



COVID-19 Diagnosis via DenseNet and Optimization of Transfer Learning Setting

Yu-Dong Zhang¹ · Suresh Chandra Satapathy² · Xin Zhang³ · Shui-Hua Wang^{4,5}

Received: 30 June 2020 / Accepted: 29 September 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

COVID-19 is an ongoing pandemic disease. To make more accurate diagnosis on COVID-19 than existing approaches, this paper proposed a novel method combining DenseNet and optimization of transfer learning setting (OTLS) strategy. Preprocessing was used to enhance, crop, and resize the collected chest CT images. Data augmentation method was used to increase the size of training set. A composite learning factor (CLF) was employed which assigned different learning factor to three types of layers: frozen layers, middle layers, and new layers. Meanwhile, the OTLS strategy was proposed. Finally, precomputation method was utilized to reduce RAM storage and accelerate the algorithm. We observed that optimization setting “201-IV” can achieve the best performance by proposed OTLS strategy. The sensitivity, specificity, precision, and accuracy of our proposed method were 96.35 ± 1.07 , 96.25 ± 1.16 , 96.29 ± 1.11 , and 96.30 ± 0.56 , respectively. The proposed DenseNet-OTLS method achieved better performances than state-of-the-art approaches in diagnosing COVID-19.

Keywords COVID-19 · DenseNet · Transfer learning · Optimization · Data augmentation · Composite learning factor · Precomputation

Introduction

The coronavirus pandemic is an ongoing global pandemic disease, which is also called COVID-19. World Health Organization (WHO) declared the COVID-19 as a public health emergency of international concern on 01/30/2020,

and as a pandemic on 03/11/2020. Till 17/09/2020, COVID-19 has caused 29.87 million confirmed cases and 940.72 thousand death tolls.

In practice, there are two main diagnosis methods. One is real-time reverse-transcription polymerase chain reaction (PCR), which is an RNA testing of respiratory secretions sampled with the help of nasopharyngeal swab. The other is the imaging methods, among which chest computed tomography (CT) can get better performance than chest X-ray. Studies have shown that chest CT is faster than more sensitive than PCR methods [1].

For the chest CT, the main biomarkers differentiating COVID-19 from healthy people are the asymmetric peripheral ground-glass opacity (GGO) without pleural effusions. Manual interpretation by radiologists is tedious and easy to be influenced by fatigue, emotion, and other factors. A smart diagnosis system via computer vision and artificial intelligence can benefit patients, radiologists, and hospitals.

Traditional artificial intelligence (AI) and modern deep learning (DL) methods have achieved excellent results in analyzing medical images; e.g., [2] proposed a radial-basis-function neural network (RBFNN) to detect pathological brains. [3] presented a kernel-based extreme learning classifier (K-ELM) to create a novel pathological brain detection system. Their method was robust and effective. [4] proposed a novel extreme learning machine trained by the bat algorithm (ELM-BA) approach.

✉ Yu-Dong Zhang
yudongzhang@ieee.org

Suresh Chandra Satapathy
sureshsatapathy@ieee.org

Xin Zhang
973306782@qq.com

Shui-Hua Wang
shuihuawang@ieee.org

¹ Department of Informatics, University of Leicester, LE1 7RH Leicester Leicestershire, UK

² School of Computer Engg, KIIT Deemed To University, Bhubaneswar, India

³ School of Architecture Building and Civil Engineering, Loughborough University, Loughborough LE11 3TU, UK

⁴ School of Mathematics and Actuarial Science, University of Leicester, Leicester, LE1 7RH, UK

⁵ Department of Medical Imaging, The Fourth People's Hospital of Huai'an, Huai'an 223002, China

[5] used a six-layer convolutional neural network (6L-CNN) to recognize sign language fingerspelling. [6] presented the GoogleNet. [7] suggested the use of ResNet18 for mammogram abnormality detection. [8] presented a weakly labeled data augmentation method on COVID-19 chest X-ray images. [9] used a deep learning approach to characterize COVID-19 pneumonia in chest CT images. [10] combined support vector machine and convolutional neural network to detect COVID-19 from chest X-ray images.

The results based on deep learning methods, particularly, convolutional neural networks (CNNs), are significantly better than those of orthodox computer-vision methods. However, two reasons limited the wide usage of CNNs: (i) Tuning the hyperparameters of CNN is boring and time-consuming. (ii) Small medical dataset may cause overfitting to CNN.

A solution to (i) accelerate developing CNN and (ii) to avoid overfitting is “transfer learning (TL).” The TL techniques can develop a deep neural network quickly with comparable or even better performance than recent deep learning methods. TL stores knowledge gained from solving another problem and applies that gained knowledge to a different but related problem.

This paper aims to apply a relatively new transfer learning framework—DenseNet to solve the task of COVID-19 diagnosis. The contributions of this study are six folds: (i) DenseNet was introduced as the backbone pre-trained model, and we modified it to our task. (ii) A composite learning factor strategy was used for training DenseNet. (iii) Data augmentation was used to enhance the training set. (iv) An optimization of transfer learning setting (OTLS) strategy was proposed. (v) Precomputation was introduced to save memory. (vi) We compared our method with state-of-the-art COVID-19 diagnosis approaches and proved it gives better performances.

Materials

We selected 142 patients with COVID-19 pneumonia from local hospitals. Another 142 cases were randomly selected from healthy medical people (tested negative). The COVID-19 patients were in the observation group: 95 males and 47 females. The healthy checkup was the control group: 88 males and 54 females. Inclusion criteria for confirmed COVID-19: (1) positive nucleic acid test and (2) the CT image data is complete. Table 1 lists the demographic statistics of all involved subjects.

Table 1 Demographic statistics of subjects

	Subject Number (male/female)	Image Number
COVID-19	142 (95/47)	320
HC	142 (88/54)	320

Image acquisition CT configuration and method: Philips Ingenity64 row spiral CT machine, KV: 120, MAS: 240, layer thickness 3 mm, layer spacing 3 mm, screw pitch 1.5: lung window (W: 1500 Hounsfield unit (HU), L: − 500 HU), Mediastinum window (W: 350 HU, L: 60 HU), thin-layer reconstruction according to the lesion display, layer thickness, and layer distance are 1-mm lung window image. The patients were placed in a supine position, breathing deeply after holding in, and conventionally scanned from the lung tip to the costal diaphragm angle. All the acquired images are with resolution of 1024×1024 .

All images are transmitted to the medical image PACS for observation, and two junior radiologists (J_1, J_2) with chest diagnostic experience collectively read the radiographs and recorded the distribution, size, and morphology of the CT manifestations of the lesions. About 1–4 slices were chosen. Those 142 COVID-19 subjects and 142 healthy people finally generate $320 + 320 = 640$ images.

When there are differences between the two analyses, a senior doctor S was consulted to reach a consensus by majority voting.

$$\mathcal{L} = \begin{cases} J_1 & J_1 = J_2 \\ \text{MV}(J_1, J_2, S) & J_1 \neq J_2 \end{cases} \quad (1)$$

where J_1, J_2 , and S represent the opinions (COVID-19 or healthy) by corresponding radiologists. MV represents majority voting, \mathcal{L} is the label result.

Slice level selection method was used to extract slice images. For COVID-19 pneumonia patients, the slice showing the largest size and number of lesions was selected. For healthy control group, any level of the image can be selected.

Methodology

For the ease of understanding, Table 13 shows the abbreviations used in this paper. In what below, we shall expatiate each steps of our proposed method.

Improvement I: Preprocessing

The chest CT images \mathbb{X} is a combination of both COVID-19 subjects \mathbb{C} and healthy controls \mathbb{H}

$$\mathbb{X} = \text{concat}(\mathbb{C}, \mathbb{H}) \quad (2)$$

Table 2 RHO setting

Set	COVID-19	Healthy	Total
Training (50%)	160	160	320
Validation (20%)	64	64	128
Test (30%)	96	96	192
Total	320	320	640

\mathbb{X} were collected from a variety of sources and different scanning machines. Suppose there are n images in the dataset, then we can write in short as $\mathbb{X} = \{x_1, x_2, \dots, x_n\}$.

The first step: A contrast normalization technique, histogram stretching (HS) [11], is chosen to preprocess all the images.

$$\mathbb{Y} = \text{HS}(\mathbb{X}) = \{y_1, y_2, \dots, y_n\} \quad (3)$$

Suppose x denotes the original chest CT image and y stands for the contrast-normalized image. Histogram stretching operation is defined below:

$$y(a, b) = \frac{[x(a, b) - x_l]}{(x_h - x_l)} \quad (4)$$

where

$$x_l = \min_{(a,b)}[x(a, b)] \quad (5)$$

$$x_h = \max_{(a,b)}[x(a, b)] \quad (6)$$

x_l and x_h represent the lowest and highest intensity gray-levels of the chest CT image x . The (a, b) stands for the coordinates.

The second step: We crop the image. The reasons are as follows: (i) There are rulers on the right side of the image and (ii) there are texts and checkup beds on the bottom of the image, as shown in Fig. 1. Those will impair the following classification performance, since the contents at bottom and right sides are unrelated with our diagnosis. After the crop, we get the image set \mathbb{Z} .



Fig. 1 Why we need to crop the chest CT image (the right and bottom show unrelated contents, outlined by red boxes)

$$\mathbb{Z} = \text{Crop}(\mathbb{Y}) = \{z_1, z_2, \dots, z_n\} \quad (7)$$

The third step: We resize the image to $[224, 224]$. The reasons are the following: (i) Original size is too large and contains redundant spatial information, (ii) reduced size can accelerate deep neural network processing, and (iii) 224×224 is the standard input for following DenseNet input. After resizing, we get the image set \mathbb{R} from \mathbb{Z} . Figure 2 shows an illustration of the preprocessed images in our dataset we shall use in this study.

$$\begin{aligned} \mathbb{R} &= \text{Resize}(\mathbb{Z}, [224, 224]) \\ &= \{r, r_2, \dots, r_n\} \end{aligned} \quad (8)$$

The resizing procedure can save storage memory significantly. Suppose we store each image in single-precision floating-point (SPFP) format, i.e., 32 bits = 4 byte the original size for each image is $1024 \times 1024 \times 3 \times 4 = 12,582,912$ byte. After resizing, the storage for each image is $224 \times 224 \times 3 \times 4 = 602,112$ byte. Using simple math, we find the resizing procedure can save $1 - \frac{602,112}{12,582,912} = 95.21\%$ storage, and therefore, we can store all the chest CT images in the RAM, so as to accelerate the running speed of proposed algorithms.

Improvement II: Data Augmentation on Training Set

To avoid overfitting, five disparate data augmentation (DA) techniques were utilized to expand training set. Suppose the dataset \mathbb{R} is divided as three parts: training set X and validate set Y and test set Z via random hold-out (RHO) method

$$\mathbb{R} \xrightarrow{\text{RHO}} \{X, Y, Z\} \quad (9)$$

where

$$\begin{aligned} X &= \{x(i)\} \\ Y &= \{y(i)\} \\ Z &= \{z(i)\} \end{aligned} \quad (10)$$

and their sizes obey following equation

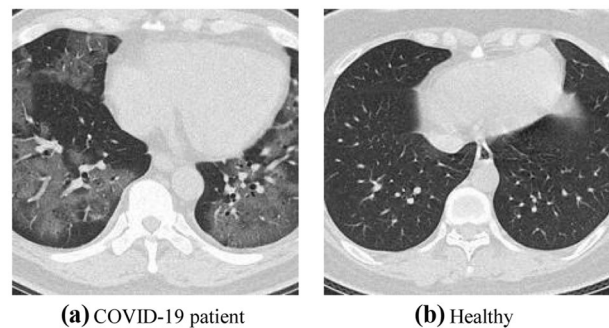


Fig. 2 Illustration of preprocessed COVID-19 dataset

$$|X| + |Y| + |Z| = |\mathbb{R}| \quad (11) \quad m_1 = m_2 = \dots = m_{30} = 0 \quad (17)$$

where $|\cdot|$ means the cardinality of the set.

In total, we have collected 640 lung window images from chest CT. Afterwards, we use the random hold-out (RHO) method. About 50% were used for training randomly, 20% for validation, and the rest 30% were used for test. The summary of the dataset is listed in Table 3.

For each image x_i , we perform following five DA operations: (i) Scaling. Chest CT images are scaled with scaling factor s from 0.7 to 1.3 with increase of 0.02, skipping $s = 1$.

$$\overrightarrow{x^{\text{scale}}(i)} = \text{scale}[x(i)] \quad (12)$$

$$= [x_1^{\text{scale}}(i, s_1), \dots, x_{30}^{\text{scale}}(i, s_{30})]$$

where

$$s_1 = 0.7, s_2 = 0.72, \dots, s_{15} = 0.98 \quad (13)$$

$$s_{16} = 1.02, \dots, s_{30} = 1.3$$

(ii) Rotation. Angle θ was in the value from -30 to 30° in increase of 2° , skipping $\theta = 0$.

$$\overrightarrow{x^{\text{rot}}(i)} = \text{rotate}[x(i)] \quad (14)$$

$$= [x_1^{\text{rot}}(i, \theta_1), x_2^{\text{rot}}(i, \theta_2), \dots, x_{30}^{\text{rot}}(i, \theta_{30})]$$

Where

$$\theta_1 = -30^\circ, \theta_2 = -28^\circ, \dots, \theta_{15} = -2^\circ \quad (15)$$

$$\theta_{16} = 2^\circ, \dots, \theta_{30} = 30^\circ$$

(iii) Noise injection. The m -mean n -variance Gaussian noises [12] are added to the chest CT images x_i to produce 30 new noised images.

$$\overrightarrow{x^{\text{noise}}(i)} = \text{noiseinject}[x(i)] \quad (16)$$

$$= [x_1^{\text{noise}}(i, m_1, n_1), \dots, x_{30}^{\text{noise}}(i, m_{30}, n_{30})]$$

where

$$n_1 = n_2 = \dots = n_{30} = 0.01 \quad (18)$$

(iv) Random translation. The chest CT image x_i is translated 30 times with random shift, of which the horizontal and vertical values $t = [t_x, t_y]$ are in the range of $[-15, 15]$ pixels, and obey uniform distribution \mathcal{U} .

$$\overrightarrow{x^{\text{translate}}(i)} = \text{translate}[x(i)] \quad (19)$$

$$= [x_1^{\text{translate}}(i, t_1), \dots, x_{30}^{\text{translate}}(i, t_{30})]$$

where

$$t_x \sim \mathcal{U}[-15, 15] \quad (20)$$

$$t_y \sim \mathcal{U}[-15, 15]$$

(v) Gamma correction. The gamma correction factor r varies from 0.4 to 1.6 with increase of 0.04, skipping the value of 1

$$\overrightarrow{x^{\text{Gamma}}(i)} = \text{Gamma_correct}[x(i)] \quad (21)$$

$$= [x_1^{\text{Gamma}}(i, r_1), \dots, x_{30}^{\text{Gamma}}(i, r_{30})]$$

where

$$r_1 = 0.4, r_2 = 0.44, \dots, r_{15} = 0.96 \quad (22)$$

$$r_{16} = 1.04, \dots, r_{30} = 1.6$$

Finally, horizontal mirror [13] was inducted to all the 150 new generated images.

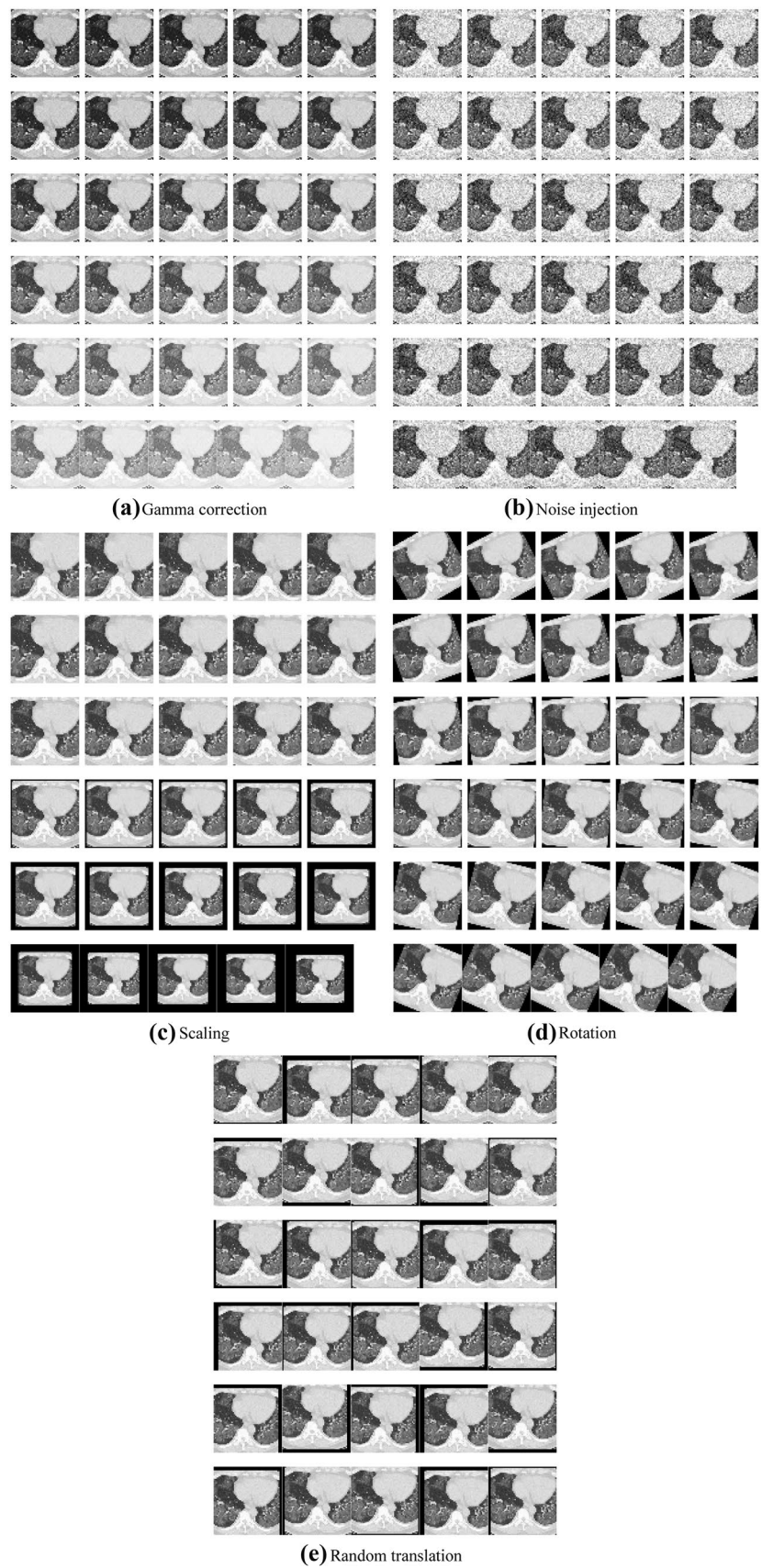
$$\overrightarrow{x^{\text{mirror}}(i)} = \text{horizontal_mirror}(\overrightarrow{x^{\text{DA5}}(i)}) \quad (23)$$

where $\overrightarrow{x^{\text{DA5}}}$ means the concatenation of five temporary DA results.

Table 3 Modification of layers of DenseNet

Layer	Original	Replaced
Third from last	FCL (1000) with pre-trained weights and biases	FCL (2) with random initialization
Last	Classification Layer 1000 classes: recreational vehicle, printer, coho, milk can, Irish wolfhound, parallel bars, tree frog, dhole, Gila monster, toucan, spider web, organ, walking stick, broccoli, loggerhead, bassoon, colobus, racket, schooner, and Kerry blue terrier, and 980 other classes	Classification Layer Two classes: (i) COVID-19; (ii) Healthy control

Fig. 3 Data-augmented training samples (mirror results were not shown)



$$\overrightarrow{x^{DA5}(i)} = \text{concat} \begin{bmatrix} \overrightarrow{x^{\text{scale}}(i)} \\ \overrightarrow{x^{\text{rot}}(i)} \\ \overrightarrow{x^{\text{noise}}(i)} \\ \overrightarrow{x^{\text{translate}}(i)} \\ \overrightarrow{x^{\text{Gamma}}(i)} \end{bmatrix} \quad (24)$$

Thus, one original image will lead to 301 images. Suppose size function is S , we have $S[x(i)] = 1$, $S[\overrightarrow{x^{DA5}(i)}] = 5 \times 30 = 150$, and $S[\overrightarrow{x^{\text{mirror}}(i)}] = 150$. Thus, adding up we get $1 + 150 + 150 = 301$.

$$x(i) \xrightarrow{\text{DA}} \text{concat} [x(i), \overrightarrow{x^{DA5}(i)}, \overrightarrow{x^{\text{mirror}}(i)}] \quad (25)$$

There are other DA techniques, such as color jittering and shear transform. In this study, we choose above five DA techniques, because they can efficiently improve the performances of our established deep learning model. Using other DA techniques may even further improve the performance, and we will leave it as our future studies.

Using Fig. 2a as an example, we can generate in total 150 new images and another 150-image of horizontally flipped images. Figure 3a–e show the $\overrightarrow{x^{DA5}}$ results. For the page limit, we do not display the horizontally flipped images. After data augmentation, the training set will be $301 \times$ of its original size. That means, we have $320 \times 301 = 96,320$ training images in the augmented training set.

Basics of Transfer learning

The basic ideas of transfer learning (abbreviated as TL) [14] are utilizing a complicated and successfully pre-trained model (DenseNet in this study), taught from a sizable amount of source data, viz., ImageNet, and then “transfer” the learnt knowledge to the relatively simple task (classify COVID-19 from HC) with a tiny quantity of data. Mathematically, suppose the source data is D_S representing ImageNet, and the source label L_S the 1000-category labeling

$$S = \{D_S, L_S, f_S\} \quad (26)$$

where f_S means the source objective-predictive function, i.e., DenseNet in this study. Now we have the target triple: target data D_T represents the augmented training set, L_T presents the 2-class labeling (COVID-19 or health), and f_T represents the classifier to be established.

$$T = \{D_T, L_T, f_T\} \quad (27)$$

Using transfer learning, the classifier to be created can be written as $f_T(D_T, L_T | D_S, L_S, f_S) = f_T(D_T, L_T | S)$. Without transfer learning, the classifier is written as $f_T(D_T, L_T)$. $f_T(D_T, L_T | S)$ is expected to be much closer to the ideal

classifier f_T^{Ideal} than using only the target domain, viz. suppose there is a large number of samples X , then

$$\begin{aligned} & |f_T(D_T, L_T | S)(X) - f_T^{\text{Ideal}}(X)| \\ & < |f_T(D_T, L_T)(X) - f_T^{\text{Ideal}}(X)| \end{aligned} \quad (28)$$

where $||$ is some error function over all the samples X .

In practice, three elements are vital to help the transfer: (i) The triumph of PTM helps the user get rid of hyper-parameter tuning. (ii) The initial layers in PTM can be thought as feature descriptors which extract low-level features, e.g., tints, edges, blobs, shades, and textures. (iii) The target model may only need to re-train the last several layers of pre-trained model, since we believe the last several layers carry out the complex identification tasks. The basic idea of transfer learning is shown in Fig. 4.

Improvement III: Use DenseNet as Backbone

Pretrained models (PTMs) are useful tools to help the AI practitioners to swiftly develop a deep neural network suitable for a specific task. Common PTMs include AlexNet, VGGNet, ResNet, GoogleNet, and DenseNet. AlexNet was proposed by [15], which overwhelmed the other competitor models in the competition of ImageNet Large Scale Visual Recognition Challenge (ILSVRC) at year 2012. The structure of AlexNet is similar to LeNet but with some new techniques like local response normalization (LRN), max pooling, and ReLU nonlinearity. [16] presented VGGNet that was a 19-layer deep neural network and won the 2nd place of ILSVRC-2014. GoogleNet was proposed by [17] combined some new ideas including 1×1 convolutions, reducing size of feature maps and making the network deeper and wider. The winner ResNet [18] of ILSVRC-2015 presented a deeper network with 152 layers, which includes a new idea of a shortcut connection. Then in 2016, [19] developed a new idea so called “DenseBlock (DB),” which introduced the feature reuse into the whole network.

In this study, we choose DenseNet as the backbone for developing a smart COVID-19 diagnosis system, as DenseNet provides the best performance based on the ImageNet classification task. In the traditional CNN shown in Fig. 5a, all layers are gradually connected as

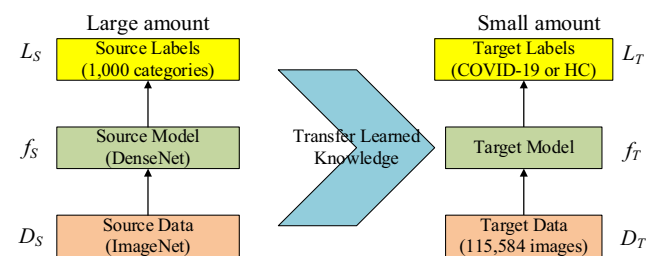


Fig. 4 Idea of transfer learning

$$\text{CNN} : x_l = N_l(x_{l-1}) \quad (29)$$

where l stands for the layer index and N means the non-linear operation. x_l represents the feature from of the l_{th} layer. CNN makes the network challenging to go wider and deeper, as it may face problems of either exploding or gradient vanishing. Then, ResNet employed the shortcut connection by skipping at least two layers.

$$\text{ResNet} : x_l = N_l(x_{l-1}) + x_{l-1} \quad (30)$$

Figure 5b shows its structure, of which the input is x_{l-1} and the output after two conv layers $N_l(x_{l-1})$ is added with the shortcut to input layer x_{l-1} , and thus, the summation is the output of l -th layer.

Further, DenseNet block shown in Fig. 5c revises the model by concatenating all the feature maps $[x_0, x_1, \dots, x_{l-1}]$ sequentially instead of summation of the output feature maps from all previous layers

$$\text{DenseNet} : x_l = N_l(\text{concat}[x_0, x_1, \dots, x_{l-1}]) \quad (31)$$

DenseNet offers concatenations of all feature maps from previous layers [20], which means, all the feature maps propagate to the later layers and connected to the newly generated feature maps. The new developed DenseNet introduce some advantages like feature reuse, decrease the problem of either exploding or gradient vanishing [21].

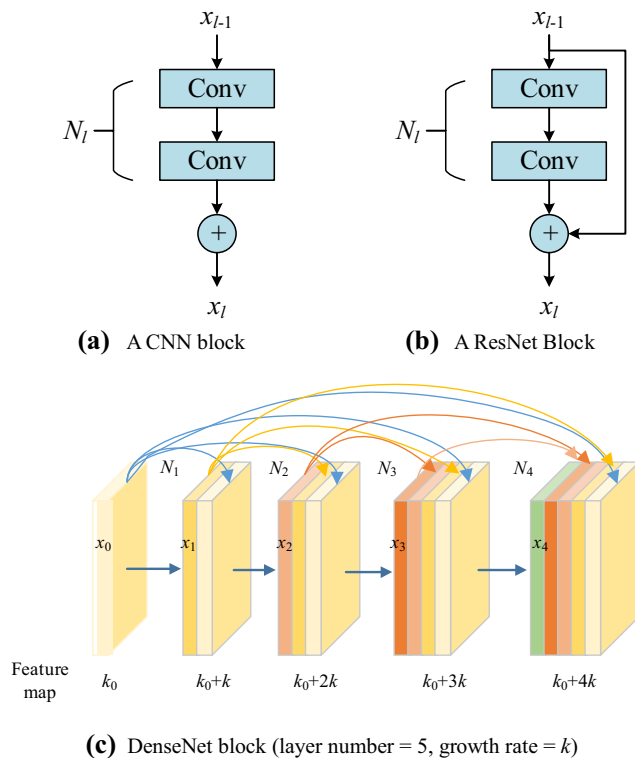


Fig. 5 Comparison of plain CNN block, ResNet block, and DenseNet block

From Fig. 5c, we find that k feature maps are generated for each operation N_l .

$$\begin{aligned} x_1 &= N_1(x_0) \\ x_2 &= N_2(\text{concat}[x_0, x_1]) \\ x_3 &= N_3(\text{concat}[x_0, x_1, x_2]) \\ x_4 &= N_4(\text{concat}[x_0, x_1, x_2, x_3]) \end{aligned} \quad (32)$$

As there are five layers in Fig. 5c, we can get $k_0 + 4k$ feature maps finally as the final feature map is $[x_0, x_1, x_2, x_3, x_4]$; k_0 stands for the number of feature map (x_0) from previous layer. The default value of k in this study is 32.

Note that there are a large number of inputs of the network; a bottleneck layer was introduced to the DenseNet, which is implemented by a 1×1 convolution before the 3×3 convolution layer, which is helpful in reducing the feature maps and saving the computation cost [22].

Transition layers (TLs) were proposed among DenseBlocks. TL has two advantages: (i) It can compress the number of feature maps: suppose k feature maps are generated by a DenseBlock and assume the compression factor as $\theta \in (0, 1]$. Then the feature maps N_{FM} will be reduced to $\lfloor \theta \times k \rfloor$.

$$N_{FM} = \begin{cases} k & \text{before TL} \\ \lfloor \theta \times k \rfloor & \text{after TL} \end{cases} \quad (33)$$

If $\theta = 1$, the number of feature maps will be the same.

(ii) It can downsample the feature maps within the transition layers, i.e., 1×1 conv followed by 2×2 pooling between two consecutive DenseBlock. Feature map sizes are the same within each dense block so that they can be concatenated together easily.

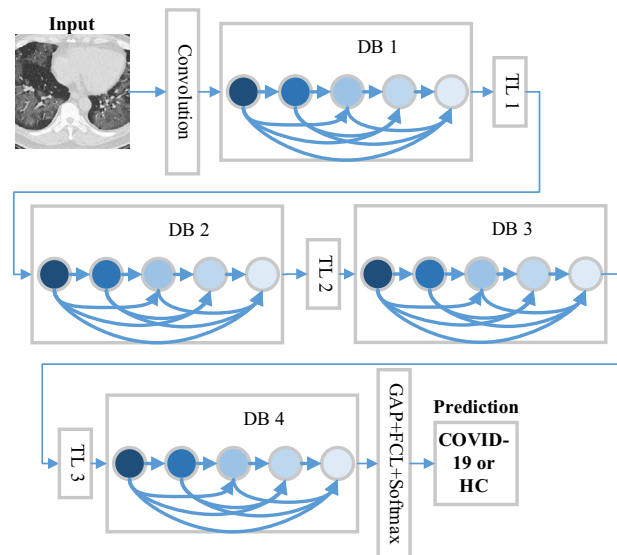


Fig. 6 How DenseNet classify chest CT images (TL transition layer, DB DenseBlock, GAP global average pooling, FCL fully connected layer)

Figure 6 shows the operations in transition layers: batch normalization, convolution, and pooling operations.

Figure 6 illustrates the structure of DenseNet, which includes three DBs, input layer, TLs, and global average pooling (GAP) layer. The TLs consist of a batch normalization layer, a 1×1 conv layer and a 2×2 average pooling layer with stride of 2. Particularly, the GAP is similar to traditional maximum pooling (MP) and average pooling (AP) methods, but it undertakes a more extreme feature map reduction, that reduces the size of feature map S_{FM} from $\omega \times \omega \times c$ to $1 \times 1 \times c$.

$$S_{FM} = \begin{cases} \omega \times \omega \times c & \text{before GAP} \\ 1 \times 1 \times c & \text{after GAP} \end{cases} \quad (34)$$

That means, GAP layer reduces the whole slice into a single digit.

We needed to modify the structure of DenseNet before making it feasible for our COVID-19 diagnosis task. The last dense layer, viz., FCL was modified, since original last FCL was created to classify 1000 categories, of which 20 randomly ones are listed below: recreational vehicle, printer, coho, milk can, Irish wolfhound, parallel bars, tree frog, dhole, Gila monster,

toucan, spider web, organ, walking stick, broccoli, loggerhead, bassoon, colobus, racket, schooner, and Kerry blue terrier. We can observe that none of those 20 categories are related to chest CT or COVID-19, which were the main classification task in this study.

Because the size of output neurons in standard DenseNet (1000-way) does not equal the number of classes in this study (two-way as COVID-19 and healthy control), it is necessary to modify the last FCL and the classification layer.

The modification is presented in Table 2. In this TL environment, a new randomly initialized FCL with 2 output neurons, and a new classification layer with 2 categories (COVID-19 and HC), was used to replace the previous cognate layers. The parameters of softmax layer were updated accordingly.

Improvement IV: Optimization of TL Setting

How to optimize the DenseNet transfer learning setting? There are many hyperparameters need to determine, e.g., do we use DenseNet-121, DenseNet-169, or DenseNet-201? How to we divide the frozen layers, middle layers, and new layers? To solve

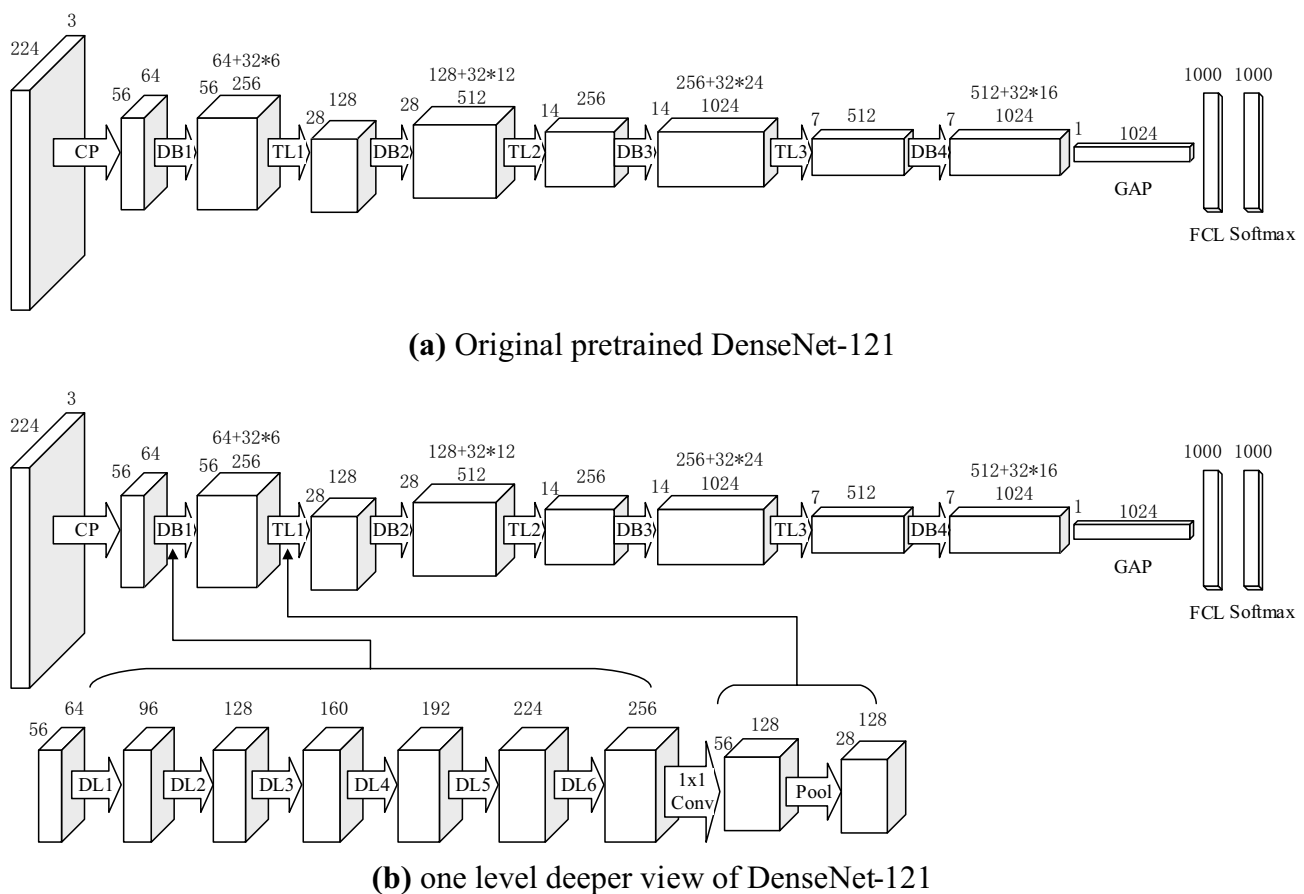


Fig. 7 Structure of DenseNet-121 (CP means the first block of conv layer and pooling layer, DB means the dense block, TL means the transition layer, GAP means global average pooling, FCL means fully connected layer, DL means dense layer)

above issues, we propose an optimization of transfer learning setting (OTLS) framework in this study.

Take DenseNet-121 shown in Fig. 7a as an example, it contains the 121 learnable layers: $(6 + 12 + 24 + 16) * 2 = 116$ layers in the dense blocks and 5 layers from first conv layer, last FCL, and three TLs. The details of DenseNet-121 are shown in Table 4. Figure 7b shows the inner structure of blocks DB1 and TL1.

For DenseNet-201, traditional transfer learning uses simple learning factor (SLF), which employs the same learning factor across all the layers within a neural network. This study employs an advanced strategy called composite learning factor (CLF) [23]: the early transferred layers are frozen with learning factor $LF_{\text{Frozen}} = 0$, i.e., no update, the middle transferred-layers are updated slowly with LF of 1, and the final added new layers learn fast with LF of 10. From the overall viewpoint, there are four different composite-learning-factor-settings (CLFSs) for DenseNet. Table 5 shows the detailed information of CLFSs.

The LF of final new layers is ten times of that of middle transferred-layer, for the middle transferred-layers are with pre-trained weights/biases and new layers are with random-initialized weights/biases. Its structure is shown in Fig. 8a. Here CLF Setting III means the layers from CP to DB3 are frozen. The layers from TL3 to DB4 are transferred directly with pre-trained weights, and they are updated with learning factor $LF_{\text{Middle}} = 1$; note that the learning rate (LR) equals global learning rate (GLR) times learning factor (LF).

$$LR = GLR \times LF \quad (35)$$

The final new layers of FCL are randomly initialized with $LF_{\text{New}} = 10$. In all, LF is chosen as

$$LF = \begin{cases} 0 & \text{Frozen Layers} \\ 1 & \text{Middle Layers} \\ 10 & \text{New Layers} \end{cases} \quad (36)$$

We can observe that LF has three possible associated values to frozen layers, middle layers, and new layers. The LF value configurations of $LF_{\text{Frozen}} = 0$, $LF_{\text{Middle}} = 1$, and $LF_{\text{New}} = 10$ are obtained from Wang [23]. To test other combination values of $(LF_{\text{Frozen}}, LF_{\text{Middle}}, LF_{\text{New}})$ is one of our further research directions.

Table 4 Detailed information of each layer/block in DenseNet-121

Index	Layers	DenseNet-121	Output
1	Input		$224 \times 224 \times 3$
2	Conv	$7 \times 7/2$ conv,	$112 \times 112 \times 64$
3	Pooling	$3 \times 3/2$ MP	$56 \times 56 \times 64$
4	DB1	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$56 \times 56 \times 256$
5	TL1	1×1 conv, $2 \times 2/2$ AP	$28 \times 28 \times 128$
6	DB2	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$28 \times 28 \times 512$
7	TL2	1×1 conv, $2 \times 2/2$ AP	$14 \times 14 \times 256$
8	DB3	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$14 \times 14 \times 1024$
9	TL3	1×1 conv, $2 \times 2/2$ AP	$7 \times 7 \times 512$
10	DB4	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$7 \times 7 \times 1024$
11	Pooling	$7 \times 7/7$ GAP	$1 \times 1 \times 1024$
12	FCL	1000 days	$1 \times 1 \times 1000$

DB dense block, TL transition layer, FCL fully connected layer

Except to optimize the CLF setting, we also need to seek for the optimal DenseNet structure. There are currently three stable DenseNet models: DenseNet-121, DenseNet-169, and DenseNet-201. Those three models have similarity structures with DenseNet-121, except that the number of 1×1 and 3×3 layers. For DenseNet-121, it contains 24 and 16 subblocks in Dense Block 3 and 4, respectively. For DenseNet-169, as shown in Fig. 8b, it includes 32 subblocks in Dense Block both 3 and 4. For DenseNet-201, the numerals change to 48 and 32 within Dense Block 3 and 4, respectively. Figure 8 shows the CLF settings of DenseNet-121 and DenseNet-169 adapted to our COVID-19 diagnosis task.

Precomputation Analysis

Precomputation is employed here. After freezing the layers by setting their cognate learning factor to 0, we can calculate the activation maps at the last frozen layer for all the images in the dataset. Then we save the feature maps to hard drive storage. Those feature maps are used as input images to train the

Table 5 Composite-learning factor setting

CLFS	Frozen layers	Middle layers	New layers
	$LF_{\text{Frozen}} = 0$	$LF_{\text{Middle}} = 1$	$LF_{\text{New}} = 10$
I	CP, DB1, TL1, DB2	TL2, DB3, TL3, DB4	FCL
II	CP, DB1, TL1, DB2, TL2	DB3, TL3, DB4	FCL
III	CP, DB1, TL1, DB2, TL2, DB3	TL3, DB4	FCL
IV	CP, DB1, TL1, DB2, TL2, DB3, TL3	DB4	FCL

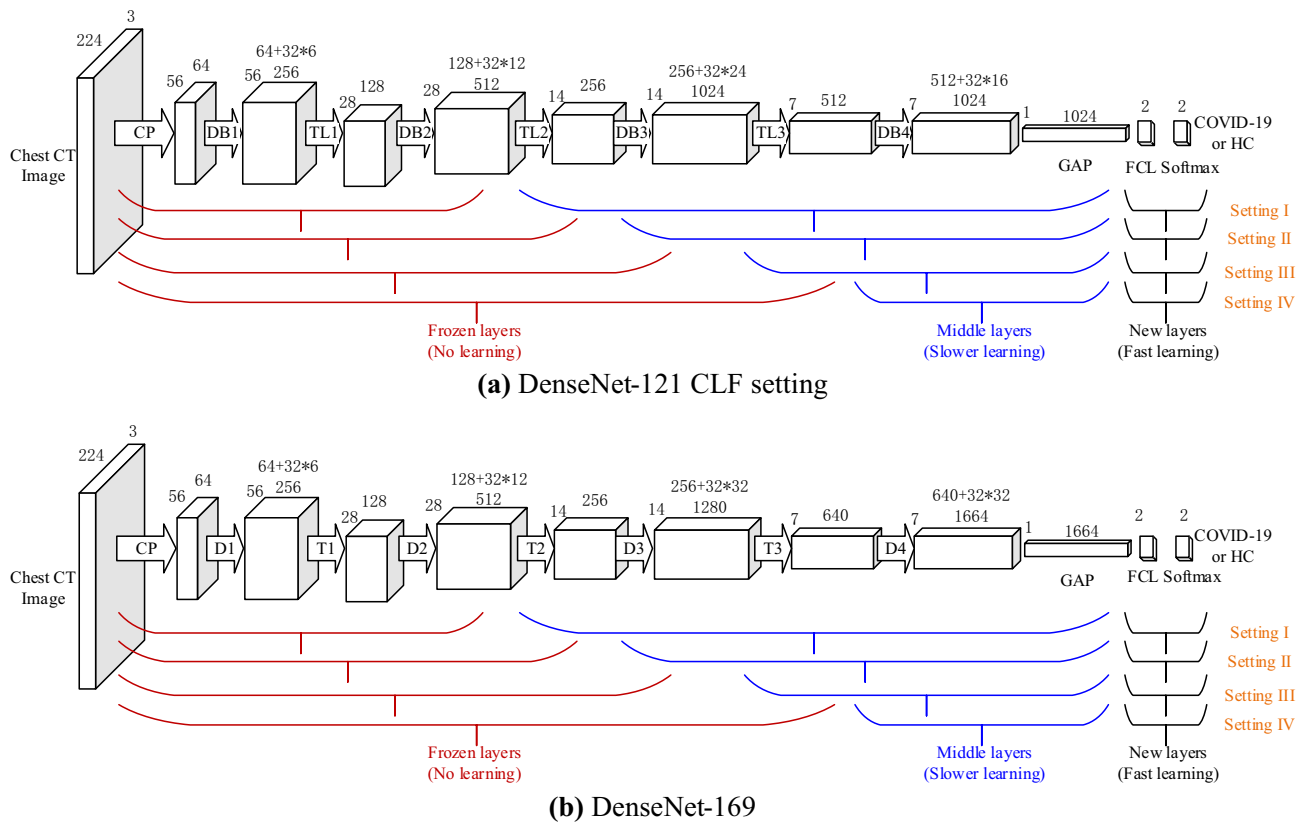


Fig. 8 CLF Setting

trainable layers (middle layers and new layers in Table 5), which can be regarded as a smaller standalone neural network.

Using DenseNet-201 as example, we check the RAM storage. Again, we assume parameters are stored in the format of SPFP format (4 bytes). We can have the storage comparison of all four CLF settings (considering weights, biases, offsets, scales, etc.) in Fig. 9, where y-axis uses log scale for ease of view. We can see CLFS-IV only costs parameters of 6.98 million, and the memory storage of around 27.95 MB, which are the least values of all CLF settings.

Implementation

The experiment ran ten times. At each time, the training-validation hold-out division was reset at random. The test set was kept away from training at the very beginning, so its information would not be leaked to the training procedure. The training stopped when either it reached given maximum epoch, or the validation performance \mathcal{P}_{val} decreased over pre-set training epochs. Stochastic gradient descent with momentum (SGDM) approach was selected as the training algorithm.

Table 6 shows the pseudocode of proposed DenseNet-OTLS method. In Phase I, we use the preprocessing approaches described in “Improvement I: Preprocessing”

to make the data tractable and fit our deep neural network model. In Phase II, we use the proposed OTLS frame to seek the optimal base smodel (BM) and composite learning factor setting (CLFS). The core function here is “TrainNetwork” with four arguments: (i) BM M , (ii) CLFS S , (iii) train data t , and (v) validation data v .

$$\mathbb{M} = \text{TrainNetwork}(\text{BM}, \text{CLFS}, t, v) \quad (37)$$

Using the trained model and new data, we can get the prediction results γ on particular data d by the model \mathbb{M} .

$$\gamma = \text{Predict}(\mathbb{M}, d) \quad (38)$$

With the ground truth labels \mathcal{L} , we can calculate the performance \mathcal{P} .

$$\mathcal{P} = \text{Compare}(\gamma, \mathcal{L}) \quad (39)$$

The performance can be training performance $\mathcal{P}_{\text{train}}$, or validation performance \mathcal{P}_{val} , or test performance $\mathcal{P}_{\text{test}}$, based on the properties of data d . The optimal BM M^* and optimal CLSF S^* can be obtained on validation set by

$$[M^*, S^*] = \text{argmax}[\mathcal{P}_{\text{val}}(M, S)] \quad (40)$$

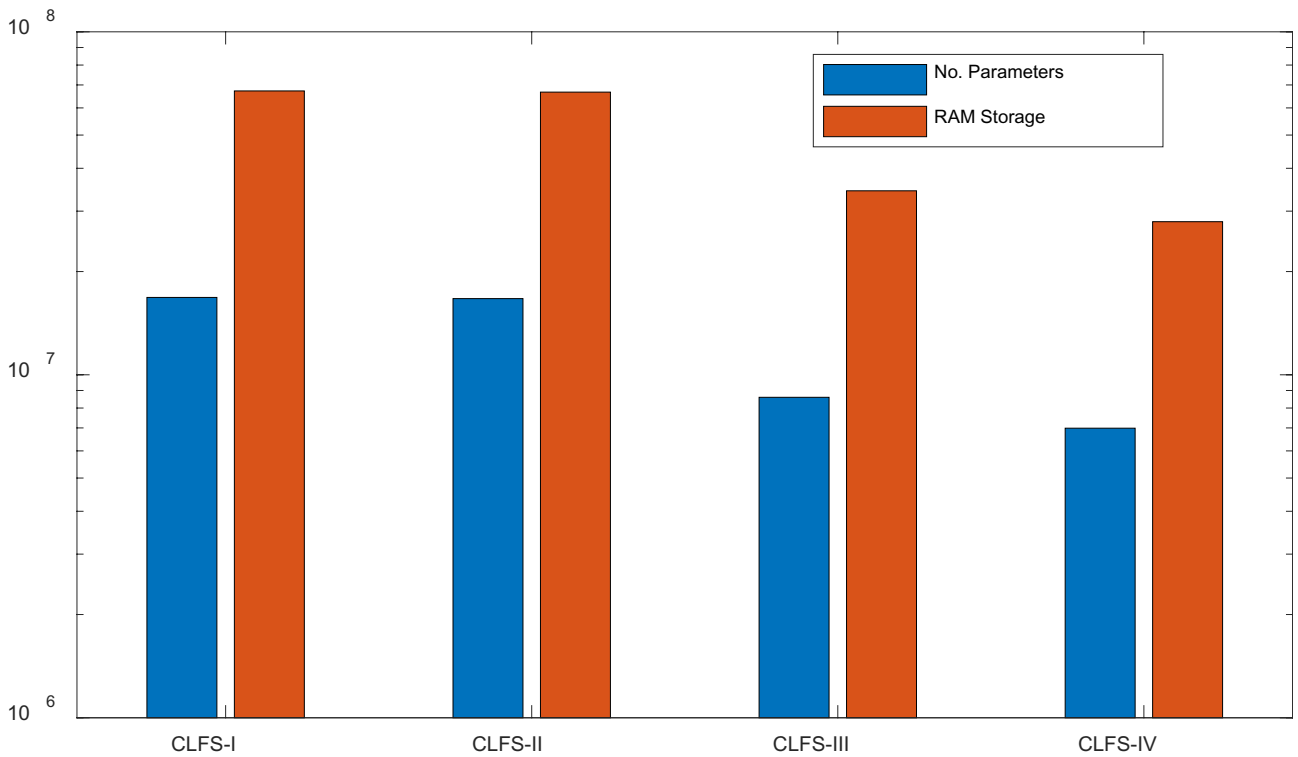


Fig. 9 Storage comparison of four transfer learning settings in DenseNet-201

In Phase III, “Predict” function was conducted on the test set, and we will finally get the test performance $\mathcal{P}_{\text{test}}$.

Indicators

The test performances across all 10 runs were noted, and the six indicators were assessed: sensitivity (SEN), specificity (SPC), accuracy (ACC), precision (PRC), F1 score, and Matthews correlation coefficient (MCC). We used those six indicators because they are widely reported in recent literature. There are some other indicators, but we did not use them, since we believed these six indicators were sufficient to measure the performance of proposed classifiers.

Assume positive means COVID-19, and negative means healthy control. The first four measures were defined as

$$\text{SEN} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (41)$$

$$\text{SPC} = \frac{\text{TN}}{\text{TN} + \text{FP}} \quad (42)$$

$$\text{PRC} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (43)$$

$$\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (44)$$

where TP, FP, TN, and FN represent true positive, false positive, true negative, and false negative, respectively. F1 and MCC are defined as

$$F_1 = \left(\frac{\frac{\text{SEN}^{-1} + \text{PRC}^{-1}}{2}}{2 \times \text{TP}} \right)^{-1} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}} \quad (45)$$

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP}) \times (\text{TP} + \text{FN}) \times (\text{TN} + \text{FP}) \times (\text{TN} + \text{FN})}} \quad (46)$$

The MCC was used in machine learning as an indicator for binary classification since 1975. MCC itself is a correlation coefficient between observation and prediction. Its value is between -1 and 1 , i.e., $-1 \leq \text{MCC} \leq 1$. When MCC equals to -1 , 0 , and 1 , the corresponding meanings are shown below as

$$\text{MCC} = \begin{cases} 1 & \text{perfect classification} \\ 0 & \text{random prediction} \\ -1 & \text{totally wrong classification} \end{cases} \quad (47)$$

Table 6 Pseudocode of our proposed DenseNet-OTLS method

Phase I: Preprocessing	
Step 1	Collect chest CT image dataset \mathbb{X} ;
Step 2	Histogram Stretching: $\mathbb{Y} = \text{HS}(\mathbb{X})$;
Step 3	Crop: Crop the peripheral texts and labels and items: $\mathbb{Z} = \text{Crop}(\mathbb{Y})$;
Step 4	Resizing: Resize the image to [224, 224], get dataset \mathbb{R} , See equation (8)
Step 5	Ground Truth \mathcal{L} : See equation (1).
Phase II: Optimization of Transfer Learning Setting (OTLS)	
Step 6	Split Dataset \mathbb{R} into non-test set $\bar{\mathbb{Z}}$, and test set \mathbb{Z} ;
Step 7	for $S = [\text{CLFS-I}, \text{CLFS-II}, \text{CLFS-III}, \text{CLFS-IV}]$
	<div style="border-left: 1px solid black; padding-left: 10px;"> CLFS = S; for $L = [121, 169, 201]$ <div style="border-left: 1px solid black; padding-left: 10px;"> $M = \text{DenseNet}(L)$; for $r = 1:10$ <div style="border-left: 1px solid black; padding-left: 10px;"> Split dataset non-test set $\bar{\mathbb{Z}}$ into training set X_r, validation set Y_r; $\text{Model}(M, S, r) = \text{TrainNetwork}[M, S, X_r, Y_r]$; $\text{Result}(M, S, r) = \text{Predict}[\text{Model}(M, S, r), Y_r]$; $\mathcal{P}_{\text{val}}(M, S, r) = \text{Compare}(\text{Result}(M, S, r), \mathcal{L})$; end $\mathcal{P}_{\text{val}}(M, S) = \text{mean}_r[\mathcal{P}_{\text{val}}(M, S, r)]$, end end $\mathcal{P}_{\text{val}}(M, S) = \text{mean}_r[\mathcal{P}_{\text{val}}(M, S)]$, end end </div> </div> </div>
Step 8	$[M^*, S^*] = \text{argmax}[\mathcal{P}_{\text{val}}(M, S)]$,
Phase III: Output the unbiased performance	
Step 9	for $r = 1:10$
	<div style="border-left: 1px solid black; padding-left: 10px;"> Split dataset non-test set $\bar{\mathbb{Z}}$ into training set X_r, validation set Y_r; $\text{Model}(r) = \text{TrainNetwork}[M^*, S^*, X_r, Y_r]$; $\text{Result}(r) = \text{Predict}[\text{Model}(r), \mathbb{Z}]$; $\mathcal{P}_{\text{test}}(r) = \text{Compare}[\text{Result}(r), \mathcal{L}]$; end </div>
Step 10	Output $\mathcal{P}_{\text{test}} = \text{mean}_r[\mathcal{P}_{\text{test}}(r)]$.

The average and standard deviation (SD) of six indicators of ten runs on the test set were analyzed and used for comparison. Two important global hyperparameters were set below: (i) the number of training epochs was set to 10, since the whole training procedure of a transfer learning should be swift. (ii) The global learning rate (GLR) was set to a trivial value of 10^{-4} to slow down learning, because transfer learning will be carried out on a pre-trained DenseNet model.

Results and Discussions

Optimization of Transfer Learning Setting

For the three DenseNet variants and four CLF settings, the comparison results were listed in Table 7. The best two are DenseNet-201-CLFS-III and DenseNet-201-CLFS-IV. The former obtains a sensitivity of $96.41 \pm 1.86\%$, a specificity of $96.88 \pm 1.21\%$, a precision of $96.90 \pm 1.18\%$, an accuracy of $96.64 \pm 1.21\%$, an F1 score of $96.63 \pm 1.24\%$, and an MCC of $93.32 \pm 2.42\%$. The latter one produces a sensitivity of $96.88 \pm 1.85\%$, a specificity of $96.72 \pm 1.47\%$, a precision of $96.76 \pm 1.39\%$, an accuracy of $96.80 \pm 0.82\%$, an F1 score of $96.79 \pm 0.84\%$, and an MCC of $93.65 \pm 1.60\%$. Considering CLFS IV will use less storage, we finally choose “201-IV” as our best model.

Figure 10 shows the error bar plot of \mathcal{P}_{val} , showing that DenseNet-201 yields better performances than DenseNet-121 and DenseNet-169 overall. The reason may be because DenseNet-201 has the deepest neural structure; thus, it can map more complicated patterns, such as the ground-glass opacity (GGO) lesions of a chest CT image of COVID-19 patients.

The detailed information of 10 runs of the best model “201-IV” is shown in Table 8. Each row shows the result of one run. The last row shows the mean and standard deviation of all 10 runs. From the last row, we can see the standard deviation values of accuracy and F1 are much smaller than those of other four indicators.

Performance on Test Set via “201-IV”

Since we have determined from the validation set that the optimal combination is “201-IV,” we run this proposed model on the test set for 10 new runs, with results of each run shown in Table 9.

The comparison between Table 9 with Table 7 is shown in Fig. 11, from which we can see the following:

(i) The performance on test set $\mathcal{P}_{\text{test}}$ is a bit lower than that performance on validation set \mathcal{P}_{val} in terms of all six measures. The reason is the test set is brand new data to the models, but validation set contributes to the trained model. So \mathcal{P}_{val} looks better than $\mathcal{P}_{\text{test}}$. (ii) We find the SD on test set is smaller than

Table 7 Validation performance based on the best configuration (unit: %)

Run	Sen	Spc	Prc	Acc	F1	MCC
R1	98.44	96.88	96.92	97.66	97.67	95.34
R2	93.75	98.44	98.39	96.09	96.01	92.30
R3	98.44	93.75	94.03	96.09	96.18	92.30
R4	96.88	96.88	96.88	96.88	96.88	93.75
R5	98.44	95.31	95.45	96.88	96.92	93.80
R6	98.44	98.44	98.44	98.44	98.44	96.88
R7	95.31	98.44	98.44	96.88	96.82	93.84
R8	93.75	96.88	96.77	95.31	95.21	90.71
R9	96.88	96.88	96.87	96.88	96.85	93.80
R10	98.44	95.31	95.45	96.88	96.92	93.80
Mean \pm SD	96.88 \pm 1.85	96.72 \pm 1.47	96.76 \pm 1.39	96.80 \pm 0.82	96.79 \pm 0.84	93.65 \pm 1.60

that on validation set. The reason is because the size of test set is larger than validation set, as can be found in Table 3.

Comparison to State-of-the-Art Approaches

We compare our method “DenseNet-OTLS” with other COVID-19 classification approaches: RBFNN [2], K-ELM [3], ELM-BA [4], 6L-CNN [5], GoogLeNet [6],

and ResNet-18 [7]. The comparison results based on test set of 10 runs are shown in Table 10, where it is clear that proposed approach is significantly better than all the six state-of-the-art methods:

In all, the comparison results of eight methods are shown in Fig. 12. This picture indicates that this proposed DenseNet-OTLS approach can achieve the highest performance among all state-of-the-art approaches.

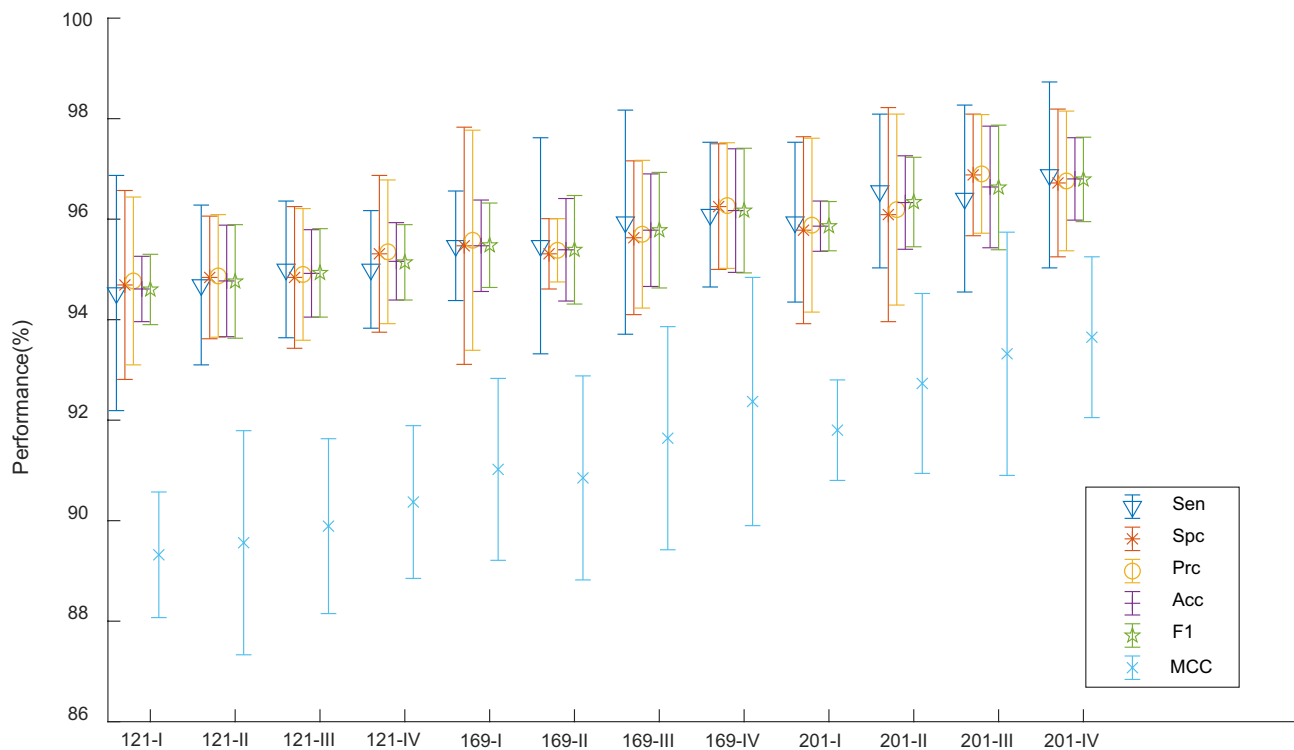
**Fig. 10** Error bar of validation performances

Table 8 Validation performance \mathcal{P}_{val} (Unit: %)

Model	CLFS	Sensitivity	Specificity	Precision	Accuracy	F1	MCC
DenseNet-121	I	94.53 \pm 2.34	94.69 \pm 1.88	94.77 \pm 1.67	94.61 \pm 0.65	94.60 \pm 0.70	89.32 \pm 1.25
	II	94.69 \pm 1.59	94.84 \pm 1.22	94.87 \pm 1.22	94.77 \pm 1.11	94.76 \pm 1.13	89.56 \pm 2.23
	III	95.00 \pm 1.36	94.84 \pm 1.41	94.90 \pm 1.31	94.92 \pm 0.87	94.93 \pm 0.88	89.89 \pm 1.74
	IV	95.00 \pm 1.17	95.31 \pm 1.56	95.35 \pm 1.43	95.16 \pm 0.77	95.14 \pm 0.75	90.37 \pm 1.52
DenseNet-169	I	95.47 \pm 1.09	95.47 \pm 2.36	95.58 \pm 2.19	95.47 \pm 0.91	95.48 \pm 0.84	91.02 \pm 1.81
	II	95.47 \pm 2.15	95.31 \pm 0.70	95.38 \pm 0.63	95.39 \pm 1.02	95.39 \pm 1.08	90.85 \pm 2.03
	III	95.94 \pm 2.23	95.63 \pm 1.53	95.70 \pm 1.47	95.78 \pm 1.12	95.78 \pm 1.15	91.64 \pm 2.22
	IV	96.09 \pm 1.44	96.25 \pm 1.25	96.27 \pm 1.25	96.17 \pm 1.23	96.17 \pm 1.24	92.37 \pm 2.47
DenseNet-201	I	95.94 \pm 1.59	95.78 \pm 1.86	95.88 \pm 1.73	95.86 \pm 0.50	95.86 \pm 0.49	91.80 \pm 1.00
	II	96.56 \pm 1.53	96.09 \pm 2.13	96.19 \pm 1.90	96.33 \pm 0.93	96.34 \pm 0.89	92.73 \pm 1.79
	III	96.41 \pm 1.86	96.88 \pm 1.21	96.90 \pm 1.18	96.64 \pm 1.21	96.63 \pm 1.24	93.32 \pm 2.42
	IV	96.88 \pm 1.85	96.72 \pm 1.47	96.76 \pm 1.39	96.80 \pm 0.82	96.79 \pm 0.84	93.65 \pm 1.60

Table 9 Test performance \mathcal{P}_{test} based on best model “201-IV” found by proposed DenseNet-OTLS (unit: %)

Run	Sen	Spc	Prc	Acc	F1	MCC
R1	96.88	96.88	96.94	96.88	96.90	93.77
R2	97.92	94.79	94.98	96.35	96.42	92.76
R3	95.83	97.92	97.92	96.88	96.85	93.79
R4	95.83	97.92	97.87	96.88	96.84	93.77
R5	97.92	95.83	95.96	96.88	96.92	93.79
R6	96.88	94.79	94.90	95.83	95.88	91.69
R7	94.79	96.88	96.83	95.83	95.79	91.71
R8	95.83	94.79	94.81	95.31	95.30	90.68
R9	94.79	96.88	96.78	95.83	95.77	91.70
R10	96.88	95.83	95.88	96.35	96.37	92.72
Mean \pm SD	96.35 \pm 1.07	96.25 \pm 1.16	96.29 \pm 1.11	96.30 \pm 0.54	96.30 \pm 0.56	92.64 \pm 1.08

Composite Learning Rate Versus Simple Learning Rate

We compare this composite learning factor (CLF) strategy with traditional simple learning factor (SLF), as described in “Improvement IV: Optimization of TL Setting.” SLF is designed here as that we set the frozen layers with learning factor of 0, and the rest layers with learning factor of 1. We compare our CLF result with SLF and show the comparison in Table 11 and Fig. 13.

The results indicate that CLF yields better performance than SLF. The reason is all the layers are divided into three types (FL, ML, and NL) using CLF: (i) frozen layers (FLs) inherit structure and weights from pretrained models; (ii) middle layers (MLs) inherit network structure, and use pretrained weights as initial; and (iii) newly layers (NLs) have no relevancy with PTMs.

Effect of Preprocessing

We justify the effectiveness of preprocessing in this experiment. Remember our preprocessing consists of three steps: (i) HS, (ii) crop, and (iii) resize. Suppose

we do not do this three-step preprocessing, i.e., we only carry out the resizing due to the transfer learning requirement. The comparison is carried out over test set and the results are itemized in Table 12.

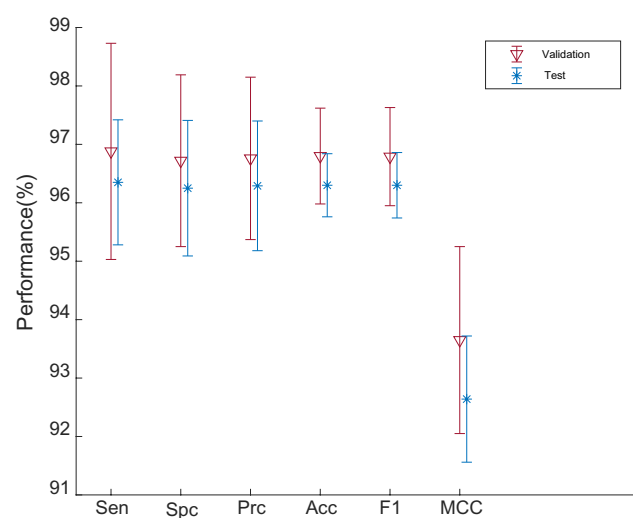
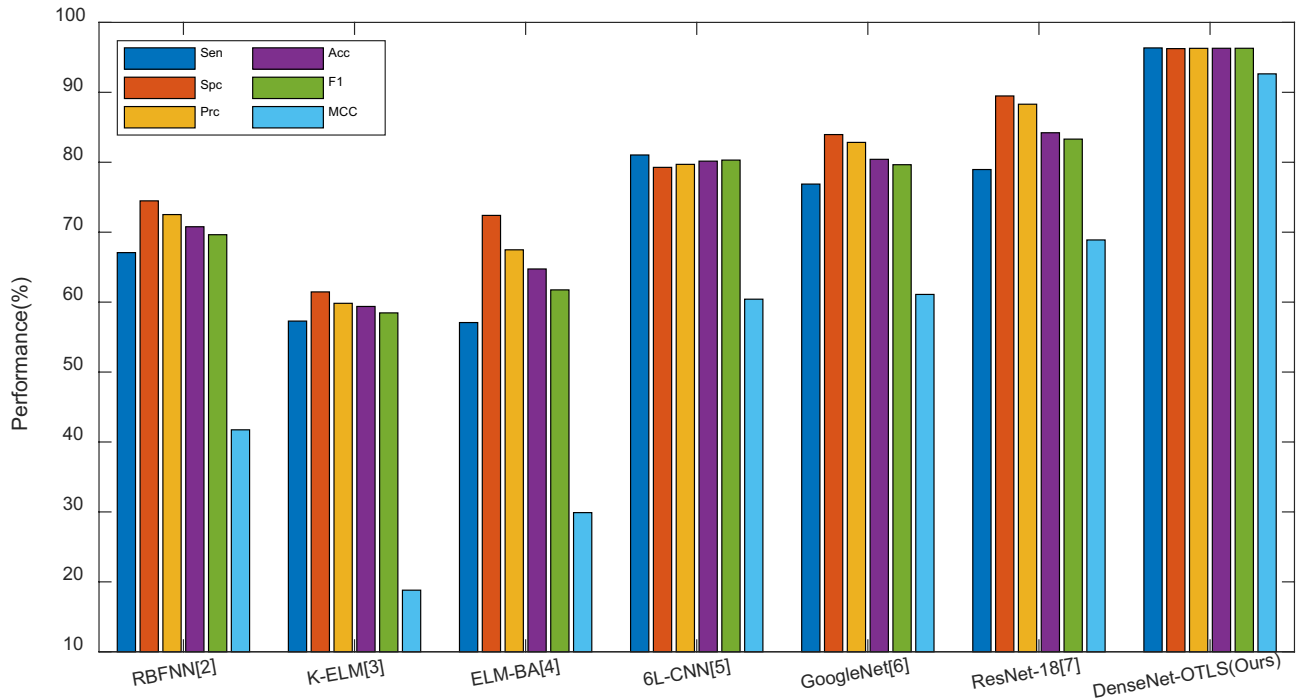
**Fig. 11** Comparison of validation performance and test performance

Table 10 Comparison with COVID-19 identification approaches (Unit: %)

Approach	Sen	Spc	Prc	Acc	F1	MCC
RBFNN [2]	67.08	74.48	72.52	70.78	69.64	41.74
K-ELM [3]	57.29	61.46	59.83	59.38	58.46	18.81
ELM-BA [4]	57.08 \pm 3.86	72.40 \pm 3.03	67.48 \pm 1.65	64.74 \pm 1.26	61.75 \pm 2.24	29.90 \pm 2.45
6L-CNN [5]	81.04 \pm 2.90	79.27 \pm 2.21	79.70 \pm 1.27	80.16 \pm 0.85	80.31 \pm 1.13	60.42 \pm 1.73
GoogLeNet [6]	76.88 \pm 3.92	83.96 \pm 2.29	82.84 \pm 1.58	80.42 \pm 1.40	79.65 \pm 1.92	61.10 \pm 2.62
ResNet-18 [7]	78.96 \pm 2.90	89.48 \pm 1.64	88.30 \pm 1.50	84.22 \pm 1.23	83.31 \pm 1.53	68.89 \pm 2.33
DenseNet-OTLS (Ours)	96.35 \pm 1.07	96.25 \pm 1.16	96.29 \pm 1.11	96.30 \pm 0.54	96.30 \pm 0.56	92.64 \pm 1.08

**Fig. 12** Comparison of our method with seven state-of-the-art approaches**Table 11** Composite learning factor versus simple learning factor (Unit: %)

Setting	Sen	Spc	Prc	Acc	F1	MCC
SLF	94.37 \pm 1.06	94.58 \pm 1.60	94.62 \pm 1.50	94.48 \pm 0.78	94.48 \pm 0.76	89.00 \pm 1.56
CLF(Ours)	96.35 \pm 1.07	96.25 \pm 1.16	96.29 \pm 1.11	96.30 \pm 0.54	96.30 \pm 0.56	92.64 \pm 1.08

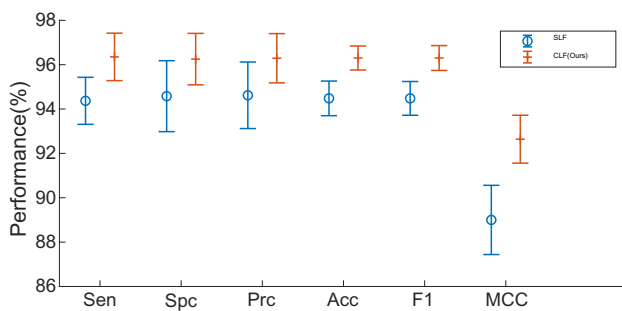
**Fig. 13** Error bar plot of CLF versus SFL

Table 12 indicate that without proper preprocessing (HS and Crop), the performance of the system will decrease to a sensitivity of $93.33 \pm 2.21\%$, a specificity of $91.77 \pm 2.75\%$, and an accuracy of $92.55 \pm 2.07\%$. This comparison clearly indicates the effectiveness of this proposed three-step preprocessing.

Conclusion

This paper proposed a novel COVID-19 diagnosis method based on DenseNet and optimization of transfer learning setting (OTLS) framework. The OTLS includes optimization the

Table 12 Effectiveness of proposed three-step preprocessing

Setting	Sen	Spc	Prc	Acc	F1	MCC
Only resizing	93.33 \pm 2.21	91.77 \pm 2.75	91.94 \pm 2.55	92.55 \pm 2.07	92.61 \pm 2.02	85.14 \pm 4.13
Preprocessing (Ours)	96.35 \pm 1.07	96.25 \pm 1.16	96.29 \pm 1.11	96.30 \pm 0.54	96.30 \pm 0.56	92.64 \pm 1.08

composite learning factor setting (CLFS) and optimization the DenseNet structure. The experiments showed our method DenseNet-OTLS is superior to six state-of-the-art approaches.

The shortcomings of this research are two-fold: (i) We did not validate the optimal combination configuration of DA techniques. We shall quantify the effect of each DA technique. (ii) We did not validate the optimal values of (LF_{Frozen} , LF_{Middle} , LF_{New}). We will try to develop some automatic learning factor optimization method.

Furthermore, we shall try to increase COVID-19's diagnosis performance further. One solution way is to make combination of different transfer learning setting, creating an ensemble DenseNet deep neural network. Another research direction is to output the localization the lesions of COVID-19, which can assist the chest radiologists to make more accurate diagnosis. (see. Table 13)

Highlights

- DenseNet was introduced as the backbone pre-trained model, and we modified it to this COVID-19 diagnosis task.
- Composite learning factor strategy was used for training DenseNet.
- Data augmentation was used to enhance the training set.
- An optimization of transfer learning setting (OTLS) was proposed to search for the optimal optimization setting.
- Precomputation was introduced to save memory.
- We compared proposed DenseNet-OTLS method with state-of-the-art COVID-19 diagnosis approaches.

Funding This paper is partially supported by Royal Society International Exchanges Cost Share Award, UK (RP202G0230); Medical Research Council Confidence in Concept Award, UK (MC_PC_17171); Hope Foundation for Cancer Research, UK (RM60G0680); Henan Key Research and Development Project (182102310629); Natural Science Foundation of China (61602250, 11502090, U1711263, U1811264); Guangxi Key Laboratory of Trusted Software (kx201901); Natural Science Foundation of Zhejiang Province (Y18F010018); British Heart Foundation Accelerator Award, UK; Fundamental Research Funds for the Central Universities (CDLS-2020-03); and Key Laboratory of Child Development and Learning Science (Southeast University), Ministry of Education.

Compliance with Ethical Standards

Informed Consent Informed consent was obtained from all individual participants included in the study.

Research Involving Human Participants and/or Animals All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Helsinki declaration and its later amendments or comparable ethical standards.

Table 13 Abbreviations and their full names

Abbreviation	Full name
AP	Average pooling
BM	Base model
CLF	Composite learning factor
CLFS	Composite learning factor setting
CNN	Convolutional neural network
CT	Computed Tomography
DA	Data augmentation
DB	DenseBlock
FCL	Fully connected layer
FL	Frozen layer
GAP	Global average pooling
GGO	Ground-glass opacity
GLR	Global learning rate
HC	Healthy control
HS	Histogram stretching
HU	Hounsfield unit
ILSVRC	ImageNet large scale visual recognition challenge
LF	Learning factor
LR	Learning rate
LRN	Local response normalization
MCC	Matthews correlation coefficient
ML	Middle layer
MP	Maximum pooling
MV	Majority voting
NL	New layer
OTLS	Optimization of transfer learning setting
PCR	Polymerase chain reaction
PTM	Pre-trained model
RHO	Random hold-out
SD	Standard deviation
SGDM	Stochastic gradient descent with momentum
SLF	Simple learning factor
SPFP	Single-precision floating-point
TL	Transfer learning
TL	Transition layer

Appendix

References

1. Ai T, et al. Correlation of chest CT and RT-PCR testing in coronavirus disease 2019 (COVID-19) in China: a report of 1014 cases. *Radiology*. 2020: Article ID. 200642.
2. Lu Z. A pathological brain detection system based on radial basis function neural network. *J Med Imaging Health Infor*. 2016;6(5):1218–22.
3. Yang J. A pathological brain detection system based on kernel based ELM. *Multimed Tools Appl*. 2018;77(3):3715–28.
4. Lu S. A pathological brain detection system based on extreme learning machine optimized by bat algorithm. *CNS & Neurological Disorders - Drug Targets*. 2017;16(1):23–9.
5. Jiang X. Chinese sign language fingerspelling recognition via six-layer convolutional neural network with leaky rectified linear units for therapy and rehabilitation. *J Med Imaging Health Infor*. 2019;9(9):2031–8.
6. Szegedy C, et al. Going deeper with convolutions. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015. p. 1–9.
7. Yu X, et al. Abnormality diagnosis in mammograms by transfer learning based on ResNet18. *Fundam Inform*. 2019;168(2–4):219–30.
8. Rajaraman S, et al. Weakly Labeled Data Augmentation for Deep Learning: A Study on COVID-19 Detection in Chest X-Rays. *Diagnostics*. 2020;10(6): p. 17: Article ID. 358.
9. Ni QQ, et al. A deep learning approach to characterize 2019 coronavirus disease (COVID-19) pneumonia in chest CT images. *Eur Radiol*. 2020: p. 11.
10. Novitasari DCR, et al. Detection of covid-19 chest x-ray using support vector machine and convolutional neural network. *Commun Math Biol Neurosci*. 2020: p. 19: Article ID. 42.
11. Veluchamy M, et al. Image contrast and color enhancement using adaptive gamma correction and histogram equalization. *Optik*. 2019;183:329–37.
12. Adesuyi TA, et al. A neuron noise-injection technique for privacy preserving deep neural networks. *Open Comput Sci*. 2020;10(1):137–52.
13. Fischetti MV, et al. Mermin-Wagner theorem, flexural modes, and degraded carrier mobility in two-dimensional crystals with broken horizontal mirror symmetry. *Phys Rev B*. 2016;93(15): p. 13: Article ID. 155413.
14. Shivakumar PG. et al. Transfer learning from adult to children for speech recognition: evaluation, analysis and recommendations. *Comput Speech Lang*. 2020;63: p. 22: Article ID. Unsp 101077.
15. Krizhevsky A, et al. ImageNet classification with deep convolutional neural networks, in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*. 2012, Curran Associates Inc.: Lake Tahoe, Nevada. p. 1097–1105.
16. Simonyan K, et al. Very deep convolutional networks for large-scale image recognition. in *International Conference on Learning Representations (ICLR)*. 2015. San Diego, CA, USA: Computational and Biological Learning Society. p. 1–14.
17. Szegedy C, et al. Going deeper with convolutions. in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015. p. 1–9.
18. He K, et al. Deep Residual Learning for Image Recognition, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016. p. 9.
19. Huang G, et al. Densely Connected Convolutional Networks. in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017. p. 2261–2269.
20. Saleh K, et al. Spatio-temporal DenseNet for real-time intent prediction of pedestrians in urban traffic environments. *Neurocomputing*. 2020;386:317–24.
21. Heo WH, et al. Source Separation Using Dilated Time-Frequency DenseNet for Music Identification in Broadcast Contents. *Appl Sci-Basel*. 2020. 10(5): p. 18: Article ID. 1727.
22. Jalali A, et al. High cursive traditional Asian character recognition using integrated adaptive constraints in ensemble of DenseNet and Inception models. *Pattern Recogn Lett*. 2020;131:172–7.
23. Wang SH. DenseNet-201-based deep neural network with composite learning factor and precomputation for multiple sclerosis classification. *ACM Trans Multimed Comput Commun Appl*. 2020. 16(2s): Article ID. 60.