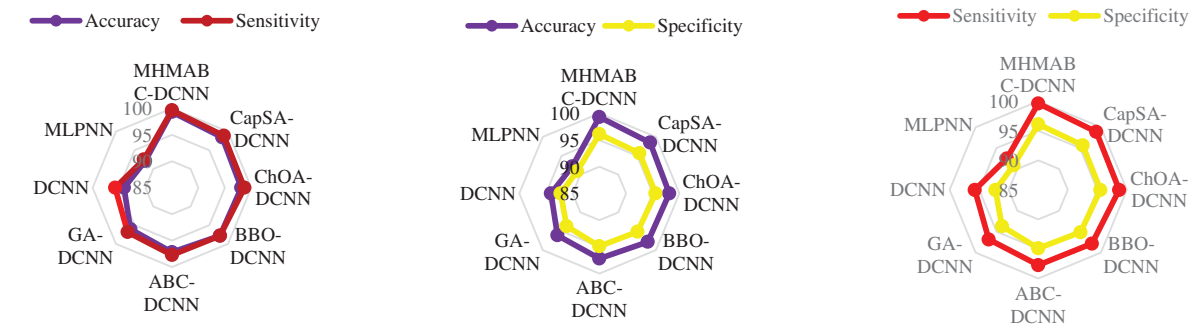


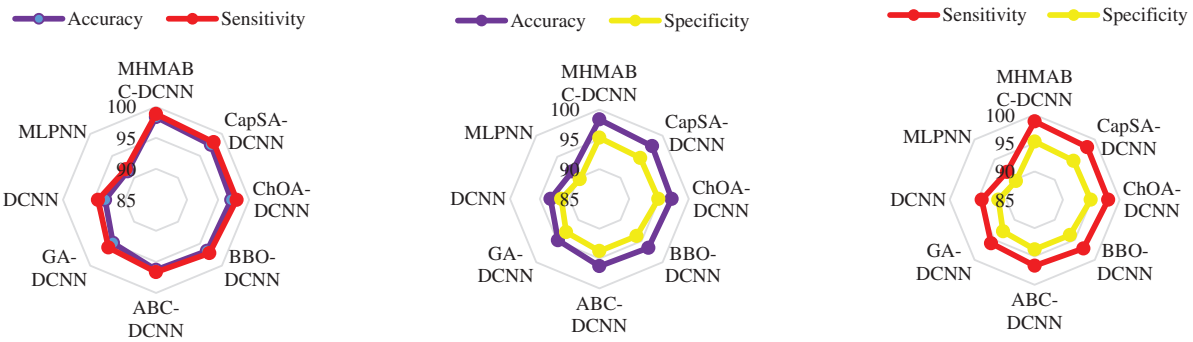
Figure 5: Six stochastic sample images from the COVID-X-ray-5 k dataset

Table 5: The results of architectures for COVID-19 X-ray images classification

Regression model	Train			Test			Run time (s)
	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity	Accuracy	
MHMABC-DCNN	0.9963	0.9612	0.9932	0.9882	0.9526	0.9832	628
CapSA-DCNN	0.9889	0.9566	0.9846	0.9812	0.9471	0.9751	692
ChOA-DCNN	0.9871	0.9553	0.9811	0.9798	0.9489	0.9712	719
BBO-DCNN	0.9786	0.9512	0.9783	0.9716	0.9381	0.9662	736
ABC-DCNN	0.9773	0.9489	0.9719	0.9665	0.9375	0.9629	729
GA-DCNN	0.9686	0.9368	0.9610	0.9586	0.9289	0.9482	768
DCNN	0.9574	0.9225	0.9406	0.9435	0.9140	0.9321	896
MLPNN	0.9253	0.9089	0.9216	0.9193	0.8966	0.9149	915



(a) Training dataset



(b) Validation dataset

Figure 6: The radar plot of the machine learning approaches

Tab. 6 also demonstrates the comparison of the different deep architectures in MSE criteria. It can be seen that the proposed deep MHMABC-DCNN architecture has a fewer MSE than the other approaches. Therefore, the proposed MHMABC-DCNN has been useful to solve this problem. The rank of the algorithms in this comparison are: MHMABC-DCNN, CapSA-DCNN, ChOA-DCNN, BBO-DCNN, ABC-DCNN, GA-DCNN, DCNN, and MLPNN, respectively. To have a more comprehensive comparison, the convergence curve of the architectures has been shown in Fig. 7. As shown in this figure, MHMABC-DCNN, CapCA-DCNN, and ChOA-DCNN architectures converge faster than others. All architectures have been trained by meta-heuristic algorithms show higher convergence rates.

Table 6: Comparison of the architecture in MSE values

Architecture	MSE	
	Training dataset	Validation dataset
MHMABC-DCNN	0.00012	0.00249
CapSA-DCNN	0.00656	0.04586
ChOA-DCNN	0.01143	0.08961
BBO-DCNN	0.09419	0.12589
ABC-DCNN	0.12985	0.40189
GA-DCNN	0.24712	0.41526

(Continued)

Table 6: Continued

Architecture	MSE	
	Training dataset	Validation dataset
DCNN	0.49521	0.63259
MLPNN	0.87652	0.98742

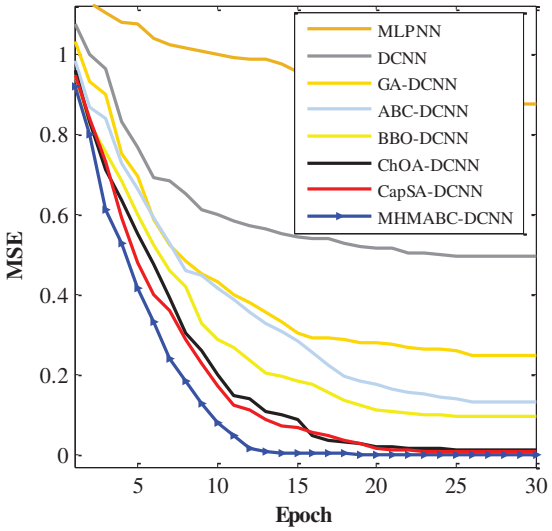


Figure 7: The convergence curve of the architectures

6 Conclusion

This paper proposed a real-time Covid-19 detector based on a novel deep learning architecture. In order to overcome the drawbacks of the PCR tests, we took the advantage of using chest X-ray images to better diagnose positive Covid-19 cases. After that, we proposed a novel algorithm named MHMABC to improve the exploitation and exploration of the ABC algorithm. We then optimally train the fully connected in our deep architecture by using the proposed MHMABC algorithm. The experiment results on Covid-19 X-ray image dataset showed that the proposed scheme (MHMABC-DCNN) could obtain better accuracy and convergence rate while reducing the execution cost. Furthermore, the performance evaluating on some well-known benchmark datasets showed that the proposed MHMABC algorithm outperformed the state-of-the-art algorithms.

Funding Statement: This work has been supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) under the High-Potential Individuals Global Training Program under Grant 2021-0-01532 (50%), and in part by the National Research Foundation of Korea (NRF) under Grant 2020R1A2B5B01002145 (50%), all funded by the Korean Government through Ministry of Science and ICT (MSIT).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] X. Li, Z. Q. Dong, P. Yu, L. P. Wang, X. D. Niu *et al.,* "Effect of self-assembly on fluorescence in magnetic multiphase flows and its application on the novel detection for COVID-19," *Physics of Fluids*, vol. 33, no. 4, pp. 042004, 2021.

[2] M. Abdel-Basset, R. Mohamed, M. Elhoseny, R. K. Chakrabortty and M. Ryan, "A hybrid COVID-19 detection model using an improved marine predators algorithm and a ranking-based diversity reduction strategy," *IEEE Access*, vol. 8, pp. 79521–79540, 2020.

[3] W. Wang, H. Liu, J. Li, H. Nie and X. Wang, "Using CFW-net deep learning models for X-ray images to detect COVID-19 patients," *International Journal of Computational Intelligence Systems*, vol. 14, no. 1, pp. 199–207, 2021.

[4] D. Zhang, J. Hu, F. Li, X. Ding, A. K. Sangaiah *et al.,* "Small object detection via precise region-based fully convolutional networks," *Computers, Materials and Continua*, vol. 69, no. 2, pp. 1503–1517, 2021.

[5] S. Wacharapluesadee, T. Kaewpom, W. Ampoot, S. Ghai, W. Khamhang *et al.,* "Evaluating the efficiency of specimen pooling for PCR-based detection of COVID-19," *Journal of Medical Virology*, vol. 92, no. 10, pp. 2193–2199, 2020.

[6] A. M. Karim, H. Kaya, V. Alcan, B. Sen and I. A. Hadimlioglu, "New optimized deep learning application for COVID-19 detection in chest X-ray images," *Symmetry*, vol. 4, no. 5, pp. 1003, 2022.

[7] M. Khishe, F. Caraffini and S. Kuhn, "Evolving deep learning convolutional neural networks for early COVID-19 detection in chest X-ray images," *Mathematics*, vol. 9, no. 9, pp. 1002, 2021.

[8] J. Wang, Y. Wu, S. He, P. K. Sharma, X. Yu *et al.,* "Lightweight single image super-resolution convolution neural network in portable device," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 15, no. 11, pp. 4065–4083, 2021.

[9] S. M. J. Jalali, M. Ahmadian, S. Ahmadian, R. Hedjam, A. Khosravi *et al.,* "X-ray image based COVID-19 detection using evolutionary deep learning approach," *Expert Systems with Applications*, vol. 201, pp. 116942, 2022.

[10] M. Kaveh, M. Khishe and M. R. Mosavi, "Design and implementation of a neighborhood search biogeography-based optimization trainer for classifying sonar dataset using multi-layer perceptron neural network," *Analog Integrated Circuits and Signal Processing*, vol. 100, no. 2, pp. 405–428, 2019.

[11] F. Najafi, M. Kaveh, D. Martin and M. R. Mosavi, "Deep PUF: A highly reliable DRAM PUF-based authentication for iot networks using deep convolutional neural networks," *Sensors*, vol. 21, no. 6, pp. 2009, 2021.

[12] J. Wang, Y. Zou, P. Lei, R. S. Sherratt and L. Wang, "Research on recurrent neural network based crack opening prediction of concrete dam," *Journal of Internet Technology*, vol. 21, no. 4, pp. 1161–1169, 2020.

[13] M. Kaveh and M. S. Mesgari, "Improved biogeography-based optimization using migration process adjustment: An approach for location-allocation of ambulances," *Computers & Industrial Engineering*, vol. 135, pp. 800–813, 2019.

[14] A. Lotfy, M. Kaveh, M. R. Mosavi and A. R. Rahmati, "An enhanced fuzzy controller based on improved genetic algorithm for speed control of DC motors," *Analog Integrated Circuits and Signal Processing*, vol. 105, no. 2, pp. 141–155, 2020.

[15] M. Khishe, M. R. Mosavi and M. Kaveh, "Improved migration models of biogeography-based optimization for sonar dataset classification by using neural network," *Applied Acoustics*, vol. 118, pp. 15–29, 2017.

[16] S. He, Z. Li, Y. Tang, Z. Liao, F. Li *et al.,* "Parameters compressing in deep learning," *Computers Materials & Continua*, vol. 62, no. 1, pp. 321–336, 2020.

[17] M. Kaveh, M. Kaveh, M. S. Mesgari and R. S. Paland, "Multiple criteria decision-making for hospital location-allocation based on improved genetic algorithm," *Applied Geomatics*, vol. 12, no. 3, pp. 291–306, 2020.

[18] N. Kianfar, M. S. Mesgari, A. Mollalo and M. Kaveh, "Spatio-temporal modeling of COVID-19 prevalence and mortality using artificial neural network algorithms," *Spatial and Spatio-Temporal Epidemiology*, vol. 40, pp. 100471, 2022.

[19] O. Rostami and M. Kaveh, "Optimal feature selection for SAR image classification using biogeography-based optimization (BBO), artificial bee colony (ABC) and support vector machine (SVM): A combined approach of optimization and machine learning," *Computational Geosciences*, vol. 25, no. 3, pp. 911–930, 2021.
[20] J. Wang, M. Khishe, M. Kaveh and H. Mohammadi, "Binary chimp optimization algorithm (BChOA): A new binary meta-heuristic for solving optimization problems," *Cognitive Computation*, vol. 13, no. 5, pp. 1297–1316, 2021.
[21] S. R. Zhou and B. Tan, "Electrocardiogram soft computing using hybrid deep learning CNN-ELM," *Applied Soft Computing*, vol. 86, pp. 105778, 2020.
[22] M. Ali, L. T. Jung, A. H. Abdel-Aty, M. Y., Abubakar, M. Elhoseny *et al.,* "Semantic-k-NN algorithm: An enhanced version of traditional k-NN algorithm," *Expert Systems with Applications*, vol. 151, pp. 113374, 2020.
[23] L. Zhu, L. Kong and C. Zhang, "Numerical study on hysteretic behaviour of horizontal-connection and energy-dissipation structures developed for prefabricated shear walls," *Applied Sciences*, vol. 10, no. 4, pp. 1240, 2020.
[24] X. R. Zhang, X. Sun, W. Sun, T. Xu and P. P. Wang, "Deformation expression of soft tissue based on BP neural network," *Intelligent Automation & Soft Computing*, vol. 32, no. 2, pp. 1041–1053, 2022.
[25] Y. Wei, M. M. Zhao, M. Hong, M. J. Zhao and M. Lei, "Learned conjugate gradient descent network for massive MIMO detection," *IEEE Transactions on Signal Processing*, vol. 68, pp. 6336–6349, 2020.
[26] X. R. Zhang, W. F. Zhang, W. Sun, X. M. Sun and S. K. Jha, "A robust 3-D medical watermarking based on wavelet transform for data protection," *Computer Systems Science & Engineering*, vol. 41, no. 3, pp. 1043–1056, 2022.
[27] C. Zhang, M. Abedini and J. Mehrmashhadi, "Development of pressure-impulse models and residual capacity assessment of RC columns using high fidelity arbitrary lagrangian-eulerian simulation," *Engineering Structures*, vol. 224, pp. 111219, 2020.
[28] M. Jiang, B. Gallagher, J. Kallman and D. Laney, "A supervised learning framework for arbitrary lagrangian-eulerian simulations," in *Proc. of the 2016 IEEE 15th Int. Conf. on Machine Learning and Applications (ICMLA)*, Anaheim, CA, USA, pp. 977–982, 2016.
[29] W. Wang, Y. Jiang, Y. Luo, J. Li, X. Wang *et al.,* "An advanced deep residual dense network (DRDN) approach for image super-resolution," *International Journal of Computational Intelligence Systems*, vol. 12, no. 2, pp. 1592–1601, 2019.
[30] J. Martens, "Deep learning via hessian-free optimization," in *Proc. of the 27th ICML'10 Int. Conf. on Machine Learning*, Haifa, Israel, vol. 27, pp. 735–742, 2010.
[31] O. Vinyals and D. Povey, "Krylov subspace descent for deep learning," *Artificial Intelligence and Statistics*, vol. 22, pp. 1261–1268, 2012.
[32] B. Cao, J. Zhao, P. Yang, P. Yang, X. Liu *et al.,* "Multiobjective feature selection for microarray data via distributed parallel algorithms," *Future Generation Computer Systems*, vol. 100, pp. 952–981, 2019.
[33] A. Lotfy, M. Kaveh, M. R. Mosavi and A. R. Rahmati, "An enhanced FPGA-based implementation of fuzzy controller using a personalized microcontroller," in *Proc. of the 2019 IEEE 34th Int. Conf. on Power System (PSC)*, Tehran, Iran, pp. 1–4, 2019.
[34] M. Kaveh, D. Martín and M. R. Mosavi, "A lightweight authentication scheme for V2G communications: A PUF-based approach ensuring cyber/physical security and identity/location privacy," *Electronics*, vol. 9, no. 9, pp. 1479, 2020.
[35] A. Lotfy, M. Kaveh, D. Martín and M. R. Mosavi, "An efficient design of anderson PUF by utilization of the xilinx primitives in the SLICEM," *IEEE Access*, vol. 9, pp. 23025–23034, 2021.
[36] W. Wang, Y. Yang, J. Li, Y. Hu, Y. Luo *et al.,* "Woodland labeling in Chenzhou, China, via deep learning approach," *International Journal of Computational Intelligence Systems*, vol. 13, no. 1, pp. 1393–1403, 2020.
[37] G. Rosa, J. Papa, A. Marana, W. Scheirer and D. Cox, "Fine-tuning convolutional neural networks using harmony search," in *Proc. of the 20th Iberoamerican Int. Conf. on Pattern Recognition*, Montevideo, Uruguay, pp. 683–690, 2015.

[38] H. Fan, L. Zheng, C. Yan and Y. Yang, "Unsupervised person re-identification: Clustering and fine-tuning," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 14, no. 4, pp. 1–18, 2018.
[39] Y. Sun, B. Xue, M. Zhang, G. G. Yen and J. Lv, "Automatically designing CNN architectures using the genetic algorithm for image classification," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3840–3854, 2020.
[40] A. A. Ewees, M. Abd El Aziz and M. Elhoseny, "Social-spider optimization algorithm for improving ANFIS to predict biochar yield," in *Proc. of the 2017 IEEE 8th Int. Conf. on Computing, Communication and Networking Technologies (ICCCNT)*, Delhi, India, pp. 1–6, 2017.
[41] K. Shankar, M. Elhoseny, S. Lakshmanaprabu, M. Ilayaraja, R. M. Vidhyavathi *et al.,* "Optimal feature level fusion based ANFIS classifier for brain MRI image classification," *Concurrency and Computation-Practice & Experience*, vol. 32, no. 1, pp. 1–12, 2020.
[42] R. M. Rizk-Allah, A. E. Hassanien and M. Elhoseny, "A multi-objective transportation model under neutrosophic environment," *Computers & Electrical Engineering*, vol. 69, pp. 705–719, 2018.
[43] I. M. El-Hasnony, S. I. Barakat, M. Elhoseny and R. R., Mostafa, "Improved feature selection model for big data analytics," *IEEE Access*, vol. 8, pp. 66989–67004, 2020.
[44] M. Elhoseny, "Intelligent firefly-based algorithm with levy distribution (FF-L) for multicast routing in vehicular communications," *Expert Systems with Applications*, vol. 140, pp. 112889, 2020.
[45] X. Wang, C. Gong, M. Khishe, M. Mohammadi and T. A. Rashid, "Pulmonary diffuse airspace opacities diagnosis from chest X-ray images using deep convolutional neural networks fine-tuned by whale optimizer," *Wireless Personal Communications*, vol. 124, no. 2, pp. 1–20, 2021.
[46] M. Turkoglu, "COVIDetectioNet: COVID-19 diagnosis system based on X-ray images using features selected from pre-learned deep features ensemble," *Applied Intelligence*, vol. 51, no. 3, pp. 1213–1226, 2021.
[47] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," *Technical Report-tr06*, vol. 200, pp. 1–10, 2005.
[48] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.
[49] M. Braik, A. Sheta and H. Al-Hiary, "A novel meta-heuristic search algorithm for solving optimization problems: Capuchin search algorithm," *Neural Computing and Applications*, vol. 33, no. 7, pp. 2515–2547, 2021.
[50] M. Khishe and M. R. Mosavi, "Chimp optimization algorithm," *Expert Systems with Applications*, vol. 149, pp. 113338, 2020.
[51] M. M., Ali, C. Khompatraporn and Z. B. Zabinsky, "A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems," *Journal of Global Optimization*, vol. 31, no. 4, pp. 635–672, 2005.
[52] W. Kaidi, M. Khishe and M. Mohammadi, "Dynamic levy flight chimp optimization," *Knowledge-Based Systems*, vol. 235, pp. 107625, 2022.
[53] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus *et al.,* "Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison," in *Proc. of the 33th AAAI Int. Conf. on Artificial Intelligence*, Honolulu, Hawaii, USA, vol. 33, no. 1, pp. 590–597, 2019.