Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Computers in Biology and Medicine 150 (2022) 106151

Contents lists available at ScienceDirect

Computers in Biology and Medicine

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

NSCGCN: A novel deep GCN model to diagnosis COVID-19

Chaosheng Tang [a,1], Chaochao Hu [a,1], Junding Sun [a,***], Shui-Hua Wang [a,b,c,**], Yu-Dong Zhang [a,c,d,*]

[a] School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo, Henan, 454000, PR China
[b] School of Architecture Building and Civil Engineering, Loughborough University, Loughborough, LE11 3TU, UK
[c] School of Computing and Mathematical Sciences, University of Leicester, Leicester, LE1 7RH, UK
[d] Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, 21589, Saudi Arabia

ARTICLE INFO

Keywords:
COVID-19
Deep graph convolutional network
Densely connection
Transfer learning
Disease diagnosis

ABSTRACT

Aim: Corona Virus Disease 2019 (COVID-19) was a lung disease with high mortality and was highly contagious. Early diagnosis of COVID-19 and distinguishing it from pneumonia was beneficial for subsequent treatment.
Objectives: Recently, Graph Convolutional Network (GCN) has driven a significant contribution to disease diagnosis. However, limited by the nature of the graph convolution algorithm, deep GCN has an over-smoothing problem. Most of the current GCN models are shallow neural networks, which do not exceed five layers. Furthermore, the objective of this study is to develop a novel deep GCN model based on the DenseGCN and the pre-trained model of deep Convolutional Neural Network (CNN) to complete the diagnosis of chest X-ray (CXR) images.
Methods: We apply the pre-trained model of deep CNN to perform feature extraction on the data to complete the extraction of pixel-level features in the image. And then, to extract the potential relationship between the obtained features, we propose Neighbourhood Feature Reconstruction Algorithm to reconstruct them into graph-structured data. Finally, we design a deep GCN model that exploits the graph-structured data to diagnose COVID-19 effectively. In the deep GCN model, we propose a Node-Self Convolution Algorithm (NSC) based on feature fusion to construct a deep GCN model called NSCGCN (Node-Self Convolution Graph Convolutional Network).
Results: Experiments were carried out on the Computed Tomography (CT) and CXR datasets. The results on the CT dataset confirmed that: compared with the six state-of-the-art (SOTA) shallow GCN models, the accuracy and sensitivity of the proposed NSCGCN had improve 8% as sensitivity (Sen.) = 87.50%, F1 score = 97.37%, precision (Pre.) = 89.10%, accuracy (Acc.) = 97.50%, area under the ROC curve (AUC) = 97.09%. Moreover, the results on the CXR dataset confirmed that: compared with the fourteen SOTA GCN models, sixteen SOTA CNN transfer learning models and eight SOTA COVID-19 diagnosis methods on the COVID-19 dataset. Our proposed method had best performances as Sen. = 96.45%, F1 score = 96.45%, Pre. = 96.61%, Acc. = 96.45%, AUC = 99.22%.
Conclusion: Our proposed NSCGCN model is effective and performed better than the thirty-eight SOTA methods. Thus, the proposed NSC could help build deep GCN models. Our proposed COVID-19 diagnosis method based on the NSCGCN model could help radiologists detect pneumonia from CXR images and distinguish COVID-19 from Ordinary Pneumonia (OPN). The source code of this work will be publicly available at https://github.com/TangChaosheng/NSCGCN.

* Corresponding author. School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo, Henan, 454000, PR China.
** Corresponding author. School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo, Henan, 454000, PR China.
*** Corresponding author.
E-mail addresses: tcs@hpu.edu.cn (C. Tang), chaos-hu@home.hpu.edu.cn (C. Hu), sunjd@hpu.edu.cn (J. Sun), shuihuawang@ieee.org (S.-H. Wang), yudongzhang@ieee.org (Y.-D. Zhang).
[1] Those two authors contributed equally to this paper.

https://doi.org/10.1016/j.compbiomed.2022.106151
Received 9 July 2022; Received in revised form 5 September 2022; Accepted 24 September 2022
Available online 30 September 2022

1. Introduction

The COVID-19 global pandemic is a public health event [1] that triggered a global health crisis. As of 10 December 2021, more than 267, 865,289 cases of infection have been confirmed worldwide, and greater than 5,285,888 patients died. COVID-19 is an infection caused by a new type of coronavirus, which can be spread through respiratory droplets, close contact, and high-concentration aerosols. COVID-19 can cause breathing difficulties in severe cases and rapidly develop into acute respiratory distress syndrome, septic shock, metabolic acidosis, and even death by multiple organ failure. Consequently, early diagnosis and treatment are vital in preventing diseases from becoming severe.

COVID-19 is usually diagnosed by reverse transcription-polymerase chain reaction (RT-PCR), which can detect novel coronavirus [2]. However, its sensitivity is not high enough [3]. As supplementary diagnosis methods, CT and CXR can improve the diagnosis rate of COVID-19 and reduce the false-positive rate. Because of the low-cost and low-radiation dose, the CXR images have been widely used in COVID-19 diagnosis [4].

Nonetheless, the manual interpretation of images by radiologists is a time-consuming task and is susceptible to the internal factors of the experts (such as fatigue, emotions, etc.). In addition, the X-ray images between COVID-19 and OPN have similar image features. Therefore, it is necessary to solve this problem by auxiliary diagnosis technology combined with computer vision and artificial intelligence.

The GCN are a hot subject in artificial neural networks research. It can effectively extract features based on a non-Euclidean structure. Recent research has suggested that GCN achieved excellent results in medical image analysis. For example, Zhang et al. [5] proposed a memory-based GCN model to diagnose Parkinson's disease. Song et al. [6] used the topology of brain nerves to generate cognitive state category labels and then fulfil the task of advertising diagnosis by GCN. Chu et al. [7] considered the potential complementary topological information on different spatial scales and proposed a multi-scale graph representation learning framework, which uses multi-scale GCN representation learning for Autism identification. Song et al. [8] proposed a multi-centre attention map with each node representing a topic to consider the influence of data source, gender, acquisition equipment and disease status of these training samples in GCN, which improve the diagnostic accuracy of early AD. Ye et al. [9] used a GCN to capture the topology of the region of interest (ROI) images and complete breast cancer screening. Elazab et al. [10] proposed a multi-site (centre) graph convolutional network with a supervision mechanism for COVID-19 diagnosis from X-ray radiographs.

In deep learning predictors, deeper neural networks help improve the performance of the model [11–13]. Furthermore, as the research went further and more detailed, the deep GCN caused extensive concern in the academic circle. However, some experiments have further concluded that, because of the excess layers stacked in the GCN models, all nodes' representation converges to a fixed point, independent of node features. The excess layers lead to the problem of gradient disappearance [14]. As a result, most of the latest GCN models do not exceed five layers. Thomas N. Kipf and Welling [15] tried to use Dropout to solve the problem, but it could not effectively prevent the occurrence of over-smoothing. Li et al. [16] proposed applying the residual connection and densely connected of CNN to GCN and made some progress in the point cloud segmentation task, but they did not solve the problem of growing volumes of parameters caused by densely connected. Although 56 layers of DenseGCN can be trained, it can only be run on a small dataset.

We propose the NSCGCN model to distinguish Healthy Control (HC), COVID-19, and OPN. In our proposed model, NSC has been designed to compress the number of feature maps. It was significant for guaranteeing the integrality and authentication of information transmitted by the feature. Furthermore, we have trained a deep GCN model with more than 200 layers, and the growth rate of DenseGCN parameters was reduced. Our proposed NSCGCN model has been applied to the public X-ray dataset to demonstrate its effectiveness. The experimental results showed that the NSCGCN could achieve better than other GCN and deep CNN models.

We aim to create a deep GCN model with the proposed NSC algorithm to distinguish between HC, COVID-19, and OPN. The contributions are as follows:

(i) A novel deep GCN has been created for medical image classification for the first time. Deep GCN has a solid nonlinear fitting ability, which can extract the invisible relationship between features and improve the classification performance;
(ii) A node-self convolution algorithm based on the graph has been proposed in our proposed model. It not only realizes the interaction and integration of cross-channel information but also reduces the dimension of the feature and the parameters of the convolution kernel;
(iii) A new feature reconstruction algorithm is proposed. It can retain the spatial structure information of the original feature maps;
(iv) Deep CNN and DenseGCN have been introduced as the backbone model and modified for COVID-19 distinguish task;
(v) We have compared the proposed COVID-19 diagnosis method with SOTA GCN, deep CNN, and COVID-19 diagnosis methods.

The arrangement of this paper is as follows: First, the related work of GCN is briefly introduced in Section 2. Section 3 introduces the framework proposed and the dataset in detail. Then the experimental settings are introduced in detail, and the experimental results and discussions are shown in Section 4. Finally, the conclusion is shown in Section 5.

2. Related work

2.1. Graph-structured data acquisition

There are two difficult problems in Deep GCN. One of the most significant challenges is that the inputs of GCN should be graph-structured data [17]. Two standard methods have been presented to convert medical image data structure. The direct way to collect medical graph-structured data is to use extremely specialized medical image instruments and equipment. It can ensure the integrity of data features. In the population graph [18], the nodes were patients' MRIs processed by professional equipment such as the software connectome calculation scheme, configurable pipeline for the study of connectomes, and data analysis for resting-state fMRI and neuroimaging analysis kit. And the edges were the patient's phenotypic data. Song et al. [6] applied the software MedlNRIA, FSL toolbox, and Freesurfer Desikan-Killiany atlas to get diffusion images with segmentation from the brain MRIs collected from GE scanners, Philips scanners, and Siemens scanners. Then, they regarded the connection strengths between the paired ROIs obtained by fibre counting as the adjacency matrix of the graph-structured data. Marzullo et al. [19] obtained the symmetric connection matrix between voxels from MRIs as the adjacency matrix of graph-structured data by medical imaging tools, such as software Eddy-current distortions, MRtrix spherical deconvolution algorithm, and probabilistic streamline tractography algorithm.

However, the cost of the data-acquisition process mentioned above is an issue. The following method is generally adopted. At first, a CNN was employed to extract the medical image feature, and then the graph-structured data can be obtained by feature reconstruction algorithms. For example, Du et al. [20] first used CNN to extract ROI features. GCN has been used to simulate the amplification mechanism of the radiologist operation to determine whether the ROI was amplified. The ROI of lesions could be amplified automatically. Secondly, the breast cancer detection of X images has been completed by GCN. Ye et al. [9] divided the image into different ROI blocks. They adopted U-net [21] to segment the ROI and then captured the topology of the ROI image by GCN.

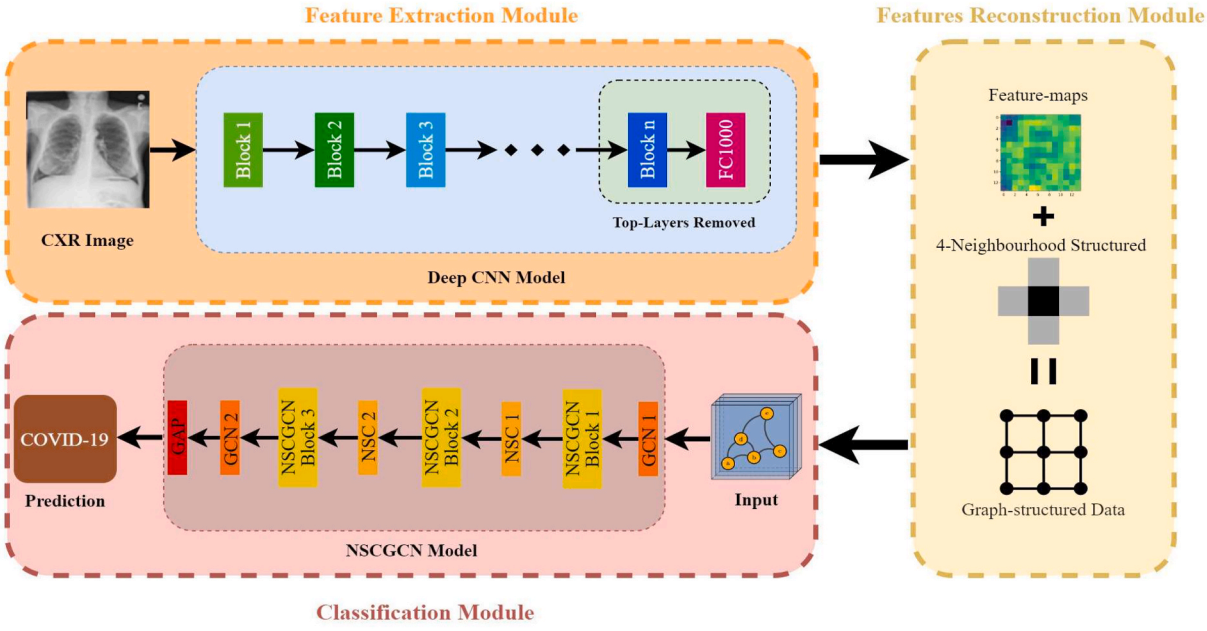


Fig. 1. The flow chart of our proposed COVID-19 diagnosis method.

Finally, a fully connected network (FC) was used to classify the feature vector and finish the diagnosis task of breast cancer. In conclusion, it is a mainstream hybrid technology with CNN as a feature extraction module and GCN as a classification module.

2.2. Deep GCN

Although GCN has excellent potential in medical image deep learning, the number of graph convolutional layers of the GCN mentioned above does not exceed five layers. A deeper model implies better nonlinear expression skills and can absorb more multifaceted transformations, which is able to fit more complex feature inputs. However, the research conducted by Kipf and Welling [15] showed that when the number of model layers exceeds five layers, excessive smoothness appears in the deep GCN model, seriously affecting the model performance. Li et al. [14] used the eigendecomposition technique to prove that the graph convolution of the GCN model is simply a special form of Laplacian smoothing. The GCN over-smoothing is that the representation of nodes within the same connected component tends to converge to the same eigenvector with increasing the number of layers. Huang et al. [22] and Yang et al. [23] also proved the over-smoothing problem caused by the features of the GNNs. There is a general recognition of the urgent need to address the over-smoothing caused by deepening GCN.

Kipf and Welling [15] add a residual connection to solve the smoothing problem, which can directly transfer the features of the node itself from the upper layer to the next layer and ensure the nodes do not tend to converge to the same eigenvector:

$$\begin{aligned} \boldsymbol{Z}^{(l+1)} &= \widehat{\boldsymbol{D}}^{-\frac{1}{2}}\widehat{\boldsymbol{A}}\widehat{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{X}^{(l)}\boldsymbol{W}^{(l)}, \\ \boldsymbol{X}^{(l+1)} &= \sigma\left(\boldsymbol{Z}^{(l+1)}\right) + \boldsymbol{X}^{(l)} \end{aligned} \tag{1}$$

where $\boldsymbol{A}$ is the adjacency matrix of $\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E})$, $\boldsymbol{D}$ is the degree matrix of $\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E})$, $\boldsymbol{D}_{ii} = \sum_j(\boldsymbol{A}_{i,j})$, $\boldsymbol{E}$ is an edge-set, $\boldsymbol{X}$ is the feature vector of node-set $\boldsymbol{V}$, $\boldsymbol{W}$ represents the learnable parameter matrix, σ is an activation function, l is layers number.

Chiang et al. [24] believe that the residual module ignores the weights of neighbour nodes. So, they strengthened the feature weights of the nodes themselves on the basis:

$$\boldsymbol{X}^{(l+1)} = \sigma\left(\left(\widehat{\boldsymbol{D}}^{-\frac{1}{2}}\widehat{\boldsymbol{A}}\widehat{\boldsymbol{D}}^{-\frac{1}{2}} + \boldsymbol{I}\right)\boldsymbol{X}^{(l)}\boldsymbol{W}^{(l)}\right) \tag{2}$$

Beyond over-smoothing comes the over-fitting problem as the number of network layers increases.

Yang et al. [23] proved that the deep GCN model could learn to resist over-smoothing during the training process. The reason that affected the model's performance lies in the over-fitting of the model. They thought the mean-subtraction trick could solve this problem.

Li et al. [16] thought that densely connected could alleviate over-fitting and strengthen the transfer of features. They proposed a deep GCN model based on densely connected and obtained SOTA results in the point cloud segmentation task. The jumping knowledge (JK) framework proposed by Xu et al. [25] can be combined with various GCN, effectively improving the model's performance.

The formula of the convolutional layer in the DenseGCN model proposed by Li et al. [16] is as follows:

$$\begin{aligned} \boldsymbol{G}_{n+1} &= F(\boldsymbol{G}_n, \boldsymbol{W}_n) \\ &= F(L(\boldsymbol{G}_n, \boldsymbol{G}_{n+1}, \ldots, \boldsymbol{G}_0), \boldsymbol{W}_n) \end{aligned} \tag{3}$$

where F is a graph convolution operation, and L is a feature connection function, which densely connects the features of the input graph $\boldsymbol{G}_0$ with the output features of all the intermediate GCN layers.

However, the deep graph learning algorithm in the field of medical images is still a struggle. We adopted a pre-trained model to extract advanced features and proposed a neighbourhood algorithm to construct the graph-structure. We proposed a novel algorithm based on feature fusion, which can solve the problems of over-fitting, over-smoothing, and memory consumption caused by the deep GCN.

2.3. Deep CNN

Deep CNN, such as GoogLeNet [26], VGG [27], ResNet [28], and DenseNet [29], have shown excellent performance in medical image classification. Most downstream applications still use ResNet and its variants as the backbone network. Cheng et al. [30] proposed a modular group attention block that captures feature dependencies in medical images in both channel and spatial dimensions and stacked these group attention blocks in the ResNet style to improve model classification performance. Extensive experiments by Rathore et al. [31] on ADNI [32]

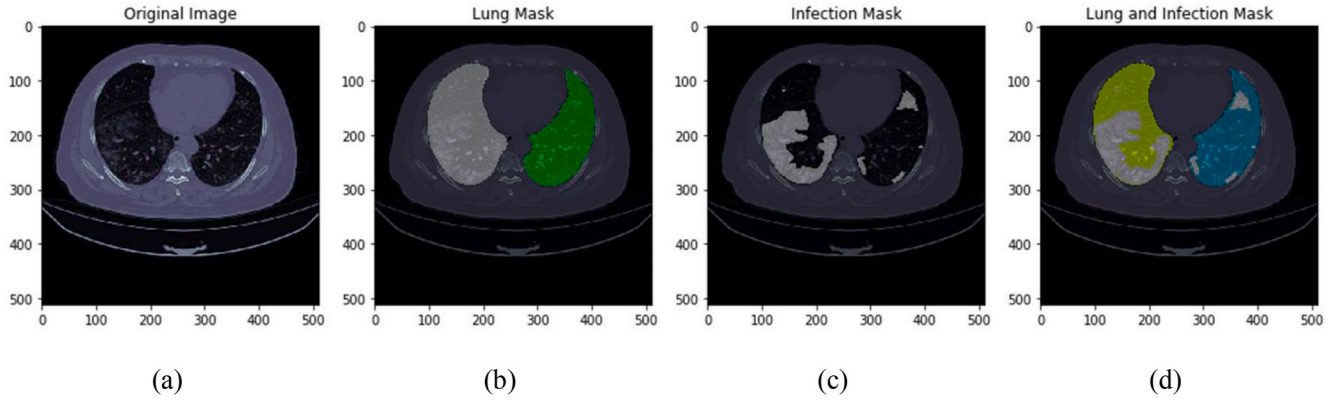


Fig. 2. Sample in the CT dataset.

dataset showed that the DenseNet model improved classification accuracy by about 9% compared to traditional machine learning, which proved the usefulness of the DenseNet model. Kong and Cheng [33] fused DenseNet and VGG, introduced an attention mechanism (global attention machine block and category attention block) to extract depth features and used ResNet to segment effective image information to achieve fast and accurate classification.

2.4. Transfer learning

Transfer learning allows CNN to rapidly transition from one domain to another to improve the reuse rate of models and reduce the cost of repeated training, which proves to be an efficient and low-cost learning technique. For example, Karim et al. [34] migrated relevant knowledge from the source data set to the target data set, which solved the problem of the small amount of available data for small molecule compounds. Parvin et al. [35] proposed the modality adaptation technique to effectively transfer and fuse the domain knowledge of multimodality and improve the performance of multimodal applications. Besides, the pre-trained convolutional neural network has been successfully used as a preprocessing tool and is widely used in medical image classification [36,37]. Ay et al. [38] used six different CNN architectures as benchmark models for performance evaluation, used the weights and architectures of pre-trained models trained on ImageNet and tuned to the medical image domain. In the pressure-injury image classification task, they froze the weights of all pre-trained network layers except the last layer and only updated the weights of new layers in the backward pass. By freezing the network weights, overfitting is prevented. Ahmed [39] based on a pre-trained CNN and image noise filter without data augmentation and fine-tuning settings, used a denoising CNN as a preprocessing tool and combined the denoising preprocessing stage with a classification method for medical image classification. Jones et al. [40] extracted ROIs around suspicious lesions and computed the radiomics features from each ROI and the automated features from the VGG16 network using transfer learning. Then they converted a single-channel ROI image to a three-channel pseudo-ROI image by superimposing the original, bilaterally filtered, and histogram equalized image. Two VGG16 models using pseudo-ROIs and three unprocessed stacked raw ROIs were used to extract automated features. Finally, they used a linear support vector machine for classification based on the extracted features.

3. Research method

In this section, the dataset and the proposed NSCGCN have been illustrated in detail. As shown in Fig. 1, the proposed framework consists of three modules: (i) The features extraction module has been built to extract advanced input image features by the deep CNN models pre-trained in ImageNet. (ii) The features reconstruction module has been built to convert the feature into graph-structured data by the proposed neighbourhood feature reconstruction algorithm. (iii) The classification module has been built to classify the input graph-structured data by our proposed NSCGCN model.

3.1. Dataset and preprocessing

We first introduce the small CT dataset to justify the performance of the deep GCN model, and then use the large CXR dataset to verify the performance of our proposed COVID-19 diagnosis method.

3.1.1. CT dataset

We use the publicly available CT dataset covid19-ct-scans[2] to justify the performance of the deep GCN model NSCGCN. The dataset contains CT images of 20 patients with covid-19 and segmentation results of lung and infection by experts. All images are three-dimensional images with sizes ranging from $400 \times 400 \times 300$ to $800 \times 800 \times 45$. As shown in Fig. 2, Fig. 2(a) represents the original medical image, Fig. 2(b) represents the lung mask image marked by the expert, Fig. 2(c) represents the lung infection mask marked by the expert, and Fig. 2(d) represents the mask superimposed by the lung and infection.

We first use the lung segmentation image labelled by experts to separate the lung region from the original image. And then, apply the sliding window with the size 50×50 to the original, the lung infection mask and the lung region mask images simultaneously. Taking all pixels in the sliding window as an input and then setting the number of infected pixels in the sliding window in the lung infection mask image as m, the number of lung pixels in the lung area mask image as n and the threshold $j = 125$. If the lung pixel n exceeds this threshold, the image cropped out of the original image by the sliding window will be used as the experimental sample image. If the number of infected pixels m exceeds 125, the sample image cropped out by the sliding window is an infected image labelled as 1. Otherwise, it is a normal image labelled as 0. This process can be expressed as:

$$l = \begin{cases} 1, m \geq 125 \\ 0, m < 125 \end{cases} \tag{4}$$

The sliding window covers the image from left to right and from top to bottom, with steps of 50 in width and height and 5 in depth. Finally, 85,725 cropped images were obtained, including 15,059 infected and 70,666 normal images. As shown in Table 1, 1500 images of each category were randomly selected for testing, and the rest were used for training.

The number of normal images is much larger than that of infected

[2] https://www.kaggle.com/andrewmvd/covid19-ct-scans.

Table 1
Sample division.

Category	Training	Testing
Infection image	13,559	1500
Normal image	69,166	1500

Table 2
Dataset details.

Subsets	HC	COVID-19	OPN	Total
dataset1	462	462	462	1386
dataset2	462	462	462	1386
dataset3	463	463	463	1389
dataset4	463	463	463	1389
dataset5	463	463	463	1389
Total	2313	2313	2313	6939

dataset was randomly divided into five subsets to ensure that the data division used in all experiments was the same. The number of each category sample is the same, and the positive and negative samples are balanced. Examples of HC, OPN, and COVID-19 samples are shown in Fig. 3. The second row is 2 CXR images of COVID-19, which showed the multiple small patch shadows and interstitial changes, and the third row is CXR images of OPN, which showed the shadow of significant leaf consolidation.

Furthermore, the sizes of the RGB images range from $2721 \times 2438 \times 3$ to $1336 \times 1128 \times 3$. The images were standardized before being input into the pre-trained model. The image sizes were scaled to $224 \times 224 \times 3$. The pixel-values were divided by 255 and thus normalized to $[0, 1]$. Meanwhile, the pixel values were normalized by the mean value and standard deviation from ImageNet. The distribution of the pixel-values was transformed into standard Gauss distribution $N(\mu, \sigma^2)$:

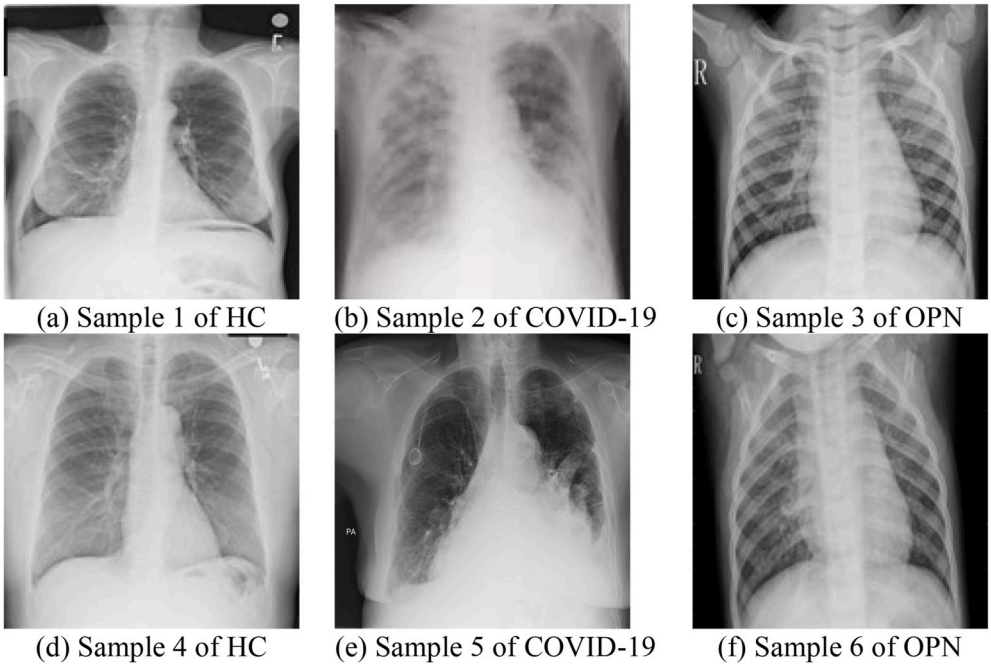


Fig. 3. Sample in the CXR dataset.

images, which will cause the classifier to be more inclined toward normal images. Therefore, it may cause false positive and false negative problems, which means that it is difficult for the model to find the infection image. Therefore, we introduce the cost matrix, and the cost ratio c_t can be calculated by Eq (5), which is being set to 5:

$$c_t = N_{infection} / N_{normal} \tag{5}$$

3.1.2. CXR dataset

We use the publicly available CXR dataset COVID19_Pneumonia_-Normal_Chest_Xray_PA_Dataset[3] to verify the validity of our proposed COVID-19 diagnosis method. X-ray samples of COVID-19 in the dataset were retrieved from different sources for the unavailability of a large specific dataset. As shown in Table 2, the dataset contains 2313 HC, 2313 OPN, and 2313 COVID-19 case samples, totalling 6939 samples. At the same time, a 5-fold cross-validation method is employed to assess the stability and reliability of the NSCGCN. Before the experiments, the

$$x = \frac{x - \mu}{\sigma} \tag{6}$$

where x denotes individual image data, $\mu = [0.485, 0.456, 0.406]$, $\sigma = [0.229, 0.224, 0.225]$.

3.2. Feature extraction module

We consider the limited number of labelled training samples insufficient to retrain the entire model from scratch. To attain high classification accuracy, we follow the transfer learning method [4] and propose a feature extraction module based on the deep CNN models trained on the ImageNet to extract the pixel-level features from medical images more effectively.

As shown in Table 3, to explore the best feature extraction model, we propose different schemes to remove the top-layers parts of DenseNet201, DenseNet121, ResNet101, ResNet50, ResNet18, Vgg16 and GoogLeNet to obtain different feature extraction model. The removed parts are labelled as A, B, or C for different top-layers removal schemes for each model.

[3] https://www.kaggle.com/amanullahasraf/covid19-pneumonia-normal-chest-xray-pa-dataset.

Table 3
Top-layers removal scheme of different deep CNN.

Model	SchemeID	Top-layers removal scheme	Feature-maps size
DenseNet201	A	The classifier layers after the Dense Block 4	7 × 7 × 1920
	B	The layers after the Transition Layer 3	7 × 7 × 896
	C	The layers after the Dense Block 3	14 × 14 × 1792
DenseNet121	A	The classifier layers after the Dense Block 4	7 × 7 × 1024
	B	The layers after the Transition Layer 3	7 × 7 × 512
	C	The layers after the Dense Block 3	14 × 14 × 1024
ResNet101	A	The layers after the conv5_x layer	7 × 7 × 2048
	B	The layers after the conv4_x layer	14 × 14 × 1024
ResNet50	A	The layers after the conv5_x layer	7 × 7 × 2048
	B	The layers after the conv4_x layer	14 × 14 × 1024
ResNet18	A	The layers after the conv5_x layer	7 × 7 × 512
	B	The layers after the conv4_x layer	14 × 14 × 256
Vgg16	A	The layers after the 5th maxpool layer	7 × 7 × 512
	B	The layers after the 4th maxpool layer	14 × 14 × 512
GoogLeNet	A	The layers after the inception5b	7 × 7 × 1024
	B	The layers after the inception5a	7 × 7 × 832

As shown in Fig. 4, take DenseNet201 as an example. To transfer the parameters of the pre-trained model to our feature extraction model, we modify the original DenseNet201 model to get different feature maps. We propose three different top-layers removal schemes. In Scheme A, we can get a feature extraction module by removing the classification layer after Dense Block 4. The features-maps A can be obtained when the CXR images have been fed into the model. The other two of our proposed removal schemes are similar to scheme A. We can obtain Scheme B by removing the top-layers after Transition Layer 3, and Scheme C by removing the top-layers after Dense Block 3. Furthermore, we can get features-maps B and C depending on the schemes accordingly.

The size of the output feature-maps of the convolutional layer is:

$$H_{out} = \frac{H + 2p_h - K_h}{s_h} + 1 \tag{7}$$

where H represents the size of the input image, P_h is the padding size, K_h is the convolution kernel size, and S_h is the stride size.

The number of the output feature-maps channels of the convolutional layers in DenseNet is:

$$c_{l+1} = c_0 + k \times (l-1) \tag{8}$$

where c_0 represents the number of channels of the input feature in each Dense Block, l represents the number of layers in each Dense Block, k is the growth rate of the DenseNet, $k = 32$.

As shown in Table 4, we obtained three feature-maps sets with sizes $7 \times 7 \times 1920$, $7 \times 7 \times 896$, and $14 \times 14 \times 1792$. Then, those sets are fed into the feature reconstruction module to get graph-structured data.

3.3. Feature reconstruction module

It was found that using a grid structure to convert images into graph-structured data can obtain better classification results than such traditional ways as down-sampling [41]. So we propose a feature reconstruction algorithm based on the neighbourhood structure of the image.

First, we introduce the neighbourhood structure of the image. The 8-neighbourhood nodes of the pixel node $p(0,0)$ can be described as follow:

$$\begin{aligned} &p_1(1,1), p_2(0,1), \\ &p_3(-1,1), p_4(1,0), \\ &p_5(-1,0), p_6(1,-1), \\ &p_7(0,-1), p_8(-1,-1) \end{aligned} \tag{9}$$

Then, the 8-neighbourhood set $N_8(p)$, 4-neighbourhood set$N_4(p)$, and D-neighbourhood set $N_D(p)$ of the node $p(x,y)$ can be obtained:

$$\begin{aligned} N_8(p) &= \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8\} \\ N_4(p) &= \{p_2, p_4, p_5, p_7\} \\ N_D(p) &= \{p_1, p_3, p_6, p_8\} \end{aligned} \tag{10}$$

As shown in Fig. 5, The 8-neighbourhood graph comprises the 4-neighbourhood graph and the D-neighbourhood graph. The D-neighbourhood graph consists of two subgraphs.

Table 4
DenseNet201 structure.

Layer	Output Size	DenseNet201
Convolution	112 × 112 × 64	7 × 7 conv
Pooling	56 × 56 × 64	3 × 3 max pool
Dense Block 1	56 × 56 × 256	$6 \times \begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix}$
Transition Layer 1	56 × 56 × 128	1 × 1 conv
	28 × 28 × 128	2 × 2 average pool, stride 2
Dense Block 2	28 × 28 × 512	$12 \times \begin{bmatrix} 1 \times 1\text{conv} \\ 3 \times 3\text{conv} \end{bmatrix}$
Transition Layer 2	28 × 28 × 256	1 × 1 conv
	14 × 14 × 256	2 × 2 average pool
Dense Block 3	14 × 14 × 1792	$48 \times \begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix}$
Transition Layer 3	14 × 14 × 896	1 × 1 conv
	7 × 7 × 896	2 × 2 average pool
Dense Block 4	7 × 7 × 1920	$32 \times \begin{bmatrix} 1 \times 1\text{conv} \\ 3 \times 3\text{conv} \end{bmatrix}$
Classification	1 × 1 × 1000	7 × 7 average pool 1000D FC, softmax

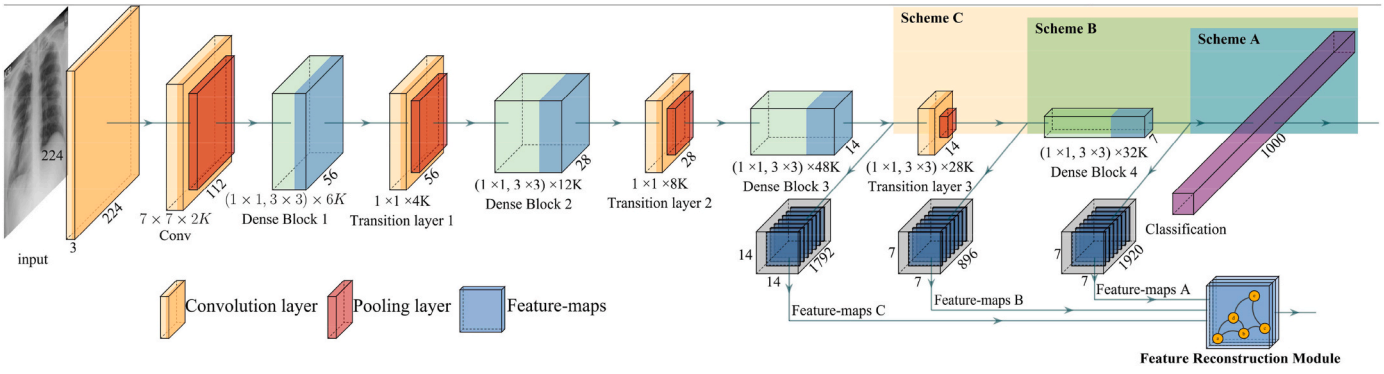


Fig. 4. The scheme of removing the top-layers in DenseNet201.

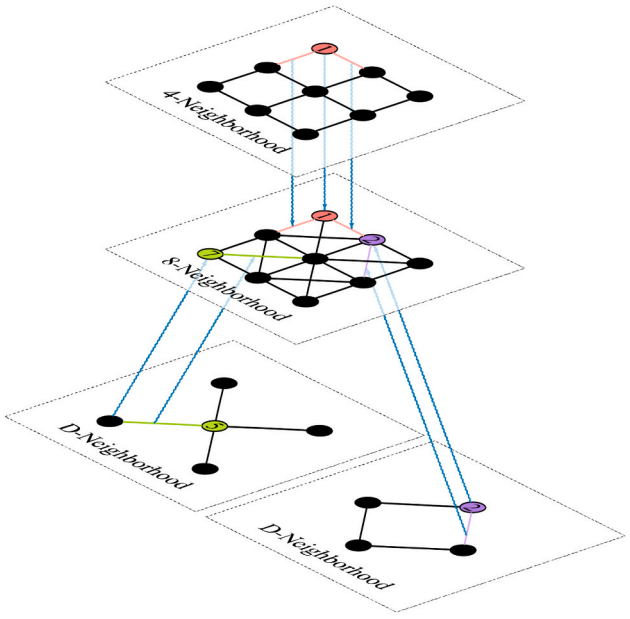


Fig. 5. Neighbourhood structure.

Table 5
Pseudocode of neighbourhood feature reconstruction algorithm.

```
Input: feature-maps P[n][n] , char type
Output: Graph G(V,E)
Initialization: int V[n^2]=0 , E[n^2][n^2]=0 , temp=0
for p in P do
|   V[temp]=p
|   if type=='4' or type=='8' then
|   |   if (temp-n)>=0 then
|   |   |   E[temp][temp-n]=1
|   |   end
|   |   if (temp-1)%n!=(n-1) and (temp-n)>=0 then
|   |   |   E[temp][temp-1]=1
|   |   end
|   |   if (temp+1)%n!=0 then
|   |   |   E[temp][temp+1]=1
|   |   end
|   |   if (temp+n)<n^2 then
|   |   |   E[temp][temp+n]=1
|   |   end
|   end
|   if type=='D' or type=='8' then
|   |   if (temp-n-1)%n!=(n-1) and (temp-n-1)>=0 then
|   |   |   E[temp][temp-n-1]=1
|   |   end
|   |   if (temp-n+1)%n!=0 and (temp-n+1)>=0 then
|   |   |   E[temp][temp-n+1]=1
|   |   end (temp+n-1)%n!=(n-1)
|   |   if and (temp+n-1)<n^2 then
|   |   |   E[temp][temp+n-1]=1
|   |   end
|   |   if (temp+n+1)%n!=0 and (temp+n+1)<n^2 then
|   |   |   E[temp][temp+n]=1
|   |   end
|   end
|   temp=temp+1
end
```

And then, the Neighbourhood Feature Reconstruction Algorithm is described in Table 5. The algorithm's input is a feature-maps, and the algorithm's output is a graph $\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E})$. The graph $\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E})$ is an unweighted undirected graph, the nodes $\boldsymbol{V}$ represent the feature vector of the feature maps, and the edges $\boldsymbol{E}$ calculated by the neighbourhood feature reconstruction algorithm represent the edges between nodes.

In the Neighbourhood Feature Reconstruction Algorithm, for input feature-maps $\boldsymbol{P}$ of size $n \times n$, the feature vector of the feature-maps are looped, and each feature vector of the feature-maps is treated as a node in the output graph $\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E})$. At the same time, the neighbourhood nodes are constructed for each node. If the corresponding neighbourhood nodes legally exist in the image range, the connection between every two nodes is added to $\boldsymbol{E}$. The feature reconstruction module is composed of the neighbourhood feature reconstruction algorithm.

The input of the feature reconstruction module is the feature maps of size $n \times n$ obtained from the feature extraction module. Furthermore, the output is a graph $\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E})$, the number of $|\boldsymbol{V}|$ is n^2. The number of $\boldsymbol{E}$ in the 8-neighbourhood structure is:

$$\begin{aligned}|\boldsymbol{E}| &= (n-1)*n*2+2*(n-1)*(n-1)\\ &= 2(n-1)(2n-1)\end{aligned} \tag{11}$$

The feature maps obtained from the feature extraction module can be converted into the graph-structure. In our proposed model, the number of nodes $|\boldsymbol{V}|$ is 49, 49, 196, the number of edges $|\boldsymbol{E}|$ is 156, 156, 702, and the number of each node feature-channels $|c|$ is 1920, 896, and 1792.

3.4. Classification module

We elaborated on the function and implementation details of the proposed NSC in this section. Then NSCGCN Block was built based on the NSC, and NSCGCN was built based on the NSCGCN Block. NSCGCN was the backbone network of the classification module.

3.4.1. Proposed node-self convolution algorithm

We found that the feature fusion algorithm plays the role of the bottleneck layer in DenseNet. The role of the bottleneck layer is to reduce the number of input feature maps, integrate the features on each channel, and reduce the amount of computation. There are other methods to reduce the amount of computation in GCN, such as neighbourhood sampling and graph pooling. However, they are only suitable for shallow GCN. In the deep GCN, neighbourhood sampling will sample the entire graph, and graph pooling will lose essential node information. There is a lack of an efficient method to solve the memory storage problem caused by increased feature dimensions in deep GCN.

Thus, we propose a new feature fusion algorithm based on the graph-structured data named node-self convolution (NSC). The detail of the proposed NSC is shown in Fig. 6: Each node is a features vector with five channels in the graph $\boldsymbol{G}^l(\boldsymbol{X}^l,\boldsymbol{A}^l)$ First, remove the connection of the graph $\boldsymbol{G}^l(\boldsymbol{X}^l,\boldsymbol{A}^l)$ and only retain the self-connection of the nodes (the node connects to the node itself) to obtain the graph $\widetilde{\boldsymbol{G}}^l(\boldsymbol{X}^l,\boldsymbol{I})$. Secondly, perform a graph convolution operation on the graph $\widetilde{\boldsymbol{G}}^l(\boldsymbol{X}^l,\boldsymbol{I})$ to obtain the feature maps $\boldsymbol{X}^{l+1}$ and then update the feature maps to complete the feature fusion and dimensionality reduction operation. Finally, by restoring the original graph-structure of $\boldsymbol{G}^l(\boldsymbol{X}^l,\boldsymbol{A}^l)$, we can get a new graph $\boldsymbol{G}^{l+1}(\boldsymbol{X}^{l+1},\boldsymbol{A}^l)$ while maintaining the structure unchanged. The proposed NSC ensures that each node only aggregates information with itself. By controlling the number of filters in the GCN layer, the dimensionality reduction and dimensionality increase operations of the feature dimension are completed. In our proposed model, we set each NSC to produce $4K$ feature maps. The convolution result of NSC is:

$$\boldsymbol{Z}^{(l+1)} = \widehat{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{I}\widehat{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{X}^{(l)}\boldsymbol{W}^{(l)} \tag{12}$$

3.4.2. NSCGCN

NSCGCN Block regards the NSC as the bottleneck layer controlling the number of feature-map channels. As shown in Fig. 7(a), the NSCGCN block was built based on the densely connected NSC and GCN. Each NSC layer takes all output feature maps of preceding GCN layers as inputs. Furthermore, an NSC layer is added before each GCN layer. The number of the feature-map channels is:

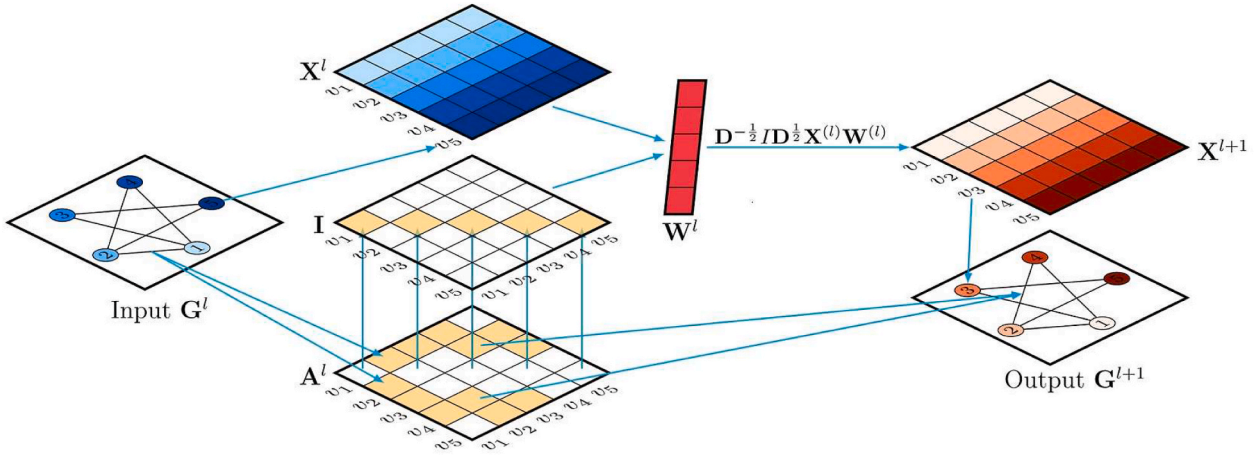


Fig. 6. The transformation of the adjacency matrix and feature in NSC.

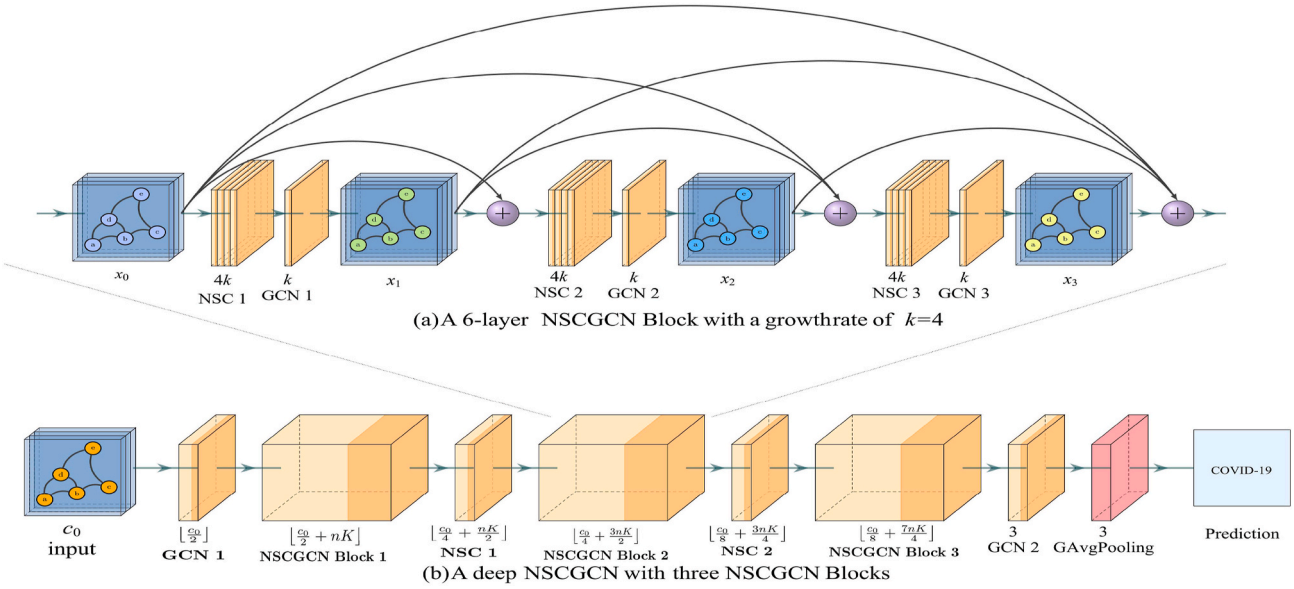


Fig. 7. The structure of NSCGCN.

Table 6
NSCGCN structure.

Layer	Method
Convolution 1	GCN
NSCGCN Block 1	$\begin{bmatrix} \text{NSC} \\ \text{GCN} \end{bmatrix} \times n$
NSC Layer 1	NSC
NSCGCN Block 2	$\begin{bmatrix} \text{NSC} \\ \text{GCN} \end{bmatrix} \times n$
NSC Layer 2	NSC
NSCGCN Block 3	$\begin{bmatrix} \text{NSC} \\ \text{GCN} \end{bmatrix} \times n$
Convolution 2	GCN
Global Pooling	GAvgPooling

$$C_n = c_0 + nK \tag{13}$$

where c_0 is the number of original feature-map channels, K is the growth rate that determines the number of output feature-map channels, and n is the number of GCN layers in the NSCGCN Block.

As shown in Fig. 7 (b), our proposed deep NSCGCN model contains three NSCGCN blocks, and there is an NSC layer between every two

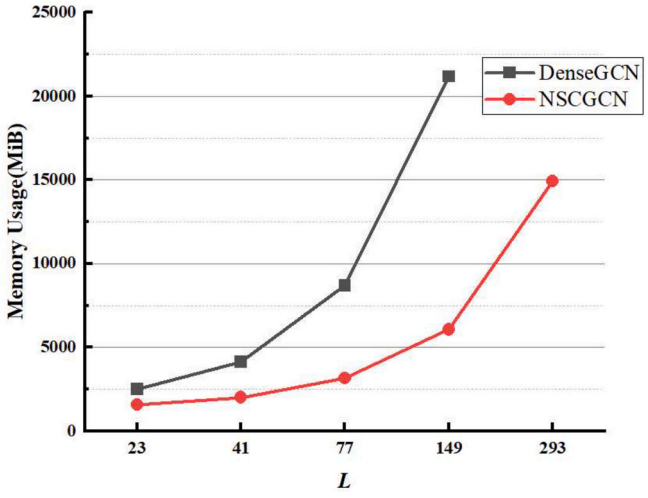


Fig. 8. Comparison of memory usage when $K = 24$ and batch size $= 32$.

NSCGCN Blocks. The GraphAvgPooling layer is applied to classification. In our experiments, the NSC layer has been set to halve the number of feature maps.

As shown in Table 6, the number of layers L of the NSCGCN is:

$$L = 6 * n + 5 \tag{14}$$

where n represents the number of GCN layers in the NSCGCN Block.

The memory usage of DenseGCN and NSCGCN is shown in Fig. 8. It can be seen that the memory usage of DenseGCN is out of memory when $K = 24$ and $L = 293$. The main reason is that the growth rate of the feature dimension of the input graph is related to the dimension of the output graph in the DenseGCN. By adding the proposed NSC, the growth rate of memory usage slows down, and the memory consumption is much lower than that of DenseGCN with the same number of layers.

3.5. Evaluation index

The evaluation indexes such as Sen., F1-score, Pre., AUC, and Acc. are used to measure the performance of the models. The following evaluation index measures the overall performance of the classifier:

$$accuracy = \frac{TH + TC + TO}{TN + FHC + FHO + FCH + TC + FCO + FOH + FOC + TO} \tag{15}$$

$$\begin{aligned} precision &= \left(\frac{TH}{TH + FHC + FHO} + \frac{TC}{FCH + TC + FCO} + \frac{TO}{FOH + FOC + TO}\right) \Big/ 3 \\ &= (presision_H + presision_C + presision_O) / 3 \end{aligned} \tag{16}$$

$$\begin{aligned} sensitivity &= \left(\frac{TH}{TH + FCH + FOH} + \frac{TC}{FHC + TC + FOC} + \frac{TO}{FOH + FOC + TO}\right) \Big/ 3 \\ &= (sensitivity_H + sensitivity_C + sensitivity_O) / 3 \end{aligned} \tag{17}$$

$$\begin{aligned} F1 &= \left(\frac{2 * precision_H * sensitivity_H}{precision_H + sensitivity_H} + \frac{2 * precision_C * sensitivity_C}{precision_C + sensitivity_C} + \frac{2 * precision_O * sensitivity_O}{precision_O + sensitivity_O}\right) \\ &= (F1_H + F1_C + F1_O) / 3 \end{aligned} \tag{18}$$

As shown in Table 7, TH, TC, and TO represent the correct predicted results for HC, COVID-19, and OPN samples, and FHC, FHO, FCH, FCO, FOH, and FOC represent the mispredicted results for HC, COVID-19, and OPN samples respective.

AUC is defined as the area under the receiver operating characteristic (ROC) curve and the coordinate axis. ROC is drawn by True Positive Rate (TPR) and False Positive Rate (FPR), where TPR represents the probability that a positive example can be paired, and FPR represents the probability that a negative example can be classified as a positive example. In our proposed method, the final data obtained through the model is the probability (score) when the samples may be in three categories, respectively. When the "Score" value is set as the threshold. Once the probability of the sample is greater than or equal to this threshold, it is considered that the sample is a certain category. Each time a different threshold is selected, a set of FPR and TPR can be added as a point to the ROC curve. We calculate the evaluation value for each category and take the average value as the performance index of the model.

$$\begin{aligned} TPR &= \left(\frac{TH}{TH + FCH + FOH} + \frac{TC}{FHC + TC + FOC} + \frac{TO}{FHO + FCO + TO}\right) \Big/ 3 \\ &= (TPR_H + TPR_C + TPR_O) / 3 \end{aligned} \tag{19}$$

$$\begin{aligned} FPR = &\left(\frac{FHC + FHO}{FHC + TC + FOC + FHO + FCO + TO}\right. \\ &+ \frac{FCH + FCO}{TH + FCH + FOH + FHO + FCO + TO} + \left.\frac{FOH + FOC}{TH + FCH + FOH + FHC + TC + FOC}\right) \Big/ 3 \\ &= (FPR_H + FPR_C + FPR_O) / 3 \end{aligned} \tag{20}$$

4. Experiments results and discussions

In this section, the experimental environment and settings were described in detail. First, the ablation experiments on the CT dataset to find the best value of layers L and the best value of convolution kernels K were shown in section 4.2. Then, the ablation experiments on the CXR dataset to find the best feature reconstruction algorithm, the best feature extractor, the best value of layers L, and the best value of convolution kernels K were shown in section 4.3. The comparisons of SOTA deep CNN, GCN and COVID-19 diagnosis methods were shown to verify the performance of our proposed COVID-19 diagnosis method.

4.1. Experimental setting

Our experiments have been carried out on a server with 64 GB RAM, CPU Intel Xeon Silver 4214, and GPU Tesla M40 24 GB. All algorithms have been programmed based on Python 3.8.8 and PyTorch 1.9.0.

For the CT dataset, we first use the graph sparse pruning algorithm

Table 7
Definition of TH, TC, to, FHC, FHO, FCH, FCO, FOH and FOC.

Name	Definition
TH	HC samples were predicted as HC by the model
TC	COVID-19 samples were predicted as COVID-19 by the model
TO	OPN samples were predicted as OPN by the model
FHC	HC samples were predicted as COVID-19 by the model
FHO	HC samples were predicted as OPN by the model
FCH	COVID-19 samples were predicted as HC by the model
FCO	COVID-19 samples were predicted as OPN by the model
FOH	OPN samples were predicted as HC by the model
FOC	OPN samples were predicted as COVID-19 by the model

Table 8
Performance of NSCGCN acc on hyper-parameters (Unit:%).

Learning rate	Weight decay				
	0.1	0.05	0.01	0.005	0.001
	Acc.				
0.005	92.01	95.46	96.40	96.26	95.75
0.001	97.77	97.62	97.37	97.04	96.76
0.0005	97.84	97.04	**98.34**	97.41	96.77
0.0001	96.47	96.11	96.62	97.70	96.69
0.000005	96.47	95.97	96.62	97.62	96.69

[42] to convert images into graph-structure data and then use the GCN model for classification. The optimization algorithm used is Adam. Furthermore, the regularization L2 (weight decay) is used to overcome the model over-fitting issue, and the weight is set to 5E-4. The initial learning rate is 0.001, the batch size is 16, the training epochs are 50, the learning rate adjustment function is the cosine annealing function, and the loss function is the cross-entropy loss function. After calculating the loss between the predicted values and ground truth, the parameters of the model are optimized by the optimization algorithm according to the loss. At the same time, the cost matrix is used to pay attention to more information about false positives and omissions to overcome the data imbalance problem.

To verify the advanced performance of our proposed deep model, we compare our proposed NSCGCN model with several shallow GCN models on the CT dataset, such as GraphSAGE [43], GCN [15], GIN [44], Graclus [45], SAGPool [46], GlobalAttentionNet [47]. The dataset used is divided uniformly. Furthermore, all GCNs are constructed based on two convolution layers, and the number of convolution kernels is 24. The hyperparameter setting is the same as that of the original article. The learning rate adjustment function is also a cosine annealing function, and the training period is also set to 50.

Table 9
Performance of NSCGCN with different K when $L = 77$(Unit:%).

K	Sen.	F1	Pre.	Acc.	AUC
3	84.27	84.02	86.56	84.27	95.82
6	85.93	85.75	87.86	85.93	96.73
12	86.30	86.12	88.33	86.30	97.01
24	**86.83**	**86.68**	**88.58**	**86.83**	**97.04**

Table 10
Performance of NSCGCN with different L when $K = 24$(Unit:%).

L	Sen.	F1	Pre.	Acc.	AUC
23	84.47	84.22	86.80	84.47	95.95
41	86.20	86.02	88.11	86.20	96.51
77	86.83	86.68	88.58	86.83	97.04
149	**87.50**	**87.37**	**89.10**	**87.50**	**97.09**

The above experimental results show that when $L = 149$, $K = 24$, our proposed NSCGCN model performs the best, and its corresponding optimal performance is as: Sen. = 87.50%, F1 = 87.37%, Pre. = 89.10%, Acc. = 87.50%, AUC = 97.08%. In the subsequent section, $K = 24$ and $L = 149$ were set on the CT dataset.

Table 11
Comparison with shallow GCN models (Unit:%).

model	Sen.	F1	Pre.	Acc.	AUC
GraphSAGE [43]	63.83	58.90	76.59	63.83	91.46
GCN [15]	77.57	76.67	82.59	77.57	93.00
GIN [44]	77.73	76.88	82.53	77.73	94.37
Graclus [45]	78.97	78.32	82.87	78.97	93.30
SAGPool [46]	79.43	78.83	83.23	79.43	93.54
GlobalAttentionNet [47]	78.47	77.89	81.77	78.47	92.71
NSCGCN (Ours)	**87.50**	**87.37**	**89.10**	**87.50**	**97.09**

For the CXR dataset, the optimization algorithm is Sharpness-Aware Minimization (SAM) [48] based on stochastic gradient descent (SGD). The cross-entropy loss function is used for gradient updating. Similarly, we use the regularization L2 (weight decay) to overcome the model over-fitting issue. As shown in Table 8, we use the grid search technique to select the best hyperparameters and choose an optimal learning rate of 0.01 and weight decay of 5E-4 by achieving the highest prediction accuracy of 98.34%. Moreover, it can be seen that both higher and lower learning rates decrease performance. Furthermore, the trained model with dynamic learning rate adjustment can effectively handle local minima and overfitting issues. The learning rate tuning function is the cosine annealing function. The parameters optimization process on the CXR dataset is the same as on the CT dataset. Moreover, the momentum is 0.9, the training epochs are 200, and the batch size is 32.

To verify the advanced performance of our proposed COVID-19 diagnosis method, we trained such SOTA GCN as GraphSAGE [43], GCN [15], GIN [44], Graclus [45], SAGPool [46], EdgePool [49], GlobalAttentionNet [47], Set2SetNet [50], SortPool [5] and JK [25]. Under the same CXR dataset, all GCN were built with three layers and 32 convolution kernels. In addition, to prove the performance of our proposed COVID-19 diagnosis method, we selected sixteen SOTA deep CNN and eight SOTA COVID-19 diagnosis methods for comparison. The hyperparameter settings are the same as the source articles' hyperparameter settings. The learning rate tuning function is the cosine annealing function, and the training period is set to 200.

4.2. Exploration of best L and K on the CT dataset

The layer number L and convolution kernel number K of NSCGCN are the key hyperparameter parameters that determine the performance of our proposed model. Therefore, in this section, hyperparametric optimization is carried out through ablation experiments to obtain the optimal L and K values.

As shown in Table 9, the value of convolution kernel number K will significantly affect the classification performance of NSCGCN. When the number of convolution kernels K increases from 3 to 6, the performance is improved by 1.7%. When the number of convolution kernels K is increased from 12 to 24, the Pre. value of the NSCGCN is slowly improved by 0.2%. Furthermore, when $K = 24$, the performance of the NSCGCN is the best and then set $K = 24$. It can be concluded that when K increases, the performance of NSCGCN shows an upward trend.

As shown in Table 10, the value of layer L significantly affects the classification performance of NSCGCN. Our proposed model performance rises with L increases. This is because the nonlinear transformation of our proposed model will become more complex as L increases, which brings more implicit information and improves the model's performance. Moreover, the layer number L has a more significant impact on the model's performance than the kernel number K.

4.3. Comparison of shallow GCN models on the CT dataset

This section compares our proposed deep GCN model NSCGCN with other shallow GCN models. As shown in Table 11, NSCGCN provides the best accuracy and robustness. Compared with the shallow GCN model, the Sen. of the deep GCN model NSCGCN is improved by at least 8%. It

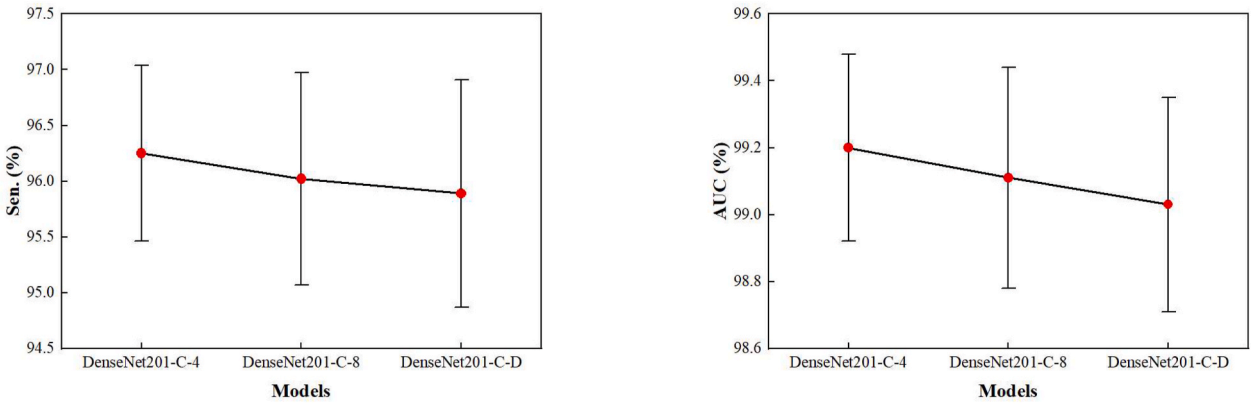


Fig. 9. Performance of DenseNet201-C on different neighbourhood structures when $L = 77$ and $K = 12$.

Table 12
Performance of NSCGCN on different feature extractors (Unit: %).

Model	Sen.	F1	Pre.	Acc.	AUC
DenseNet201-A	94.87 ± 0.61	94.87 ± 0.61	94.99 ± 0.62	94.87 ± 0.61	98.79 ± 0.11
DenseNet201-B	95.69 ± 0.73	95.69 ± 0.73	95.88 ± 0.68	95.69 ± 0.73	99.12 ± 0.18
DenseNet201-C	**96.25 ± 0.79**	**96.25 ± 0.79**	**96.42 ± 0.71**	**96.25 ± 0.79**	**99.20 ± 0.28**
DenseNet121-A	95.55 ± 0.56	95.54 ± 0.56	95.65 ± 0.56	95.55 ± 0.56	99.16 ± 0.18
DenseNet121-B	95.48 ± 0.57	95.47 ± 0.57	95.62 ± 0.55	95.48 ± 0.57	99.13 ± 0.20
DenseNet121-C	95.88 ± 0.54	95.87 ± 0.54	96.03 ± 0.46	95.88 ± 0.54	99.17 ± 0.20
ResNet101-A	94.58 ± 0.32	94.58 ± 0.32	94.69 ± 0.32	94.58 ± 0.32	98.83 ± 0.17
ResNet101-B	95.66 ± 0.52	95.66 ± 0.53	95.78 ± 0.50	95.66 ± 0.52	99.15 ± 0.28
ResNet50-A	95.04 ± 0.42	95.04 ± 0.42	95.18 ± 0.42	95.04 ± 0.42	98.85 ± 0.11
ResNet50-B	95.50 ± 0.27	95.50 ± 0.28	95.63 ± 0.24	95.50 ± 0.27	99.16 ± 0.06
ResNet18-A	94.94 ± 0.32	94.93 ± 0.32	95.06 ± 0.28	94.94 ± 0.32	98.85 ± 0.17
ResNet18-B	95.71 ± 0.79	95.70 ± 0.79	95.88 ± 0.71	95.71 ± 0.79	99.15 ± 0.27
Vgg16-A	94.91 ± 0.32	94.90 ± 0.32	94.99 ± 0.29	94.91 ± 0.32	98.78 ± 0.23
Vgg16-B	95.42 ± 0.58	95.41 ± 0.58	95.53 ± 0.55	95.42 ± 0.58	99.06 ± 0.24
Google-A	93.44 ± 0.65	93.44 ± 0.65	93.49 ± 0.65	93.44 ± 0.65	98.50 ± 0.26
Google-B	95.07 ± 0.67	95.07 ± 0.68	95.16 ± 0.67	95.07 ± 0.67	98.87 ± 0.35

*model-X means remove X from the model.

demonstrates the effectiveness of our work on deepening GCN and demonstrates that deep GCNs are a viable research direction. And it can be concluded that NSC can effectively deepen GCN and reduce over-fitting and over-smoothing problems.

4.4. Ablation experiments on the CXR dataset

We attempt to find the best settings of our proposed model through the ablation experiments, such as the different feature reconstruction algorithms, feature extractors, and different values of L and K.

4.4.1. Exploration of best feature reconstruction model

The feature reconstruction algorithm can reconstruct the feature maps extracted by the pre-training model, but its actual effect depends mostly on the neighbourhood structure. Our proposed NSCGCN model was compared and tested under three different neighbourhood structure settings for optimum results. The other hyperparameters of our proposed NSCGCN model are set to $L = 77$ and $K = 12$.

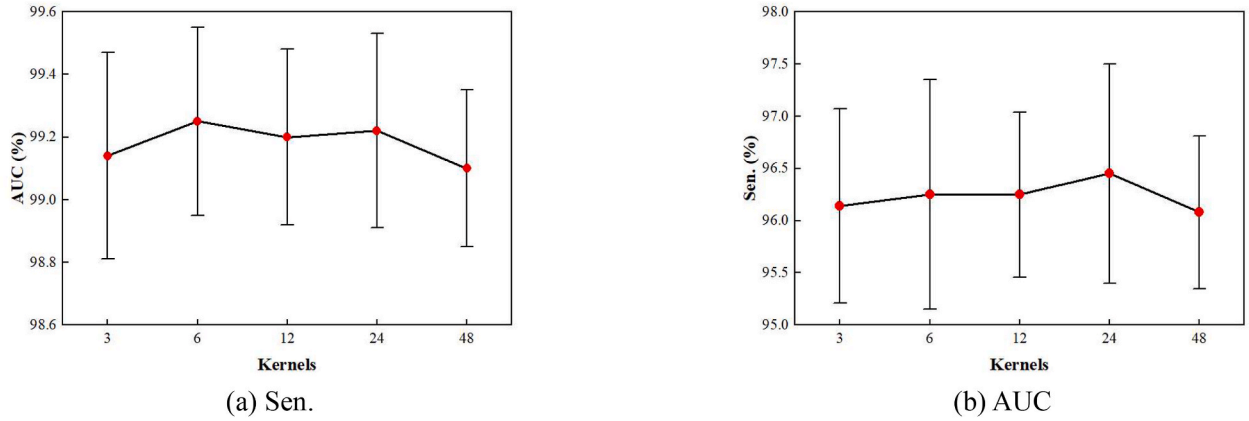
The performance of Sen. and AUC are shown in Fig. 9. DenseNet201-C-4 represents our proposed NSCGCN model that employed DenseNet201-C as the feature extractor with the 4-neighbourhood structure. It can be seen that the performance of DenseNet201-C-4 is better than others, which means that the 4-neighbourhood structure fits better in our proposed NSCGCN model compared to other neighbourhood structures. The performance of DenseNet201-C-D is the lowest among the three models. The main reason is that the feature reconstruction algorithm based on the D-neighbourhood structure will convert the feature maps into two independent subgraphs. During graph convolution, the feature information between those two subgraphs will not interact with each other, which reduces the NSCGCN performance. Therefore, the 4-neighbourhood structure is set as the neighbourhood structure in the feature reconstruction algorithm for subsequent experiments.

4.4.2. Exploration of best feature extractor

To reduce the model training cost and maximize the use of existing research results, we use a pre-trained deep CNN model based on transfer learning. In addition, to gain the best feature extractor, our proposed NSCGCN model is compared and tested under different top-layers removal schemes of different models proposed in Section 3.2. Other parameters of NSCGCN were $L = 77$, $K = 12$, and the feature reconstruction algorithms with a 4-neighbourhood structure.

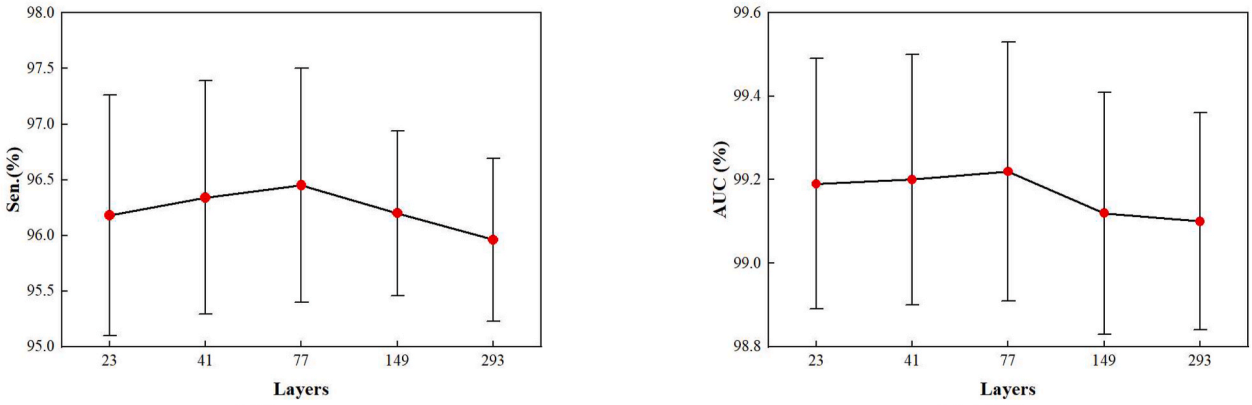
The results are shown in Table 12. By comparing the experimental results of different top-layers removal schemes based on the same pre-training model, it can be found that the more layers are removed in the pre-training model, the higher the Sen. of our proposed NSCGCN model. As in the DenseNet201 model, the Sen. value of scheme C of 96.25% is 1.3% higher than the Sen. value of scheme A of 94.87%. In the ResNet101 model, the Sen. value of scheme B of 95.66% is 1% higher than the Sen. value of scheme A of 94.58%. In the Vgg16 model, the Sen. value of scheme B of 95.42% is 0.5% higher than the Sen. value of scheme A of 94.91%. In the GoogLeNet model, the Sen. value of scheme B of 95.02% is 1.5% higher than the Sen. value of scheme A of 93.44%.

The experimental results suggest that our proposed NSCGCN model delivers excellent performance in different removal schemes. Our proposed NSCGCN model performed best using the DenseNet201-C model among all these feature extraction extractors. Compared with other models, the DenseNet201 has the largest number of convolutional layers and the maximum depth. It means that DenseNet201 has the power to extract more pixel-level features than other deep CNN models and discard the redundant and useless features in the original image.



(a) Sen. (b) AUC

Fig. 10. Performance of NSCGCN when $L = 77$ and $K = 3,6,12,24,48$



(a)Sen. (b)AUC

Fig. 11. Performance of NSCGCN when $K = 24$ and $L = 23,41,77,149,293$.

Table 13
Confusion matrix of NSCGCN.

True	predict on dataset1			predict on dataset2			predict on dataset3			predict on dataset4			predict on dataset5		
	HC	COV	OPN	HC	COV	OPN	HC	COV	OPN	HC	COV	OPN	HC	COV	OPN
HC	456	3	3	452	2	8	449	6	8	459	2	2	452	3	8
COV	3	457	2	0	460	2	0	462	1	8	454	1	4	454	4
OPN	38	1	423	36	0	426	48	2	413	41	6	416	3	0	460

Table 14
Five confusion matrix metrics of NSCGCN (Unit: %).

Performance	dataset1	dataset2	dataset3	dataset4	dataset5	avg	std
Sen.	96.39	96.54	95.32	95.68	98.34	96.45	1.05
F1	96.39	96.53	95.30	95.68	98.34	96.45	1.05
Pre.	96.57	96.63	95.50	95.97	98.35	96.61	0.97
Acc.	96.39	96.54	95.32	95.68	98.34	96.45	1.05
AUC	99.10	99.14	99.10	98.93	99.83	99.22	0.31

Therefore, we choose DenseNet201-C as the feature extractor of the NSCGCN.

4.4.3. *Exploration of best L and K*

The number of convolution kernels K and network layers L of our proposed NSCGCN are key hyperparameters that determine the model's performance.

First, to determine the optimal convolution kernel number K, that is, the optimal width of the NSCGCN, we conduct a comparative experiment of NSCGCN under different K values. The number of layers L of our proposed NSCGCN model is set to 12, and K is selected from 3, 6, 12, 24, and 48.

The experimental results are shown in Fig. 10. Our proposed model performance has improved with the kernel increase, especially when $K = 24$ and the main performance indexes reach a peak. Significantly, the value of AUC is slightly less than 99.25 as $K = 6$. The slight difference is only 0.03%, which may be raised mainly by stochastic errors in the data processing system. It can be concluded that different from the

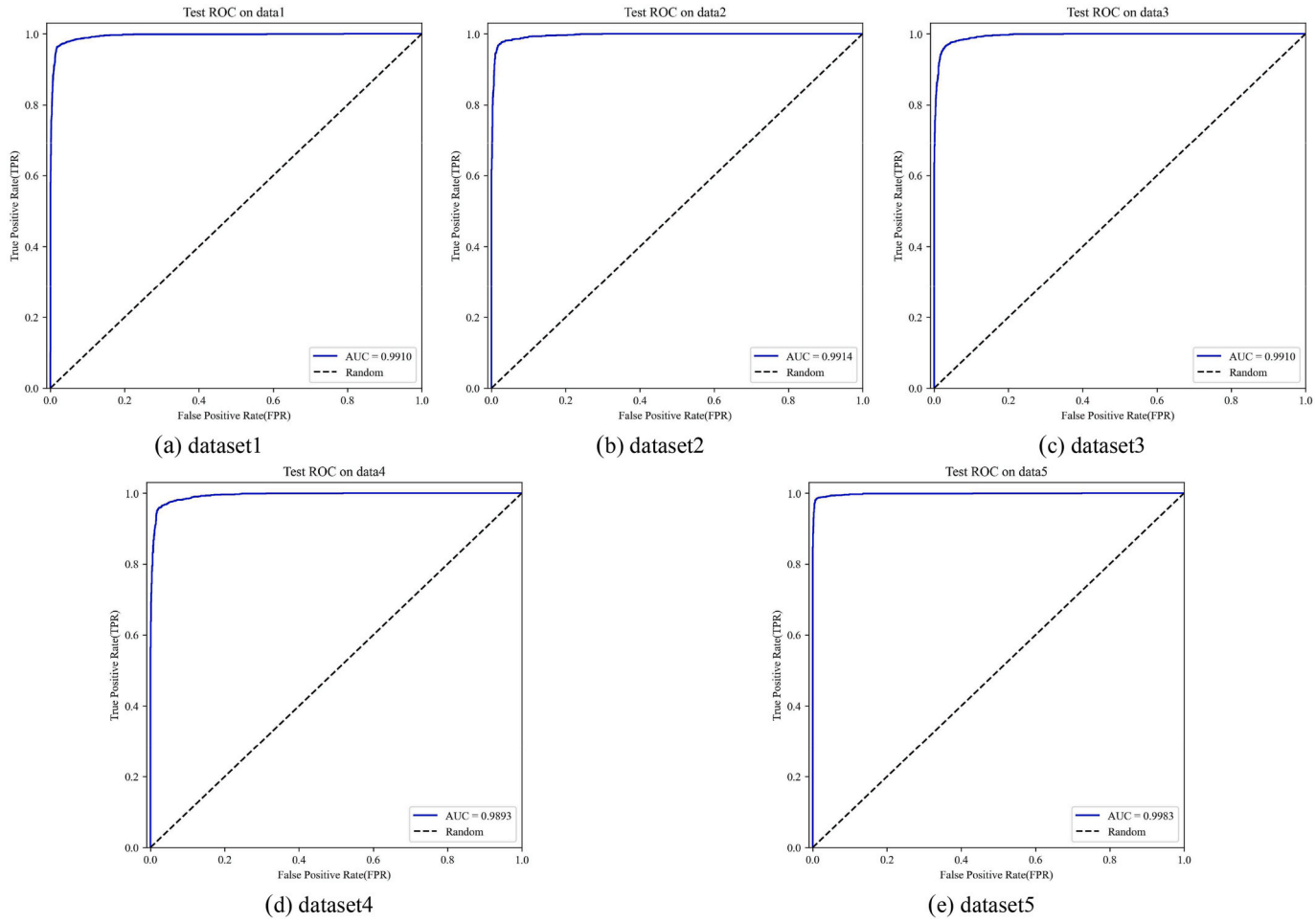


Fig. 12. ROC analyses of NSCGCN.

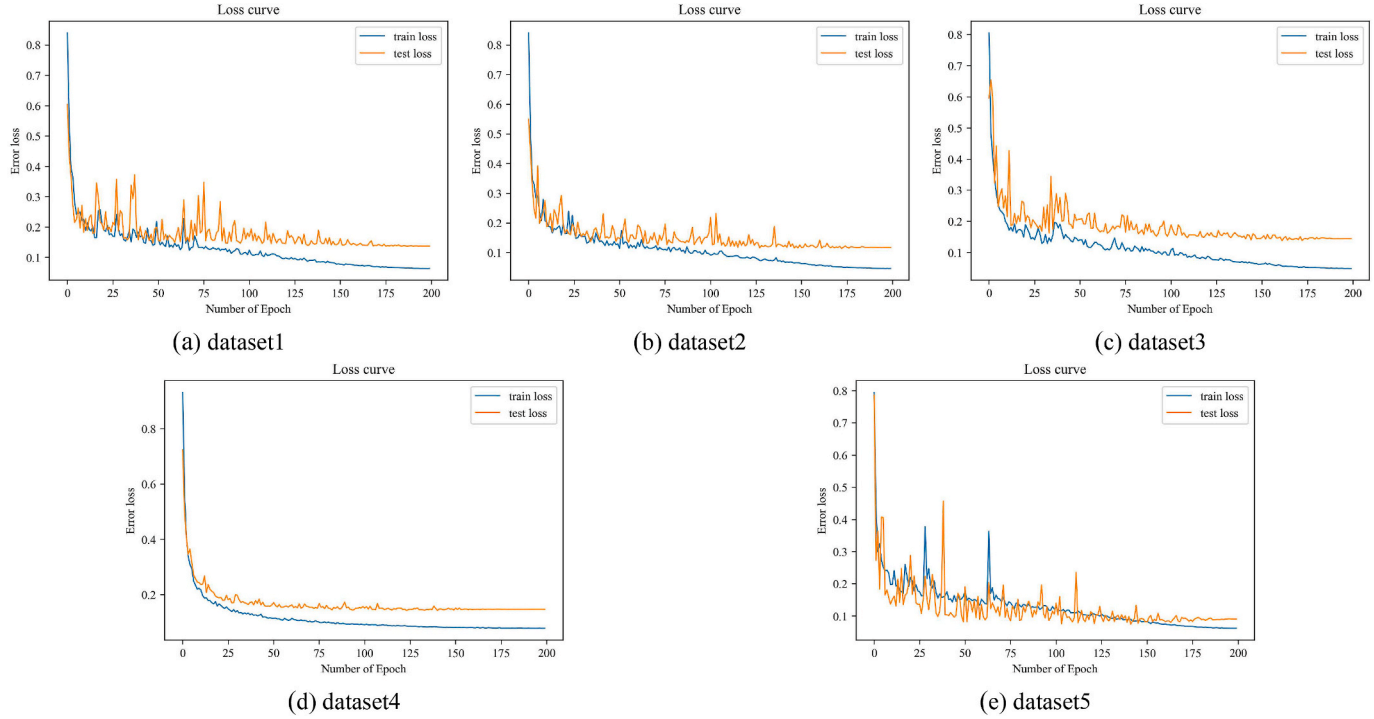


Fig. 13. Error loss vs epochs of NSCGCN.

Table 15
The HeatMap, HeatMap++, Grad-CAM, Grad-CAM++, LIME and SHAP of NSCGCN.

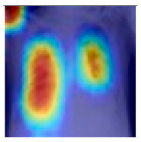
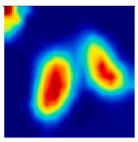
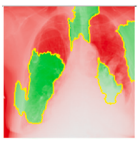
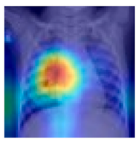
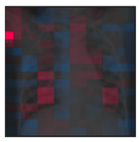
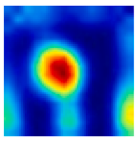
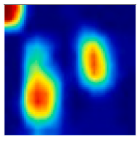
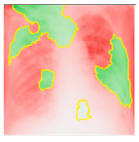
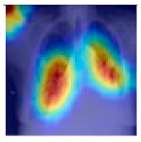
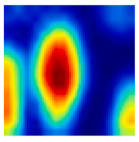
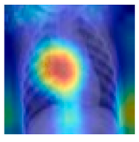
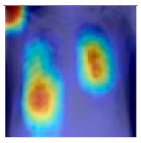
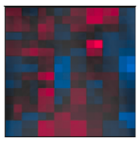
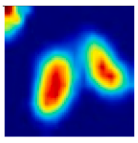
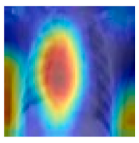
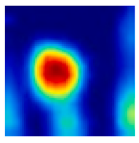
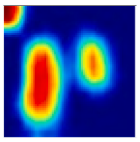
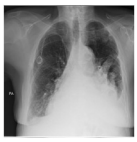
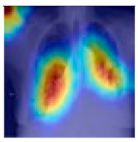
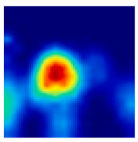
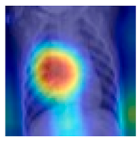
Sample	original	HeatMap	Grad-CAM	HeatMap ++	Grad-CAM ++	LIME	SHAP
COV 1							
COV 2							
OPN 1							
OPN 2							

Table 16
Performance of SOTA GCN models with the DenseNet201-C (Unit: %).

Model	Sen.	F1	Pre.	Acc.	AUC
GraphSAGE [43]	94.51 ± 1.33	94.51 ± 1.32	94.67 ± 1.26	94.51 ± 1.33	98.42 ± 0.67
GraphSAGEWithJK [25]	94.98 ± 1.13	94.98 ± 1.14	95.09 ± 1.10	94.98 ± 1.13	98.68 ± 0.58
GCN [15]	95.00 ± 1.02	94.99 ± 1.02	95.10 ± 0.98	95.00 ± 1.02	98.69 ± 0.48
GCNWithJK [25]	95.04 ± 0.66	95.04 ± 0.66	95.18 ± 0.63	95.04 ± 0.66	98.91 ± 0.40
GIN0 [44]	94.84 ± 0.71	94.84 ± 0.70	94.92 ± 0.69	94.84 ± 0.71	99.10 ± 0.22
GIN0WithJK [25]	95.32 ± 0.38	95.31 ± 0.38	95.37 ± 0.40	95.32 ± 0.38	99.13 ± 0.26
GIN [44]	94.84 ± 0.71	94.84 ± 0.70	94.95 ± 0.71	94.84 ± 0.71	99.07 ± 0.22
GINWithJK [25]	95.43 ± 0.38	95.43 ± 0.38	95.51 ± 0.38	95.43 ± 0.38	99.19 ± 0.23
Graclus [45]	94.63 ± 1.02	94.62 ± 1.02	94.74 ± 1.05	94.63 ± 1.02	98.83 ± 0.50
SAGPool [46]	95.17 ± 1.23	95.17 ± 1.23	95.31 ± 1.17	95.17 ± 1.23	98.66 ± 0.74
EdgePool [49]	94.93 ± 0.61	94.92 ± 0.61	95.03 ± 0.57	94.93 ± 0.61	98.76 ± 0.47
GlobalAttentionNet [47]	94.14 ± 0.95	94.13 ± 0.96	94.27 ± 0.96	94.14 ± 0.95	98.52 ± 0.57
Set2SetNet [50]	94.11 ± 1.08	94.10 ± 1.08	94.16 ± 1.07	94.11 ± 1.08	98.24 ± 0.57
SortPool [5]	94.25 ± 0.38	94.25 ± 0.38	94.39 ± 0.23	94.25 ± 0.38	98.69 ± 0.40
NSCGCN (Ours)	**96.45 ± 1.05**	**96.45 ± 1.05**	**96.61 ± 0.97**	**96.45 ± 1.05**	**99.22 ± 0.31**

performance change trend in the binary classification task when the K value increases, our proposed model performance does not increase all the time but shows a trend of first increasing and then decreasing. The reason is that when the value of K increases, more parameters need to be learned. Especially when dense connections are used, the number of parameters to be learned increases quadratically compared to the increase in the value of K. And more learnable parameters mean that more data is needed to train our proposed model, so the number of labelled samples limits it. Generally, when $K = 24$, our proposed model performance is the best.

We conduct comparative experiments of NSCGCN under different L to determine the optimal network layer. The number of convolution kernels K is set to 24 and the number of graph convolution layers n in each NSCGCN Block is selected from 3, 6, 12, 24, and 48. Thus, the number of convolution layers L is 23, 41, 77, 149, and 293.

The experimental results are shown in Fig. 11. The AUC of our proposed NSCGCN increases with increasing L, and the model with $K = 24$ and $L = 77$ perform the best. As L increases, the nonlinear transformation of the model becomes more complex, which is in favour of getting more implicit information. However, the existing amount of dataset cannot support a training model with a depth of more than 77 layers. These results suggest that our proposed NSCGCN model performs best when L is 77 and K is fixed at 24.

4.4.4. *Model performance analysis on the CXR dataset*

To analyze the performance of our proposed model and better demonstrate the role of the model in classification. We use confusion matrix and visualization techniques to analyze the best model of NSCGCN.

The confusion matrix of the NSCGCN with $K = 24$ and $L = 77$ is shown in Table 13. Here, COV represents COVID-19. Table 14 shows the five confusion matrix metrics of NSCGCN($L = 77$, $K = 24$) according to the confusion matrix. Since the number of samples of each type is the same, the two indicators, sensitivity and accuracy, have the same values. The experimental results show that the Acc of the dataset5 is the highest. Since the CXR images of OPN are similar to the CXR images of HC, our proposed model will recognize OPN as HC on dataset1, dataset2, dataset3 and dataset4. Additionally, our proposed model has a very high detection rate of COVID-19; the average detection rate for COVID-19 is 98.88%.

Moreover, instead of the prediction rates, we also draw the AUC as given in Fig. 12. ROC curve is the primary measuring tool for analyzing the stability and consistency of the model. The results show that our proposed NSCGCN has excellent performance. In addition, the error-loss-vs-epochs figures of NSCGCN for multiclass classification are depicted in Fig. 13. The error loss can be significantly reduced by increasing the number of training epochs. For example, in Fig. 13 (a), NSCGCN reported the maximum test error loss of 0.6 in the first epoch that is continuously reduced to 0.14 by increasing the number of training epochs to 150.

Table 17
Performance of SOTA deep CNN and different transferred deep CNN models (Unit: %).

Model	Sen.	F1	Pre.	Acc.	AUC
DenseNet201 [29]	93.75 ± 0.39	93.75 ± 0.39	93.89 ± 0.42	93.75 ± 0.39	98.40 ± 0.21
DenseNet201 (A)	91.99 ± 0.25	91.97 ± 0.25	92.07 ± 0.33	91.99 ± 0.25	98.13 ± 0.29
DenseNet201 (B)	94.03 ± 0.14	94.03 ± 0.13	94.11 ± 0.18	94.03 ± 0.14	98.86 ± 0.17
DenseNet201 (C)	94.19 ± 1.01	94.18 ± 1.03	94.26 ± 1.06	94.19 ± 1.01	98.97 ± 0.25
DenseNet121 [29]	93.44 ± 0.55	93.44 ± 0.55	93.48 ± 0.55	93.44 ± 0.55	98.49 ± 0.32
DenseNet121 (A)	91.79 ± 0.68	91.78 ± 0.68	91.84 ± 0.63	91.79 ± 0.68	98.15 ± 0.28
DenseNet121 (B)	94.29 ± 0.38	94.30 ± 0.38	94.36 ± 0.38	94.29 ± 0.38	98.82 ± 0.43
DenseNet121 (C)	94.08 ± 0.64	94.08 ± 0.64	94.20 ± 0.54	94.08 ± 0.64	98.76 ± 0.22
ResNet101 [28]	92.51 ± 1.06	92.50 ± 1.06	92.60 ± 1.10	92.51 ± 1.06	98.11 ± 0.54
ResNet101(A)	92.84 ± 0.64	92.84 ± 0.67	92.99 ± 0.63	92.84 ± 0.64	98.24 ± 0.21
ResNet101(B)	94.25 ± 0.77	94.26 ± 0.78	94.52 ± 0.67	94.25 ± 0.77	98.40 ± 0.56
ResNet50 [28]	92.36 ± 0.54	92.35 ± 0.54	92.40 ± 0.55	92.36 ± 0.54	98.06 ± 0.18
ResNet50(A)	92.88 ± 0.73	92.87 ± 0.74	92.99 ± 0.76	92.88 ± 0.73	98.40 ± 0.26
ResNet50(B)	94.78 ± 0.34	94.78 ± 0.34	94.93 ± 0.32	94.78 ± 0.34	98.67 ± 0.38
ResNet18 [28]	92.84 ± 0.58	92.83 ± 0.58	92.89 ± 0.56	92.84 ± 0.58	98.12 ± 0.35
ResNet18(A)	90.10 ± 1.13	90.13 ± 1.10	90.39 ± 1.05	90.10 ± 1.13	97.64 ± 0.54
ResNet18(B)	94.12 ± 1.37	94.12 ± 1.38	94.16 ± 1.42	94.12 ± 1.37	98.44 ± 0.59
Vgg16 [27]	93.87 ± 0.88	93.87 ± 0.88	93.95 ± 0.87	93.87 ± 0.88	97.85 ± 0.86
Vgg16(A)	94.39 ± 0.56	94.38 ± 0.56	94.50 ± 0.53	94.39 ± 0.56	98.90 ± 0.28
Vgg16(B)	94.75 ± 0.43	94.75 ± 0.43	94.84 ± 0.41	94.75 ± 0.43	98.64 ± 0.48
GoogLeNet [26]	94.71 ± 0.63	94.70 ± 0.63	94.81 ± 0.60	94.71 ± 0.63	98.67 ± 0.51
GoogLeNet(A)	91.38 ± 0.81	91.42 ± 0.79	91.65 ± 0.82	91.38 ± 0.81	97.92 ± 0.41
GoogLeNet(B)	94.32 ± 0.64	94.32 ± 0.63	94.49 ± 0.58	94.32 ± 0.64	99.00 ± 0.38
NSCGCN (Ours)	**96.45 ± 1.05**	**96.45 ± 1.05**	**96.61 ± 0.97**	**96.45 ± 1.05**	**99.22 ± 0.31**

*model(X) means only train X in the model.

The Grad-CAM [51], Grad-CAM++ [52], LIME [53], and SHAP [54] of the NSCGCN with $K = 24$ and $L = 77$ are shown in Table 15. For the COVID-19 samples COV 1 and COV 2, infection in both lungs was detected, and the HeatMaps almost completely covered both lung areas. Comparing the Grad-CAM and Grad-CAM++ of COV 1, it can be found that the HeatMaps of Grad-CAM++ cover more comprehensively. However, due to the presence of orientation markers in some of the images in the dataset, the HeatMaps covers the upper left regions of the image. For the OPN samples OPN 1 and OPN 2, the infection of the right lung close to the heart area was detected, and the HeatMaps also concentrated on the infected area. Comparing Grad-CAM and Grad-CAM++, it can be found that Grad-CAM++ covers more areas but also more invalid areas. Comparing the Grad-CAM and Grad-CAM++ of COV and OPN, the area covered by the HeatMaps is different. Similarly, the LIME analysis of NSCGCN is shown in Table 15. For the COVID-19 samples, the LIME shows that the model paid more attention to local information. For the OPN samples, the LIME shows that the model paid more attention to global information. Furthermore, in the SHAP analysis of NSCGCN, the red pixels represent positive SHAP values that increase the class probability, while blue pixels represent negative SHAP values that decrease class probability.

Table 18
Performance of SOTA COVID-19 diagnosis methods (Unit: %).

Method	Sen.	F1	Pre.	Acc.	AUC
Ozturk et al. (DarkCovidNet) [55]	95.06 ± 0.54	95.06 ± 0.54	95.17 ± 0.50	95.06 ± 0.54	98.51 ± 0.43
Afshar et al. (COVID-CAPS) [56]	92.04 ± 0.75	92.03 ± 0.78	92.17 ± 0.83	92.04 ± 0.75	97.62 ± 0.46
Elbishlawi et al. (CORONANET) [57]	95.30 ± 0.49	95.30 ± 0.48	95.45 ± 0.45	95.30 ± 0.49	98.93 ± 0.35
Wang et al. (COVIDNET) [58]	94.64 ± 0.36	94.64 ± 0.36	94.75 ± 0.33	94.64 ± 0.36	98.59 ± 0.44
Arias-Londono et al. (COVIDNET-DE) [59]	94.98 ± 0.54	94.99 ± 0.54	95.16 ± 0.46	94.98 ± 0.54	98.65 ± 0.46
Arias-Londono et al. (COVIDNET-Grad-CAM) [59]	94.83 ± 0.40	94.83 ± 0.40	94.98 ± 0.34	94.83 ± 0.40	98.59 ± 0.46
Redie et al. (Modified DarkCovidNet) [60]	94.38 ± 0.70	94.38 ± 0.70	94.43 ± 0.70	94.38 ± 0.70	98.61 ± 0.36
Bhowal et al. (Choquet Integral-Based Ensemble) [61]	94.67 ± 0.84	94.64 ± 0.85	95.13 ± 0.65	94.67 ± 0.85	97.78 ± 0.87
Ours (NSCGCN)	**96.45 ± 1.05**	**96.45 ± 1.05**	**96.61 ± 0.97**	**96.45 ± 1.05**	**99.22 ± 0.31**

4.5. Comparison of SOTA GCN models on the CXR dataset

In this section, we compare our proposed NSCGCN model with other SOTA GCN models on the DenseNet201-C. As shown in Table 16, the results show that our proposed NSCGCN model is the best method compared with the SOTA GCN models. The proposed NSCGCN model is best because our proposed NSCGCN is a deep model, which can effectively improve the reusability of features and the accuracy of the final prediction results. Our proposed model comprises GCN and NSC without involving more advanced graph convolution layers like SAGPool. Our NSCGCN significantly improves Acc. and Pre. by 1.45% and 1.5%, compared to the GCNs. Furthermore, the dense ship connection in our proposed model works better than the JK module, which is used in some SOTA GCN models [22]. All these results demonstrate the effectiveness of NSC in deep GCN.

4.6. Comparison of SOTA deep CNN models on the CXR dataset

In this section, to verify the effectiveness of our proposed model, we train SOTA deep CNN models, such as GoogleNet [26], ResNet18 [28], ResNet101 [28], DenseNet201 [29], and Vgg16 [27] on the same dataset, and only train the top-layers removed of them in Section 3.2. under the same condition. To make these deep CNN suit the three classification requirements, we build the new SoftMax and three-classification layers o replace the original classification layers.

The performance of different deep CNN is shown in Table 17; the best performance of SOTA deep CNN is ResNet50(B) as Sen. = 94.78%, F1 = 94.78%, Pre. = 94.93%, Acc. = 94.78%, AUC = 98.67%. It is mainly due to the limited size of the dataset that DenseNet201 is not sufficiently learned. However, the architectural superiority of pre-trained DenseNet201 helps NSCGCN extract enough representative features, and the modelling ability of GCN helps NSCGCN extract underlying relationships between features. They complete each other; hence the performance of NSCGCN is better than the SOTA deep CNN.

4.7. Comparison of SOTA COVID-19 diagnosis methods on the CXR dataset

Now some works have been done to study the high-precision pneumonia diagnosis system. To provide a fair comparison, we compare our proposed COVID-19 diagnosis method with the other methods validated

on the same dataset. For OPN, COVID-19, and HC three-classification tasks, the performance of SOTA methods is shown in Table 18. The performance of CORONANET is the best in the SOTA networks as Sen. = 95.30%, F1 = 95.30%, Pre. = 95.45%, Acc. = 95.30%, AUC = 98.93%. It is higher than the ResNet50(B) but lower than some GCN models.

And then, it can be seen that NSCGCN is superior to all the latest methods. Furthermore, our proposed method achieves the highest sensitivity, which means that our proposed method can show a more robust ability of pneumonia image discrimination while ensuring high accuracy.

5. Conclusion and future directions

In this paper, we propose a novel feature fusion algorithm NSC and construct a novel deep GCN NSCGCN to complete pneumonia diagnosis. Experiments show that our COVID-19 diagnosis method is better than fourteen SOTA GCN, sixteen SOTA deep CNN, and eight SOTA COVID-19 diagnosis models. The reason why our NSCGCN has the best performance is (i) because NSCGCN is a deep GCN model. It is more expressive and can be used to represent more complex situations. (ii) The proposed NSC includes graphics transformation and feature fusion process. It helps significantly increase the nonlinear characteristics of the model. (iii) Transferred networks help reduce training time and improve training efficiency.

This research has two shortcomings: (i) We did not achieve the best structure for feature reconstruction. We turn feature reconstruction into an adaptive process in the future. (ii) The model still has an over-fitting phenomenon. We will try to collect more data and further improve the diagnostic performance of COVID-19.

The future work directions are: (i) Combine different feature structures to create an integrated width GCN. (ii) Fuse multimodal data information to improve diagnosis accuracy. (iii) Expand the dataset and test our model on different sources of COVID-19.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

Key Science and Technology Program of Henan Province, China (212102310084); Key Scientific Research Projects of Colleges and Universities in Henan Province, China (22A520027); National Natural Science Foundation of China (62276092); Henan Association for Science and Technology, China (HNKJZK-2022-11A); Hope Foundation for Cancer Research, UK (RM60G0680); Royal Society International Exchanges Cost Share Award, UK (RP202G0230); Medical Research Council Confidence in Concept Award, UK (MC_PC_17171); British Heart Foundation Accelerator Award, UK (AA/18/3/34220); Sino-UK Industrial Fund, UK (RP202G0289); Global Challenges Research Fund (GCRF), UK (P202PF11); LIAS Pioneering Partnerships award, UK (P202ED10); Data Science Enhancement Fund, UK (P202RE237).

References

[1] WHO COVID-19 Dashboard, Geneva: World Health Organization, 2020. Available online: https://covid19.who.int/. (Accessed 10 December 2021).
[2] R. Pu, S. Liu, X. Ren, D. Shi, Y. Ba, Y. Huo, W. Zhang, L. Ma, Y. Liu, Y. Yang, N. Cheng, The screening value of RT-LAMP and RT-PCR in the diagnosis of COVID-19: systematic review and meta-analysis, J. Virol Methods 300 (2022), 114392.
[3] Y. Artik, A.B. Coşğun, N.P. Cesur, N. Hızel, Y. Uyar, H. Sur, A. Ayan, Comparison of COVID-19 laboratory diagnosis by commercial kits: effectivity of RT-PCR to the RT-LAMP, J. Med. Virol. 94 (2022) 1998–2007.
[4] V. Kumar, A. Zarrad, R. Gupta, O. Cheikhrouhou, COV-DLS: prediction of COVID-19 from X-rays using enhanced deep transfer learning techniques, Journal of Healthcare Engineering 2022 (2022).
[5] M. Zhang, Z. Cui, M. Neumann, Y. Chen, An end-to-end deep learning architecture for graph classification, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[6] T.-A. Song, S.R. Chowdhury, F. Yang, H. Jacobs, G. El Fakhri, Q. Li, K. Johnson, J. Dutta, Graph convolutional neural networks for Alzheimer's disease classification, in: 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019), IEEE, 2019, pp. 414–417.
[7] Y. Chu, G. Wang, L. Cao, L. Qiao, M. Liu, Multi-scale graph representation learning for autism identification with functional MRI, Front. Neuroinf. 15 (2022), 802305.
[8] X. Song, F. Zhou, A.F. Frangi, J. Cao, X. Xiao, Y. Lei, T. Wang, B. Lei, Multi-center and multi-channel pooling GCN for early AD diagnosis based on dual-modality fused brain network, IEEE Trans. Med. Imag. (2022), https://doi.org/10.1109/TMI.2022.3187141. In press.
[9] H. Ye, D.-H. Wang, J. Li, S. Zhu, C. Zhu, Improving histopathological image segmentation and classification using graph convolution network, in: Proceedings of the 2019 8th International Conference on Computing and Pattern Recognition, 2019, pp. 192–198.
[10] A. Elazab, M.A. Elfattah, Y. Zhang, Novel multi-site graph convolutional network with supervision mechanism for COVID-19 diagnosis from X-ray radiographs, Appl. Soft Comput. 114 (2022), 108041.
[11] S. Akbar, S. Khan, F. Ali, M. Hayat, M. Qasim, S. Gul, iHBP-DeepPSSM, Identifying hormone binding proteins using PsePSSM based evolutionary features and deep learning approach, Chemometr. Intell. Lab. Syst. 204 (2020), 104103.
[12] A. Ahmad, S. Akbar, S. Khan, M. Hayat, F. Ali, A. Ahmed, M. Tahir, Deep-AntiFP: prediction of antifungal peptides using distanct multi-informative features incorporating with deep neural networks, Chemometr. Intell. Lab. Syst. 208 (2021), 104214.
[13] S. Akbar, M. Hayat, M. Tahir, S. Khan, F.K. Alarfaj, cACP-DeepGram: classification of anticancer peptides via deep neural network and skip-gram-based word embedding model, Artif. Intell. Med. 131 (2022), 102349.
[14] Q. Li, Z. Han, X.-M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[15] M. Welling, T.N. Kipf, Semi-supervised classification with graph convolutional networks, in: J. International Conference on Learning Representations, ICLR 2017, 2018.
[16] G. Li, M. Muller, A. Thabet, B. Ghanem, Deepgcns: can gcns go as deep as cnns?, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9267–9276.
[17] C.-S. Tang, C.-C. Hu, J.-D. Sun, H.-F. Sima, Deep learning-based medical images analysis evolved from convolution to graph convolution, Journal of Image and Graphics 26 (2021) 2078–2093.
[18] S. Parisot, S.I. Ktena, E. Ferrante, M. Lee, R. Guerrero, B. Glocker, D. Rueckert, Disease prediction using graph convolutional networks: application to autism spectrum disorder and Alzheimer's disease, Med. Image Anal. 48 (2018) 117–130.
[19] A. Marzullo, G. Kocevar, C. Stamile, F. Durand-Dubief, G. Terracina, F. Calimeri, D. Sappey-Marinier, Classification of multiple sclerosis clinical profiles via graph convolutional neural networks, Front. Neurosci. 13 (2019) 594.
[20] H. Du, J. Feng, M. Feng, Zoom in to where it Matters: a Hierarchical Graph Based Model for Mammogram Analysis, 2019.
[21] O. Ronneberger, P. Fischer, T. Brox, U-net, Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.
[22] W. Huang, Y. Rong, T. Xu, F. Sun, J. Huang, Tackling Over-smoothing for General Graph Convolutional Networks, 2020. ArXiv Preprint ArXiv:2008.09864.
[23] C. Yang, R. Wang, S. Yao, S. Liu, T. Abdelzaher, Revisiting Over-smoothing in Deep GCNs, 2020. ArXiv Preprint ArXiv:2003.13663.
[24] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, C.-J. Hsieh, Cluster-gcn: an efficient algorithm for training deep and large graph convolutional networks, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 257–266.
[25] K. Xu, C. Li, Y. Tian, T. Sonobe, K. Kawarabayashi, S. Jegelka, Representation learning on graphs with jumping knowledge networks, in: International Conference on Machine Learning, PMLR, 2018, pp. 5453–5462.
[26] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[27] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, 2014. ArXiv Preprint ArXiv:1409.1556.
[28] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[29] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.
[30] J. Cheng, S. Tian, L. Yu, C. Gao, X. Kang, X. Ma, W. Wu, S. Liu, H. Lu, ResGANet: residual group attention network for medical image classification and segmentation, Med. Image Anal. 76 (2022), 102313.
[31] Y.K. Rathore, R.R. Janghel, Prediction of stage of alzheimer's disease DenseNet deep learning model, in: Next Generation Healthcare Systems Using Soft Computing Techniques, CRC Press, 2022, pp. 105–121.
[32] C.R. Jack Jr., M.A. Bernstein, N.C. Fox, P. Thompson, G. Alexander, D. Harvey, B. Borowski, P.J. Britson, J.L. Whitwell, C. Ward, A.M. Dale, J.P. Felmlee, J. L. Gunter, D.L.G. Hill, R. Killiany, N. Schuff, S. Fox-Bosetti, C. Lin, C. Studholme, C. S. DeCarli, G. Krueger, H.A. Ward, G.J. Metzger, K.T. Scott, R. Mallozzi, D. Blezek, J. Levy, J.P. Debbins, A.S. Fleisher, M. Albert, R. Green, G. Bartzokis, G. Glover,

J. Mugler, M.W. Weiner, The Alzheimer's disease neuroimaging initiative (ADNI): MRI methods, J. Magn. Reson. Imag. 27 (2008) 685–691.
[33] L. Kong, J. Cheng, Classification and detection of COVID-19 X-Ray images based on DenseNet and VGG16 feature fusion, Biomed. Signal Process Control 77 (2022), 103772.
[34] K. Abbasi, A. Poso, J. Ghasemi, M. Amanlou, A. Masoudi-Nejad, Deep transferable compound representation across domains and tasks for low data drug discovery, J. Chem. Inf. Model. 59 (2019) 4528–4539.
[35] P. Razzaghi, K. Abbasi, M. Shirazi, N. Shabani, Modality adaptation in multimodal data, Expert Syst. Appl. 179 (2021) 115–126.
[36] X. Yu, J. Wang, Q.-Q. Hong, R. Teku, S.-H. Wang, Y.-D. Zhang, Transfer learning for medical images analyses: a survey, Neurocomputing 489 (2022) 230–254.
[37] L. Wang, H. Wang, Y. Huang, B. Yan, Z. Chang, Z. Liu, M. Zhao, L. Cui, J. Song, F. Li, Trends in the application of deep learning networks in medical image analysis: evolution between 2012 and 2020, Eur. J. Radiol. 146 (2022), 110069.
[38] B. Ay, B. Tasar, Z. Utlu, K. Ay, G. Aydin, Deep transfer learning-based visual classification of pressure injuries stages, Neural Comput. Appl. (2022) 1–12.
[39] A. Ahmed, Classification of gastrointestinal images based on transfer learning and denoising convolutional neural networks, in: Proceedings of International Conference on Data Science and Applications, Springer, 2022, pp. 631–639.
[40] M.A. Jones, R. Faiz, Y. Qiu, B. Zheng, Improving mammography lesion classification by optimal fusion of handcrafted and deep transfer learning features, Phys. Med. Biol. 67 (2022), 054001.
[41] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, M.M. Bronstein, Geometric deep learning on graphs and manifolds using mixture model cnns, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5115–5124.
[42] Y. Zhan, H. Pan, Q. Han, X. Xie, Z. Zhang, X. Zhai, Medical image clustering algorithm based on graph entropy, in: 2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), IEEE, 2015, pp. 1151–1157.
[43] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 1024–1034.
[44] K. Xu, W. Hu, J. Leskovec, S. Jegelka, How Powerful Are Graph Neural Networks?, 2018. ArXiv Preprint ArXiv:1810.00826.
[45] I.S. Dhillon, Y. Guan, B. Kulis, Weighted graph cuts without eigenvectors a multilevel approach, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2007) 1944–1957.
[46] J. Lee, I. Lee, J. Kang, Self-attention graph pooling, in: International Conference on Machine Learning, PMLR, 2019, pp. 3734–3743.
[47] Y. Li, R. Zemel, M. Brockschmidt, D. Tarlow, Gated graph sequence neural networks, in: Proceedings of ICLR'16, 2016.
[48] P. Foret, A. Kleiner, H. Mobahi, B. Neyshabur, Sharpness-aware Minimization for Efficiently Improving Generalization, 2020. ArXiv Preprint ArXiv:2010.01412.
[49] F. Diehl, Edge Contraction Pooling for Graph Neural Networks, 2019. ArXiv Preprint ArXiv:1905.10990.
[50] O. Vinyals, S. Bengio, M. Kudlur, Order Matters: Sequence to Sequence for Sets, 2015. ArXiv Preprint ArXiv:1511.06391.
[51] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-cam: visual explanations from deep networks via gradient-based localization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 618–626.
[52] A. Chattopadhay, A. Sarkar, P. Howlader, V.N. Balasubramanian, Grad-cam++: generalized gradient-based visual explanations for deep convolutional networks, in: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2018, pp. 839–847.
[53] M.T. Ribeiro, S. Singh, C. Guestrin, " Why should i trust you?" Explaining the predictions of any classifier, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1135–1144.
[54] S.M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, Adv. Neural Inf. Process. Syst. (2017) 30.
[55] T. Ozturk, M. Talo, E.A. Yildirim, U.B. Baloglu, O. Yildirim, U.R. Acharya, Automated detection of COVID-19 cases using deep neural networks with X-ray images, Comput. Biol. Med. 121 (2020), 103792.
[56] P. Afshar, S. Heidarian, F. Naderkhani, A. Oikonomou, K.N. Plataniotis, A. Mohammadi, Covid-caps: a capsule network-based framework for identification of covid-19 cases from x-ray images, Pattern Recogn. Lett. 138 (2020) 638–643.
[57] S. Elbishlawi, M.H. Abdelpakey, M.S. Shehata, M.M. Mohamed, CORONA-net: diagnosing COVID-19 from X-ray images using Re-initialization and classification networks, Journal of Imaging 7 (2021) 81.
[58] L. Wang, Z.Q. Lin, A. Wong, Covid-net: a tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images, Sci. Rep. 10 (2020) 1–12.
[59] J.D. Arias-Londoño, J.A. Gomez-Garcia, L. Moro-Velázquez, J.I. Godino-Llorente, Artificial Intelligence applied to chest X-Ray images for the automatic detection of COVID-19. A thoughtful evaluation approach, IEEE Access 8 (2020) 226811–226827.
[60] D.K. Redie, A.E. Sirko, T.M. Demissie, S.S. Teferi, V.K. Shrivastava, O.P. Verma, T. K. Sharma, Diagnosis of COVID-19 Using Chest X-Ray Images Based on Modified DarkCovidNet Model, Evolutionary Intelligence, 2022, pp. 1–10.
[61] P. Bhowal, S. Sen, J.H. Yoon, Z.W. Geem, R. Sarkar, Choquet integral and coalition game-based ensemble of deep learning models for covid-19 screening from chest x-ray images, IEEE Journal of Biomedical and Health Informatics 25 (2021) 4328–4339.