Published in final edited form as:
Biocell. 2023 ; 47(2): 373–384. doi:10.32604/biocell.2021.0xxx.

PSTCNN: Explainable COVID-19 diagnosis using PSO-guided self-tuning CNN

Wei Wang[#1], **Yanrong Pei**[#2], **Shui-Hua Wang**[1], **Juan manuel Gorrz**[3], **Yu-Dong Zhang**[1,*]

[1]School of Computing and Mathematical, University of Leicester, Leicester, LE1 7RH, UK

[2]Huai'an Tongji Hospital, Huai'an, Jiangsu 223000, China

[3]Department of Signal Theory, Networking and Communications, University of Granada, Granada, 52005, Spain

[#] These authors contributed equally to this work.

Abstract

Since 2019, the coronavirus disease-19 (COVID-19) has been spreading rapidly worldwide, posing an unignorable threat to the global economy and human health. It is a disease caused by severe acute respiratory syndrome coronavirus 2, a single-stranded RNA virus of the genus Betacoronavirus. This virus is highly infectious and relies on its angiotensin-converting enzyme 2-receptor to enter cells. With the increase in the number of confirmed COVID-19 diagnoses, the difficulty of diagnosis due to the lack of global healthcare resources becomes increasingly apparent. Deep learning-based computer-aided diagnosis models with high generalisability can effectively alleviate this pressure. Hyperparameter tuning is essential in training such models and significantly impacts their final performance and training speed. However, traditional hyperparameter tuning methods are usually time-consuming and unstable. To solve this issue, we introduce Particle Swarm Optimisation to build a PSO-guided Self-Tuning Convolution Neural Network (PSTCNN), allowing the model to tune hyperparameters automatically. Therefore, the proposed approach can reduce human involvement. Also, the optimisation algorithm can select the combination of hyperparameters in a targeted manner, thus stably achieving a solution closer to the global optimum. Experimentally, the PSTCNN can obtain quite excellent results, with a sensitivity of 93.65%±1.86%, a specificity of 94.32%±2.07%, a precision of 94.30%±2.04%, an accuracy of 93.99%±1.78%, an F1-score of 93.97%±1.78%, Matthews Correlation Coefficient of 87.99%±3.56%, and Fowlkes-Mallows Index of 93.97%±1.78%. Our experiments demonstrate

[*]Address correspondence to: Yu-Dong Zhang, yudongzhang@ieee.org.

Author Contribution

Conceptualization: Wei Wang, Yanrong Pei, Shui-Hua Wang; Methodology: Wei Wang, Yu-Dong Zhang; Software: Wei Wang, Yanrong Pei, Juan Manuel Gorriz; Validation: Wei Wang, Yanrong Pei, Shui-Hua Wang, Juan Manuel Gorriz, Yu-Dong Zhang; Investigation: Wei Wan, Shui-Hua Wang, Juan Manuel Gorriz; Writing - Original Draft: Wei Wang, Yanrong Pei, Yu-Dong Zhang; Writing – Review & Editing: Wei Wang, Yanrong Pei, Yu-Dong Zhang; Visualisation: Wei Wang, Yanrong Pei, Yu-Dong Zhang; Formal analysis: Shui-Hua Wang; Resources: Shui-Hua Wang, Juan Manuel Gorriz, Yu-Dong Zhang; Supervision: Shui-Hua Wang, Juan Manuel Gorriz, Yu-Dong Zhang; Project administration: Shui-Hua Wang, Juan Manuel Gorriz, Yu-Dong Zhang; Funding acquisition: Shui-Hua Wang, Juan Manuel Gorriz, Yu-Dong Zhang.

Ethics Approval

Not applicable.

Conflicts of Interest

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

Europe PMC Funders Author Manuscripts

that compared to traditional methods, hyperparameter tuning of the model using an optimisation algorithm is faster and more effective.

Keywords

COVID-19; SARS-CoV-2; Particle Swarm Optimisation; Convolutional Neural Network; Hyperparameters Tuning

Europe PMC Funders Author Manuscripts

Introduction

COVID-19 is a new global epidemic characterised by high infectivity and variability, posing a significant threat to human life and the global economy (Chakraborty and Maity 2020). The pathogen of COVID-19 is severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) (Hasöksüz, Kiliç et al. 2020), a single-stranded RNA virus of the genus Beta coronavirus and shares 79% nucleotide sequence identity with SARS-CoV, the causative agent of the 2002 SARS epidemic. Other viruses in the same genus include human coronavirus HCoV-HKu1, HCoV-OC43, and Middle East respiratory syndrome coronavirus (MERS-CoV). These coronavirus particles are composed of four structural proteins, including nucleocapsid (N), membrane (M), envelope (E), and spike (S) proteins. The S protein mediates the attachment and fusion of the coronavirus to the host cell membrane. The S protein consists of two non-covalently related subunits: (1) the S2 subunit anchors the S protein to the host cell membrane, and (2) the S1 subunit binds to its specific angiotensin-converting enzyme 2 (ACE2) receptor to enter the cell and infect patients (Jackson, Farzan et al. 2022). The new coronavirus is highly infectious. The number of confirmed COVID-19 diagnoses has been growing since first identified in 2019. As of April 2022, the cumulative number of confirmed COVID-19 diagnoses is close to five billion, with over six million deaths (Organization 2022). At the same time, the novel coronavirus continues to mutate at a much faster rate than the vaccine development. In addition, it has undergone significant changes in its characteristics, making it challenging to sustain efforts to combat the disease.

Symptoms of COVID-19 vary significantly among individuals with a continuous cough, fever, taste loss, and, in severe cases, death, thus making COVID-19 even more dangerous. In general, the spread of infectious diseases can be stopped by isolating the source of infection and blocking the transmission route. However, because of (1) the mutation of the new coronavirus, (2) the increase in the number of asymptomatic patients, and (3) the difficulties of diagnosis, isolating the source of COVID-19 infection becomes a challenge (Kronbichler, Kresse et al. 2020).

The most widely used test for COVID-19 is the reverse transcription polymerase chain reaction (RT-PCR) (van Kasteren, van der Veer et al. 2020), which has improved considerably in recent years, allowing results to be obtained within tens of minutes. However, these tests are often associated with high false negatives and a high potential for underdiagnosis (Arevalo-Rodriguez, Buitrago-Garcia et al. 2020). Moreover, the availability of RT-PCR reagents is highly challenging due to the high number of infections and

the population's fears of a global disease. Therefore, exploring more effective ways of diagnosing COVID-19 for outbreak control is essential.

COVID-19 is an infectious emergency respiratory disease, and the disease state is usually reflected in the lungs. X-ray and CT, the most common medical imaging techniques in modern medicine, can play a vital role in diagnosing COVID-19 by reflecting changes in the lung tissue through their chest impact. As a relatively new technology, computed tomography (CT) imaging allows multi-layered photography of the target area to form a three-dimensional image, providing multi-angle image data with a higher resolution than X-ray images (Berrimi, Hamdi et al. 2021). Therefore, the diagnosis of COVID-19 by CT imaging of the chest is significant research and practical value. However, the diagnosis of chest CT is often dependent on medical specialists, making it difficult to be helpful in the face of the shortage of medical resources associated with a COVID-19-like global epidemic.

As one of the most influential frontier technologies of the 20th century, artificial intelligence (AI) significantly impacts human work and life (Liang, On et al. 2021), especially in the medical field (Ning, Li et al. 2020, Han, Liu et al. 2022, Miller, Panneerselvam et al. 2022). Artificial intelligence-based computer-aided diagnosis (CAD) is one of the research areas receiving the most attention (Chen, Yang et al. 2022, Guan, Chen et al. 2022, Yu, Liu et al. 2022, Yu, Han et al. 2022). In the face of the global pandemic of COVID-19, many researchers have risen to fight against it, hoping to contribute to its eradication. As a result, many studies on COVID-19 are rapidly emerging (BOBERMIN, MEDEIROS et al. 2022, CARRETTA, DOMENICO et al. 2022, OZTURK and KARA 2022), especially on COVID-19 CAD systems.

Chen (2021) used the effectiveness of the greyscale co-occurrence matrix (GLCM) for texture feature extraction to extract features from chest CT images and a support vector machine (SVM) to perform binary classification on these features to achieve the diagnosis of COVID-19. Pi and Lima (2021) and Pi (2021) also used GLCM as a feature extraction method. Among them, Pi and Lima (2021) used the extreme learning machine (ELM) as a classifier to classify the features extracted from chest CT by GLCM, and Pi (2021) used the Schmitt Neural Network as the classifier of the model. These methods achieved promising performance, which demonstrates the effectiveness of GLCM for feature extraction from chest CT images. Some methods extract features using wavelet entropy. The study by Wang (2021) used Jaya algorithm-based training algorithm to train the model and achieved an encouraging performance. A further attempt by Wang, Zhang et al. (2022) to use the Self-Adaptive Jaya (SAJ) algorithm to train their model. Their model (WE-SAJ) has achieved further performance improvements. On the other hand, Field Wu (Wu 2020) used a particular wavelet entropy, Wavelet Renyi Entropy, as the feature extraction method and a three-stage biogeographic optimisation algorithm as the training algorithm, achieving excellent performance. Khan (2021) proposed a novel deep-learning approach to construct a diagnostic tool for COVID-19. They used Pseudo-Zernike moment derived from Zernike moment as feature values and deep sparse autoencoder as the classifier for chest CT image classification. Their model obtained an accuracy higher than 90% and achieved similarly excellent performance in other performance metrics. Some studies have also used models based on classical deep-learning methods to diagnose COVID-19. Hou and Han (2022)

used a six-layer convolutional neural network (CNN) for chest CT images based COVID-19 diagnosis task. Their model has achieved about 90% accuracy. Yu, Wang et al. (2020) improved one of the most classical CNNs, GoogleNet, for the COVID-19 chest CT image-based diagnosis task. They replaced the last two layers of GoogleNet with a dropout layer, two fully connected layers, and an output layer. Their model also achieved close to 90% accuracy. Zhang, Zhang et al. (2021) proposed a multi-input deep convolutional attention network constructed using a convolutional block attention module. Their model has achieved more than 90% accuracy. These models are well-designed but require hyperparameter tuning to achieve fast convergence and high performance, often requiring expert intervention. At the same time, manual tuning of hyperparameters often relies on empirical knowledge, which is subjective and requires a high level of expertise. These factors have primarily hindered the cross-domain generalisation of CAD methods. We believe an optimisation algorithm could be a solution to make the hyperparameter tuning process automatic.

This paper uses the particle swarm optimisation algorithm (PSO) to optimise the three hyperparameters and gradient-based localisation of CNN to generate visual explanations. The proposed approach uses particle swarm optimisation algorithms to perform auto hyperparameters tuning, reducing the dependence of model construction on machine learning experts. In addition, PSO purposefully finds the hyperparameters closest to the optimal solution more consistently. Our method has achieved a promising performance in COVID-19 diagnosis.

Our contributions to this study are i) we experimentally demonstrated the possibility of using optimisation algorithms for hyperparameter tuning, ii) we proposed a high-performance COVID-19 diagnostic method with a visual explanation based on CT chest images, and iii) we further explored the potential of AI-based techniques in medical image processing. In the rest of the paper, Section 2 introduces the data set used, Section 3 introduces the background of the methods involved in the experiment, Section 4 describes the experimental workflow of PSO-guided self-tuning CNN (PSTCNN), and Section 5 presents the experiment results and discusses it in detail.

Dataset

The experiment used a publicly available chest CT image slice dataset proposed by (Zhang, Lu et al. 2022). The dataset was a binary classification dataset with the categories positive (COVID- 19) and negative (Health Control). The dataset contains a total of 296 slice images. The positive category contains 148 slice images taken from chest CT images of 66 COVID-19-infected subjects, assessed as positive by nucleic acid test. The negative category contains 148 slice images taken from chest CT images of 66 uninfected subjects from COVID-19. These subjects included 77 males and 55 females. The detailed statistic of the dataset is shown in TABLE 1, and two sample images are illustrated in Fig. FIGURE 1.

Europe PMC Funders Author Manuscripts

Methodology

10-Fold Cross Validation

The dataset used for the study was small, so we introduced the 10-Fold Cross Validation to train and evaluate the model. Specifically, we divided the dataset into ten groups and performed ten runs, with different groups selected as the test set and all other groups as the training set for each run. The model was thoroughly trained and evaluated for each run to obtain performance metric values. The final performance of the model was obtained by calculating the sum, the mean and standard deviation (MSD) of all ten sets of performance metric values. This approach allowed for efficient use of the samples in the dataset and effectively avoided overfitting.

Feature Learning Through Convolutional Neural Network

CNN is one of the trendiest research directions in computer-aided diagnosis tasks. It comprises different network layers, e.g., the input layer, convolutional layer, activation layer, pooling layer, and output layer. By combining these network layers, CNNs can effectively solve the problems of spatial information loss when expanding images into vectors, inefficiency training and network overfitting caused by high parameter volume when processing large images with fully connected neural networks (Gu, Wang et al. 2018).

Many existing deep learning-based CAD methods are based on CNNs. Polsinelli, Cinque et al. (2020) have made a simple modification to SqueezeNet by adding a batch normalisation layer between the squeeze convolution layer and the activation layer and replacing the original rectifier linear unit (ReLU) function with an exponential linear unit (ELU) function to build a lightweight CNN model. They tested the unmodified SqueezeNet and their proposed modified model separately using a COVID-19 chest CT dataset. Their proposed modification effectively improved the performance of the model. Horry, Chakraborty et al. (2020) used pre-trained VGG-19 based on transfer learning techniques to classify each of the three COVID-19 datasets, x-ray, CT, and ultrasound, and achieved promising results (precision up to 86% on x-ray, 100% on ultrasound and 84% on CT scans). Ismael and Şengür (2021) tested a variety of CNN models based on the COVID-19 x-ray dataset, including ResNet18 (accuracy of 88.42%), ResNet50 (accuracy of 92.63%), ResNet101 (accuracy of 87.37%), VGG16 (accuracy of 85.26%), and VGG19 (accuracy of 89.47%). Their validation of multiple pre-trained CNN models with the excellent performance of their custom CNNs further demonstrated the effectiveness of deep learning techniques for the COVID-19 diagnostic task.

Our method uses a five-layer neural network, including three convolution layers and two fully connected layers.

Convolutional layers are based on the concept of convolution. It is the core component of CNNs and generates most of the computation in the network. A convolutional layer contains a number of learnable filters (kernels), where the kernels are usually squares with smaller widths and lengths but the same depth as the input image. The convolution is the process by which the kernel slides over the image. The sliding direction is from left to right for the width direction and then repeats sliding following the width direction from the top

rows to the bottom rows of the image until reaching the bottom edge of the image. For each step of the sliding process of the convolution, each pixel of the region mapped by the kernel in the input image is multiplied by the information at the location corresponding to the flipped kernel. All the results generated are summed to aggregate the information. The region mapped by the kernel in the input image is called the sliding window. The step size of the sliding, $\boldsymbol{S} = (S_{\mathrm{l}}, S_{\mathrm{w}})$, is a customisable hyperparameter, in which S_{l} represents the step size of the sliding across the length direction and S_{w} represents the step size of the sliding across the width direction.

In this process, the filters scan across the input image, so every neighbourhood of the image is processed by the same filter. This sharing weight feature reduces the number of parameters, thus reducing computational costs and preventing overfitting due to too many parameters. Assume the size of kernel $\boldsymbol{K}$ is $F \times F$, and the size of a two-dimension single-channel image $\boldsymbol{I}$ is $N_{\mathrm{l}} \times N_{\mathrm{w}}$, where N_{l} represents the length of $\boldsymbol{I}$, and N_{w} represents the width of $\boldsymbol{I}$.

The output $o_{(m,n)}$ of the sliding process of the convolution when the sliding window centred at (m, n) can be defined as presented in Equation (1). Let $\boldsymbol{O}$ be the output of the convolution, $o \in \boldsymbol{O}$. Repeating the calculation to slide throughout the entire image will generate the output $\boldsymbol{O}$ of the convolution. If there are residuals (the sliding window cannot reach the edge of the image), the residual image pixels are abandoned. The output shape, $\boldsymbol{Z} = (Z_{\mathrm{l}}, Z_{\mathrm{w}}, Z_{\mathrm{c}})$, of a convolutional layer can be calculated by Equation (2).

$$o_{(m,n)} = \sum_{a-0}^{F-1} \sum_{b=0}^{F-1} \boldsymbol{K}[a,b] \cdot \boldsymbol{I}\left[n + \left\lfloor \frac{F}{2} \right\rfloor - a, m + \left\lfloor \frac{F}{2} \right\rfloor - b\right], \tag{1}$$

where (n, m) is the coordinate of the pixel point corresponding to the position of the centre of the sliding window in the input image. The range of $n \in \mathbb{N}$ is $\left(\frac{F}{2}, N_{\mathrm{w}} - \frac{F}{2} - 1\right)$, and the range of $m \in \mathbb{N}$ is $\left(\frac{F}{2}, N_{\mathrm{l}} - \frac{F}{2} - 1\right)$.

$$\boldsymbol{Z} = (Z_{\mathrm{l}}, Z_{\mathrm{w}}, Z_{\mathrm{c}}) = \left(\left\lfloor \frac{N_{\mathrm{l}} - F}{s_{\mathrm{l}}} \right\rfloor + 1, \left\lfloor \frac{N_{\mathrm{w}} - F}{s_{\mathrm{w}}} \right\rfloor + 1, Z_{\mathrm{c}}\right), \tag{2}$$

where Z_{l}, Z_{w}, and Z_{c} represent the length, width, and the number of channels of the output $\boldsymbol{O}$, respectively. N_{l} and N_{w} represent the length and width of the input image, respectively. $F \times F$ is the size of the kernel.

Activation functions play an essential role in CNNs, bringing a non-linear factor to the neural network, enhancing its expressive power, and thus improving the final classification performance. Working with high-dimensional input data such as images would be computationally prohibitive for each neuron in a network layer to fully connect to all the neurons in the previous layer (Najafabadi, Villanustre et al. 2015). Therefore, neurons in a CNN are usually related to only one local region of the input data. The spatial size of this connected region is called the neuron's receptive field, which depends on the kernel's size and is a hyperparameter. Also, to give the network the ability to handle non-linear tasks, the convolutional layer often includes an activation function as a non-linear factor. The most

common activation function is the ReLU function. In addition, the ReLU activation function can be used to construct sparse matrices to remove redundancy from the data and retain the maximum possible features of the data. The formula for the ReLU function is shown in Equation (3).

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{cases} \boldsymbol{x}, & \boldsymbol{x} > \boldsymbol{0}, \\ \boldsymbol{0}, & \text{otherwise.} \end{cases} \tag{3}$$

Applying Equation (3) to every pixel of $\boldsymbol{O}$ generates $\boldsymbol{O}'$, the output of the ReLU activation layer, as shown in Equation (4).

$$\boldsymbol{O}'(a, b) = f(\boldsymbol{O}(a, b)), \tag{4}$$

where (a, b) is the coordinate of pixels in $\boldsymbol{O}$, in which $0 \le a < O_w$, $a \in \mathbb{N}$, $0 \le b < O_l$, $b \in \mathbb{N}$. Each pixel in $\boldsymbol{O}'$ corresponds to a pixel in $\boldsymbol{O}$ but with a new value generated by the ReLU function with an input of the pixel's value in $\boldsymbol{O}$. $\boldsymbol{O}'$ has the same size and shape as $\boldsymbol{O}$.

Padding can regulate the size of the output of the network layer. In the convolution process, pixels at the edges of input images are never located in the centre of the kernel. These pixels are used far less than the pixels in the centre of the image, resulting in a significant loss of information at the image boundaries. In addition, the output image from the convolution process often does not maintain the same size as the input image, and different kernel sizes result in various degrees of image shrinkage. Padding is designed to address this issue, which has two modes, VALID mode and SAME mode.

In the VALID mode, padding does not perform any operations, and convolution performs a basic convolution operation, where the output image size is smaller than the input image. In the SAME mode, additional pixels are padded around the input image according to the kernel size (the padding value is usually 0). It allows the kernel to extend beyond the original image boundaries, thus allowing the output image to remain the same size as the original and avoiding losing information from the edges of the input image. FIGURE 2 illustrates convolution samples under VALID and SAME padding, respectively. Our method uses the SAME padding to preserve the image edge information, keeping the same size as the original image.

Training Algorithm

Training algorithms can determine the way how neural networks run. The essence of deep learning is to (1) take the loss function as the objective function, (2) input a large amount of data, (3) calculate the value of the objective function, and then (4) adjust and optimise the learnable parameters in the model to obtain the model that will give the closest result to the true value. In this process, the optimiser algorithm was followed. The choice of optimiser plays a vital role in deep learning training, as it affects the speed of convergence of the model and the final performance achieved.

Adam (Kingma and Ba 2014) is one of the most popular deep learning training algorithms that control the updating of model parameters. For each parameter, Adam uses the

hyperparameters β_1 and β_2 to calculate the exponential decay rates of the past gradient g_t and the square of the past gradient g_t^2, respectively, d_g and d_{gs}, using first-order moment estimates and second-order moment estimates of the gradient to dynamically adjust the learning rate of each parameter. The calculation is shown in Equation (5). As d_g and d_{gs} are initialised to 0, their values are biased towards 0. A bias correction is introduced in Adam to bias d_g and d_{gs} to obtain $\widehat{d_g}$ and $\widehat{d_{gs}}$, and further offset these initial biases. The bias correction is calculated as shown in Equation (6). Ultimately, Adam calculates the update step through the d_g and d_{gs} angles to update the parameters, as shown in Equation (7).

$$\begin{aligned} d_g^t &= \beta_1 d_g^{t-1} + (1-\beta_1) g_t, \\ d_{gs}^t &= \beta_2 d_{gs}^{t-1} + (1-\beta_2)(g_t)^2, \end{aligned} \tag{5}$$

$$\begin{aligned} \widehat{d_g} &= \frac{d_g}{1-(\beta_1)^t}, \\ \widehat{d_{gs}} &= \frac{d_{gs}}{1-(\beta_2)^t}, \end{aligned} \tag{6}$$

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\widehat{d_{gs}} + \in}} \widehat{d_g}. \tag{7}$$

Adam can adaptively adjust the learning rate of the model parameter updates, implement the step annealing process naturally, and makes the model parameter updates independent of the gradient scaling transformation. In addition, Adam has the advantages of high computational efficiency and low memory consumption (Saad Hikmat and Adnan Mohsin 2021). In simple words, Adam can achieve typically high levels of robustness and performance in various situations (Dogo, Afolabi et al. 2018). Therefore, our method uses Adam as the training algorithm.

Visual Explanation via Gradient-based Localisation (Grad-CAM)

Grad-CAM is a method for visualising the basis of CNN decisions. It uses a heat map to mark how much attention the neural network pays to different regions when classifying data, thus highlighting the regions on which the neural network focuses its attention. In detail, Grad-CAM uses the global average of the gradients to calculate the weight a_k^c of the feature map k corresponding to the class c (as shown in Equation (8). then, it calculates the weighted sum of the weight and the last layer of the feature map, combining it with the ReLU activation function to produce the heat map L_{GradCAM}^c. The procedure for calculating the heat map is shown in Equation (9).

$$a_k^c = \frac{1}{z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}, \tag{8}$$

where z is the number of pixels in the feature map, y^c is the fraction corresponding to class c, and A_{ij}^k denotes the pixel values at the coordinate (i, j) of the k feature maps.

$$L_{\text{GradCAM}}^c = f\left(\sum_k a_k^c A^k\right), \tag{9}$$

where f represents the ReLU function.

Hyperparameter Tuning

Almost all deep learning optimisers have customisable hyperparameters that can significantly influence the performance of the optimiser, hence the speed of convergence and the ultimate performance of the model. Many studies try to optimise hyperparameter tuning for better COVID-19 diagnostics. Ezzat, Hassanien et al. (2021) optimised the three hyperparameters OF DenseNet121, the batch size, the rate of dropout layer, and the number of neurons of the first dense layer and trained their proposed model using two sets of X-ray data from COVID-19. They could achieve 98.38% accuracy, a significant improvement over DenseNet121(94% accuracy) and Inception-v3(95% accuracy), which they used as controls. Monshi, Poon et al. (2021) tested the function loss, the number of epochs, and the batch size hyperparameters with different value combinations. Then, they selected the combinations of hyperparameters with the highest performance and used these combinations for the different models to test performances. In their approach, the accuracy of VGG19 improved by 11.93% and that of ResNet-50 by 4.97%. Kiziloluk and Sert (2022) used the gradient-based optimiser (GBO) and the Quasi-Newton algorithm (Q-N) to optimise the hyperparameters of several CNN models, including Alexnet, Darknet-19, Inception-v3, MobileNet, Resnet-18, and ShuffleNet. Their results show that GBO improves the classification performance of the original CNN model by 6.22-13.29%, and the Q-N algorithm improves the performance of the original CNN model by 2.92% to 8.40%. These studies demonstrate the critical impact of hyperparameter optimisation on the performance of CNN models in COVID-19 diagnostic tasks.

However, the hyperparameters can have an infinite number of possible combinations. Therefore, hyperparameter tuning becomes a challenging, time-consuming and computationally expensive stage in training deep learning models. There is no straightforward and efficient way to accurately and quickly find the optimal hyperparameters. The most commonly used hyperparameter selection methods are Random Search and Grid Search.

Grid Search—tries all the candidate hyperparameter combinations by loop traversal. Then it calculates the performance of the model under all the hyperparameter combinations and selects the parameter combination with the highest performance in the solution. However, the combination of the hyperparameters can be nearly infinite, so it is difficult to traverse all the possibilities in practical application and requires enormous time cost and calculation costs, which is inefficient.

Random Search—computes a neural network with a configuration of candidate parameters by randomly sampling the parameter space and stops the search when the

Europe PMC Funders Author Manuscripts

maximum number of iterations is completed, selecting the combination of hyperparameters with the highest performance. Although random search can be optimised to prevent repeated calculations of the same hyperparameter combinations, the search process is too random. Therefore, the large number of hyperparameter combinations makes it difficult to ensure that optimal hyperparameters are in the search range. As a result, the performance of the final model derived from random search can vary considerably for the same number of iterations.

Most hyperparameter tuning methods require many aimless attempts to find the most suitable hyperparameters, which are inefficient and ineffective, resulting in high consumption of time and computational resources. Optimisation algorithms could be a solution to this issue. This paper aims to discover the possibility of PSO in hyperparameter tuning. TABLE 2 discusses the advantages and disadvantages of PSO and traditional hyperparameter tuning methods from a theoretical perspective. Then, Section TABLE 2 introduces the experiment design of PSO-guided hyperparameter tuning.

Particle Swarm Optimisation-guided Self-Tuning CNN

The Original Particle Swarm Optimisation

To discover more possibilities of optimisation algorithm-based hyperparameter tuning. Our experiments employed the PSO algorithm to adjust the three hyperparameters in the Adam optimiser. PSO (Kennedy and Eberhart 1995) is an easy-to-understand and easy-to-implement optimisation algorithm with global solid search capability. Because of these advantages, it is one of the most typical optimisation algorithms. The core idea of PSO is to keep all particles moving and update their position to find the optimal solution through collaboration and information sharing among particles in the population. Hyperparameter tuning involves purposefully trying out different hyperparameter configurations and calculating the corresponding performance to find the optimal combination of hyperparameters.

Encoding Scheme of Particle Properties

In PSO, a swarm represents the collection of all particles. Each particle p_i in the swarm has two properties, a vector of velocity $\boldsymbol{V}_i = (\mathscr{V}_{i1}, \mathscr{V}_{i2}, \ldots, \mathscr{V}_{in})$ represents how fast a particle is moving, and a vector of position $\boldsymbol{X}_i = (\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{in})$ represents the direction in which a particle is moving. The algorithm finds the best position of a particle, $\boldsymbol{x}_{\text{pbest}}$, and the best position of the swarm, $\boldsymbol{x}_{\text{gbest}}$. In each iteration, the particles update their velocities in a direction for moving to new positions closer to $\boldsymbol{x}_{\text{pbest}}$ and $\boldsymbol{x}_{\text{gbest}}$.

The three essential variables of the Adam training algorithm to tune to obtain the highest performance of the model (as shown in Equation (5)) are the learning rate α, coefficient β_1, which controls the exponential decay rates of the past gradient g_t, and coefficient β_2, which controls the exponential decay rates of the square of the past gradient g_t^2. The first step of our experiment is to encode these three variables into the PSO algorithm to perform tuning. Assume an n dimension search space, each particle p_i has n positions $(\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{in})$ and n velocities $(\mathscr{V}_{i1}, \mathscr{V}_{i2}, \ldots, \mathscr{V}_{in})$. For every position $\mathscr{V}_{in}$ stores a set of hyperparameter combinations and is encoded as a vector $(\alpha_{in}, \beta_{1in}, \beta_{2in})$. For every velocity, $\mathscr{V}_{in}$ is encoded

as a vector ($\mathscr{V}_{1in}, \mathscr{V}_{2in}, \mathscr{V}_{3in}$), in which, $\mathscr{V}_{1in}$, $\mathscr{V}_{2in}$, and $\mathscr{V}_{3in}$ are for movements of, α_{in}, β_{1in} and β_{2in}, respectively. FIGURE 3 illustrates the encoding scheme.

Calculations of Velocities and Positions

At the beginning of PSO, all particles are initialised with random positions (hyperparameter configurations) and random velocity vectors. The CNN runs iteratively in each iteration with different hyperparameter configurations. It calculates the mean squared errors (MSE) for every hyperparameter configuration as its fitness value ($\mathscr{F}$), according to which it finds the personal best hyperparameter configuration, $\boldsymbol{x}_{\text{pbest}}$, for each particle, and the global best hyperparameter configuration, $\boldsymbol{x}_{\text{gbest}}$, of the entire particle population, which is the current best hyperparameter configuration. The formulates of finding $\boldsymbol{x}_{\text{pbest}}$ and $\boldsymbol{x}_{\text{gbest}}$ are shown in Equation (10) and Equation (11), respectively, where $\boldsymbol{x}_{\text{cur}}$ represents the current hyperparameter configuration.

$$\boldsymbol{x}_{\text{pbest}}^{k+1} = \begin{cases} \boldsymbol{x}_{\text{cur}}, & \text{if } \mathscr{F}(\boldsymbol{x}_{\text{cur}}) < \mathscr{F}\left(\boldsymbol{x}_{\text{pbest}}^{k}\right), \\ \boldsymbol{x}_{\text{pbest}}^{k}, & \text{otherwise.} \end{cases} \tag{10}$$

$$\boldsymbol{x}_{\text{pbest}}^{k+1} = \begin{cases} \boldsymbol{x}_{\text{pbest}}^{k+1}, & \text{if } \mathscr{F}\left(\boldsymbol{x}_{\text{pbest}}^{k+1}\right) < \mathscr{F}\left(\boldsymbol{x}_{\text{gbest}}^{k}\right), \\ \boldsymbol{x}_{\text{gbest}}^{k}, & \text{otherwise,} \end{cases} \tag{11}$$

where k represents iteration count. At the end of each iteration, the velocities of all particles are iteratively updated to directions that are closer to $\boldsymbol{x}_{\text{pbest}}$ and $\boldsymbol{x}_{\text{gbest}}$, following Equation (12). Also, the hyperparameter configurations of each particle are updated to newer values according to the new velocities, following Equation (13).

$$\mathscr{V}_{ij}^{k+1} = \mathscr{V}_{ij}^{k} + c_p r_p\left(\boldsymbol{x}_{i\text{pbest}}^{k} - \boldsymbol{x}_{ij}^{k}\right) + c_g r_g\left(\boldsymbol{x}_{\text{gbest}}^{k} - \boldsymbol{x}_{ij}^{k}\right), \tag{12}$$

where k represents iteration count, c_p refers to the cognitive acceleration coefficient, c_g refers to the social acceleration coefficient, r_p and r_g are random numbers.

$$\boldsymbol{x}_{ij}^{k+1} = \boldsymbol{x}_{ij}^{k} + \mathscr{V}_{ij}^{k+1}. \tag{13}$$

Repeating the above steps, all particles keep moving to hyperparameter configurations that can obtain better performance until the algorithm reaches the maximum number of iterations.

Structure of Particle Swarm Optimisation-guided Self-Tuning CNN

In our research, a number of experiments were performed in incremental steps to find out the neural network structure with the best performance. The final neural network is a five-layer CNN consisting of three convolution layers and two fully connected layers. A softmax function is introduced to classify the extracted features. All trainable parameters are updated following the Adam optimiser during the training process. In the tuning process, the neural networks are trained with a number of hyperparameter combinations generated or updated

by PSO to obtain the performance of different combinations. Finally, the output of PSO is the hyperparameter combination of the final model with the best performance. FIGURE 4 illustrates the overall structure of PSO-guided Self-Tuning CNN (PSTCNN). Theoretically, the framework can be generalised to other CNNs for image classification tasks in different domains.

Results and Discussion

The following values were used for various performance indicators to evaluate the model's performance comprehensively. (1) True Positive (TP) represents the number of positive samples that the model correctly predicts as the positive class, (2) True Negative (TN) represents the number of negative samples that the model correctly predicts as the negative class, (3) False Positive (FP) represents the number of negative samples that the model incorrectly predicted as the positive class, and (4) False Negative (FN) represents the number of positive samples that the model incorrectly predicts as the negative class.

The seven performance metrics used to assess the model's performance are accuracy, precision, sensitivity, specificity, F1-score, Matthews correlation coefficient, and the Fowlkes-Mallows index, which provide a comprehensive evaluation of the model from a variety of perspectives to ensure a comprehensive performance evaluation.

Accuracy is one of the most common metrics used to evaluate the performance of a model. The core idea is to calculate the number of correct predictions as a percentage of the total number of samples, covering both positive and negative samples. The formula for accuracy is shown in Equation (14).

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} . \tag{14}$$

Although accuracy can assess the overall performance of a model with a dataset containing both positive and negative samples, it is not rigorous for an unbalanced dataset. For example, suppose there is a dataset with 90% positive samples and only 10% negative samples. A model that predicts all samples as positive can achieve 90% accuracy, but this is not an accurate representation of the model's performance. In short, accuracy is not an effective way to evaluate the predictive performance of a model for the positive and negative samples separately, so three performance metrics, precision, sensitivity, and specificity, were introduced to provide a more comprehensive evaluation of the model.

Precision evaluates model performance primarily based on prediction results by calculating the number of samples correctly predicted as positive as a proportion of all samples predicted as positive to assess the probability that the model is correctly predicted in the samples where the model is predicted as positive. The precision of a model increases as the FP decreases, which can be a guide for finding the lowest FP. The formula of precision is shown in Equation (15).

Europe PMC Funders Author Manuscripts

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}. \quad (15)$$

Specificity is another metric that can be used to measure the performance of a model for positive samples. The difference is that specificity is calculated based on the true labels of the data rather than the model predictions, and the 'performance of the model is assessed by calculating the number of samples predicted to be negative as a proportion of the total number of negative samples in the dataset. The specificity of a model also increases as the FP decreases and is calculated as shown in Equation (16).

$$\text{Specificity} = \frac{\text{TP}}{\text{TP+FP}}. \quad (16)$$

Sensitivity is also calculated based on the true label of the data, except that it evaluates the model's performance by calculating the proportion of samples predicted to be positive to the total number of positive samples in the data set. The phenomenon that sensitivity reflects is somewhat the opposite of precision. It increases as FN decreases, which can guide finding the lowest FN. Its formula is shown in Equation (17).

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP+FN}}. \quad (17)$$

F1-score is the performance metric that considers both precision and sensitivity, tries to find the balance between these two metrics, and simultaneously makes them the highest possible values. The calculation of the F1-score is as shown in Equation (18).

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Sensitivity}}{\text{Precision+Sensitivity}}. \quad (18)$$

Matthews Correlation Coefficient (MCC) compensates that the four elements TP, TN, FP, and FN are not fully considered in the abovementioned metrics. It considers the true and predicted values as two variables and calculates the correlation coefficient. The higher the correlation between the true and predicted values, the better the model performance. An MCC value of 1 indicates a perfect positive correlation between the predicted and true results (FP = FN = 0), meaning that the model performs perfectly. An MCC value of -1 indicates a negative correlation between the predicted and true results (TP = TN = 0), meaning that the model predicts the exact opposite of the true results. An MCC of 0 means that the classifier cannot provide meaningful results. The calculation of MCC is shown in Equation (19).

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP+FP})(\text{TP+FN})(\text{TN+FP})(\text{TN+FN})}}. \quad (19)$$

Europe PMC Funders Author Manuscripts

The Fowlkes-Mallows Index (FMI) is a performance metric that considers both precision and sensitivity. Its calculation is as shown in Equation (20). When FMI is 0, the value of TP is 0, which means that the model will mispredict all positive samples, and the model is not considered a valid classifier. On the other hand, if FMI is 1, the model is deemed to have perfect classification ability.

$$\mathrm{FMI} = \sqrt{\mathrm{precision} \times \mathrm{sensitivity}} = \sqrt{\frac{\mathrm{TP}}{\mathrm{TP+FP}} \times \frac{\mathrm{TP}}{\mathrm{TP+FN}}}, \tag{20}$$

Area Under Curve (AUC) is an important performance metric for evaluating binary classification models and is derived by calculating the area under the receiver operating characteristic (ROC) curve. The vertical and horizontal axes are true positive rate (TPR) = sensitivity and false positive rate (FPR) = specificity. The ROC curves were obtained by traversing m thresholds used to differentiate the classification results to obtain different TPR and FPR values. The more significant the decrease in FPR with increasing TPR in the curve (i.e., the steeper the ROC curve), the better the model performance. As the AUC is calculated from the area under the ROC curve, the higher the AUC, the better the model performance. As both TPR and FPR are considered, the ROC curve has the excellent quality of not changing with sample proportion, and the AUC inherits this same advantage. The AUC is calculated as shown in Equation (21), where **TPR** and **FPR** represent a set of TPRs and a set of FPRs, respectively.

$$\mathrm{AUC} = \frac{1}{2}\sum_{i=1}^{m-1}(\mathbf{TPR}_i + \mathbf{TPR}_{i-1}) \times (\mathbf{FPR}_i - \mathbf{FPR}_{i-1}). \tag{21}$$

Statistical Analysis

To minimise the bias in the model performance evaluation results, we used the 10-fold cross-validation to test the model's performance under the optimal hyperparameter configuration obtained by the optimisation algorithm. As a result, we obtained a sensitivity (Sen) of 93.65%±1.86%, a specificity (Spc) of 94.32%±2.07%, a precision (Prc) of 94.30%±2.04%, an accuracy (Acc) of 93.99%±1.78%, an F1-score (F1) of 93.97%±1.78%, Matthews correlation coefficient (MCC) of 87.99%±3.56%, and Fowlkes-Mallows index (FMI) of 93.97%±1.78%. The 10-runs 10-fold cross-validation results are shown in TABLE 3.

Visual Explanation

FIGURE 5 illustrates three sample heatmaps generated from ten runs using Grad-CAM. The annotated parts are the basis of CNN decisions (the warmer colour and higher attention level), which correspond to COVID-19 lung infection areas. These reflect, to some extent, the soundness of the decision basis of the model.

Receiver Operating Characteristic Curve & Area Under Curve

FIGURE 6 illustrates the ROC curve and AUC of PSTCNN. Each of the points in the figure corresponds to a threshold value. The value of the AUC can be obtained by calculating the area under the curve formed by connecting these points. The figure shows that the PSTCNN

Europe PMC Funders Author Manuscripts

can achieve a high TPR with a low FPR and an AUC close to 0.96. These results indicate a very promising performance of the model.

Comparison to the State-of-the-Art (SOTA)

FIGURE 7 illustrates the performance comparison between our method (PSTCNN) and the deep learning-based SOTA COVID-19 diagnostic methods, where the height of the bars represents the model performance; the higher bar, the better performance. PSTCNN shows superior performance in all metrics compared to deep learning-based SOTA COVID-19 diagnostic methods. The detailed comparison is shown in TABLE 4. This demonstrates the feasibility of the optimisation algorithm for hyperparameter tuning. Compared to other methods, the proposed method's comprehensive hyperparameter tuning can cover more possible combinations of hyperparameters. Furthermore, its purposeful search ensures that the global-optimal solution is approached step by step. Therefore, it can achieve better performance than manual hyperparameter tuning while automating hyperparameter tuning.

Conclusion

Since 2019, the global economy and human health have continuously received threats from COVID-19. In addition, the COVID-19 pandemic has highlighted the global shortage of healthcare resources. AI technologies to aid diagnosis are one of the vital viable options to alleviate this problem. This paper explores the possibility of further automation based on traditional AI techniques. It confirms this possibility by automating hyperparameter tuning using an optimisation algorithm and the excellent performance achieved by this method.

However, our method was only experimentally tested for three hyperparameters of the neural network training process, i.e., the learning rate, the coefficient that controls the exponential decay rates of the past gradient, and the coefficient that controls the exponential decay rates of the square of the past gradient. A few more hyperparameters can be tuned that we did not cover in this report, e.g., the number of network layers and the type of network layers. It means that the proposed method is insufficient as a final solution for self-tuning neural networks. In addition, the proposed method takes advantage of the purposeful movement of particles in the particle swarm optimisation algorithm for the hyperparameters to approach the optimal solution in each iteration. However, the movement of particles in each iteration is updated according to the direction of the best solution in the previous iteration, which may deviate from the global optimal solution and fall into the local optimal solution if the best solution is not in the path between the particles and the global optimal solution.

In future research, we will further explore the applicability of other optimisation algorithms to this task and attempt to avoid locally optimal solutions when obtaining combinations of hyperparameters while covering more hyperparameters in the experiment. We believe that reducing the dependence of model training on machine learning experts can effectively accelerate the generalisation of AI technologies across different domains. AI techniques will therefore become an essential tool in human life and industries in the near future.

Funding Statement

This paper is partially supported by the Medical Research Council Confidence in Concept Award, UK (MC_PC_17171); Royal Society International Exchanges Cost Share Award, UK (RP202G0230); British Heart Foundation Accelerator Award, UK (AA\18\3\34220); Hope Foundation for Cancer Research, UK (RM60G0680); Global Challenges Research Fund (GCRF), UK (P202PF11); Sino-UK Industrial Fund, UK (RP202G0289); LIAS Pioneering Partnerships award, UK (P202ED10); Data Science Enhancement Fund, UK (P202RE237); Guangxi Key Laboratory of Trusted Software, CN (kx201901).

Availability of Data and Materials

Raw data were generated at Huai'an Tongji Hospital. Derived data supporting the findings of this study are available from the corresponding author on request.

References

Arevalo-Rodriguez I, Buitrago-Garcia D, Simancas-Racines D, Zambrano-Achig P, Campo RD, Ciapponi A, Sued O, Martinez-García L, Rutjes AW, Low N, Bossuyt PM, et al. False-negative results of initial RT-PCR assays for COVID-19: A systematic review. PLOS ONE. 2020; 15 (12) 1–19.

Berrimi, M; Hamdi, S; Cherif, RY; Moussaoui, A; Oussalah, M; Chabane, M. COVID-19 detection from Xray and CT scans using transfer learning; 2021 International Conference of Women in Data Science at Taif University (WiDSTaif); Taif, Saudi Arabia. 2021. 1–6.

Bobermin L-D, Medeiros L-S, Weber F, Oliveira G-T-D, Santi L, Beys-Da-Silva W-O, Gonçalves C-A, Quincozes-Santos A. Impact of SARS-CoV-2 infection during pregnancy on postnatal brain development: The potential role of glial cells. BIOCELL. 2022; 46 (12) 2517–2523.

Carretta D-M, Domenico M-D, Lovero R, Arrigoni R, Wegierska A-E, Boccellino M, Ballini A, Charitos I-A, Santacroce L. SARS-CoV-2 induced myocarditis: Current knowledge about its molecular and pathophysiological mechanisms. BIOCELL. 2022; 46 (8) 1779–1788.

Chakraborty I, Maity P. COVID-19 outbreak: Migration, effects on society, global environment and prevention. Science of The Total Environment. 2020; 728: 1–6.

Chen Y. Covid-19 Classification Based on Gray-Level Co-occurrence Matrix and Support Vector Machine. COVID-19: Prediction, Decision-Making, and its Impacts. 2021; 60: 47–55.

Chen Y, Yang X-H, Wei Z, Heidari AA, Zheng N, Li Z, Chen H, Hu H, Zhou Q, Guan Q. Generative Adversarial Networks in Medical Image augmentation: A review. Computers in Biology and Medicine. 2022; 144: 1–22.

Dogo, EM; Afolabi, OJ; Nwulu, NI; Twala, B; Aigbavboa, CO. A Comparative Analysis of Gradient Descent-Based Optimization Algorithms on Convolutional Neural Networks; 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS); Belgaum, India. 2018. 92–99.

Ezzat D, Hassanien AE, Ella HA. An optimized deep learning architecture for the diagnosis of COVID-19 disease based on gravitational search optimization. Applied Soft Computing. 2021; 98: 1–13.

Gu J, Wang Z, Kuen J, Ma L, Shahroudy A, Shuai B, Liu T, Wang X, Wang G, Cai J, Chen T. Recent advances in convolutional neural networks. Pattern Recognition. 2018; 77: 354–377.

Guan Q, Chen Y, Wei Z, Heidari AA, Hu H, Yang X-H, Zheng J, Zhou Q, Chen H, Chen F. Medical image augmentation for lesion detection using a texture-constrained multichannel progressive GAN. Computers in Biology and Medicine. 2022; 145: 1–14.

Han K, Liu L, Song Y, Liu Y, Qiu C, Tang Y, Teng Q, Liu Z. An Effective Semi-supervised Approach for Liver CT Image Segmentation. IEEE Journal of Biomedical and Health Informatics. 2022. 1–8.

Hasöksüz M, Kiliç S, Saraç Fy. Coronaviruses and SARS-COV-2. Turkish Journal of Medical Sciences. 2020; 50 (SI–1) 549–556. [PubMed: 32293832]

Horry MJ, Chakraborty S, Paul M, Ulhaq A, Pradhan B, Saha M, Shukla N. COVID-19 Detection Through Transfer Learning Using Multimodal Imaging Data. IEEE Access. 2020; 8: 149808–149824. [PubMed: 34931154]

Hou S, Han J. COVID-19 Detection via a 6-Layer Deep Convolutional Neural Network. Computer Modeling in Engineering & Sciences. 2022; 130 (2) 855–869.

Ismael AM, Şengür A. Deep learning approaches for COVID-19 detection based on chest X-ray images. Expert Systems with Applications. 2021; 164: 1–11.

Jackson CB, Farzan M, Chen B, Choe H. Mechanisms of SARS-CoV-2 entry into cells. Nature Reviews Molecular Cell Biology. 2022; 23 (1) 3–20. [PubMed: 34611326]

Kennedy, J; Eberhart, R. Particle swarm optimization; Proceedings of ICNN'95 - International Conference on Neural Networks; Perth, WA, Australia. 1995. 1942–1948.

Khan MA. Pseudo Zernike Moment and Deep Stacked Sparse Autoencoder for COVID-19 Diagnosis. CMC-Computers, Materials & Continua. 2021; 69 (3) 3145–3162.

Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv preprint. 2014. 1–15. arXiv:1412.6980

Kiziloluk S, Sert E. COVID-CCD-Net: COVID-19 and colon cancer diagnosis system with optimized CNN hyperparameters using gradient-based optimizer. Medical & Biological Engineering & Computing. 2022; 60 (6) 1595–1612. [PubMed: 35396625]

Kronbichler A, Kresse D, Yoon S, Lee KH, Effenberger M, Shin JI. Asymptomatic patients as a source of COVID-19 infections: A systematic review and meta-analysis. International Journal of Infectious Diseases. 2020; 98: 180–186. [PubMed: 32562846]

Liang G, On B-W, Jeong D, Heidari AA, Kim H-C, Choi GS, Shi Y, Chen Q, Chen H. A text GAN framework for creative essay recommendation. Knowledge-Based Systems. 2021; 232: 1–11.

Miller A, Panneerselvam J, Liu L. A review of regression and classification techniques for analysis of common and rare variants and gene-environmental factors. Neurocomputing. 2022; 489: 466–485.

Monshi MMA, Poon J, Chung V, Monshi FM. CovidXrayNet: Optimizing data augmentation and CNN hyperparameters for improved COVID-19 detection from CXR. Computers in Biology and Medicine. 2021; 133: 1–13.

Najafabadi MM, Villanustre F, Khoshgoftaar TM, Seliya N, Wald R, Muharemagic E. Deep learning applications and challenges in big data analytics. Journal of Big Data. 2015; 2 (1) 1–21.

Ning H, Li R, Ye X, Zhang Y, Liu L. A Review on Serious Games for Dementia Care in Ageing Societies. IEEE Journal of Translational Engineering in Health and Medicine. 2020; 8: 1–11.

Organization, W. H. WHO Coronavirus (COVID-19) Dashboard. 2022. Retrieved Apr 01, 2022, from https://covid19.who.int

Ozturk A, Kara M. Diagnostic and prognostic significance of the lymphocyte/C-reactive protein ratio, neutrophil/lymphocyte ratio, and D-dimer values in patients with COVID-19. BIOCELL. 2022; 46 (12) 2625–2635.

Pi P. Gray level co-occurrence matrix and Schmitt neural network for Covid-19 diagnosis. EAI Endorsed Transactions on eLearning. 2021; 7 (22) 1–13.

Pi P, Lima D. Gray level co-occurrence matrix and extreme learning machine for Covid-19 diagnosis. International Journal of Cognitive Computing in Engineering. 2021; 2: 93–103.

Polsinelli M, Cinque L, Placidi G. A light CNN for detecting COVID-19 from CT scans of the chest. Pattern Recognition Letters. 2020; 140: 95–100. [PubMed: 33041409]

Saad Hikmat H, Adnan Mohsin A. Comparison of Optimization Techniques Based on Gradient Descent Algorithm: A Review. PalArch's Journal of Archaeology of Egypt / Egyptology. 2021; 18 (4) 2715–2743.

van Kasteren PB, van der Veer B, van den Brink S, Wijsman L, de Jonge J, van den Brandt A, Molenkamp R, Reusken CBEM, Meijer A. Comparison of seven commercial RT-PCR diagnostic kits for COVID-19. Journal of Clinical Virology. 2020; 128: 1–4.

Wang, W. Covid-19 detection by wavelet entropy and jaya; International Conference on Intelligent Computing; Shenzhen, China. 2021. 499–508.

Wang W, Zhang X, Wang S-H, Zhang Y-D. Covid-19 diagnosis by WE-SAJ. Systems Science and Control Engineering. 2022; 10 (1) 325–335.

Wu X. Diagnosis of COVID-19 by Wavelet Renyi Entropy and Three-Segment Biogeography-Based Optimization. International Journal of Computational Intelligence Systems. 2020; 13 (1) 1332–1344.

Yu H, Liu J, Chen C, Heidari AA, Zhang Q, Chen H. Optimized deep residual network system for diagnosing tomato pests. Computers and Electronics in Agriculture. 2022; 195: 1–18.

Yu M, Han M, Li X, Wei X, Jiang H, Chen H, Yu R. Adaptive soft erasure with edge self-attention for weakly supervised semantic segmentation: Thyroid ultrasound image case study. Computers in Biology and Medicine. 2022; 144: 1–11.

Yu, X; Wang, S-H; Zhang, X; Zhang, Y-D. Detection of COVID-19 by GoogLeNet-COD; 2020 Sixteenth International Conference on Intelligent Computing; Bari, Italy. 2020. 499–509.

Zhang X, Lu S, Wang S-H, Yu X, Wang S-J, Yao L, Pan Y, Zhang Y-D. Diagnosis of COVID-19 Pneumonia via a Novel Deep Learning Architecture. Journal of Computer Science and Technology. 2022; 37 (2) 330–343. [PubMed: 35496726]

Zhang Y-D, Zhang Z, Zhang X, Wang S-H. MIDCAN: A multiple input deep convolutional attention network for Covid-19 diagnosis based on chest CT and chest X-ray. Pattern Recognition Letters. 2021; 150: 8–16. [PubMed: 34276114]

Europe PMC Funders Author Manuscripts

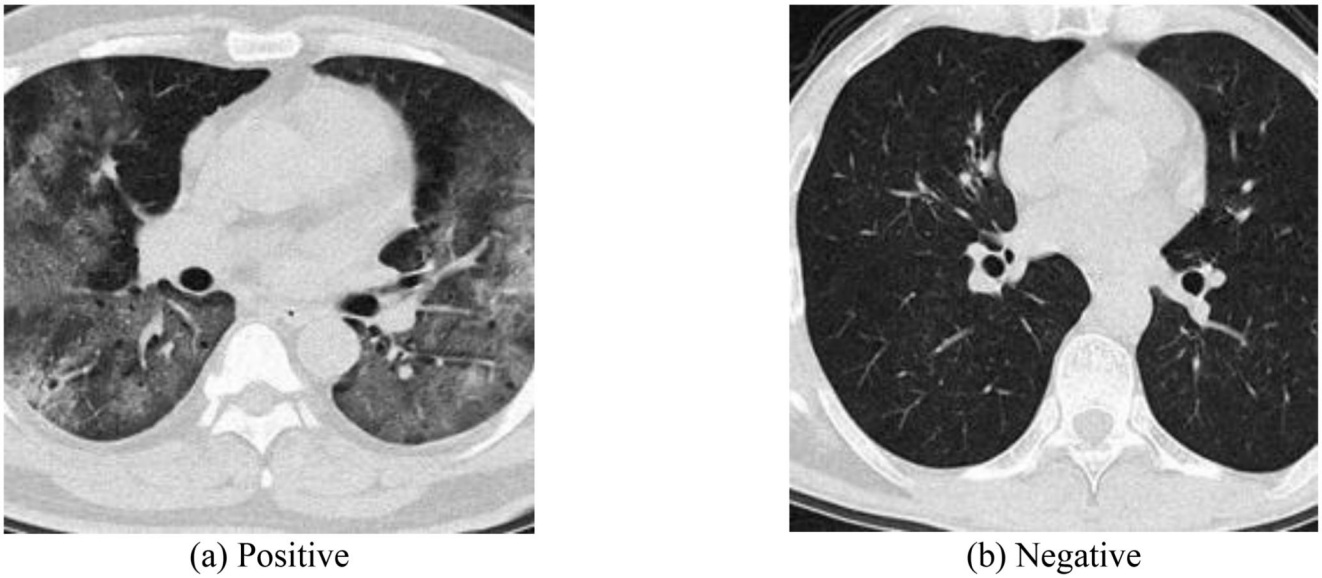

Figure 1. COVID-19 infected Chest-computed tomography (CT) slice image sample (a) and uninfected Chest-CT slice image sample (b).

Europe PMC Funders Author Manuscripts

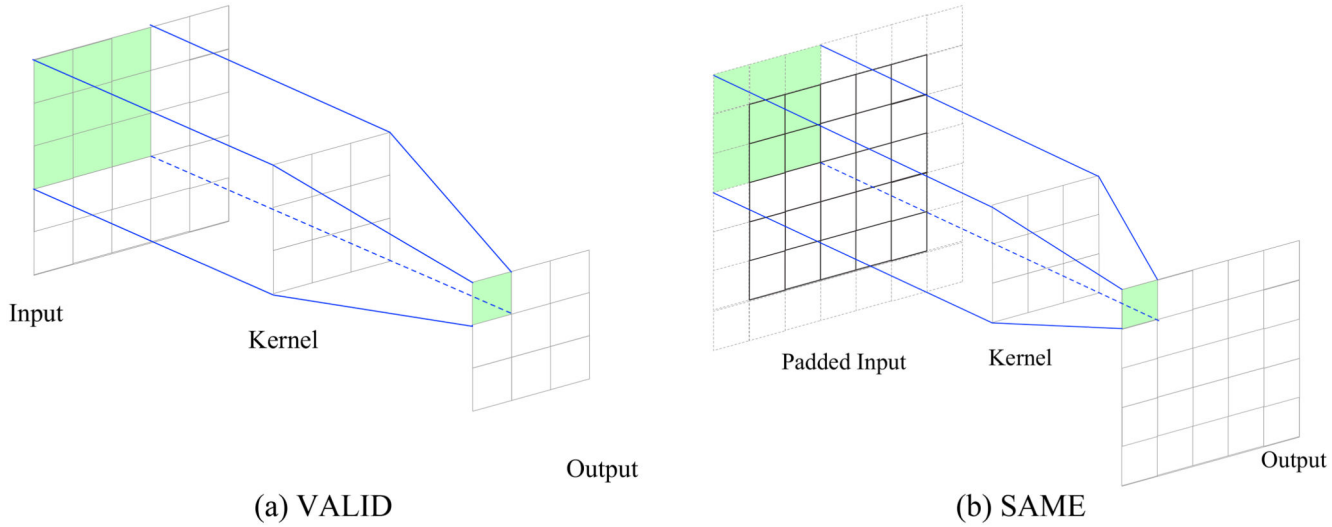

Figure 2. In the VALID mode

(a), the output size of the convolution for a 5 × 5 image is 3 × 3. In SAME mode (b), the input image is padded to size 6 × 6, and the output size remains the same as the original input, 5 × 5. The dotted blocks are the padded section.

Europe PMC Funders Author Manuscripts

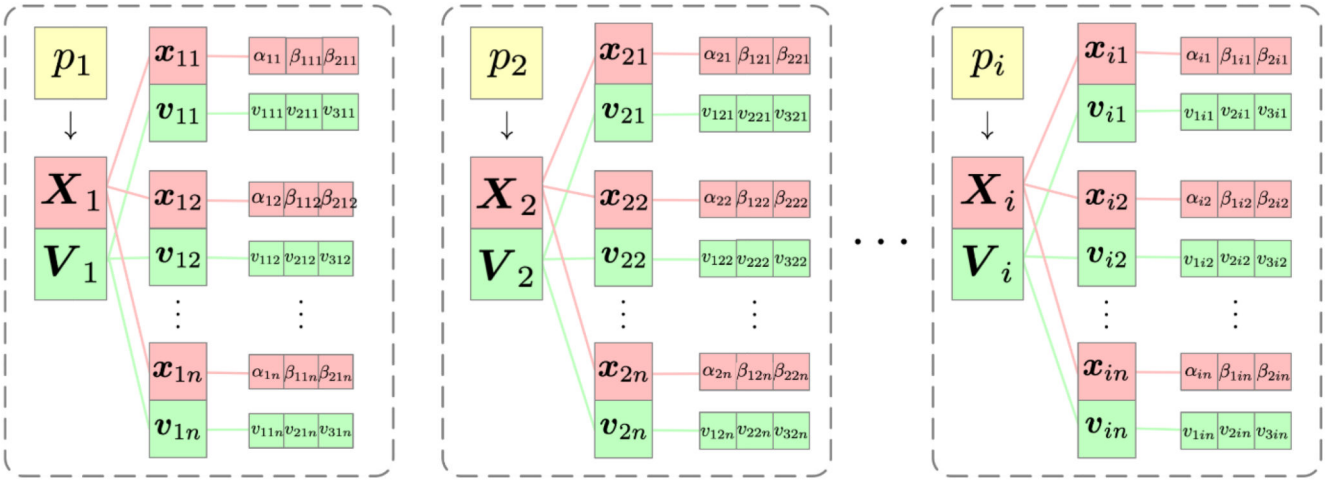

Figure 3. Illustration of Encoding Scheme.

A particle swarm contains i particles, each with n positions and n velocities. Each position is a vector of α, β, and β_2, and each velocity contains three values corresponding to α, β, and β_2 of the position.

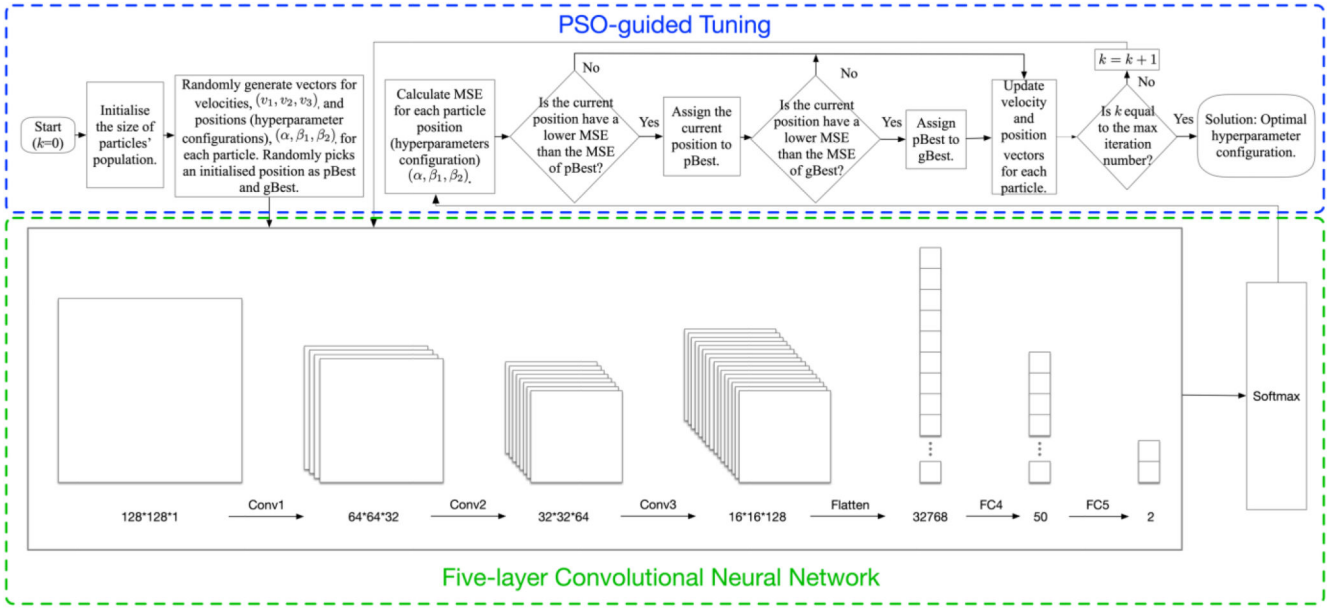

Figure 4. Overall Structure of PSO-guided Self-Tuning CNN (PSTCNN).

The framework can be divided into two main parts, PSO-guided Tuning (blue dashed box) and a Five-layer CNN (green dashed box). The PSO-guided Tuning part does the initialisation and updating of the parameter values. The Five-layer CNN is trained with the three hyperparameters from the PSO-guided Tuning and some pre-set constant hyperparameters (batch size is 128, kernel size is 3 × 3, and the number of epochs of 100). The MSE calculated from the output of the trained CNN is the fitness value of the PSO. The starting iteration index k=0, and when the value of k reaches the pre-set maximum number of iterations, The training of the model is completed.

Europe PMC Funders Author Manuscripts

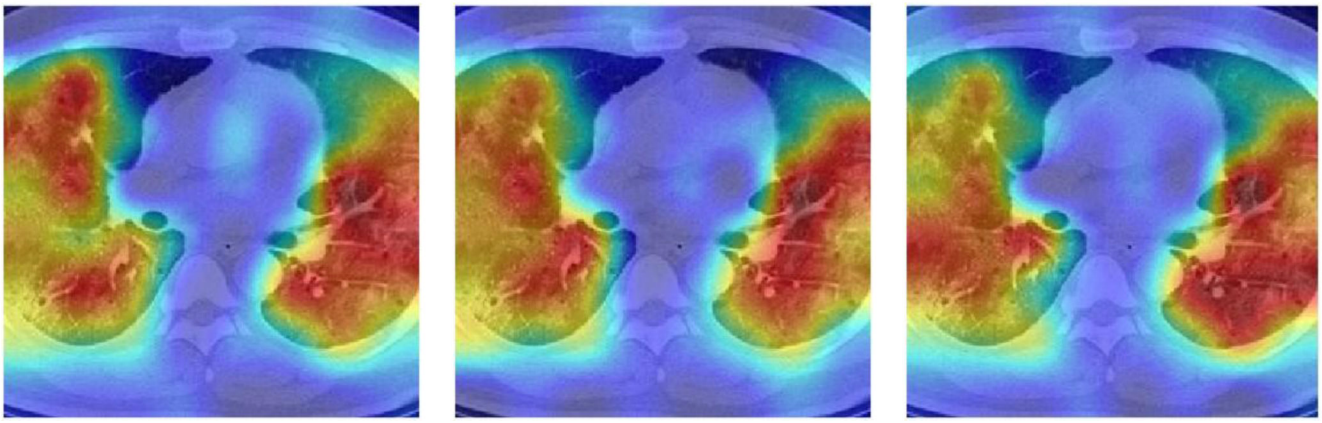

Figure 5. Heatmap examples generated by Grad-CAM.

Europe PMC Funders Author Manuscripts

Europe PMC Funders Author Manuscripts

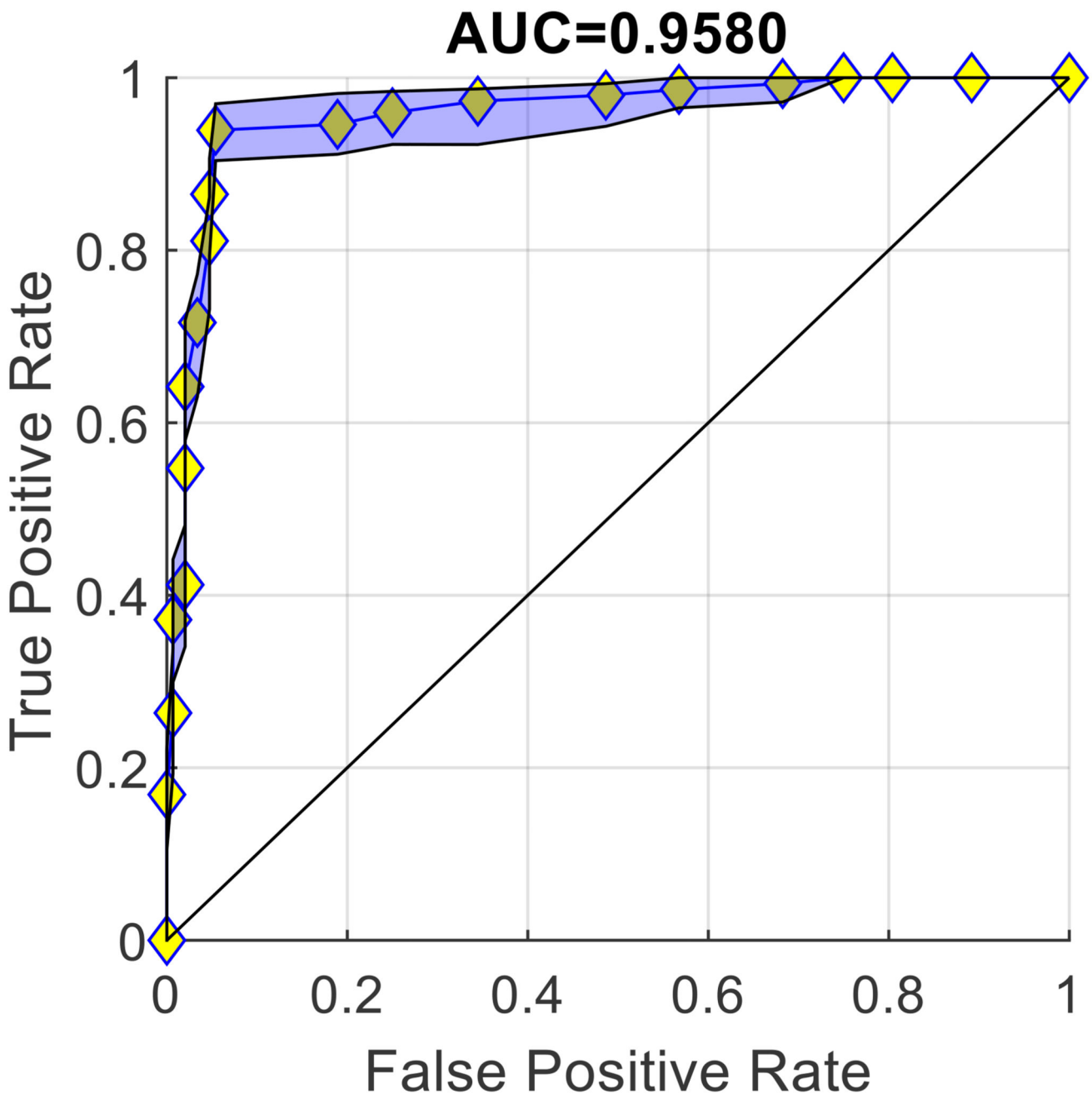

Figure 6.
Receiver Operating Characteristic Curve of Particle Swarm Optimisation-guided Self-Tuning CNN.

Europe PMC Funders Author Manuscripts

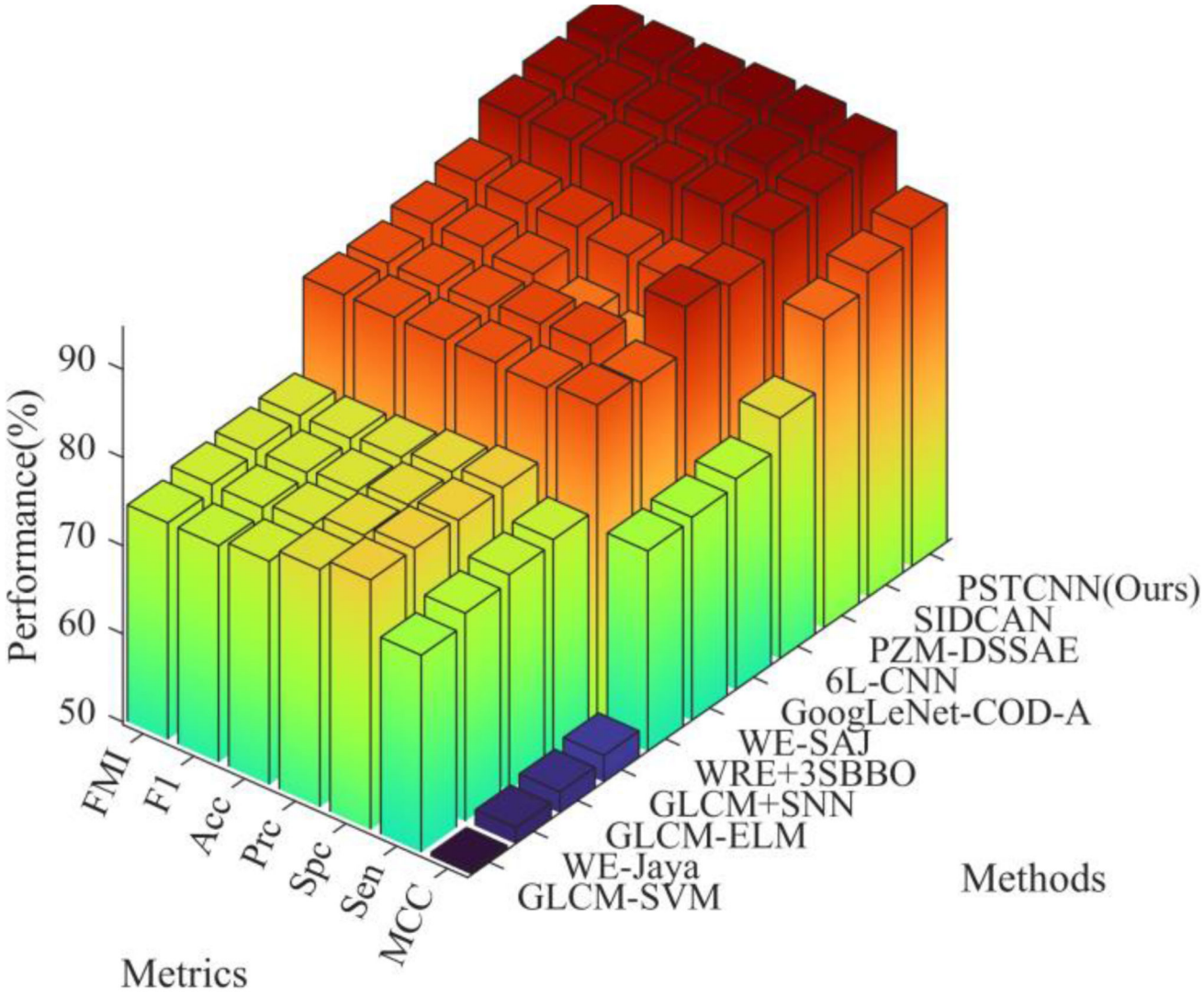

Figure 7. Comparison with State-Of-The-Art methods.

Europe PMC Funders Author Manuscripts

Table 1

Dataset Statistic

Class	Ratio	No. of Samples
Positive (COVID-19)	0.5	148
Negative (Health Control)	0.5	148

Europe PMC Funders Author Manuscripts

Table 2

Advantages and disadvantages of search methods

Search Method	Advantage	Disadvantage
Grid Search	The grid search process iterates through all possible combinations of hyperparameters without missing possible combinations as far as time and computational resources allow.	Grid search is difficult to traverse a nearly infinite number of all hyperparameter combinations, requiring huge time and computational costs, and is inefficient.
Random Search	The random search process is stochastic and can cover a much larger range of hyperparameter combinations.	The search process is so random that, even though it can be optimized to prevent repeated computation of the same hyperparameter combinations, the large number of hyperparameter combinations makes it difficult to guarantee that the optimal hyperparameter is found.
Particle Swarm Optimisation (PSO; Ours)	PSO uses many particles to purposefully find the optimal combination of hyperparameters, making the search process more efficient.	Updating particle positions requires training neural networks based on different combinations of hyperparameters, which requires high computational resources and time costs.

Europe PMC Funders Author Manuscripts

Table 3

Ten runs results among seven performance metrics (in %)

	Sen	Spc	Prc	Acc	F1	MCC	FMI
R1	89.86	91.89	91.72	90.88	90.78	81.77	90.79
R2	91.89	95.27	95.10	93.58	93.47	87.21	93.48
R3	94.59	93.92	93.96	94.26	94.28	88.52	94.28
R4	93.92	93.92	93.92	93.92	93.92	87.84	93.92
R5	92.57	91.89	81.95	92.23	92.26	84.46	92.26
R6	93.92	96.62	96.53	95.27	95.21	90.57	95.21
R7	93.92	91.89	92.05	92.91	92.98	85.83	92.98
R8	**96.62**	**97.30**	**97.28**	**96.96**	**96.95**	93.92	**96.95**
R9	93.92	93.92	93.92	93.92	93.92	87.84	93.92
R10	95.27	96.62	96.58	95.95	95.92	**97.90**	95.92
MSD	93.65	94.32	94.30	93.99	93.97	87.99	93.97
	±1.86	±2.07	±2.04	±1.78	±1.78	±3.56	±1.78

R1, R2, …, R10: Run 1, Run 2, …, Run 10; MSD: Mean ± Standard Deviation; Sen: Sensitivity; Spc: Specificity; Prc: Precision; Acc: Accuracy; F1: F1-score; Mcc: Matthews correlation coefficient; FMI: Fowlkes-Mallows index.

Europe PMC Funders Author Manuscripts

Table 4
Comparison with State-Of-The-Arts methods (in %)

Model	Sen	Spc	Prc	Acc	F1	MCC	FMI
WRE+ 3SBBO (Wu 2020)	86.40 ± 3.00	85.81 ± 3.14	86.14 ± 3.03	86.12 ± 2.75	86.16 ± 2.77	72.42 ± 5.55	86.15 ± 2.76
GoogLeNet-COD-A (Yu, Wang et al. 2020)	90.54 ±2.16	82.77 ±2.65	84.07 ±1.93	86.66 ±1.14	87.15 ±1.06	73.59 ±2.25	87.23 ±1.07
GLCM-SVM (Chen 2021)	72.03 ±2.94	78.04 ±1.72	76.66 ±1.07	75.03 ±1.12	74.24 ±1.57	50.20 ±2.17	74.29 ±1.53
6L-CNN (Hou and Han 2022)	89.47 ±1.50	87.47 ±2.11	87.75 ±1.76	88.47 ±1.05	88.59 ±0.99	76.98 ±2.09	88.60 ±0.99
SIDCAN (Zhang, Zhang et al. 2021)	92.86 ±1.59	93.64 ±2.09	93.36 ±2.02	93.26 ±0.74	93.08 ±0.71	86.55 ±1.49	93.10 ±0.72
PZM-DSSAE (Khan 2021)	92.06 ±1.54	92.56 ±1.06	92.53 ±1.03	92.31 ±1.08	92.29 ±1.10	84.64 ±2.15	92.29 ±1.10
GLCM-ELM (Pi and Lima 2021)	74.19 ±2.74	77.81 ±2.03	77.01 ±1.29	76.00 ±0.98	75.54 ±1.31	52.08 ±1.95	75.57 ±1.28
WE-Jaya (Wang 2021)	73.31 ±2.26	78.11 ±1.92	77.03 ±1.35	75.71 ±1.04	75.10 ±1.23	51.51 ±2.07	75.14 ±1.22
GLCM+SNN (Pi 2021)	74.66 ±1.87	78.00 ±1.29	77.24 ±1.15	76.33 ±1.18	75.92 ±1.31	52.70 ±2.34	75.93 ±1.30
WE-SAJ (Wang, Zhang et al. 2022)	85.47 ±1.84	87.23 ±1.67	87.03 ±1.34	86.35 ±0.70	86.23 ±0.77	72.75 ±1.38	86.24 ±0.76
PSTCNN (Ours)	**93.65 ±1.86**	**94.32 ±2.07**	**94.30 ±2.04**	**93.99 ±1.78**	**93.97 ±1.78**	**87.99 ±3.56**	**93.97 ±1.78**

Sen: Sensitivity; Spc: Specificity; Prc: Precision; Acc: Accuracy; F1: F1-score; Mcc: Matthews correlation coefficient; FMI: Fowlkes-Mallows index.