Medical & Biological Engineering & Computing (2022) 60:2931–2949
https://doi.org/10.1007/s11517-022-02637-6

ORIGINAL ARTICLE

COVID-19 diagnosis using chest CT scans and deep convolutional neural networks evolved by IP-based sine-cosine algorithm

Binfeng Xu[1] · Diego Martín[2] · Mohammad Khishe[3] · Reza Boostani[4]

Received: 16 November 2021 / Accepted: 15 June 2022 / Published online: 12 August 2022

Abstract
The prevalence of the COVID-19 virus and its variants has influenced all aspects of our life, and therefore, the precise diagnosis of this disease is vital. If a polymerase chain reaction test for a subject is negative, but he/she cannot easily breathe, taking a computed tomography (CT) image from his/her lung is urgently recommended. This study aims to optimize a deep convolution neural network (DCNN) structure to increase the COVID-19 diagnosis accuracy in lung CT images. This paper employs the sine-cosine algorithm (SCA) to optimize the structure of DCNN to take raw CT images and determine their status. Three improvements based on regular SCA are proposed to enhance both the accuracy and speed of the results. First, a new encoding approach is proposed based on the internet protocol (IP) address. Then, an enfeebled layer is proposed to generate a variable-length DCNN. The suggested model is examined over the COVID-CT and SARS-CoV-2 datasets. The proposed method is compared to a standard DCNN and seven variable-length models in terms of five known metrics, including sensitivity, accuracy, specificity, F1-score, precision, and receiver operative curve (ROC) and precision-recall curves. The results demonstrate that the proposed DCNN-IPSCA surpasses other benchmarks, achieving final accuracy of (98.32% and 98.01%), the sensitivity of (97.22% and 96.23%), and specificity of (96.77% and 96.44%) on the SARS-CoV-2 and COVID-CT datasets, respectively. Also, the proposed DCNN-IPSCA performs much better than the standard DCNN, with GPU and CPU training times, which are 387.69 and 63.10 times faster, respectively.

Keywords DCNNs · Chest CT scans · COVID-19 · Internet protocol address · Sine-cosine algorithm

1 Introduction

Due to the worldwide spread of COVID-19 and its variants in all countries, fast and precise diagnosis of COVID-19 is necessary. The first step in the COVID-19 diagnosis is to take the reverse transcription-polymerase chain reaction (RT-PCR) [1]; however, in a few cases, the RT-PCR test of a subject is negative but he/she has a problem in breathing [2]. As the second line diagnosis, in this state, taking a chest computed tomography (CT) image is recommended by specialists [3, 4]. If COVID-19 highly infects the lungs, its diagnosis can be easily made by visual inspection. Nonetheless, when the disease is in the early stage, visual diagnosis of this disease carries a degree of uncertainty. COVID-19 lung affection has recently been diagnosed using X-rays and CT scans [5, 6]. However, the accuracy of the diagnosis depends on the expert's judgment [7]. Since differences in pixels' value can be better recognized by image processing techniques, as a qualitative method, this paper is aimed to automatically and more precisely diagnose the infected CT images [8]. Deep learning (DL) methods can automatically identify this condition to solve this weakness [9].

Bones, blood arteries, and other interior organs can be seen on CT scans. Thus, they allow clinicians to see interior organ size and shape. Unlike X-rays, CT scans can cut a particular body slice without affecting adjacent slices, and they can also reveal detailed images of the patient's body in a high spatial resolution [10]. This information can be analyzed to

✉ Binfeng Xu
xbf923@163.com

✉ Mohammad Khishe
m_khishe@alumni.iust.ac.ir

1 Guangdong Food and Drug Vocational College, Guangzhou 510520, Guangdong, China

2 ETSI Telecomunicación, Universidad Politécnica de Madrid, Av. Complutense 30, 28040 Madrid, Spain

3 Department of Electrical Engineering, Imam Khomeini Marine Science University, Nowshahr, Iran

4 CSE & IT Department, Electrical and Computer Engineering Faculty, Shiraz University, Shiraz, Iran

give spatial information in millimeter resolution. Therefore, deep learning approaches have efficiently increased the usability of chest CT scans for COVID-19 diagnosis [10–12]. For example, Grewal et al. [9] employed DenseNet and RNNs to analyze the brain CT images. This hybrid combination, as well as the deficiencies of the RNN, including exploding or gradient vanishing and a very long processing time, make the proposed model unreliable and slow. Song et al. [10] deployed three deep neural models (SAE, DNN, and CNN) for lung cancer detection [13]. They found that the deep convolution neural network (DCNN) architecture outperforms the other models in terms of accuracy. In this study, SAE did not provide acceptable results because the auto-encoders did not perform well in error reconstruction for CT scan images. Gonzales et al. [11] used deep learning to study the neural system for lung disease identification and acute prediction of the respiratory problem. Although the proposed model's complexity is acceptable, the accuracy is not good enough for COVID-19 diagnosis.

It was found that CT scan–based processing approaches effectively diagnose COVID-19 disease, as the second line diagnosis tool, during the pandemic. For example, Zhao et al. [12] evaluated the relationship between chest CT scan results and pneumonia in COVID-19 participants. Since collecting plenty of COVID-19 CT scan images of COVID-19 cases is not accessible to most researchers, Zhao et al. [13] created a publicly available dataset with 363 healthy individuals and 349 COVID-19 instances from 216 patients. Gozes et al. [14] studied COVID-19 positive cases identification and tracking by processing their CT images, which provided a 95% detection rate [14]. In another attempt, Zheng et al. [15] suggested a deep learning algorithm to detect COVID-19 on 3D CT images. Ai et al. [16] used a pre-trained UNet to predict infection in 630 patients' CT scans and could achieve 97% sensitivity [15]. Most of these methods either suffer from low accuracy or high complexity.

Due to the limitation of lung CT images belonging to COVID-19 cases, the COVID-CT dataset was publicly accessible as the first public dataset in this field. After releasing this dataset, several studies have focused on using this dataset for COVID-19 detection from the CT images [16–18]. The best accuracy, Fl-score, and area under the curve (AUC) were previously attained at 85%, 86.5%, and 94% in reference [17], which is not acceptable for COVID-19 diagnosis. Soares et al. [19] recently published the SARS-CoV-2 CT-scan dataset containing 2482 CT scan lung images. Due to the volume of this dataset, training deep learning on this number of images imposes a high computational burden [20]. Therefore, a precise and time-efficient COVID-19 detector is required. After a thorough examination of different methodologies published in recent years for identifying and diagnosing COVID-19, it is evident that the DCNN is one of the most widely employed techniques [21]. Therefore, this research suggests using DCNN's exceptional capacity as a detector of COVID-19 positive cases in an efficient computing manner.

Recently, metaheuristics have aimed to reduce DCNN design complexity by constructing an architecture without human involvement. Several metaheuristic algorithms have been successfully used for this purpose including genetic programming (GP) [22], sine-cosine algorithm (SCA) [23], particle swarm optimization (PSO) [24], Chimp optimization algorithm (ChOA) [25], and genetic algorithms (GAs) [26]. However, learning from massive data sets is impractical due to high computational costs and a time-consuming learning process. The EvoCNN model [11] uses GA to evolve DCNN structures efficiently [11]. EvoCNN saves time by teaching the model in just ten epochs rather than the LEIC's 25,600 epochs.

Regardless of various metaheuristics merits, the No-Free-Lunch (NFL) theorem [27] asserts that no metaheuristic technique can satisfactorily address all optimization issues. In order to solve various optimization challenges, scientists have tried to create new metaheuristics or improve current ones. In order to solve the challenge of DCNN structure developing without human involvement, we use powerful nature-inspired algorithms called sine-cosine algorithms (SCA) [28]. In addition to the NFL theorem, another motivation for choosing SCA is that the SCA is a fast nature-inspired algorithm in which the random solutions converge to a proper solution in a predefined number of iterations compared to the other nature-inspired algorithms. In order to overcome the restrictions of fixed-length encodings, a novel flexible encoding approach is presented in this study.

As a result, the proposed model's primary purpose is to create and implement a functional SCA that automatically discovers suitable DCNN structures. To sum up, the paper has the following contributions:

(1) Creating a new SCA method that efficiently uses an innovative candid solution encoding technique to encode a DCNN architecture.
(2) Developing a technique to learn variable-length DCNN structures without the fixed-length encoding limitation of existing SCA. In this regard, a new enfeebled layer will be proposed to create a variable-length searching agent.

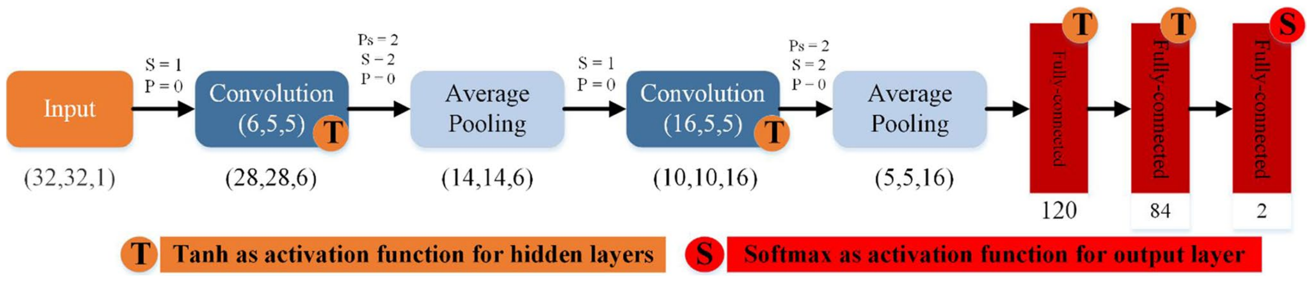


Fig. 1 The LeNet-5 architecture

(3) Developing a fitness evaluation technique that uses a subset of the dataset rather than the entire dataset to speed up the developing process.

The rest of this paper is organized as follows. Section 2 summarizes DCNN, SCA, and CT scan datasets. Section 3 presents the proposed variable-length encoding strategy. Section 4 presents the empirical results and discussion. Section 5 represents the conclusion.

2 Background and materials

This section describes the structure of the LeNet-5 DCNN, SCA, and CT scan datasets.

2.1 DCNN

Among the new DCNN models, LeNet-5 [29] is a simple and efficient one. We utilize this network because it has a simple structure and requires fewer parameters. As shown in Fig. 1, convolution, subsampling, and fully connected layers are three primary components of DCNNs. Furthermore, feature maps, i.e., FM_{ij}^{k} and sub-sampling operation, are respectively performed by Eqs. (1) and (2).

$$FM_{ij}^{k} = \tanh((W^{k} \times x)_{ij} + b_{k}) \tag{1}$$

$$\alpha_j = \tanh(\beta \sum_{N\times N} \alpha_i^{n\times n} + b) \tag{2}$$

where b and β are respectively the bias values and learning constants, and also $\alpha_i^{n\times n}$ are inputs. The last fully connected layer then performs classification. This layer has only one neuron because our problem is a binary task (COVID and non-COVID).

2.2 Sine-cosine algorithm

SCA is a meta-heuristic mathematical inspired algorithm that uses the trigonometric properties of sine and cosine functions. Compared to the other nature-inspired algorithms, SCA is a fast manner in which the random solutions converge to a proper solution in a predefined number of iterations. The algorithm is started from an initial population, and the candidate solutions are updated according to Eqs. (3-4) [28].

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 \times sin(r_2) \times \left| r_3 p_i^t - X_i^t \right| & , \ r_4 < 0.5 \\ X_i^t + r_1 \times cos(r_2) \times \left| r_3 p_i^t - X_i^t \right| & , \ r_4 \geq 0.5 \end{cases} \tag{3}$$

$$r_1 = a - t\frac{a}{T} \tag{4}$$

where X_i^t denotes the position of the current solution in the tth iteration and ith dimension. The parameter r_1 is linearly decreased by increasing the number of iterations (t), where parameter a is a fixed number and T is the final number of iterations. In addition, parameters r_2 , r_3, and r_4 denote random numbers, which are varied in the ranges of $[0, 2\pi]$, [0,2], and [0,1], respectively. p_i stands for the best solution found so far in the ith dimension, and $|.|$ is the absolute value. The value of a in Eq. (4) can control the trade-off between exploration and exploitation. As $a \gg 1$, r_1 starts from a more significant value which provides a suitable exploration, and by increasing the iteration number of t, the exploration is decreased while exploitation is increased. Parameter r_4 randomly chooses the sine or cosine function based on Eq. (3), leading to insert the randomness to the parameter r_1 to guide the solution X_i^t. This is because r_1 is a linearly decreasing parameter and suffers from the lack of randomness in both magnitude and sign [28]. Sine and cosine functions are orthogonal to each other, and the probability of being positive or negative for a random angle is independent of each other. The sine and cosine functions have a periodic structure, enabling a response to accumulating around another reaction. As a result, it is guaranteed that the specified location between the two responses will be identified. Thus, the solutions should thoroughly search in a search space to locate the target within the search region. This capability is obtained by

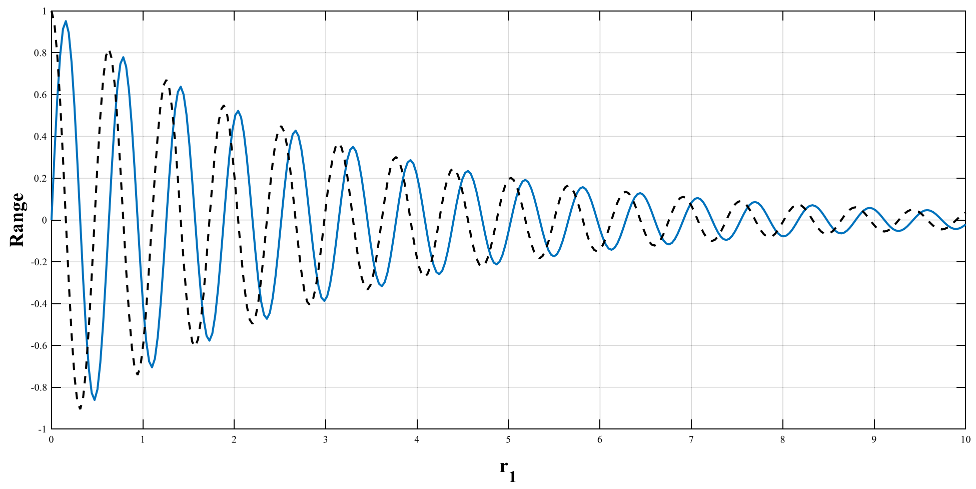


Fig. 2 Range reduction of the sine and cosine values during iterations

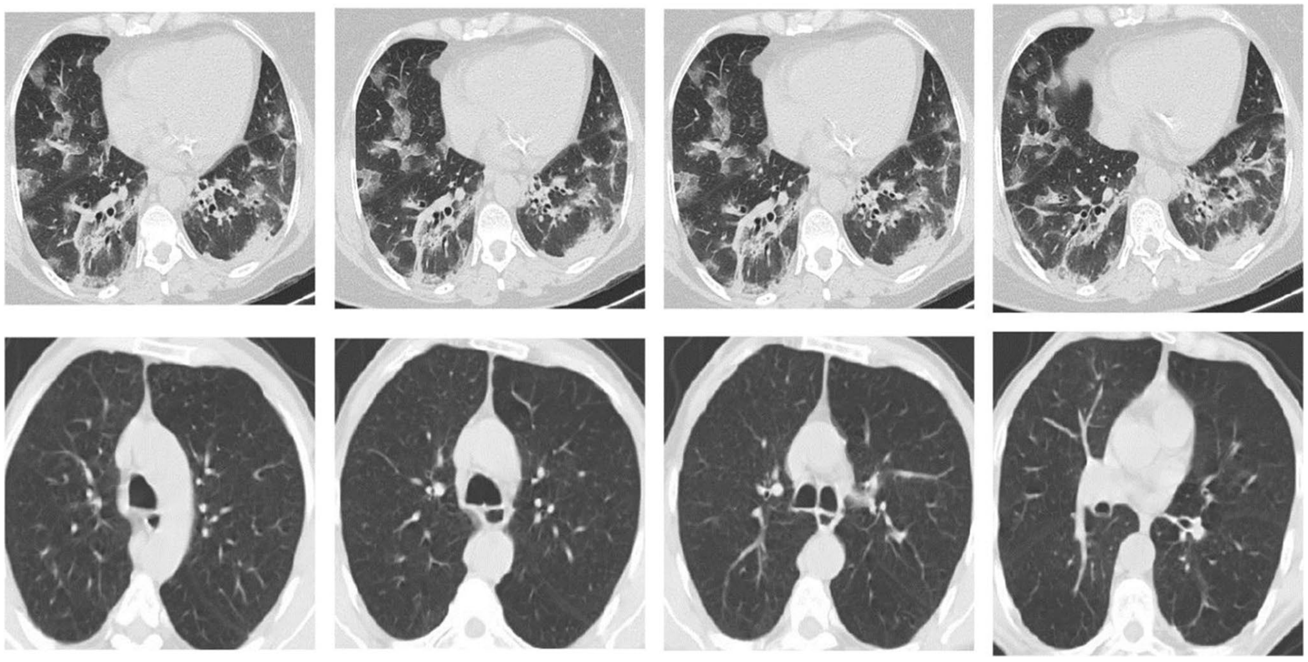


Fig. 3 Typical CT images from the SARS-CoV-2 CT-scan dataset

modifying the amplitude of the sine and cosine values. Figure 2 illustrates the cosine and sine amplitude reduction in the course of iterations.

As shown in Fig. 2, SCA traverses the searching space whenever the sine and cosine values are in ranges [−2,−1) and (1,2). On the other hand, the algorithm recognizes the searching space when they are between [−1,1].

2.3 Datasets

SARS-CoV-2 [20] and the COVID-CT [30] are two CT-scan image datasets used in this study, which their detail is explained in the following subsections.

2.3.1 SARS-CoV-2 CT-scan dataset

The SARS-CoV-2 originates from Sao Paulo hospitals. This dataset contains 2482 CT scan images from 120 individuals, where 1252 CT scan images are from SARS-CoV-2 patients, and 1230 are from non-infected individuals with various respiratory illnesses. Figure 3 illustrates positive and negative COVID-19 case samples. The first raw contains some COVID-19-infected instances, while the second one contains normal (uninfected) cases. The eXplainable Deep Neural Network technique (xDNN) has been presented as a baseline for this dataset, which may reach an F1-score of 97.31%. Everyone can access the data set in the link below:

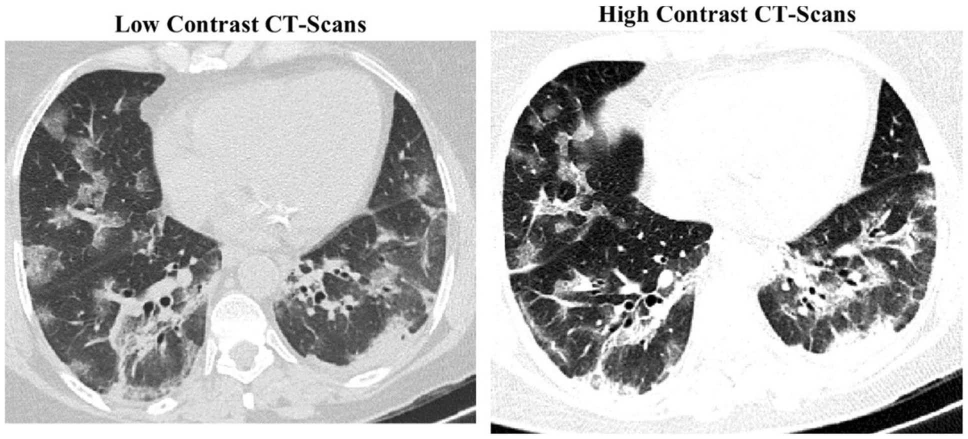


Fig. 4 The representation of two images with different sizes and contrast

www.kaggle.com/plameneduardo/sarscov2-ctscan-dataset

As demonstrated in Fig. 4, the SARS-CoV-2 has no extraordinary uniformity in terms of contrast and image sizes.

2.3.2 COVID-CT dataset

The COVID-CT dataset[1] [30] contains COVID-19-infected CT scans of patients. This dataset comprises 349 images from 216 patients. The methodology used to create the COVID-CT dataset is described as follows: First, 760 preprints on COVID-19 were acquired from medRxiv[2] and bioRxiv[3], between January 19th and March 25th. Several preprints describe patient instances with COVID-19, while some of them provide CT images, which are accompanied by subtitles that describe the clinical findings revealed by the CTs. PyMuPDF[4] is used to extract the PDF files of preprints' low-level structural information and find all embedded images. The quality factors of the images (including resolution, size, etc.) are preserved. The captions for the images are identified using structural information. All CT images are hand-picked using the derived images and captions. Then, for each CT image, the caption is reviewed to determine whether it is positive for COVID-19 or not. If the caption cannot be determined, the text evaluating this image in the draft is retrieved to make a determination. Metadata taken from the paper is also gathered for each CT image, including the patient's age, location, gender, scan time, medical history, scan time, COVID-19 severity, and radiologist report.

Finally, 349 CT images with a positive label for COVID-19 are collected. These CT scans come in a variety of sizes. The lowest, average, and highest heights are 153, 491, and 1853, respectively. The widths at the lowest, average, and maximum are 124, 383, and 1485, respectively. These images belong to 216 patients. Similar to the first dataset, the second dataset contains no predefined image contrast and size modification. As a result, we begin by improving the quality and size of the low-contrast images, which is outlined in [31], as follows:

- The scan center obtains the CT image.
- A grayscale image is created by converting the color image to a grayscale image.
- The images are adjusted to a variety of sizes.
- Various filtering techniques are used to eliminate noise from the gray photo.
- The image's contrast is increased through the use of the adaptive histogram equalization filtering technique.
- The acquired image is the output image that has been improved.

Fig. 5 shows an unedited version and its upgraded versions. Table 1 lists the datasets used in this study.

3 Methodology

This section describes the IP-based SCA (IPSCA) approach to improving DCNNs in detail.

3.1 Algorithm description

The IPSCA method is structured in Algorithm 1. The population will be updated in three steps: encoding candid solutions, updating positions, and checking for termination Fig. b.

[1] https://github.com/UCSD-AI4H/COVID-CT
[2] https://www.medrxiv.org/
[3] https://www.biorxiv.org/
[4] https://github.com/pymupdf/PyMuPDF

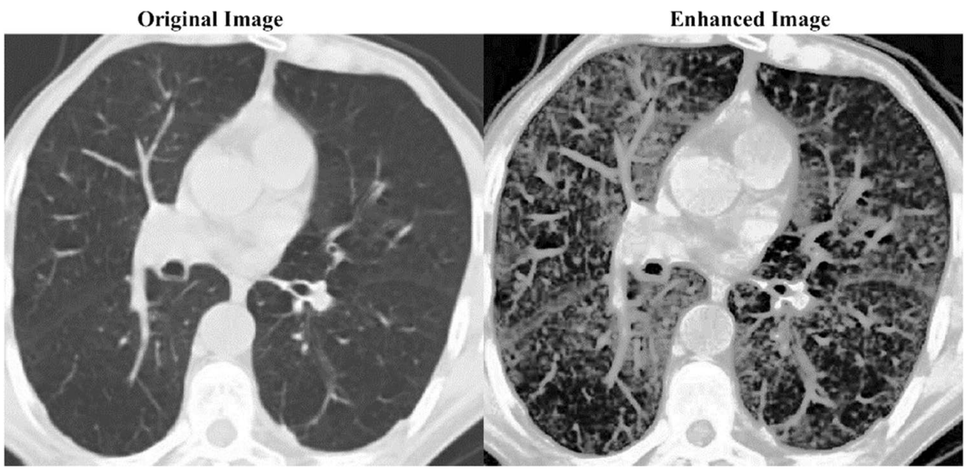


Fig. 5 An illustration of the standard image (left image) and its enhanced version (right image)

Table 1 The datasets' specification

Dataset	Number	SARS-CoV-2 CT-Scan [20]	COVID-CT [22]
COVID-19	Patients	60	216
	Images	1252	349
Non-COVID-19	Patients	60	55
	Images	1230	463

3.1.1 Candid solution encoding strategy

The Network IP address structure influences the IPSCA encoding approach. The DCNN structure contains three layers, each with different parameters. Table 2 lists the successive layers. We can create a fixed-length Network IP address and subdivides it to allow different DCNN layers.

The IPSCA encoding scheme is based on the IP address. The DCNN architecture has three types of layers:

```
Initialize the individuals with a candid solution encoding strategy  $X_i^t \leftarrow$ empty
while termination criterion is not satisfied
    update searching agents' position by Eq. 5;
    evaluate the searching agents' fitness;
    update $X_i^t$;
end while
```

Algorithm 1 Framework of IPSCA

Table 2 The information of the DCCN's layers with an example

Layer Type	Parameter	Range	No. of bits	Example
Convolutional	Filter size	[1, 8]	3	3(011)
	No. feature maps	[1,128]	7	5(000 0101)
	Stride size	[1, 4]	2	3(11)
	Summary		12	011 000 0101 11
Pooling	Kernel size	[1, 4]	2	3(11)
	Stride size	[1, 4]	2	3(11)
	Type:1(max),2(ave)	[1, 2]	1	1(1)
	Place holder	[1,128]	6	8(00 1000)
	Summary		11	11 11 1 00 1000
Fully-connected	No. Neurons	[1,2048]	11	128(00010000000)
	Summary		11	00010000000
Enfeebled	Place holder	[1,2048]	11	128(00010000000)
	Summary		11	00010000000

Table 3 Four utilized subnets for four types of DCNN's layers

Layer type	Convolution	Fully connected	Pooling	Enfeebled
Subnet (CIDR)	0.0/4	16.0/5	24.0/5	32.0/5
IP range	0.0–15.255	16.0–23.255	24.0–31.255	32.0–39.255

introduced to compensate. Using all three DCNN layers, the enfeebled subnet has an IP address range of 32.0/5, as illustrated in Table 3. A binary string of a specific layer is filled with zeros until it reaches the length of two bytes, at which point the subnet mask is applied, and it is converted to an IP address by dividing a single byte with full stops, as shown in Table 4. Accordingly, in Table 2, 011 000 1000 01, one can get the 2-byte digitized string 0000 0011 and 100001. The first byte was translated to 6 and the second to 33 in order to get the IP address 6.33.

Table 4 IP addresses' examples

Layer type	Convolution	Fully connected	Pooling	Enfeebled
Binary	(0000)011 000 1000 01	(00000)011 11111001	(00000)01 01 0 00 1011	(00000)01111111001
IP address	6.33	27.249	18.139	35.249

convolutional, pooling, and fully connected. As shown in Table 2, each type of layer has different numbers and ranges for various parameters.

An IP-based encoding scheme's length must first be determined. In terms of convolutional layers, three critical parameters, i.e., the number of feature maps, filter size, and stride size, are tabulated in the parameter column of Table 2. Second, based on datasets' size, the parameter ranges are set to [1, 8], [1,128], and [1, 4], respectively. The decimal values of 3, 5, and 3 may be translated into digitized strings of 011, 000 0101, and 11, where the digitized string is completed with zeros until the length exceeds the proper amount of bits, as shown in the value column in Table 2. Finally, the summary row of the convolutional layer in Table 2 displays the total number of bits of 12 and the sample digitized string 011 000 0101 01 by coupling the strings of the three mentioned parameters. The total amount of bits and the sample binary string can be obtained from pooling and fully-connected layers using the same technique as convolutional layers, which is shown in Table 2. Since the maximum bit number for a layer is twelve, and an IP address contains eight bits (one byte), the 12-bit IP address will require two bytes.

As stated in Table 3, all DCNN layers' subnets must be in CIDR format. Since DCNN layers are categorized into three types, three subnets must support each type. The subnet mask has the length of the IP address 0.0 plus the convolution layer's bit numbers (12 bits). The subnet mask is 0.0/4 (0.0–15.255). The pooling layer's initial IP address is 16.0, created by adding 1 to the final convolution layer's IP address. As with the convolution layer, the subnet mask's length is five, resulting in 16.0/5. As a result, subnets 16.0–23.255 reflect the pooling layer. A similar issue occurs with the fully connected layer subnet 24.0/5, covering 24.0 to 31.255. Table 3 depicts the subnets' arrangement.

To adapt DCNN topologies at differing lengths, some layers in the IP-coded candid solution vector are disabled after initialization. Therefore, the enfeebled layer and subnet are

Table 5 The list of parameters

Parameters	Parameter definition	Value
Max-length	Max. length of DCNN layers	13
N	Population size	40
Max-totally connected	Max. fully connected layers (at least one fully-connected layer is required)	3
K	The training epoch number before evaluating the trained DCNN	10
No. batch	The batch size for evaluating the DCNN	200

Before calculating searching agents, each layer must be transformed into a 2-byte IP address. First, some characteristics like max-length and max-totally-connected must be explained. Table 5 shows these parameters. The array holding position data will be double the size of a single byte, with each byte indicating one dimension of the candid solution.

As an example, consider this candid solution vector to better understand the DCNN coding and its variable-length topology. Pooling (*P*), convolution (*C*), fully connected (*F*), and enfeebled layers can all be represented by IP addresses in the DCNN architecture shown in Fig. 6. The size of the related candid solution vector is illustrated in Fig. 7.

After several SCA updates, the 9th and 10th dimensions of the candid solution vector could turn into 18 and 139, respectively, which transforms the 3rd IP address showing an enfeebled layer into a pooling layer, thereby changing the candid solution's DCNN architecture to five layers. In summary, as demonstrated in this example, the IPSCA-encoded searching agents may represent variable-length DCNN architectures, i.e., 3, 4, and 5 in this case.

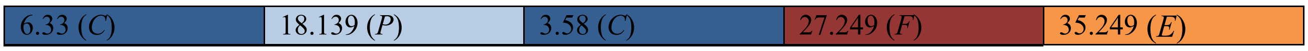


Fig. 6 An IP address from a candid solution with five DCNN layers

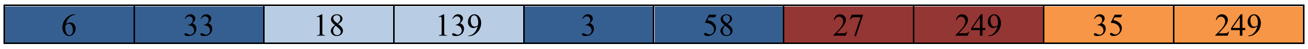


Fig. 7 The illustration of an encoded candid solution vector

```
Initialize
        The training epoch number ($k$)
        The Population ($P$)
        The batch size ($batch\_size$);
        The training set ($D_{train}$)
        The fitness evaluation dataset ($D_{fitness}$)
for candid solution $c$ in P
    $i = 1$
    $i = i + 1$
    while $i \leq k$
     Train the DCNN's connection weights described by candid solution $c$;
    end while
$accuracy$ = Batch-evaluate the trained DCNN on the $D_{fitness}$ dataset with the $batch\_size$
store each batch's accuracy;
Calculate the mean ($accuracy$);
Fitness value $\leftarrow$ mean ($accuracy$);
$P \leftarrow$ calculate the fitness value of the candid solution $c$
end for
return $P$
```

Algorithm 2 The evaluation of fitness function

3.1.2 Initializing population

To construct our population, first, the population size is decided, and then, individuals are generated until the population size is attained. Each empty vector of individuals includes a network interface comprising the IP address and subnet information for that individual. Each following layer may be loaded with a convolution, pooling, or enfeebled layer. Before the final fully connected layer, any four-layer types may be used. The final layer is composed of a fully connected layer. A random IP address will be assigned to the subnets for each layer.

3.1.3 Fitness assessment

Before fitness evaluation, it is required to choose a weight initialization method. The Xavier weight initialization method [32] is chosen because it has been shown to be effective and is used in several deep learning networks. Individuals will be trained with the settings for their DCNN architecture, which have been previously decoded, for the first half of the training dataset (as described in Algorithm 2). Once the partially trained DCNN is batch-evaluated on the second segment of the training dataset, a sequence of accuracies will be generated. Following that, we take an average over the accuracies of each individual to determine the candid solution's fitness Fig. c.

The next step is to modify the candid solution's position by applying the coefficient values for each byte in the candid solution. In order to ensure that each interface in the candid solution vector has access to the same subnet, each interface should have a unique IP address assigned to it. Each interface limitation, such as the second interface capacity to serve as a convolution layer, pooling layer, or enfeebled layer, may vary based on its position in the candid solution vector.

Table 6 The setting parameters and initial values

Algorithms	Parameters	Value
VLBBO	P_c	1
	The range of migration probability	[0, 1]
	Max (I) and Max(E)	1
	Step size	1
	Mutation probability	0.005
VLGA and VLNSGA-II	Type	Real coded
	Crossover	Single-point (1)
	Selection	Roulette wheel
	Mutation	Uniform (0.01)
VLACO	τ_0	0.000001
	Q	20
	q_0	1
	P_g	0.9
	P_t	0.5
	A	1
	β	5
VLBSO	M	5
	P_{5a}	0.2
	P_{6b}	0.8
	P_{6biii}	0.4
	P_{6c}	0.5
	K	20

4 Simulation results and discussion

In general, we attempt to incorporate IPSCA in order to improve the diagnostic accuracy of the DCNN. The number of searching agents (population size) is set to 50 for all tests, and the maximum number of iterations is set to 200. DCNN is configured with a batch size of 0.0002 and a learning rate of 11. Moreover, epoch number is chosen between 1 and 10 for each assessment, and it is emphasized that prior to feeding images to DCNNs, they must be down-sampled to 31 × 31 pixels by the bicubic down-sampling method [33]. It should be noted that the two datasets were used to train and test the model. leave-one(patient)-out cross-validation is used for model evaluation. The leave-one-out cross-validation procedure is appropriate for this kind of dataset because the size of the dataset is small, and an accurate estimation of the model performance is necessary. All experiments were conducted in MATLAB R2019b on a computer equipped with an Intel Core i7-7700HQ processor running at a maximum speed of 3.8 GHz, Windows 10, and 16 GB RAM. DCNN-performance IPSCA's is compared to that of IPPSO [24], variable-length genetic algorithm (VLGA) [34], variable-length NSGA-II (VLNSGA-II) [35], variable-length brain storm optimization algorithm (VLBSO) [36], IP-Modified PSO (IPMPSO) [37], variable-length biogeography-based optimizer (VLBBO) [5], and variable-length ant colony optimization (VLACO) [38] on the two utilized datasets. Table 6 summarizes the parameters of the SCA and additional benchmark models. The population size for all algorithms is set to 50.

4.1 Evaluation metrics

Sensitivity, accuracy, specificity, F1-score, and precision are all performance metrics used in this research. The metrics' equations are represented in Eqs. 5 to 9.

$$\text{Sensitivity } (TPR) = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{5}$$

$$\text{Specificity } (TNR) = \frac{TN}{N} = \frac{TN}{TN + FP} \tag{6}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{7}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{8}$$

$$F_1\text{-score} = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{9}$$

The number of true negative cases is denoted by TN, the number of true positive cases is denoted by TP, the number of false-positive cases is denoted by FP, and the number of false-negative cases is denoted by FN.

4.2 Performance evaluation

Although various robust deep convolutional neural network structures have been proposed recently, we employ the basic DCNN architecture because the computational cost is an essential factor in the training process. The first experiment predicts a probabilistic score for each image in both datasets, SARS-CoV-2 CT-Scan and COVID-CT, indicating the possibility of each image being diagnosed as a COVID-19 positive case. This probability is then compared to a predefined cut-off value to determine whether the inputs are infected samples or not. In the extreme case, the intended model should get a probability of one for infected samples and zero for normal samples. The EPG distributions for SARS-CoV-2 and COVID-CT are depicted in Figs. 8 and 9. By definition, infected images have a higher probability than uninfected ones. Figures 10 and 11 illustrate the confusion matrices for SARS-CoV-2 CT-Scan and COVID-CT datasets, respectively.

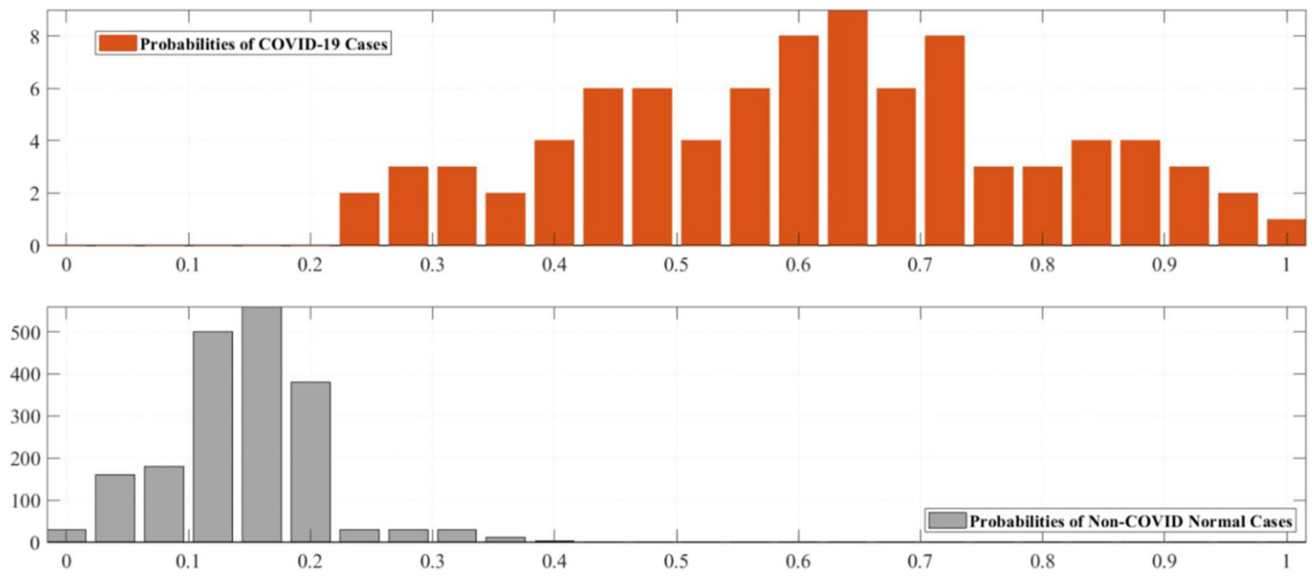


Fig. 8 The EPG for the SARS-CoV-2 CT-Scan dataset

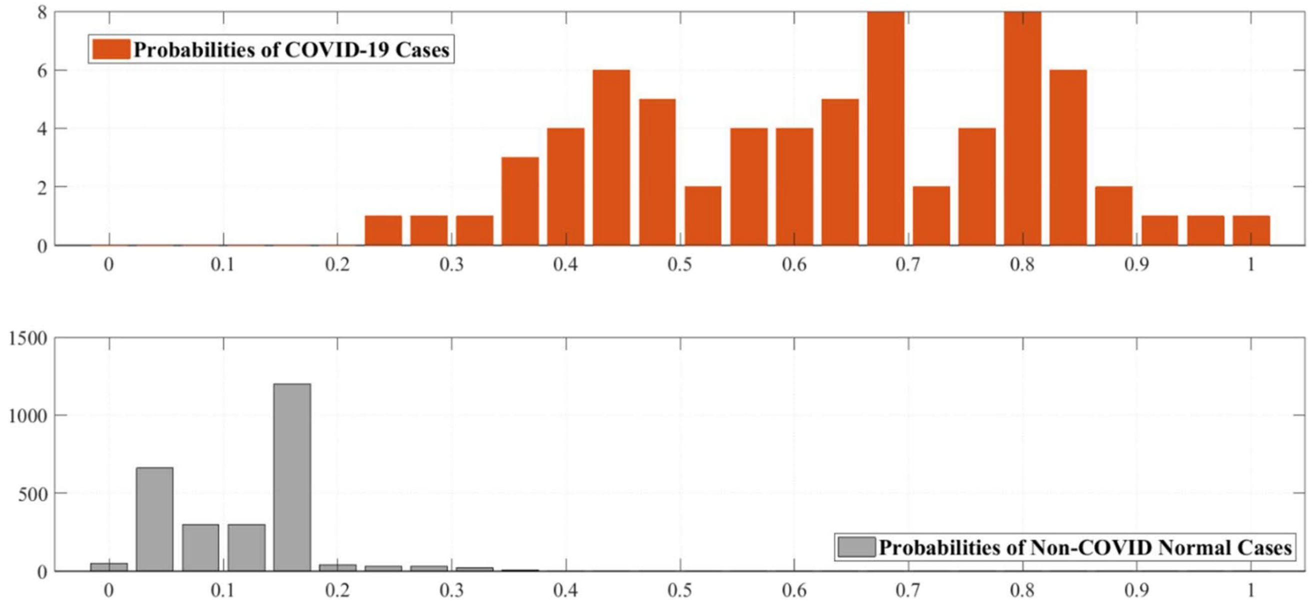


Fig. 9 The EPG for the COVID-CT dataset

Tables 7 and 8 tabulate the specificity, sensitivity, precision, accuracy, and F1-score of the proposed DCNN-IPSCA and other compared methods for the second experiment, i.e., IPPSO, VLGA VLNSGA-II, VLBSO, IPMPSO, VLBBO, and VLACO. The findings show that DCNN-IPSCA has the best accuracy score of 99.18% for the SARS-CoV-2 and 98.22% for the COVID-CT dataset, respectively. Additionally, the second-best result is 99.01% and 98.01% for SARS-CoV-2 CT-Scan and COVID-CT, respectively, obtained using DCNN-VLNSGA-II. The optimal outcome is emphasized in bold type.

Comparing pattern recognizers is done by using a variety of metrics, but only one metric can provide a big picture of the performance of benchmarks crossing a range of thresholds. We analyze the benchmarks' performance across all potential cut-off threshold values using the precision-recall curve to solve this problem. This illustration depicts the link between recall and precision rates. Figures 12 and 13 illustrate the precision-recall curves for the

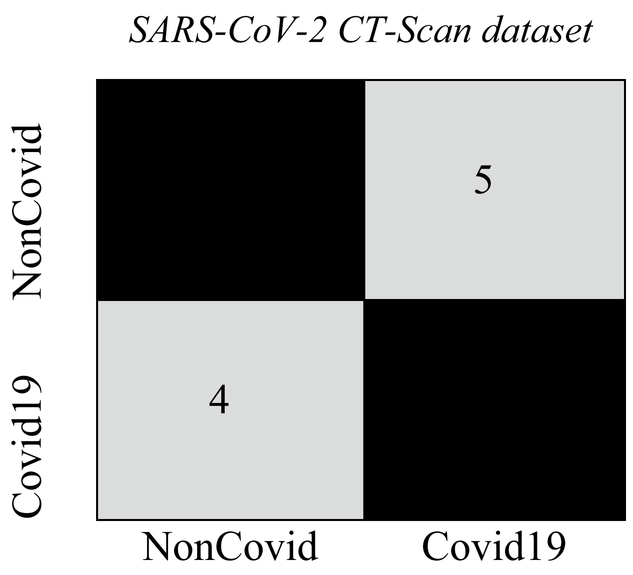


Fig. 10 The confusion matrix for the SARS-CoV-2 CT-Scan dataset

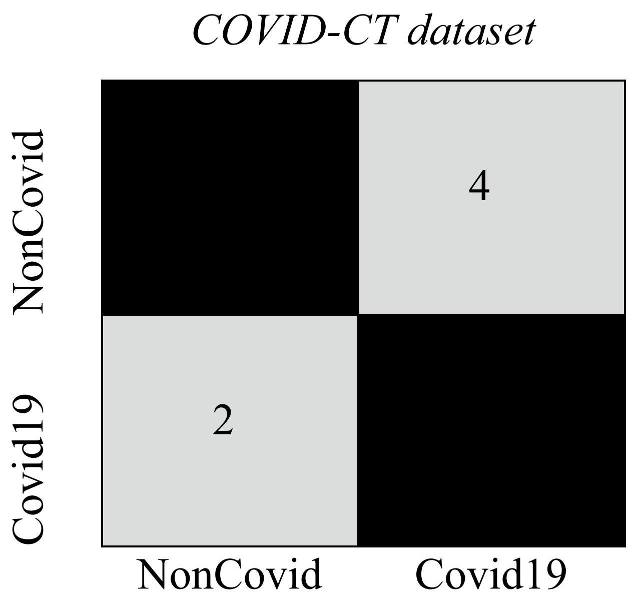


Fig. 11 The confusion matrix for the COVID-CT dataset

proposed DCNN-IPSCA and the other eight benchmarks on the SARS-CoV-2 and COVID-CT datasets. Another appropriate figure for depicting the TPR as a function of the FPR is the receiver operating characteristic (ROC) curve. Thus, Figs. 15 and 16 show the ROC plots for the proposed DCNN-IPSCA and its counterparts. These figures show that the proposed DCNN-IPSCA outperforms the other variable-length DCNNs on both datasets.

Convergence speed is a crucial metric when comparing metaheuristic algorithms. To facilitate comparisons, in addition to the plots depicted above, the convergence curves of comparative benchmarks are shown in Fig. 14.

The two preceding plots show that the proposed DCNN-IPSCA model outperforms the other competitive models. In addition, it can be concluded that the canonical DCNN has nearly the worst performance among the rivals.

4.3 Time complexity analysis

There is a trade-off between accuracy and processing time. The accuracy of the suggested model was compared to that of the other benchmarks in the preceding section. The processing time of the proposed model is compared in this section to form a complete comparison. We constructed the proposed DCNN-IPSCA and comparison networks on an Intel Core i7-7700HQ CPU and an NVidia Tesla K20 GPU. Table 9 shows the time values for the 2633 training images and 659 test images. Furthermore, the best outcome is highlighted in bold type.

The results of Table 9 indicate that the proposed variable-length DCNN (DCNN-IPSCA) performs much better than the other benchmarks, with CPU and GPU training times being 63.10 and 387.69 times faster, respectively. It is concluded that DCNN-IPSCA requires less than 0.001 seconds for each image in testing and training by examining the total number of processed images, i.e., 3392.

4.4 Sensitivity analysis

This subsection evaluates the sensitivity of the control parameters employed in the proposed DCNN-IPSCA. The parameter a specifies the SCA reduction rate, which affects the convergence process. The second and third parameters (N_{Layer} and N_{batch}) are concerned with the network structure. The analytical findings illustrate the parameters' robustness and input sensitivity. The associated tests were conducted using four different parameter settings [23]. A lot of parameter combinations were determined using an orthogonal array, as shown in Table 10. The model is trained for each possible combination of the parameters.

Additionally, Table 11 shows the mean square errors (MSEs) achieved for each test data. Figure 15 illustrates the trend in the values of the parameters in accordance with the data in Table 11. It can be observed from the results that $N_{Layer} = 5$, $a = 1$, and $N_{batch} = 10$ produce the best results.

4.5 Class activation mapping

Along with classification accuracy, we search for areas inside an image that contribute to categorization. Class activation mapping (CAM) [39] is utilized for this purpose.

Table 7 The results of benchmark models for SARS-CoV-2 dataset

Model	Precision (%)	F1-score (%)	Accuracy (%)	Sensitivity (%)	Specificity (%)
DCNN	94.86	93.75	93.47	94.44	93.32
DCNN-IPPSO	95.44	94.21	94.33	94.75	93.12
DCNN-VLBSO	95.63	93.22	94.44	93.54	92.12
DCNN-IPMPSO	96.32	95.87	95.63	95.37	95.75
DCNN-VLGA	94.44	95.01	95.88	94.32	94.55
DCNN-VLBBO	96.33	96.77	96.01	95.92	96.02
DCNN-VLACO	97.32	96.25	96.96	96.33	96.79
DCNN-VLNSGA-II	97.99	97.27	97.85	96.91	**96.83**
DCNN-IPSCA	**98.23**	**97.99**	**98.32**	**97.22**	96.77

The best results are presented in the bold type

Table 8 The results of benchmark models for COVID-CT dataset

Model	Precision (%)	F1-score (%)	Accuracy (%)	Sensitivity (%)	Specificity (%)
DCNN	95.25	94.39	94.56	94.88	94.22
DCNN-IPPSO	95.32	94.22	94.45	93.89	94.97
DCNN-VLBSO	96.48	95.36	95.53	94.65	95.75
DCNN-IPMPSO	97.47	96.25	96.32	95.87	96.02
DCNN-VLGA	95.63	94.87	96.85	95.43	94.63
DCNN-VLBBO	97.01	95.33	96.99	95.89	94.65
DCNN-VLACO	97.15	96.65	96.25	95.42	95.32
DCNN-VLNSGA-II	98.22	96.56	97.99	95.33	**96.98**
DCNN-IPSCA	**98.95**	**97.45**	**98.01**	**96.23**	96.44

The best results are presented in the bold type

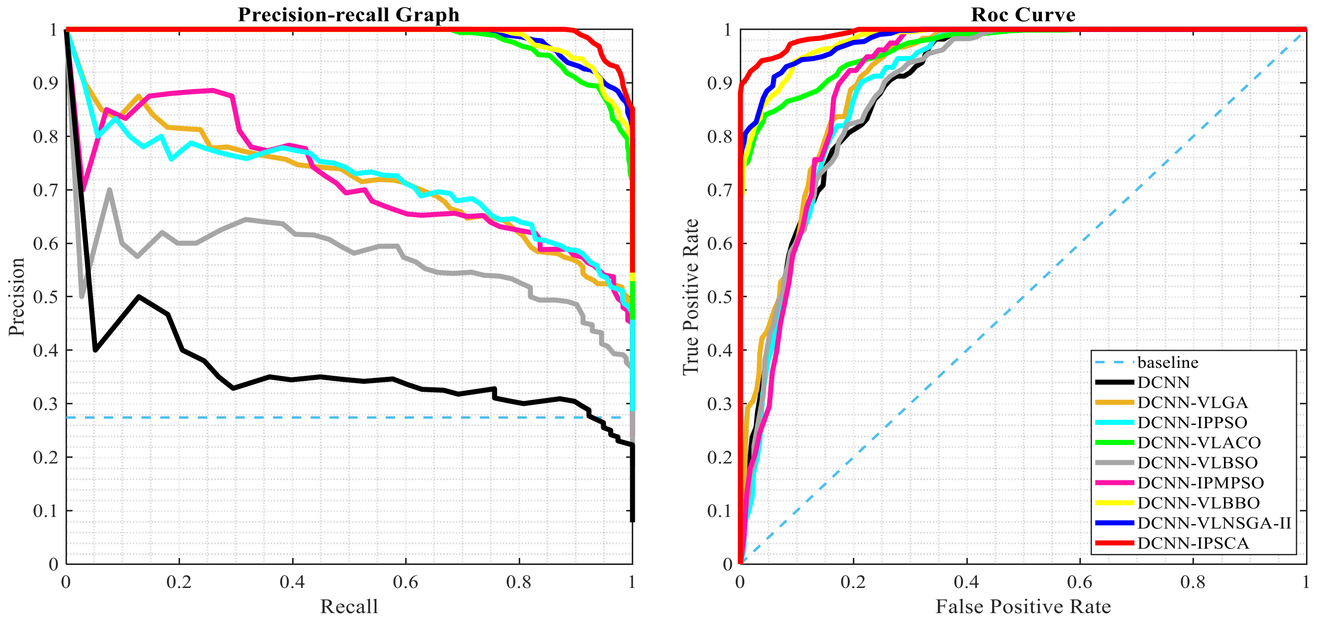


Fig. 12 The precision-recall and ROC curves for the SARS-CoV-2 dataset

As a result, the discriminative areas, which are specific to each class, are highlighted by returning the DCNN model's probability to the final convolutional layer of the associated model in this manner.

Following the final convolutional layer, the activation map of the ReLU layer yields the CAM for a given image class. Class grades are derived from the weighted average of each activation mapping. The innovation of CAM is the

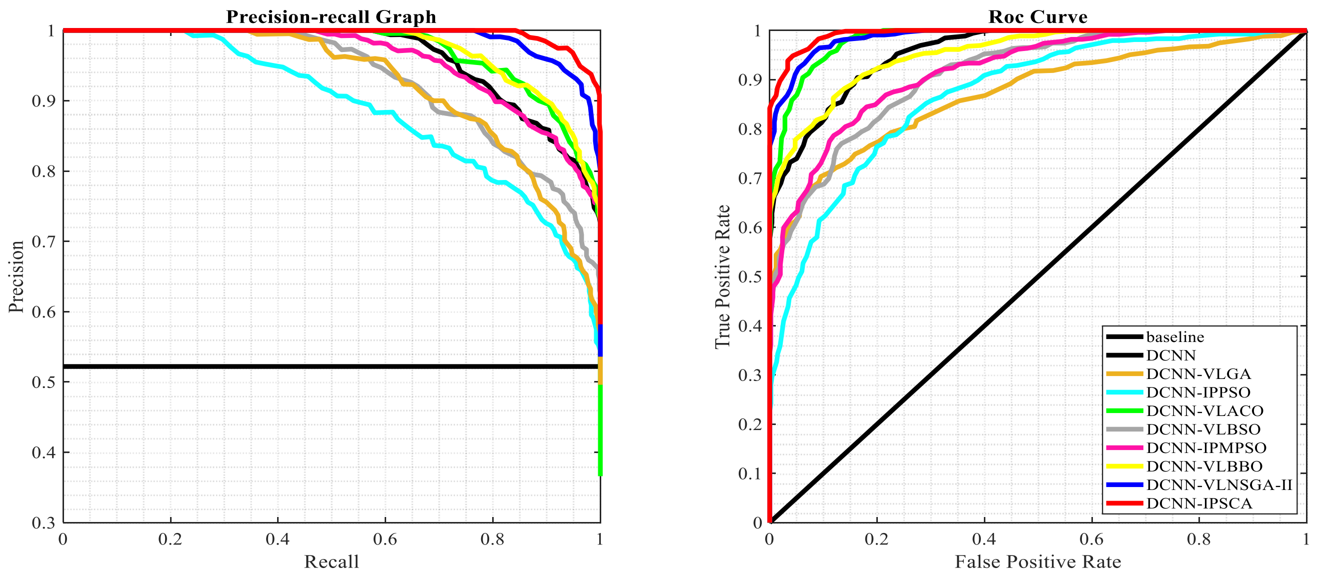


Fig. 13 The precision-recall and ROC curves for the COVID-CT dataset

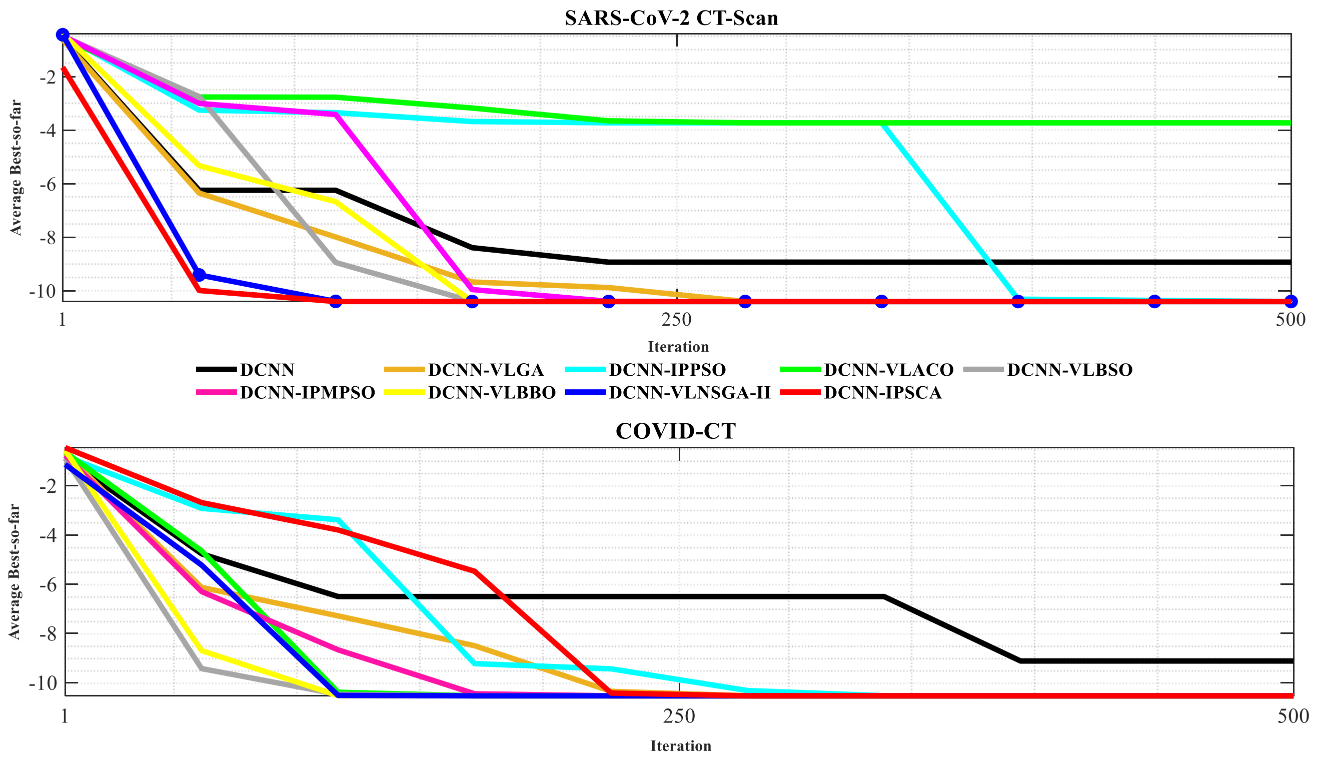


Fig. 14 The convergence curves for utilized benchmark algorithms

combined total pooling layer used after the last convolutional layer depending on the geographic location to provide the connection weights. Thereby, it allows recognizing the appropriate regions inside a CT scan that contrast the class specificity before the softmax layer, which can boost trust in the findings.

Figure 16 displays a typical example of the feature map for input images, i.e., COVID19 and non-COVID19.

Table 9 The comparison of test and training time of benchmark networks implemented on GPU and CPU

Model	CPU vs. GPU	Training time	Testing time	*P*-value
DCNN	GPU	126 sec	603 ms	0.0012
	CPU	1 h, 7 min, 2 s	55 sec	0.0031
DCNN-IPPSO	GPU	1200 ms	825 ms	0.0051
	CPU	2 min, 13 s	1 min, 56 s	0.047
DCNN-VLBSO	GPU	747.3 ms	625 ms	0.0056
	CPU	2 min, 21 s	2 min, 01 s	0.0084
DCNN-IPMPSO	GPU	2200 ms	933 ms	0.031
	CPU	3 min, 42 s	2 min, 03 s	0.0051
DCNN-VLGA	GPU	633 ms	402 ms	0.021
	CPU	2 min, 41 s	1 min, 52 s	0.0054
DCNN-VLBBO	GPU	733.7 ms	601 ms	0.0063
	CPU	2 min, 33 s	1 min, 05 s	0.012
DCNN-VLACO	GPU	1258 ms	823 ms	0.0063
	CPU	3 min, 42 s	2 min, 03 s	0.0001
DCNN-VLNSGA-II	GPU	1012 ms	871 ms	0.047
	CPU	3 min, 25 s	2 min, 01 s	0.005
DCNN-IPSCA	GPU	325 ms	216 ms	0.021
	CPU	58 s	1 min, 02 s	N/A

Table 10 The parameters' specification

Level	N_{layer}	a	N_{batch}
1	3	0.5	6
2	4	1	8
3	5	1.5	10
4	6	2	12

Table 11 The MSE for various control parameter values

Experiments	Parameters			Result (MSE)
	N_{layer}	τ	N_{batch}	
#1	1	1	1	0.0625
#2	1	0.5	2	0.0439
#3	1	1.5	3	0.0239
#4	1	0.5	4	0.0129
#5	2	2	2	0.0409
#6	2	1	1	0.0339
#7	2	2	4	0.0109
#8	2	1.5	3	0.0079
#9	3	0.5	1	0.0199
#10	3	1	4	0.0089
#11	3	1.5	2	0.0039
#12	**3**	**2**	**3**	**0.0019**
#13	4	0.5	4	0.0249
#14	4	1	3	0.0110
#15	4	2	2	0.0051
#16	4	1.5	1	0.0029

The best results are presented in the bold type

Figure 17 displays a typical example of masked images for COVID19 and non-COVID19 input. Consequently, Figs. 18 and 19 depict the discriminative zones for various common CT-scan images. Figures 18 and 19 demonstrate the CAM demonstration results for the "COVID19" and "Non-COVID19" examples, respectively, which the proposed model properly predicts and also reveals the discriminative region for its choice.

Clearly, various areas are highlighted (reddish color) in the COVID19 instances. This demonstrates that the system does a successful categorization. Medical professionals and radiologists can benefit from this visual representation of deep learning models, which provides a second viewpoint and a deeper understanding of these models.

4.6 Discussion

SCA and deep learning techniques were used for COVID-19 evaluation using CT images, and the findings showed that our model, which was developed using COVID-CT and SARS-CoV-2 datasets, had good generalizability in terms of the CT images it was trained on. On the SARS-CoV-2 and COVID-CT datasets, the proposed model yields final accuracy of (98.32% and 98.01%), the sensitivity of (97.22% and 96.23%), and specificity of (96.77% and 96.44%), respectively. Our approach outperforms the DCNN architecture search model on all of the measures we have discussed. COVID-19-negative patients are appropriately identified as negative in the great majority of instances, reducing the likelihood of incorrectly classifying COVID-19-negative cases as positive and reducing the cost to the healthcare system. Additional testing with limited data revealed that the model currently performs adequately despite its shortcomings. In the real world, where diverse and large datasets may not be easily available, this indicates that our methodology is still useful with limited data. Finally, we used the CAM visualization approach to better understand and describe the suggested deep learning model for COVID-19 testing. The model validates its performance by comparing it to the radiologist's interpretation of the COVID-19 scans. New visual markers to aid clinical practitioners in additional manual screening can be discovered by examining normal and COVID-19 CT images. Our models' performance in COVID-19 testing has been demonstrated in the results. The severity of COVID-19 will be closely monitored in the future, and we will use CT scans to gather more data that can help us against the pandemic, as well. After doing descriptive assessments on the models, we will discover critical CT image features that may be used to identify COVID-19 and facilitate screening by

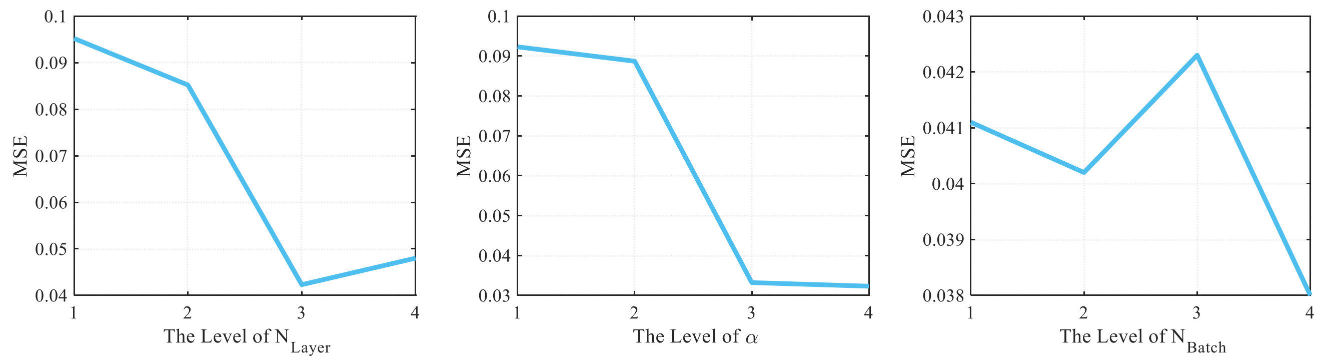


Fig. 15 The results of the control parameters

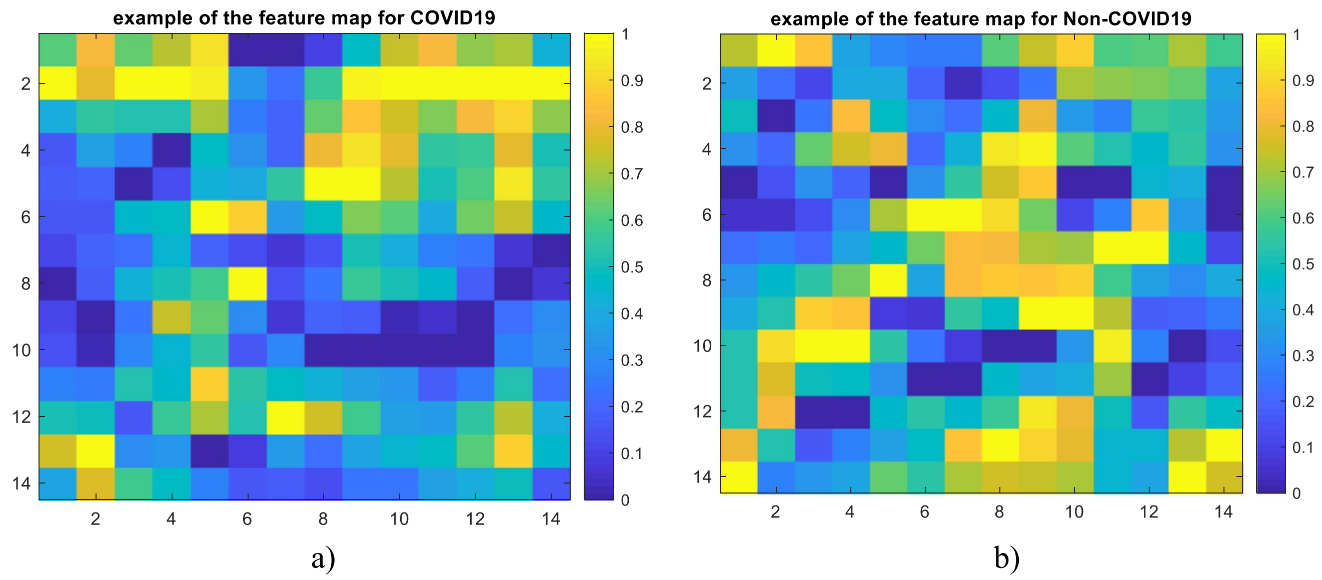
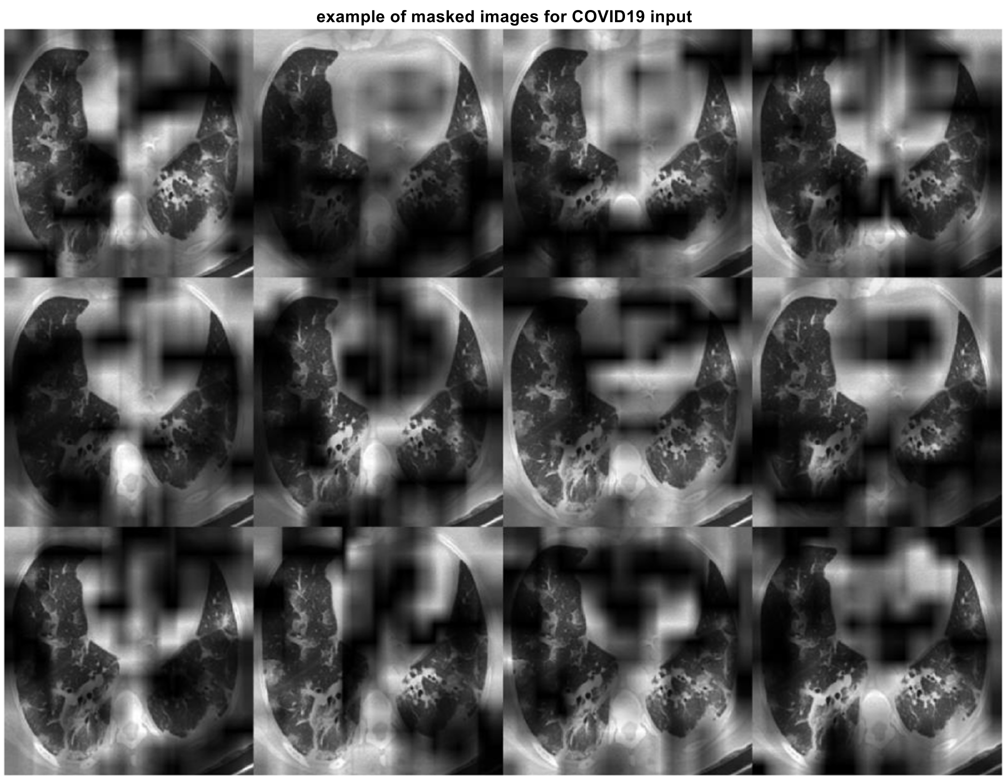


Fig. 16 Typical example of the feature map for input images: **a** COVID19 and **b** non-COVID19

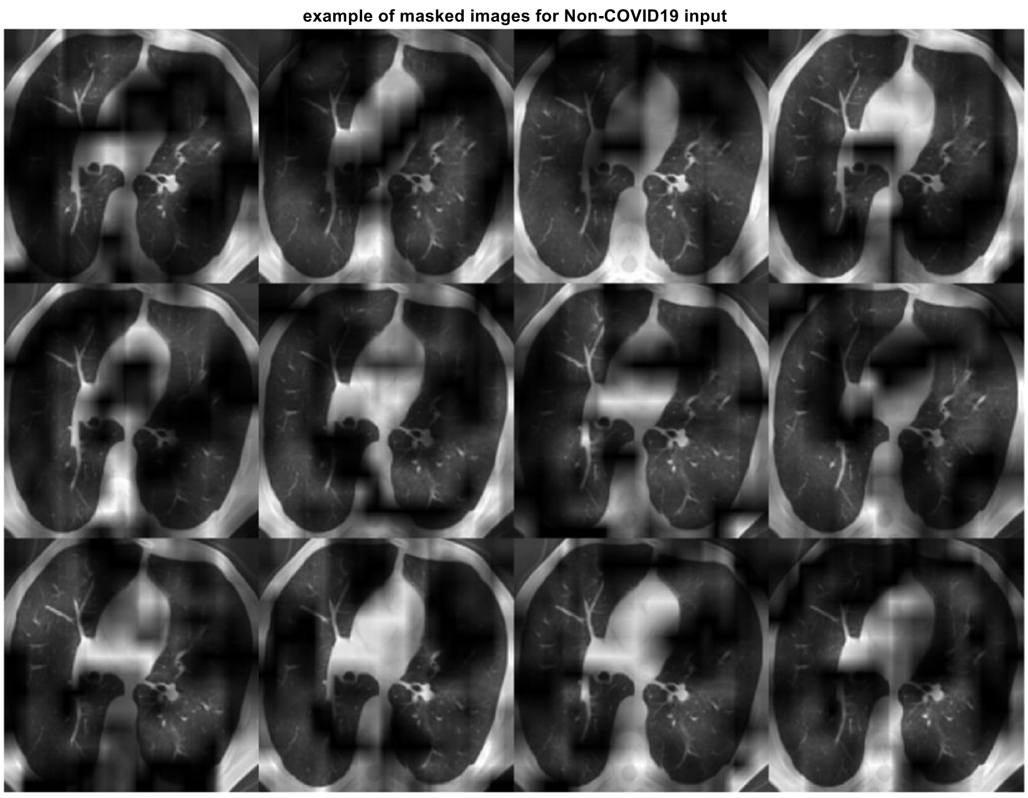
clinical practitioners. There has not been any real-world clinical validation of these models yet, even though they have done well on public datasets. Therefore, we plan to put our system through its paces in a clinical setting and talk to radiologists about how they can use it and what is their opinion about the models we have developed. As a result, our models will continue to improve in the future.

5 Conclusion

This paper investigated the efficiency of the SCA to determine the optimal structure of DCNNs without the need for manual effort to have speed and accuracy at the same time. Afterward, three improvements based on regular SCA were devised to reach the goal. First, a new encoding approach based on IP address was proposed; then, an enfeebled layer was proposed to cover specific candid solution vector dimensions. Finally, the learning process separated large datasets into smaller pieces. The proposed DCNN-IPSCA was investigated on the SARS-CoV-2 and COVID-CT datasets to have a detailed comparison. In this regard, the performance of the DCNN-IPSCA was evaluated by the standard DCNN, and DCNNs evolved by IPPSO, VLGA, VLNSGA-II, VLBSO, IP-MPSO, VLBBO, and VLACO in terms of five well-known metrics: sensitivity, accuracy, specificity, F1-score, precision, and ROC and precision-recall curves. The comparison study empirically proved that the presented system outperforms competing models, with a final accuracy of 98.32% on the SARS-CoV-2 and 98.01% on the COVID-CT dataset. Also, the proposed variable-length DCNN (DCNN-IPSCA) performed



a) COVID19



b) Non-COVID19

Fig. 17 Typical example of masked images for **a** COVID19 and **b** Non-COVID19 input

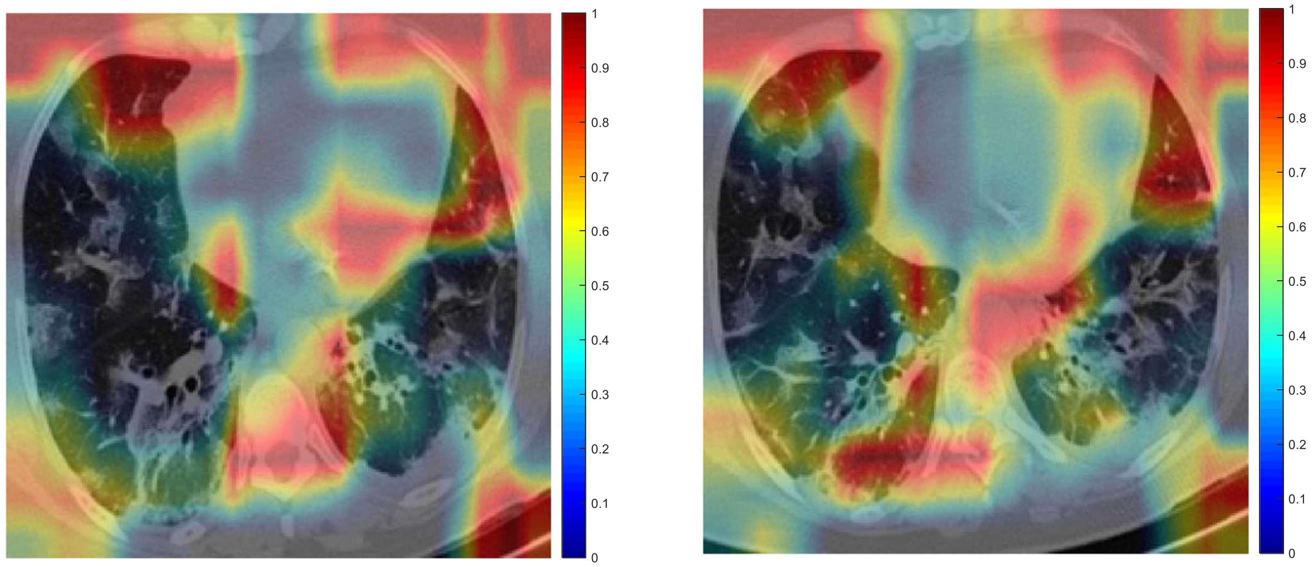


Fig. 18 The CAM demonstration results for the two typical COVID19 examples

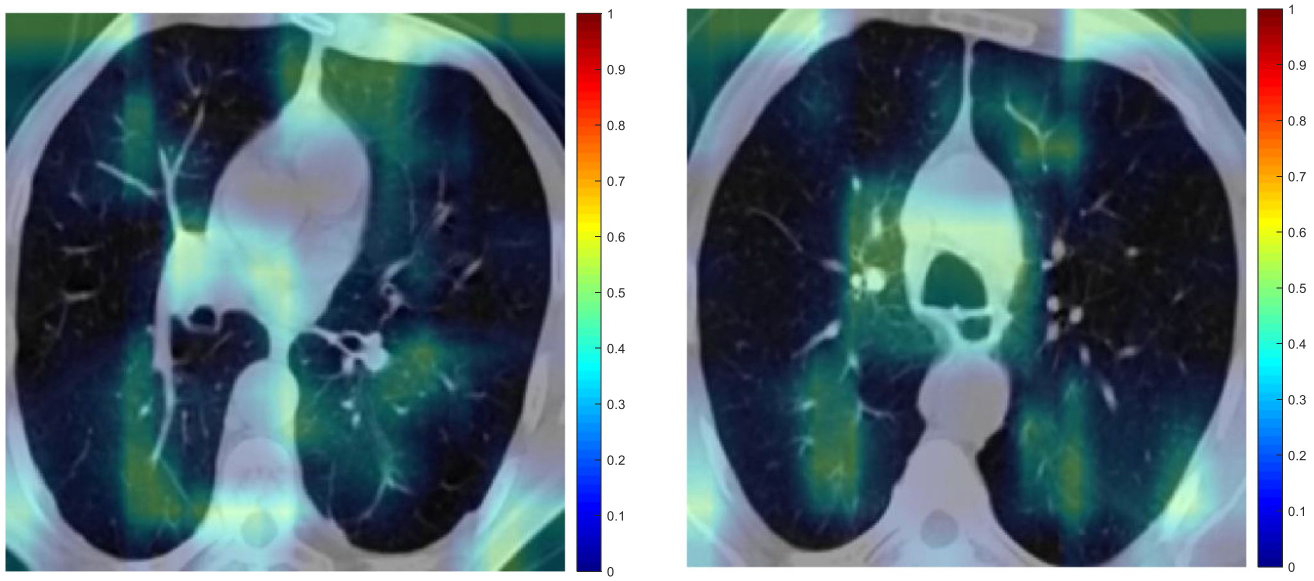


Fig. 19 The CAM demonstration results for the two typical non-COVID19 examples

much better than the standard DCNN, with GPU and CPU training times being 387.69 and 63.10 times faster, respectively. There are numerous opportunities for further research, including the use of DCNN-IPSCA in various image processing tasks. Researchers can examine IPSCA in order to solve challenges involving multi-objective optimization. Additionally, a unique fitness function can also be created to model the issue more appropriately. The usefulness of oppositional-based learning, chaotic maps, orthogonal learning, and Gaussian walks can be explored to optimize the DCNN-performance as further study directions.

Funding This work was supported by Professor Xiaogang Luo of Chongqing University.

Data availability The resource images can be downloaded using the following links: https://github.com/UCSD-AI4H/COVID-CT; http://medicalsegmentation.com/covid19/

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Panwar H, Gupta PK, Siddiqui MK, Morales-Menendez R, Singh V (2020) Application of deep learning for fast detection of COVID-19 in X-Rays using nCOVnet, *Chaos*. Solitons and Fractals. https://doi.org/10.1016/j.chaos.2020.109944
2. Zhang J, Xie Y, Li Y, Shen C, Xia Y (2020) COVID-19 screening on chest X-ray images using deep learning based anomaly detection, *aeXiv.2002.12338v1*
3. Han G et al (2019) Hybrid resampling and multi-feature fusion for automatic recognition of cavity imaging sign in lung CT. Futur. Gener. Comput. Syst. 99:558–570
4. Lella KK, Pja A (2022) Automatic diagnosis of COVID-19 disease using deep convolutional neural network with multi-feature channel from respiratory sound data: cough, voice, and breath. Alexandria Eng. J. 61(2):1319–1334
5. Khishe M, Caraffini F, Kuhn S (2021) Evolving deep learning convolutional neural networks for early COVID-19 detection in chest X-ray images. Mathematics 9(9):1002
6. Turkoglu M (2021) COVIDetectioNet: COVID-19 diagnosis system based on X-ray images using features selected from prelearned deep features ensemble. Appl. Intell. 51(3):1213–1226
7. Wang X, Gong C, Khishe M, Mohammadi M, Rashid TA (2021) Pulmonary diffuse airspace opacities diagnosis from chest X-ray images using deep convolutional neural networks fine-tuned by whale optimizer. Wirel Pers Commun. 1–20
8. Barshooi AH, Amirkhani A (2022) A novel data augmentation based on Gabor filter and convolutional deep learning for improving the classification of COVID-19 chest X-Ray images. Biomed. Signal Process. Control 72:103326
9. Mahmud T, Rahman MA, Fattah SA (2020) CovXNet: A multi-dilation convolutional neural network for automatic COVID-19 and other pneumonia detection from chest X-ray images with transferable multi-receptive feature optimization. Comput Biol Med. 122. https://doi.org/10.1016/j.compbiomed.2020.103869.
10. Shah V, Keniya R, Shridharani A, Punjabi M, Shah J, Mehendale N (2021) Diagnosis of COVID-19 using CT scan images and deep learning techniques. Emerg Radiol. 1–9
11. Dai W et al (2020) CT imaging and differential diagnosis of COVID-19. Can. Assoc. Radiol. J. 71(2):195–200
12. Afshar P et al (2021) COVID-CT-MD, COVID-19 computed tomography scan dataset applicable in machine learning and deep learning. Sci. Data 8(1):1–8
13. Song Q, Zhao L, Luo X, Dou X (2017) Using deep learning for classification of lung nodules on computed tomography images. J Healthc Eng. 2017
14. Gozes O et al (2020) Rapid AI development cycle for the coronavirus (covid-19) pandemic: initial results for automated detection & patient monitoring using deep learning CT image analysis, *arXiv Prepr. arXiv2003.05037*
15. Ai T et al (2020) Correlation of chest CT and RT-PCR testing in coronavirus disease 2019 (COVID-19) in China: a report of 1014 cases. Radiology, 296, E32-E40."
16. Liu Z et al (2020) Composition and divergence of coronavirus spike proteins and host ACE2 receptors predict potential intermediate hosts of SARS-CoV-2. J. Med. Virol. 92(6):595–601
17. Breban R, Riou J, Fontanet A (2013) Interhuman transmissibility of Middle East respiratory syndrome coronavirus: estimation of pandemic risk. Lancet 382(9893):694–699
18. Ozturk T, Talo M, Yildirim EA, Baloglu UB, Yildirim O, Rajendra Acharya U (2020) Automated detection of COVID-19 cases using deep neural networks with X-ray images. Comput Biol Med. https://doi.org/10.1016/j.compbiomed.2020.103792
19. Angelov P, Almeida Soares E (2020) SARS-CoV-2 CT-scan dataset: a large dataset of real patients CT scans for SARS-CoV-2 identification, *medRxiv*
20. He X et al (2020) Sample-efficient deep learning for covid-19 diagnosis based on ct scans, *MedRxiv*
21. Liu W, Wang Z, Liu X, Zeng N, Liu Y, Alsaadi FE (2017) A survey of deep neural network architectures and their applications. Neurocomputing 234:11–26
22. Suganuma M, Shirakawa S, Nagao T (2017) A genetic programming approach to designing convolutional neural network architectures, in Proceedings of the genetic and evolutionary computation conference. 497–504
23. Wu C, Khishe M, Mohammadi M, Karim SHT, Rashid TA (2021) Evolving deep convolutional neutral network by hybrid sine–cosine and extreme learning machine for real-time COVID19 diagnosis from X-ray images, Soft Comput. 1–20
24. Wang B, Sun Y, Xue B, Zhang M (2018) Evolving deep convolutional neural networks by variable-length particle swarm optimization for image classification, in. IEEE Congress on Evolutionary Computation (CEC) 2018:1–8
25. Hu T, Khishe M, Mohammadi M, Parvizi GR, Taher Karim SH, Rashid TA (2021) Real-time COVID-19 diagnosis from X-ray images using deep CNN and extreme learning machines stabilized by chimp optimization algorithm. Biomed Signal Process Control 68:102764. https://doi.org/10.1016/j.bspc.2021.102764
26. Stanley KO, Miikkulainen R (2002) Evolving neural networks through augmenting topologies. Evol. Comput. 10(2):99–127
27. Webb GI, Keogh E, Miikkulainen R, Miikkulainen R, Sebag M (2011) No-Free-Lunch theorem, in Encyclopedia of Machine Learning
28. Mirjalili S (2016) "SCA: a sine cosine algorithm for solving optimization problems. Knowledge-Based Syst. https://doi.org/10.1016/j.knosys.2015.12.022
29. LeCun Y (2015) LeNet-5, convolutional neural networks. 20(5):14. http://yann.lecun.com/exdb/lenet
30. Yang X, He X, Zhao J, Zhang Y, Zhang S, Xie P (2020) COVID-CT-dataset: a CT scan dataset about COVID-19
31. Perumal S, Velmurugan T (2018) Preprocessing by contrast enhancement techniques for medical images. Int. J. Pure Appl. Math. 118(18):3681–3688
32. Postel J (1980) DoD standard internet protocol. ACM SIGCOMM Comput. Commun. Rev. 10(4):12–51
33. Van Ouwerkerk JD (2006) Image super-resolution survey. Image Vis. Comput. 24(10):1039–1052
34. Qiongbing Z, Lixin D (2016) A new crossover mechanism for genetic algorithms with variable-length chromosomes for path optimization problems. Expert Syst. Appl. 60:183–189

35. Pal R, Chaudhuri TD, Mukhopadhyay S (2021) Portfolio formation and optimization with continuous realignment: a suggested method for choosing the best portfolio of stocks using variable length NSGA-II. Expert Syst. Appl. 186:115732
36 Cheng J, Chen J, Guo Y, Cheng S, Yang L, Zhang P (2021) Adaptive CCR-ELM with variable-length brain storm optimization algorithm for class-imbalance learning. Nat Comput 20(1):11–22
37. Abbas SA (2018) Internet protocol (IP) steganography using modified particle swarm optimization (MPSO) algorithm. Diyala J. Pure Sci. 14(03):220–236
38. Liao T, Socha K, de Oca MAM, Stützle T, Dorigo M (2013) Ant colony optimization for mixed-variable optimization problems. IEEE Trans. Evol. Comput. 18(4):503–518
39. Kwaśniewska A, Rumiński J, Rad P (2017) Deep features class activation map for thermal face detection and tracking, in 2017 10Th international conference on human system interactions (HSI), pp. 41–47