Biomedical Signal Processing and Control 86 (2023) 105064

Contents lists available at ScienceDirect

Biomedical Signal Processing and Control

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

Medical image mis-segmentation region refinement framework based on dynamic graph convolution☆

Haocheng Liang [a], Jia Lv [a,b,*], Zeyu Wang [a], Ximing Xu [c,*]

[a] College of Computer and Information Sciences, Chongqing Normal University, Chongqing 401331, China
[b] National Center for Applied Mathematics in Chongqing, Chongqing Normal University, Chongqing 401331, China
[c] Children's Hospital of Chongqing Medical University, National Clinical Research Center for Child Health and Disorders, Ministry of Education Key Laboratory of Child Development and Disorders, Chongqing 400014, China

ARTICLE INFO

Keywords:
Medical image segmentation
Convolutional neural network
Graph convolutional network
Potential mis-segmented region extraction network
Dynamic graph learning

ABSTRACT

In medical image segmentation tasks, it is hard for traditional Convolutional Neural Network (CNN) to capture essential information such as spatial structure and global contextual semantic features since it suffers from a limited receptive field. The deficiency weakens the CNN segmentation performance in the lesion boundary regions. To handle the aforementioned problems, a medical image mis-segmentation region refinement framework based on dynamic graph convolution is proposed to refine the boundary and under-segmentation regions. The proposed framework first employs a lightweight dual-path network to detect the boundaries and nearby regions, which can further obtain potentially misclassified pixels from the coarse segmentation results of the CNN. Then, we construct the pixels into the appropriate graphs by CNN-extracted features. Finally, we design a dynamic residual graph convolutional network to reclassify the graph nodes and generate the final refinement results. We chose UNet and its eight representative improved networks as the basic networks and tested them on the COVID, DSB, and BUSI datasets. Experiments demonstrated that the average Dice of our framework is improved by 1.79%, 2.29%, and 2.24%, the average IoU is improved by 2.30%, 3.53%, and 2.39%, and the Se is improved by 5.08%, 4.78%, and 5.31% respectively. The experimental results prove that the proposed framework has the refinement capability to remarkably strengthen the segmentation result of the basic network. Furthermore, the framework has the advantage of high portability and usability, which can be inserted into the end of mainstream medical image segmentation networks as a plug-and-play enhancement block.

1. Introduction

With the rapid development of medical imaging technology, medical images such as Computed Tomography (CT) and Ultrasound (US) have become essential ways of disease screening and medical-aided diagnosis [1]. For example, CT images play a vital role in screening for infectious diseases (*e.g.*, COVID-19), and US images are important for early diagnosis and cancer diagnosis (*e.g.*, breast tumors). Medical image segmentation is a critical basis for medical image analysis. However, manually sketching and accurately analyzing medical images is laborious and time-consuming for clinicians, requiring extensive clinical experience [2]. Therefore, it has substantial practical value to employ computer vision technology to process vast amounts of medical images.

In recent years, CNN has been widely used in different medical image segmentation tasks since its powerful capability to classify images at the pixel level. FCN [3] and UNet [4] are representative networks of encoded-decoded architecture. Both of them complement low-level spatial features to high-level semantic features through skip connection, which effectively augments the quality of the features. Based on UNet, many researchers explore improved work in various directions for

☆ This document is the results of the research project funded by The Key Project of Scientific and Technological Research of Chongqing MunicipalEducation Commission, China (KJZD-K202200511), Technology Foresight and System Innovation Project of Chongqing Science andTechnology Bureau, China (2022TFII-OFX0044), National Clinical Research Center for Child Health and Disorders (Children's Hospitalof Chongqing Medical University, Chongqing, China), China (NCRCCHD-2022-HP-01).

* Corresponding authors at: College of Computer and Information Sciences, Chongqing Normal University, Chongqing 401331, China (J. Lv). Children's Hospital of Chongqing Medical University, National Clinical Research Center for Child Health and Disorders, Ministry of Education Key Laboratory of Child Development and Disorders, Chongqing 400014, China (X. Xu).

E-mail addresses: lvjia@cqnu.edu.cn (J. Lv), ximing@hospital.cqmu.edu.cn (X. Xu).

https://doi.org/10.1016/j.bspc.2023.105064
Received 14 December 2022; Received in revised form 9 April 2023; Accepted 23 May 2023
Available online 8 June 2023
1746-8094/

different medical segmentation tasks. UNet++ [5] strengthens the learning capability by mitigating the semantic gap between encoder and decoder through dense skip connections and is validated on liver and cell datasets. UNet3+ [6] utilizes full-scale skip connections to mine more semantic information from multiple scales and achieves significant segmentation gains on liver and spleen datasets. UNeXt [7] further improves the segmentation accuracy by introducing the shifted MLP mechanism at the bottom of UNet with lower computational complexity. YNet [8] proposes to enhance the capability of learning cross-domain features by incorporating spectral domain and spatial domain features. The above algorithms significantly improve the segmentation accuracy of medical images, but these convolution-based segmentation networks still face some hard-to-overcome challenges. For instance, due to the natural characteristics of CT and US imaging, some lesion boundary regions tend to be ambiguous, which are also susceptible to motion artifacts. In addition, due to the similarity between the background and the lesion boundary regions, the pixels in the regions are easier misclassified than those inside the lesion [9]. These challenges constrain the capability of CNN to extract boundary information, making it difficult to segment the boundary accurately. However, the boundary shapes have crucial anatomical significance, providing a vital basis for clinical diagnosis.

The essence of the above problems is that traditional CNN can only process image data in Euclidean space by sliding operations with fixed-size convolution kernels. Limited by the local receptive field size, CNN is hard to capture global contextual semantic information adequately [10] and is vulnerable to background interference since it cannot holistically extract the boundary information. In addition, frequent convolution and pooling operations also damage spatial structure information [18], which is prone to blurring at the lesion boundaries. Therefore, how to effectively obtain spatial structure and global contextual semantic information is the focus of researchers in recent years. A couple of studies [11,12] deepen the layers of the network by fusing multiple CNN to obtain the global receptive field indirectly by stacking the local receptive field, which alleviates the inherent limitations of CNN to a certain extent. In another way, some studies [13,14] introduce the transformer into UNet to enhance the capability of acquiring contextual semantic information directly. UCTransNet [15] argues that pure skip connections cannot effectively complement the critical spatial structure information, so it designs a multi-scale channel cross-fusion method based on Transformer to replace UNet skip connection, which achieves SOTA on glandular datasets. However, each of those algorithms inevitably complicates the network structure, which is harder to train.

Recently, Graph Convolutional Network (GCN) has attracted much attention from researchers with the advantages of efficiently extracting global context semantic relations and irregular structural information from the graph structure. Additionally, the architecture of GCN is relatively lightweight, which can achieve higher performance without too deep layer stacking. Shin et al. [16] first apply GCN to segment retinal vessels by constructing the vessel skeleton as the graphs. The graphs are then fed into a GCN consisting of only 2-layers graph convolutional layers to acquire more spatial information. However, they do not consider the correlation between graph nodes and ignore the importance of edge weights between nodes. Lu et al. [17] construct each pixel as a graph node based on feature maps, enabling GCN to sufficiently utilize the CNN feature information. It enhances the segmentation capability of the network and performs well on lung and gastroscopy datasets. However, the adjacency matrix in this algorithm is fixed, which means the edge connectivity between nodes cannot be dynamically adjusted during GCN training. This flaw weakens the segmentation performance of GCN. Unlike the two above methods that directly incorporate GCN into CNN, several studies [18,19] propose new graph construction strategies to further exploit the potential of GCN by utilizing the node and edge information more effectively. Soberanis-Mukul et al. [18] employ Monte Carlo dropout (MCDO) to analyze the uncertainty of CNN and quadratic classifying the pixels with high uncertainty by 2-layer GCN in a semi-supervised way. However, MCDO relies on the dropout layer to analyze uncertainty, which is gradually replaced by the normalization layer with better performance since it may lead to unstable during CNN training. Tian et al. [19] extract features by CNN and utilize GCN to fit the node positions of lesion boundary by regression to gain more precise boundary segmentation results. However, in this graph construction strategy, the number of boundary nodes is limited and the shape is predetermined. When the lesion boundary is highly irregular and complex, the actual shape is difficult to fit and tends to produce overly smooth segmentation results. Based on Lu et al. [17], Liu et al. [20] fuse the features of the last two layers in UNet decoder as graph node features and construct the corresponding adjacency matrices based on the Gaussian kernel, which can cope with the variation of pancreatic size. By adding a 2-layer graph convolutional layers for additional loss supervision, the distorted spatial structure information in the decoder is better mined and complemented. However, this algorithm also constructs graph nodes with each pixel in the feature map, which faces the problem of heavy spatial and temporal consumption. Unlike the above-mentioned literature, which constructs the adjacency matrix by function calculation or manual design, Xu et al. [21] propose to combine the multi-scale features of encoder and bottleneck, feed into a convolutional layer and learn the initial adjacency matrix, which improves the representation of graph structure. However, the adjacency matrix is still static during training, which means GCN is inevitably affected by noise and redundant information during the information propagation.

To overcome the problem that CNN tends to be disturbed by background and performs poorly at the boundary regions, we propose a medical image mis-segmentation region refinement framework based on dynamic graph convolution to maximize GCN support for CNN. Our framework can obtain the potential mis-segmentation regions from the coarse segmentation results of CNN, then leverage GCN to fix the misclassified pixels in this region through its excellent capability in capturing spatial structure and global contextual semantic information. Our method is not affected by the degree of regularity of the boundary regions. Furthermore, it can guide the GCN to focus on the wrong regions, which avoids introducing too many correctly classified background pixels. It can also reduce the cost of graph construction and the computational complexity of GCN in the following step. Specifically, the main contributions of this paper are as follows:

(1) Aiming at CNN performing weakly in the boundary regions, our framework proposes a Potentially mis-segmentation Region Extraction Network (PRENet) to predict potentially misclassified pixels as Region of Interest (ROI) from the coarse segmentation results. By supervised learning of boundary and misclassified region labels, PRENet can efficiently detect the boundaries and their surrounding regions further to obtain potentially misclassified pixels as ROI.
(2) We propose a more efficient graph construction approach to model the image into graphs suitable for GCN. The pixel in the ROI is constructed to graph node pixel by pixel by utilizing the CNN-extracted feature, and the graph adjacency matrix is constructed by calculating the Euclidean distance and similarity between nodes. In this way, we convert the refinement task of coarse segmentation into a graph node classification task.
(3) We design a simple but efficient Dynamic Residual Graph Convolutional Network (DR-GCN) to capture the spatial structure and global contextual information that CNN lack. DR-GCN can reclassify the constructed graph and obtain the final segmentation refinement results. Additionally, DR-GCN can improve the anti-interference capability of the network by incorporating Dynamic Graph Learning Module (DGLM), which can dynamically adjust the edge weights during training and reduce the association between different semantic categories.

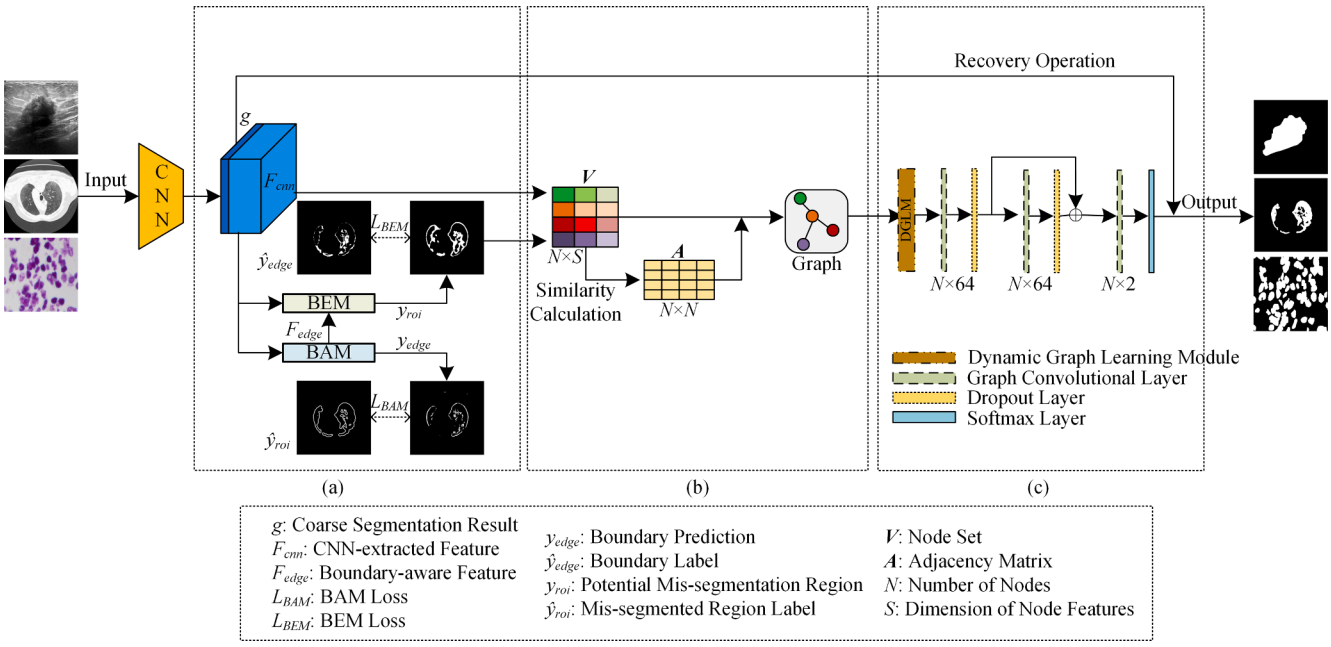

Fig. 1. The architecture of our framework: (a) PRENet; (b) GCM; (c) DR-GCN. PRENet is a dual-path structure consisting of a Boundary Aware Module (BAM) and a Boundary Enhancement Module (BEM).

In conclusion, our framework can effectively solve the problem of poor performance of CNN in segmenting the lesion boundary regions without modifying the existing basic network structure. By inserting it as a plug-and-play block into the end of mainstream medical image segmentation networks, our framework can improve segmentation performance from coarse to fine and achieve high portability and usability.

2. Methods

2.1. Overview of the framework architecture

The architecture of the proposed framework consists of three parts: PRENet, Graph Construction Module (GCM) and DR-GCN, as shown in Fig. 1. After CNN outputs the coarse segmentation result, our framework first employs PRENet to predict the potential mis-segmented regions as ROI by paying more attention to the lesion boundary regions. Then, the CNN-extracted feature and the ROI are used to construct a suitable graph by GCM. Finally, we feed the graph into DR-GCN to reclassify each graph node, which is used to replace the ROI in the coarse segmentation result to obtain the final refined segmentation result. Obviously, our framework is an enhancement block that provides a useful complement to CNN through secondary segmentation. Moreover, the refinement framework is independent of CNN, which can be used as a plug-and-play block inserted into the end of mainstream medical image segmentation network.

2.2. Potentially mis-segmented region extraction network

The mis-segmented regions are usually concentrated in the lesion boundaries and nearby regions since CNN performs weakly in these regions. Some studies [9,22–24] prove that adequate boundary information can effectively resolve segmentation blurring and ambiguity in

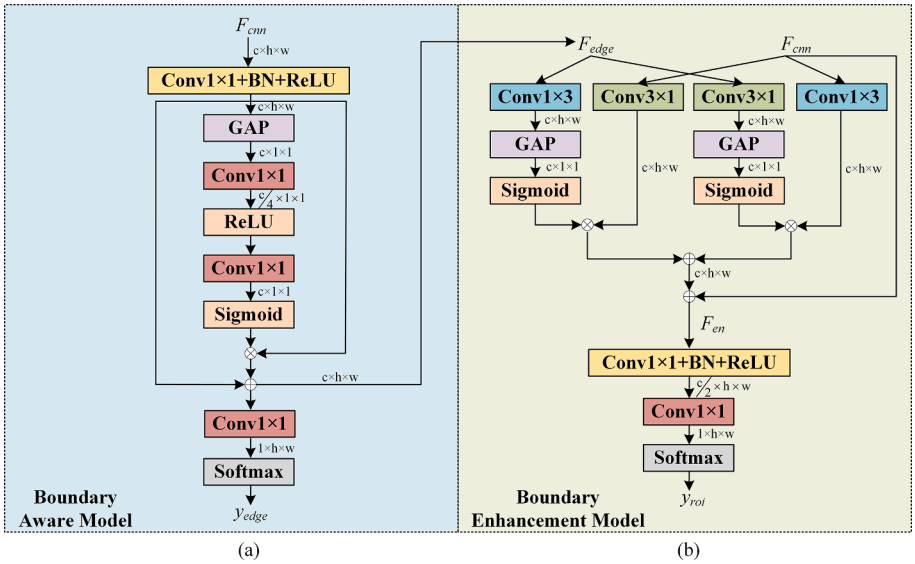

Fig. 2. The architecture of PRENet: (a) is BAM and (b) is BEM. "⊕" denotes element-wise addition, and "⊗" denotes dot product.

boundary regions, which helps for improving the overall segmentation performance. Inspired by them, we propose a lightweight network called PRENet, which is designed to locate potential mis-segmentation regions by extracting lesion boundaries and their nearby regions as much as possible. This approach supports DR-GCN to focus on the incorrect regions, which can minimize interference from the irrelevant background.

As shown in Fig. 1(a), the main body of PRENet consists of dual paths: Boundary Aware Module (BAM) and Boundary Enhancement Module (BEM). The core idea is employing BAM to capture boundary feature from CNN-extracted feature, then feed them into BEM for further enhancement, which is finally used to predict potential mis-segmentation regions from the coarse segmentation results.

2.2.1. Boundary aware module

Given the CNN-extracted feature $F_{cnn} \in \mathbb{R}^{c\times h\times w}$, where c is the channel dimension and h × w is the image resolution, BAM aims to extract the boundary-aware feature $F_{edge} \in \mathbb{R}^{c\times h\times w}$ by supervised learning the boundary label $\hat{y}_{edge} \in \mathbb{R}^{1\times h\times w}$, which is generated from the Ground True *GT* by Sobel operator [25]. We choose the input feature of the last decoder layer as the F_{cnn}. It is worth noting that the channel of F_{cnn} is dependent on various basic networks (*e.g.*, $F_{cnn} \in \mathbb{R}^{128\times h\times w}$ in UNet).

Fig. 2(a) shows the architecture and dimensional details of BAM. To be specific, we first project F_{cnn} through 1 × 1 convolution, Batch Normalization (BN), and ReLU sequences, then employ channel attention mechanism (*i.e.*, Global Average Pooling (GAP), two 1 × 1 convolution, Sigmoid, and residual connection operations) to augment the channels that contain more spatial structure information. In this way, the boundary-aware feature F_{edge} that BAM requires is produced. Furthermore, to guide BAM learning sufficient boundary information, the channel of F_{edge} is finally reduced to one by 1 × 1 convolution and outputs the lesion boundary prediction $y_{edge} \in \mathbb{R}^{1\times h\times w}$ by Softmax layer, which is used to supervise with $\hat{y}_{edge}$ by binary cross-entropy loss L_{BAM}.

2.2.2. Boundary enhancement module

The mis-segmented region can be regarded as the non-overlapping part of the coarse segmentation result *g* and the Ground True *GT*. Therefore, we can intuitively represent the mis-segmented region label as $\hat{y}_{roi} = \overline{g \cap GT}$. After obtaining F_{edge} from BAM, the ultimate target of BEM is to predict the potential mis-segmentation region $y_{roi} \in \mathbb{R}^{1\times h\times w}$ under $\hat{y}_{roi} \in \mathbb{R}^{1\times h\times w}$ supervision ($y_{roi} \in (0,1)$, pixels with a value of 1 are treated as potentially misclassified pixels in *g*).

The architecture and dimensional details of the BEM are shown in Fig. 2(b). Given the input features F_{cnn} and F_{edge}, we follow [25] to embed a pair of 1 × 3 and 3 × 1 convolutions in BEM to further enhance the quality of F_{edge}. The complementary convolution can better extract the curvilinear structure information [26], which assists PRENet in generating y_{roi} more accurately. To be specific, F_{cnn} and F_{edge} are separately fed into the same set of 1 × 3 and 3 × 1 convolutions, then we employ GAP and Sigmoid to calculate the channel attention map of two F_{edge} branches and dot product with the corresponding F_{cnn} branches. After that, we perform element-wise addition to the outputs of complementary convolution and produce the boundary-enhanced feature $F_{en} \in \mathbb{R}^{c\times h\times w}$ by residual connection. To accurately locate potential mis-segmented region in *g*, we utilize the sequences of 1 × 1 convolution, BN, and ReLU to halve the channel of F_{en}, which is then adjusted to a single channel and outputs the final result y_{roi} by 1 × 1 convolution and Softmax layer. Same as BAM, we also adopt binary cross-entropy loss to supervise y_{roi} and $\hat{y}_{roi}$, named as L_{BEM}. The total loss function of PRENet can be formulated as follow, where α is a weight coefficient and will be discussed in Section 3.5.1:

$$L_{PRENet} = (1-\alpha)L_{BAM}\left(y_{edge}, \hat{y}_{edge}\right) + \alpha L_{BEM}\left(y_{roi}, \hat{y}_{roi}\right) \tag{1}$$

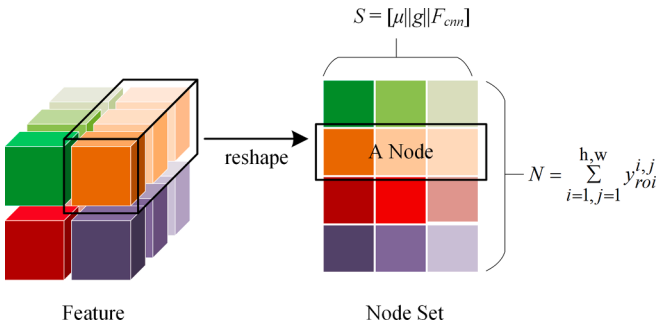

Fig. 3. Graph node construction: color pieces from deep to shallow are μ, *g*, and F_{cnn}, respectively, and || represents channel concatenation operation.

2.3. Graph construction module

2.3.1. Nodes of graph

The graph for GCN can be represented as $G = (\boldsymbol{V}, \boldsymbol{A})$, where $\boldsymbol{V}$ denotes the node set that define the number of nodes and the node features, and $\boldsymbol{A}$ denotes the adjacency matrix that defines the edge connections between nodes and the edge weights. Graph construction is a process of modeling the CNN outputs into $\boldsymbol{V}$ and $\boldsymbol{A}$. An appropriate construction method can boost the refinement performance of DR-GCN. As shown in Fig. 3, GCM constructs each pixel with a value of 1 in y_{roi} as a node individually and constructs their node features with pixel intensity μ, CNN-extracted features F_{cnn}, and coarse segmentation results *g* by channel concatenation and reshaping operations. Finally, $\boldsymbol{V}$ can be formulated as:

$$\boldsymbol{V} = [\boldsymbol{n}_1, \boldsymbol{n}_2, \cdots, \boldsymbol{n}_N]^{\mathrm{T}}, \boldsymbol{V} \in \mathbb{R}^{N\times S} \tag{2}$$

where $n_i \in \mathbb{R}^{S\times 1}$ denotes the *i*-th node, *S* is the dimension of node feature, and $N = \sum_{i=1,j=1}^{h,w} y_{roi}^{ij}$ is the total number of pixels with a value of 1 in y_{roi}.

As a supplementary note, we build a node-pixel hash map, which records the coordinate mapping of each node in $\boldsymbol{V}$ to the potentially misclassified pixels in *g*. This mapping is mainly used for recovering the classified nodes to their corresponding pixels, called Recovery Operation, as described in Section 2.4.3. Details of the map construction can be found in [9].

2.3.2. Adjacency matrix of graph

Establishing the appropriate edge connectivity between nodes can assist DR-GCN in aggregating critical information selectively during message propagation. We utilize pixel coordinates to calculate the Euclidean distance $D_{i,j}$ between nodes ($\boldsymbol{n}_i$,$\boldsymbol{n}_j$) in turn and connect the *l* nearest neighboring nodes to the central node $\boldsymbol{n}_i$. It can enhance the semantic connection between similar pixels. The value of *l* will be discussed in Section 3.5.2.

In addition, intra-class objects tend to have higher semantic similarity to each other [27]. Therefore, to establish closer associations between potentially inter-class nodes and enlarge the inter-class ones, we calculate the similarity between each pair of nodes as edge weights $\varepsilon_{i,j}$ by three parts: Euclidean distance, node features, and pixel intensity:

$$\varepsilon_{i,j} = \frac{1}{D_{i,j}} + D_{KL}\left(\boldsymbol{n}_i, \boldsymbol{n}_j\right) + \exp\left(-||\mu_i - \mu_j||_2\right) \tag{3}$$

where $D_{KL}(\boldsymbol{n}_i, \boldsymbol{n}_j) = \sum_k^S (\boldsymbol{n}_i)_k \cdot \log\frac{(\boldsymbol{n}_i)_k}{(\boldsymbol{n}_j)_k}$ denotes the KL scatter between node features, and $\exp\left(-\left|\left|\mu_i - \mu_j\right|\right|_2\right)$ denotes the variation of the pixel intensity. Finally, $\boldsymbol{A} = [a_{i,j}]_{N\times N}$ can be formulated as:

$$a_{i,j} = \begin{cases} \varepsilon_{i,j}, & l \text{ closest nodes} \\ 0, & \text{others} \end{cases} \tag{4}$$

Table 1
The architecture of Res-GCN.

Type	ChebConv	Dropout	ChebConv	Dropout	ChebConv	Softmax
Filter Size	(S, 64)	—	(64, 64)	—	(64, 1)	—
Output Size	$N \times 64$	$N \times 64$	$N \times 64$	$N \times 64$	$N \times 1$	$N \times 1$

2.4. Dynamic residual graph convolutional network

DR-GCN consists of Dynamic Graph Learning module and Residual Graph Convolutional Network (Res-GCN). The graph convolution operation of GCN is based on the spectral domain, which can propagate information efficiently according to the design of $\boldsymbol{A}$ and update the central node by weighting the sum of all its neighboring node features [28]. Therefore, GCN can explicitly explore the spatial structure information from graph structure and extract the semantic features from non-Euclidean space.

The normalized Laplacian matrix $\boldsymbol{L}$ of G is defined as $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$, $\boldsymbol{I}$ is an identity matrix, and $\boldsymbol{D}$ is the degree matrix of $\boldsymbol{A}$. $\boldsymbol{L}$ is a symmetric positive semi-definite matrix with a set of eigenvectors $\boldsymbol{U}$, which can be further diagonalized as $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathrm{T}}$. We can project the input $\boldsymbol{x} \in R^{N\times S}$ to the spectral domain by graph Fourier transform $\widehat{\boldsymbol{x}} = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{x}$. The graph convolution operation g_θ can be formulated as:

$$g_\theta \star \boldsymbol{x} = g(\boldsymbol{L})\boldsymbol{x} = \boldsymbol{U}g_\theta(\boldsymbol{\Lambda})\boldsymbol{U}^{\mathrm{T}}\boldsymbol{x} \tag{5}$$

Naive graph convolution needs to compute the eigenvectors of $\boldsymbol{L}$ frequently in Eq. (5). To mitigate the computational complexity of these operations, we utilize 1-th order of the Chebyshev polynomial approximation to replace the graph convolution filter [29], which can be formulated as $g_\theta(\boldsymbol{\Lambda}) \approx \sum_{k=0}^{K}\theta_k T_k(\boldsymbol{\Lambda})$, where T_k is the k-th order Chebyshev polynomial, and θ_k is the Chebyshev coefficient. In this way, the graph convolution filter is transformed into $g_\theta \star \boldsymbol{x} = \theta\left(\boldsymbol{I} + \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\right)\boldsymbol{x}$, and the layer-wise propagation rule of GCN is given by:

$$H^{(t+1)} = \widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}H^{(t)}\Theta^{(t)} \tag{6}$$

where $\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$, $\widetilde{\boldsymbol{D}}$ is the degree matrix of $\widetilde{\boldsymbol{A}}$, $\Theta^{(t)} \in \mathbb{R}^{S\times S'}$ represents the training parameters of the t-th layer, S' is the number of the graph convolution filters, $H^{(t+1)} \in R^{N\times S'}$ represents the output of the $(t+1)$-th layer.

2.4.1. Residual graph convolutional network

Residual connection structure can assist GCN to converge more effectively during training, which can further improve the classification performance of GCN [30]. As shown in Fig. 1(c), the architecture of Res-GCN consists of three graph convolution units. The first two units consist of a Chebyshev graph convolution layer and a Dropout layer with the residual connection between them. The last unit consists of a Chebyshev graph convolution layer and a Softmax layer. The network details are shown in Table 1, we formulate Res-GCN as follow:

$$H^{(t+1)} = \left(\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}H^{(t)}\Theta^{(t)}\right) + H^{(t)}, H^{(0)} = \boldsymbol{V} \tag{7}$$

where the size of $\boldsymbol{A}$ is fixed during training, and the channel dimension of $H^{(t+1)}$ is determined by S'.

As the layers deepen, the message propagation range of Res-GCN is also expanded. However, in contrast to CNN, which squeezes the size of features while downsampling, the node numbers of $H^{(t+1)}$ remain remains constant during Res-GCN training. It not only preserves spatial structure information but also sufficiently captures global contextual semantic information.

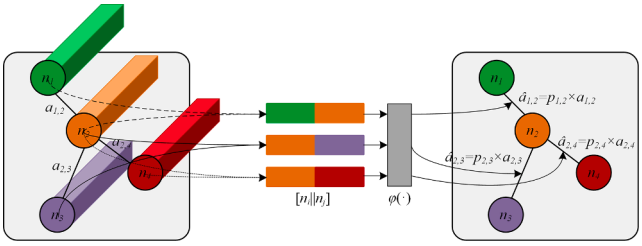

Fig. 4. The architecture of DGLM.

2.4.2. Dynamic graph learning module

The performance of GCN is highly correlated with the quality of the graph. However, the graph structures in Res-GCN are static, which means $\boldsymbol{A}$ is fixed during training. Static graphs cannot restrict noise and redundant information propagation, leaving critical features vulnerable to misinformation [31]. This flaw weakens the classification performance of Res-GCN.

Several studies [32,33] prove that dynamically learning the graph structure can alleviate the interference from noise and redundant nodes. Inspired by them, we insert a DGLM before Res-GCN. As shown in Fig. 4, DGLM can dynamically learn the attention coefficient for each edge during training. Then the edge weights between nodes can be adjusted and model a more flexible and effective graph structure.

The inputs of DGLM are $\boldsymbol{V}$ and $\boldsymbol{A}$. It can predict the attention coefficient $p_{i,j}$ for each edge by learning a function F, which calculates the correlation between a pair of nodes. The attention coefficient between nodes $\boldsymbol{n}_i$ and $\boldsymbol{n}_j$ can be calculated as:

$$p_{i,j} = F\left(\boldsymbol{n}_i, \boldsymbol{n}_j\right) = \varphi([\boldsymbol{n}_i||\boldsymbol{n}_j]) \tag{8}$$

where $||$ represents channel concatenation operation, $\varphi(\cdot)$ is a linear function used to learn $p_{i,j}$ for each edge dynamically. We optimize the training parameters of F by minimizing the following regularization:

$$L_D = \sum_{i,j=1}^{l}||\boldsymbol{n}_i - \boldsymbol{n}_j||_2^2 p_{i,j} \tag{9}$$

After calculating the attention coefficients for each pair of nodes, we multiply the attention coefficient with its original edge weight to obtain the new edge weight $\hat{a}_{i,j}$, which can be represented as $p_{i,j} \times a_{i,j}$. Finally, the edges from the same node are normalized by the Softmax function to ensure that the sum of the weight values equals 1. The final output of DGLM can be formulated as $\widehat{A} = [\widehat{a}_{i,j}]_{N\times N}$:

$$\widehat{a}_{i,j} = \frac{a_{i,j}\exp\left(F\left(n_i, n_j\right)\right)}{\sum_{j=1}^{l} a_{i,j}\exp\left(F\left(n_i, n_j\right)\right)} \tag{10}$$

DGLM can dynamically adjust the edge weights before training Res-GCN, which can guide the graph convolution layers to focus on the most relevant nodes in the neighborhood, resulting in a more accurate classification output.

2.4.3. DR-GCN loss

The loss function of DR-GCN $L_{DR\text{-}GCN}$ consists of two parts, Res-GCN loss L_G and DGLM loss L_D:

$$L_{DR-GCN} = L_G(H(x), GT) + \beta L_D \tag{11}$$

where β is a hyperparameter, which is used to control the order of magnitude of L_D (we set it to 1e-5). L_G is the binary cross-entropy loss:

$$L_G = -\sum_{i=1}^{N}\left(GT_i \log\left(H(x)_i\right) + (1 - GT_i)\log\left(1 - H(x)_i\right)\right) \tag{12}$$

where $H(x)_i$ represents the prediction of DR-GCN for the i-th node, and $GT_i \in (0, 1)$ represents the label of the i-th node. After obtaining the DR-GCN prediction results, we sequentially search the pixel in g according to the coordinate recorded in the node-pixel hash map. The values of the located pixel are replaced by the node prediction to output the final refined segmentation result. We refer to this process as the Recovery Operation. Finally, the overall procedure of our framework is summarized in Algorithm 1.

Input: Dataset $\boldsymbol{D}=\{(X^{(n)}, GT^{(n)})\}_{n=1}^{N}$, Hyperparameters α, l

Output: Refined segmentation result $\boldsymbol{Y}=\{Y^{(n)}\}_{n=1}^{N}$

Initialize mode weights W_{CNN}, W_{PRENet}, $W_{DR\text{-}GCN}$
// Train for Basic Network
while *epoch* < 80 **do**
    Sample $X^{(n)}$, $GT^{(n)}$ from $\boldsymbol{D}$
    $g^{(n)}$, $F_{CNN}^{(n)} \leftarrow \mathrm{CNN}(X^{(n)}, W_{CNN})$
    $L_{CNN} \leftarrow \mathrm{BCELoss}(g^{(n)}, GT^{(n)})$
    Back-propagate L_{CNN} to update W_{CNN}
end while
return $g=\{g^{(n)}\}_{n=1}^{N}$, $F_{CNN}=\{F_{CNN}^{(n)}\}_{n=1}^{N}$
// Train for Our Framework
while *epoch* < 100 **do**
    // Locate mis-segmentation region by PRENet
    Sample $g^{(n)}$, $F_{CNN}^{(n)}$ from g, F_{CNN}
    Create boundary label $\hat{y}_{edge}^{(n)}$ from $GT^{(n)}$ by Sobel Operator
    Create mis-segmentation label by $\hat{y}_{roi}^{(n)} = \overline{g^{(n)} \bigcap \boldsymbol{GT^{(n)}}}$
    $y_{edge}^{(n)}$, $y_{roi}^{(n)} \leftarrow \mathrm{PRENet}(F_{CNN}^{(n)}, W_{PRENet})$
    $L_{PRENet} \leftarrow \{L_{BAM}(y_{edge}^{(n)}, \hat{y}_{edge}^{(n)}), L_{BEM}(y_{roi}^{(n)}, \hat{y}_{roi}^{(n)})\}$ with α as in Eq. (1)
    // Construct graph by GCM
    Create $\boldsymbol{V}^{(n)}$ from $y_{roi}^{(n)}$ as in Eq. (2)
    Create $\boldsymbol{A}^{(n)}$ of $\boldsymbol{V}^{(n)}$ with l as in Eq. (4)
    // Refinement by DR-GCN
    $\hat{\boldsymbol{A}}^{(n)} \leftarrow \mathrm{DGLM}(\boldsymbol{V}^{(n)}, \boldsymbol{A}^{(n)})$ as in Eq. (10)
    Calculate regularization L_D as in Eq. (9)
    $H^{(n)} \leftarrow \text{Res-GCN}\{(\boldsymbol{V}^{(n)}, \hat{\boldsymbol{A}}^{(n)}), W_{DR\text{-}GCN}\}$ as in Eq. (7)
    $L_{DR\text{-}GCN} \leftarrow \{L_G(H^{(n)}, GT^{(n)}), L_D\}$ as in Eq. (11)
    Back-propagate L_{PRENet}, $L_{DR\text{-}GCN}$ to update W_{PRENet}, $W_{DR\text{-}GCN}$
    $Y^{(n)} \leftarrow (H^{(n)}, g^{(n)})$ by Recovery Operation
end while
return $\boldsymbol{Y}=\{Y^{(n)}\}_{n=1}^{N}$

Table 2
Evaluation metrics.

Dice	IoU	Se	HD
$\frac{2 \times TP}{2 \times TP + FP + FN}$	$\frac{TP}{TP + FN + FP}$	$\frac{TP}{TP + FN}$	$\max\left(\max\left\{\min_{y \in Y}\|y - gt\|\right\}, \max_{gt \in GT}\left\{\min_{y \in Y}\|gt - y\|\right\}_{y \in Y}\right)$

3. Experiments and results

3.1. Datasets

To validate that our framework can adapt well to different domains of medical segmentation tasks, we pick three public datasets for testing: the CT dataset COVID-19 CT segmentation dataset (COVID)[1], the microscopy dataset 2018 Data Science Bowl (DSB) [34], and the US dataset Breast Ultrasound Image Dataset (BUSI) [35]. The COVID dataset contains 100 axial two-dimensional CT images of 60 COVID-19 patients, which is one of the most challenging datasets for COVID-19 segmentation with insufficient data and high variability among images. The DSB dataset consists of 670 cell nuclear microscopy images of different zoom magnifications, imaging methods, and cell types. The BUSI dataset collects from 600 female patients between 25 and 75 years old, and we pick 210 malignant breast ultrasound images from it for the experiment. As shown in the first row of Fig. 7, US images have low contrast between background and foreground due to the special imaging characteristics. Moreover, the foreground is easily disturbed by the background with inhomogeneous texture distribution, exacerbating the blurring in the boundary regions. Therefore, it is challenging for the network to segment malignant tumors from US images accurately.

We split each dataset in the ratio of 8:2 for training and testing, and randomly selected 10% from the training set for validation. The input images are uniformly reshaped to 256 × 256 and applied data augmentation by random rotating. Meanwhile, we convert the DSB dataset to gray, followed by adaptive histogram equalization and adaptive gamma correction operations.

3.2. Parameter settings and implement details

We perform all experiments on Windows 10 operating system with the Intel Xeon E-2186 M CPU, 64G RAM, and the NVIDIA Quadro P5200 GPU. The CNN part is implemented by the PyTorch, and the GCN part is implemented by the Pytorch Geometric.

Our framework uses Adam optimizer with the cosine annealing strategy for parameter optimization, and the initial learning rate is set to 0.01. The dropout ratio is set to 0.5 and the batch size is 1. We train our framework for a total of 100 epochs. α is set to 0.6, and l is set to 16. Meanwhile, to fairly compare the performance of our framework under different basic networks, the training epoch of each basic network is uniformly set to 80, the batch size is 2, and the learning rate is 0.0005. The loss function is uniformly chosen as Dice loss, and the optimizer is Adam with the cosine annealing strategy.

3.3. Evaluation metrics

We adopt Dice, IoU, Sensitivity (Se) and Hausdorff Distance (HD) to evaluate the refinement performance of our framework. HD is sensitive to boundary. A better performance has smaller value of HD and larger of Dice, IoU, and Se. The formulate for each evaluation metric is shown in Table 2, where Y represents the prediction result, and GT represents the Ground True. TP, TN, FP, and FN represent the number of true positive, true negative, false positive, and false negative pixels in Y, respectively.

[1] https://medicalsegmentation.com/covid19/.

Table 3
Performance comparison in the COIVD dataset of our framework.

Dataset	Network	Dice	IoU	Se	HD
COVID	UNet (MICCAI15)	0.7423	0.5901	0.6562	30.55
	UNet + Ours	0.7664	0.6213	0.7283	27.09
	AttUNet (arXiv18)	0.7167	0.5585	0.6110	33.06
	AttUNet + Ours	0.7469	0.5960	0.6953	32.45
	CENet (TMI19)	0.7119	0.5527	0.6259	25.38
	CENet + Ours	0.7325	0.5779	0.6793	22.41
	UNet++ (TMI19)	0.7459	0.5948	0.6609	31.89
	UNet++ + Ours	0.7675	0.6227	0.7118	29.83
	UNet3+ (ICASSP20)	0.7643	0.6185	0.6923	27.00
	UNet3+ + Ours	0.7719	0.6286	0.7228	24.02
	TransUNet (arXiv21)	0.7662	0.6210	0.6961	22.65
	TransUNet + Ours	0.7720	0.6286	0.7213	22.55
	UNeXt (MICCAI22)	0.7562	0.6080	0.6862	34.57
	UNeXt + Ours	0.7666	0.6215	0.7255	32.39
	UCTransNet (AAAI22)	0.7445	0.5929	0.7193	**22.29**
	UCTransNet + Ours	0.7540	0.6051	0.7370	34.96
	YNet (MICCAI22)	0.7502	0.6003	0.6954	29.03
	YNet + Ours	**0.7819**	**0.6419**	**0.7793**	26.78
	Avg (↗)	1.79%	2.30%	5.08%	0.44
	Max (↗)	3.17%	4.16%	8.43%	3.46

3.4. Overall performance

Our framework is a plug-and-play enhancement block. To demonstrate that the framework can effectively boost the segmentation performance under different basic networks with a good generalization, we select nine representative segmentation networks based on different improve directions for comparison: (1) UNet [4] is a classic and seminal encoder-decoder network; (2) AttUNet [36] is a typical U-shaped network based on the attention mechanism designed; (3) CENet [37] incorporates the multi-kernel context extraction module, which is similar to the Inception structure; (4) UNet++ [5] is a representative

Table 4
Performance comparison in the DSB dataset of our framework.

Dataset	Network	Dice	IoU	Se	HD
DSB	UNet (MICCAI15)	0.8160	0.6892	0.7170	10.86
	UNet + Ours	0.8986	0.8158	0.8713	12.17
	AttUNet (arXiv18)	0.8601	0.7545	0.7773	10.44
	AttUNet + Ours	0.8719	0.7729	0.8028	10.44
	CENet (TMI19)	0.8847	0.7933	0.8449	**10.10**
	CENet + Ours	0.9002	0.8185	0.8730	**10.10**
	UNet++ (TMI19)	0.8805	0.7865	0.8114	11.09
	UNet++ + Ours	0.8878	0.7983	0.8299	10.44
	UNet3+ (ICASSP20)	0.9046	0.8258	0.8628	11.09
	UNet3+ + Ours	**0.9112**	**0.8368**	**0.8830**	11.18
	TransUNet (arXiv21)	0.8455	0.7324	0.7529	10.44
	TransUNet + Ours	0.8756	0.7787	0.8230	12.16
	UNeXt (MICCAI22)	0.8717	0.7726	0.8111	11.49
	UNeXt + Ours	0.8810	0.7873	0.8397	12.17
	UCTransNet (AAAI22)	0.8867	0.7965	0.8396	12.17
	UCTransNet + Ours	0.8912	0.8038	0.8534	12.17
	YNet (MICCAI22)	0.8105	0.6813	0.6973	11.22
	YNet + Ours	0.8491	0.7378	0.7685	11.22
	Avg (↗)	2.29%	3.53%	4.78%	−0.35
	Max (↗)	8.26%	12.66%	15.42%	0.65

Table 5
Performance comparison in the BUSI dataset of our framework.

Dataset	Network	Dice	IoU	Se	HD
BUSI	UNet (MICCAI15)	0.6161	0.4452	0.5117	39.62
	UNet + Ours	0.6692	0.5028	0.6230	**34.34**
	AttUNet (arXiv18)	0.5621	0.3909	0.4294	51.73
	AttUNet + Ours	0.6230	0.4525	0.5417	41.11
	CENet (TMI19)	0.6377	0.4681	0.5161	41.19
	CENet + Ours	0.6472	0.4784	0.5468	37.95
	UNet++ (TMI19)	0.5989	0.4274	0.4786	50.58
	UNet++ + Ours	0.6168	0.4459	0.5291	40.27
	UNet3+ (ICASSP20)	0.5977	0.4263	0.4727	42.44
	UNet3+ + Ours	0.6001	0.4287	0.4772	37.34
	TransUNet (arXiv21)	0.6428	0.4736	0.5383	59.36
	TransUNet + Ours	**0.6745**	**0.5089**	0.6210	44.46
	UNeXt (MICCAI22)	0.6538	0.4857	0.5330	56.38
	UNeXt + Ours	0.6668	0.5001	0.5536	52.10
	UCTransNet (AAAI22)	0.6247	0.4542	0.5060	64.69
	UCTransNet + Ours	0.6282	0.4580	0.5344	63.41
	YNet (MICCAI22)	0.6627	0.4956	0.5906	42.19
	YNet + Ours	0.6721	0.5061	**0.6274**	43.60
	Avg (↗)	2.24%	2.39%	5.31%	5.96
	Max (↗)	6.09%	6.16%	11.23%	14.90

work of dense skip connection; (5) UNet3+ [6] further mining full-scale features based on UNet++; (6) TransUNet [13] is a pioneer in integrating the transformer into UNet, enhancing the capacity for contextual semantic information capture; (7) UNetXt [7] uses the Shifted MLP module to replace the convolution layer at the bottom of UNet, achieving better performance and lightweight; (8)UCTransNet [15] redesigns the UNet skip connection based on Transformer, which narrows the semantic gap between encoder and decoder; (9)YNet [8] enhances the learning capability by exploiting cross-domain features.

The experimental results for each network in the COVID, DSB, and BUSI datasets are shown in Table 3, Table 4, and Table 5, where the best results in the same basic network are underlined, the best results in all are bolded, and (↗) represents the growth rate. The segmentation refinement results of each dataset are shown in Fig. 5, Fig. 6, and Fig. 7, where the first row to the third row represent the original image, *GT* image, and the segmentation comparison image, respectively. The white part in the third row represents the coarse segmentation result, and the green part represents the refinement result.

3.4.1. Results in the COVID dataset

In Table 3, we can find that our framework has a significant gain on the Dice, IoU, and Se over each basic network in the COVID dataset, with an average improvement of 1.79%, 2.30%, and 5.08%. HD is also gained except in UCTransNet, with an average improvement of 0.44. It is worth noting that our refinement result is particularly effective when the segmentation capability of the basic network is weak, *e.g.*, (**i**) UNet is hard to extract sufficient spatial structure information to accurately segment the boundary shapes when the lesion regions are scattered or fragmented, which hurts the performance of HD. Thanks to the unique spatial structure of the graph, DR-GCN can effectively refine the boundary segmentation results of UNet, improving HD and Se by 3.46 and 7.21%, which are comparable to the best basic network TransUNet. (**ii**) AttUNet has large development potential for Dice, IoU, and Se since its design is not focused on acquiring contextual semantic information. On this basis, DR-GCN achieves the best improvement by 3.02%, 3.75%, and 8.43%, respectively. As shown in Fig. 5(b), our framework can effectively segment more lesion regions that AttUNet ignores, which enhances the semantic association between distant lesion regions.

TransUNet and UNet3 + achieve the best and second-best performance among the basic network, which alleviates the limitation of the local receptive field and provides stronger segmentation capabilities through the transformer and dense connectivity mechanisms. Nevertheless, our framework can further boost them by efficiently extracting global contextual information, improving IoU by 3.17% and 4.16%, respectively. Moreover, our framework brings the highest improvements based on YNet with 3.17% Dice and 4.16% IoU while achieving the best COVID dataset performance with 78.19% Dice, 64.19% IoU, and 77.93% Se, respectively.

3.4.2. Results in the DSB dataset

To better illustrate the good generalization of our framework, the next experiment is performed in the microscopy dataset DSB. We can find from Table 4 that our framework also obtains significant gains based on each network, with an average improvement of 2.29%, 3.53%, and 4.78% in Dice, IoU, and Se, respectively. Moreover, our framework

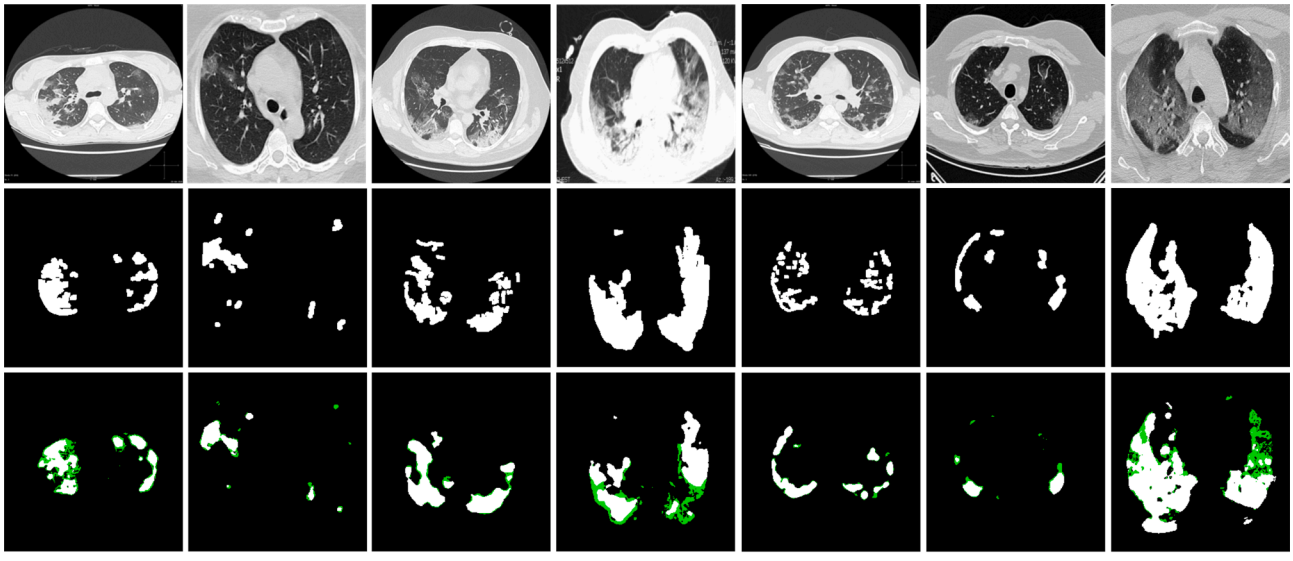

Fig. 5. COVID dataset segmentation results: (a) to (g) represent UNet, AttUNet, CENet, UNet++, TransUNet, UNeXt, and YNet, respectively.

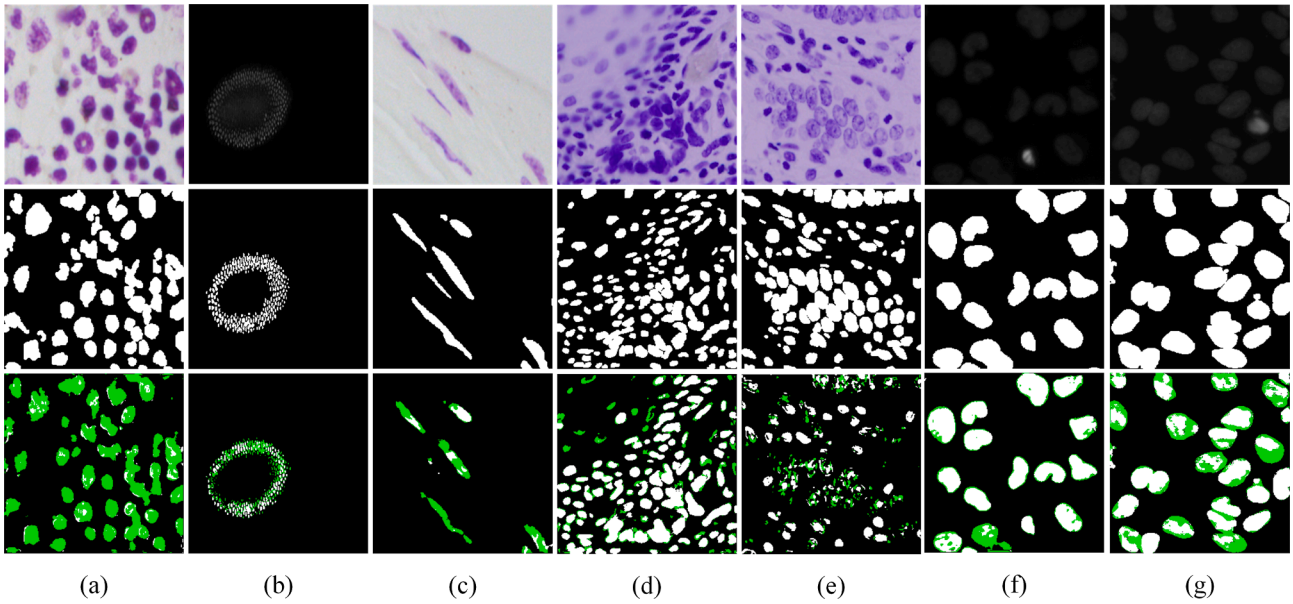

Fig. 6. DSB dataset segmentation results: (a) to (g) represent UNet, AttUNet, CENet, UNet++, TransUNet, UNeXt, and YNet, respectively.

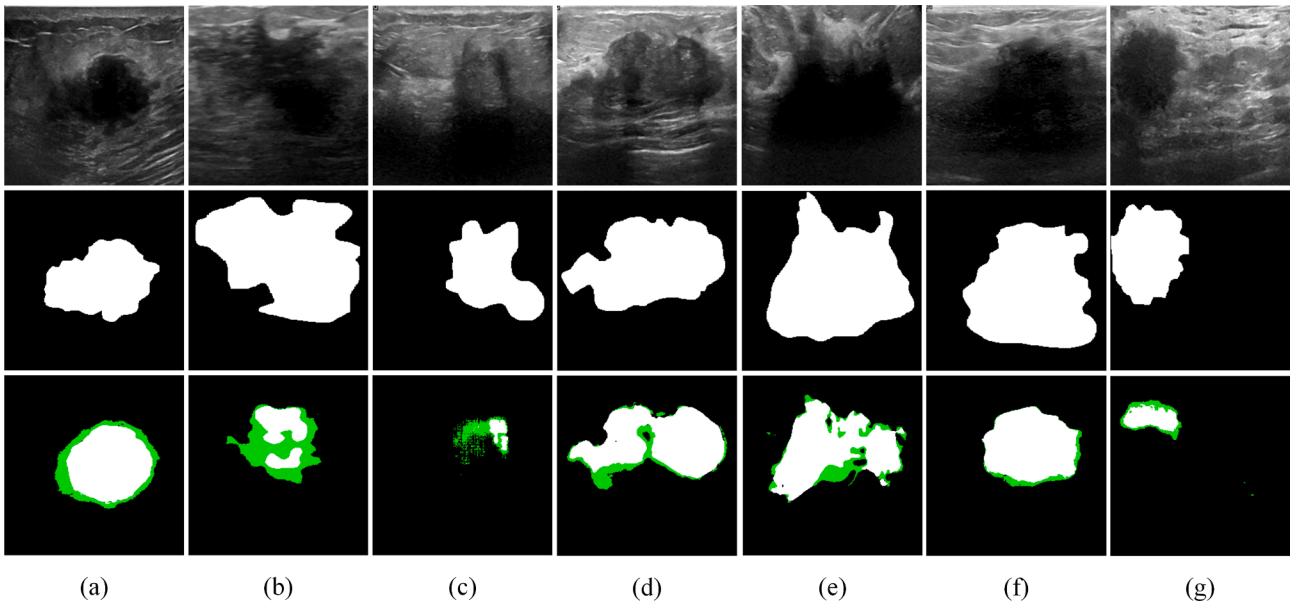

Fig. 7. BUSI dataset segmentation results: (a) to (g) represent UNet, AttUNet, CENet, UNet++, TransUNet, UNeXt, and YNet, respectively.

reaches the best progress on the basics of UNet, which achieved 8.25%, 12.66%, and 15.42%, respectively. We can observe its significant refinement results in Fig. 6(a). Similarly, our framework still improves the strongest network UNet3 + with 0.66%, 1.10%, and 2.02% in Dice, IoU, and Se, achieving the best results of 91.12%, 83.68%, and 88.30%, respectively. However, the HD of our framework is inferior in the DSB dataset, with a tiny improvement of 0.65 under UNet++ but an average decrease of 0.35 compared to each basic network. We give two following possible reasons after comparing the original images and the refinement ones: (**i**) Our framework has less potential to optimize the boundary shape since the shape of the cell nucleus is relatively regular; (**ii**) As shown in Fig. 6(e), although our framework can refine more under-segmented regions, the structure of some refinement regions may be fragmented, which instead exacerbates the boundary irregularities. It shows up as Dice, IoU, and Se getting boost, but HD could be worse. But overall, DR-GCN can not only fill the lacking spatial structure and contextual semantic information of the basic network but also owns better global modeling capability to strengthen the association between isolated cell nuclei. As shown in Fig. 6(a) and (b), DR-GCN can segment more individuals ignored by the basic network. Meanwhile, DR-GCN is more sensitive in the nuclei boundaries, which can better fulfill some broken nuclei, as shown in Fig. 6(c) and (g).

3.4.3. Results in the BUSI dataset

To further validate the refinement performance of our framework in boundary regions, we finally choose the BUSI dataset for experiments. In

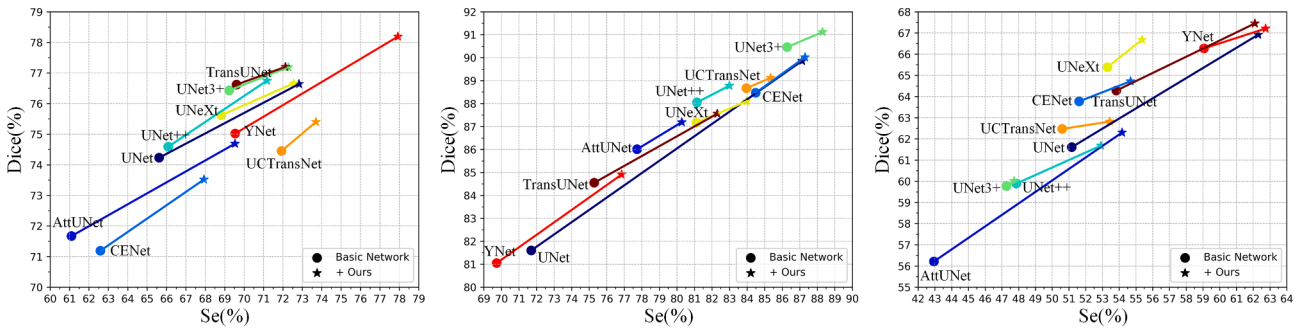

Fig. 8. The enhancement of our framework under different basic networks.

Table 6
Comparison of metrics with different weight coefficients.

Dataset	α	Dice	IoU	Se	HD
COVID	0.5	0.7644	0.6187	0.7253	27.09
	0.6	**0.7664**	**0.6213**	**0.7283**	27.09
	0.7	0.7662	0.6210	0.7276	**26.50**
	0.8	0.7638	0.6178	0.7166	27.82
DSB	0.5	0.8911	0.8035	0.8575	11.87
	0.6	0.8986	0.8158	0.8713	12.17
	0.7	**0.9001**	**0.8183**	**0.8729**	12.17
	0.8	0.8963	0.8122	0.8729	12.17
BUSI	0.5	0.6241	0.4536	0.5734	33.63
	0.6	**0.6692**	**0.5028**	**0.6230**	34.34
	0.7	0.6266	0.4563	0.5730	**32.09**
	0.8	0.6244	0.4539	0.5454	32.09

Table 5, the performance of each basic network is worse than the previous two datasets, with a maximum of only 66.27% Dice, 49.56% IoU, 59.06% Se, and 39.62 HD. It suggests that missing boundary information severely plagues the performance of basic networks when segmenting complex images. However, our framework still brings significant gains at this time, with the Dice, IoU, and Se improving by 2.24%, 2.39%, and 5.31% on average.

Furthermore, as shown in Fig. 7, we can observe that the refinement regions are mainly concentrated in the malignant tumor boundaries, which is also reflected in the HD. Our framework achieves the best HD progress among the three datasets, with an average improvement of 5.97, and the highest improvement of 14.90 based on TransUNet. For the best HD performance network UNet, our framework still improves by 5.20. It proves that DR-GCN can better address challenging tasks by exploiting sufficient spatial information from graph structure. Similar to the previous experiments, our framework achieves the best progress on the weaker network AttUNet, with improvements of 6.09%, 6.16%, and 11.23% in Dice, IoU, and Se, which also improves TransUNet, the best-performing network for Dice and IoU, reaching 67.45% and 50.89%, respectively.

In the end, to clearly show the superiority, we visualize the enhancement of our framework in three datasets. As shown in Fig. 8, the improvement trend of our framework is relatively simultaneous and better than each basic network, while Se is the most significant. Our framework performs well and displays strong generalizability in each dataset with various imaging characteristics. Furthermore, our framework owns the potential to handle complex tasks better, which can achieve higher gains in weak basic networks or c challenging datasets. However, we also find that the improvement trend of our framework is slowing down for powerful basic networks, which indicates that the refinement capability of DR-GCN still has room for further development.

3.5. Hyperparameter analysis

3.5.1. Weight coefficients

To analyze the effect of weight coefficients α in L_{PRENet}, we choose

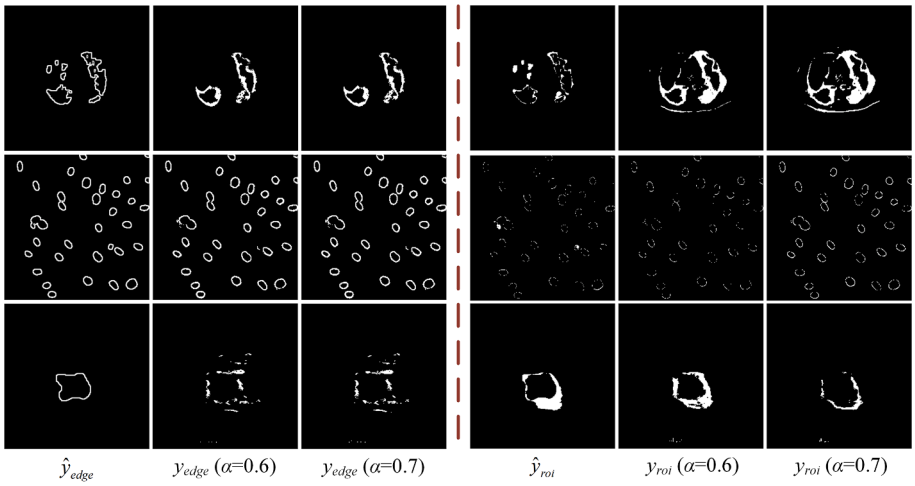

Fig. 9. Outputs comparison of PRENet under different weight coefficients: the first row represents the COVID dataset, the second row represents the DSB dataset, and the third row represents the BUSI dataset.

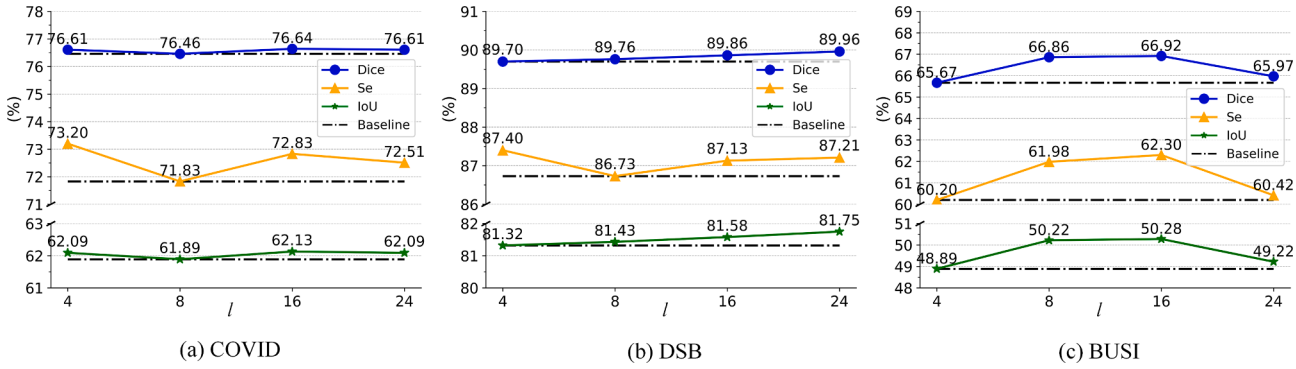

Fig. 10. Comparison of indicators with different number of edges: Baseline represents the minimum value in each group of indicators.

UNet as the basic network and compare the performance for four groups of coefficients under each dataset respectively. The experimental result is shown in Table 6, and the best results in the same dataset are bolded. In the COVID dataset, the Dice, IoU, and Se achieve the best performance when α is set to 0.6, and the variation between different coefficients is slight. In the BUSI dataset, α also achieves the best performance at 0.6, which is significantly better than the others except for the HD, with an average improvement of about 5%. However, in the DSB dataset, α gains the best performance at another weight 0.7.

To further analyze this reason, we visualize the outputs of PRENet, as shown in Fig. 9, where the left and right halves are the BAM prediction y_{edge} and the BEM prediction y_{roi} with their corresponding labels, respectively. In the second row of Fig. 9, we can find that BAM achieves accurate boundary prediction results y_{edge} both at 0.6 or 0.7 in the DSB dataset. However, the results under 0.7 can predict mis-segmentation regions more completely than 0.6. It indicates that the segmentation pressure of BAM is low when the boundary shape is homogeneous and regular, and appropriately increasing coefficients can give BEM higher supervision, promoting it to focus on more potentially mis-segmented regions as much as possible. Thus, DR-GCN can fix more potential misclassified pixels to obtain better refinement. However, in complex datasets such as COVID and BUSI, the boundary prediction results of BAM are more ambiguous, which means the quality of its extracted F_{edge} is relatively poorer. Thus, we should give BAM higher supervision to learn more boundary information, and the experimental results show that setting α to 0.6 can achieve a better balance between BAM and BEM and improve overall performance.

In conclusion, α set to 0.6 is a more suitable choice for the challenging datasets, and a higher α is better for datasets with simple boundary shape.

3.5.2. Number of edges

The second hyperparameter in this paper is the number of edges l. In order to compare the effects on DR-CCN with various l and ascertain its optimal parameter settings, we choose UNet as the basic network and set up four groups l to experiment under three datasets, which is shown in Fig. 10. Note that we omit the curves of HD since the experiment shows that different l has almost no effect on HD in each dataset, which is 27.09, 12.17, and 33.34, respectively.

Fig. 10 shows that the variation of Dice and IoU tends to be consistent across datasets. In the COVID and BUSI datasets, Dice and IoU perform best when l is set to 16. We can notice that for both challenging datasets, higher l fails to bring better performance. The reason is that GCN updates node features by aggregating neighborhood information, and excessive edges establish the connection between the central node and low-similarity nodes. It will weaken the quality of the updated node features since attracting more potential interference information. In contrast, for the relatively easy-to-segment DSB dataset, Dice and IoU perform best at $l = 24$. There are mainly two reasons as follows: (**i**) The interference caused by more edges is milder since the DSB images are more homogeneous. (**ii**) The basic network can extract more accurate features in simple datasets, indirectly improving the initial graph quality during construction. It will help to gathering more valuable information based on more edges and improve the final performance. In addition, we find that Se behaves differently than the above indicators, which performs best in the BUSI dataset at $l = 16$ while $l = 4$ for the COVID and DSB datasets. It demonstrates that for datasets with more segmented regions, such as COVID and DSB, a lower l setting is helpful for Se enhancement.

It is worth to note the literature [9] indicates that the number of edges is one of the critical factors in controlling the cost of graph construction. Furthermore, it can be found in Fig. 10(a) and (b) that the gap for Dice and IoU is minor between $l = 16$ and $l = 24$, while the Se is also closer. As shown above, setting l to 16 is a more suitable choice, which can not only promote each node to aggregate richer global context information that gains better performance of DR-GCN but also balance the growth in computational complexity.

Table 7
Ablation experiment results.

Dataset	Network	Dice	IoU	Se	HD
COVID	Res-GCN	0.7636	0.6176	0.7194	27.46
	DR-GCN	**0.7664**	**0.6213**	**0.7283**	**27.09**
DSB	Res-GCN	0.8963	0.8122	0.8687	**11.87**
	DR-GCN	**0.8986**	**0.8158**	**0.8713**	12.17
BUSI	Res-GCN	0.6585	0.4909	0.6052	**34.34**
	DR-GCN	**0.6692**	**0.5028**	**0.6230**	**34.34**

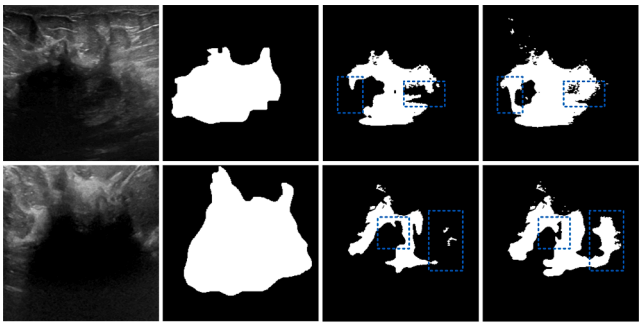

Fig. 11. Ablation experiment visualization comparison.

Table 8
Compared with the algorithm complexity of our framework in different basic networks.

Method	UNet	+ours	UNet++	+ours	TransUNet	+ours	UNeXt	+ours
Params(M)	34.53	34.67	9.16	9.31	209.52	209.66	1.47	1.62
Time(s)	0.34	0.96	0.29	0.87	0.39	0.90	0.25	0.79

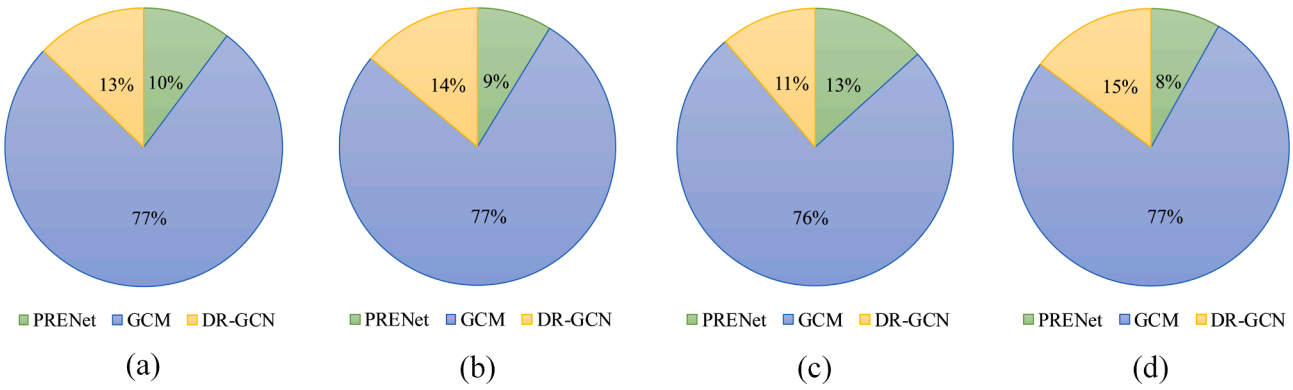

Fig. 12. Percentage of average inference time for each module under different basic networks: (a) to (d) represent UNet + Ours, UNet++ + Ours, TransUNet + Ours and UNeXt + Ours respectively.

3.6. Ablation analysis

To evaluate the impact of DGLM in our framework, we select UNet as the basic network and perform ablation experiments on three datasets respectively. The experimental results are shown in Table 7, and the best result in the same dataset is bolded.

Table 7 shows that Dice, IoU, and Se improve in all datasets with DGLM, and the more complex the dataset is, the better the improvement it achieves. The reason is that the quality of the features extracted by the network is poor when the task is complex. Nevertheless, DGLM can strengthen the association with similar semantic information and expand the distance for inter-class nodes by dynamically adjusting the edge weights of neighboring nodes during training. To highlight the effect of DGLM, we visualize the BUSI dataset with the best gain, as shown in Fig. 11. By observing the blue box in Fig. 11, we can find that DR-GCN can further segment the disconnected regions, which proves that the DGLM plays an active role in maintaining the integrity of the segmentation results.

3.7. Complexity analysis

We construct each pixel in the potentially mis-segmented regions as a single graph node. To verify the computational complexity of this strategy, we compare the parameters and average inference time of our framework in the COVID dataset, and the experimental results are shown in Table 8. It can be found that the average inference time of the proposed framework is about three times more than each basic network, but it is still in an acceptable range. It is worth noting that the parameters of our framework increase by less than 0.15 M, which prove its potential for practical applications.

To further analyze the reasons for the significant escalation of average inference time, we separately calculated the time percentage for each module under different basic networks. As shown in Fig. 12, PRENet and DR-GCN account for a low percentage and exhibit some fluctuation since the size of the mis-segmented regions suffers from the segmentation quality with different basis networks. Meanwhile, the largest fraction of the average inference time is GCM, which accounts for about 77%, indicating that the primary time cost of our framework comes from the graph construction process. Therefore, around this shortcoming, how to effectively optimize GCM will be crucial for improving the practicality of our framework.

4. Conclusions

To address the problem that traditional CNN is hard to effectively extract spatial structure and global contextual semantic information, resulting in poor segmentation in the boundary regions, we propose a medical image mis-segmentation region refinement framework based on dynamic graph convolution, which can efficiently boost the performance of the basic network. Our framework obtains potential mis-segmented pixels from the coarse segmentation results by PRENet. Then construct the appropriate graphs by GCM and feed them into DR-GCN to generate the final refinement results. We hope that the proposed plug-and-play refinement framework can effectively enhance mainstream medical image segmentation networks with small parameter costs, further satisfy physicians' requirements for accurate segmentation, and promote the development of graph convolutional segmentation methods in clinical applications.

However, our framework still exits some limitations: (**i**) DGLM tends to generate a few isolated noise pixels in some refinement results, as shown in the first row of Fig. 11. (**ii**) Our framework requires a long inference time, which is not suitable for real-time segmentation tasks. In future work, we will focus on improving GCM and DGLM to strengthen the graph construction efficiency while enhancing the anti-interference capability and further improving the generality of our framework.

CRediT authorship contribution statement

Haocheng Liang: Conceptualization, Methodology, Software, Writing – original draft. **Jia Lv:** Data curation, Resources, Supervision, Writing – review & editing. **Zeyu Wang:** Visualization, Investigation, Formal analysis, Validation. **Ximing Xu:** Funding acquisition, Resources.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: 'Jia Lv reports financial support was provided by Chongqing Municipal Education Commission (KJZD-K202200511). Jia Lv reports financial support was provided by Chongqing Science and Technology Bureau (2022TFII-OFX0044). Ximing Xu reports financial support was provided by Children's Hospital of Chongqing Medical University (NCRCCHD-2022-HP-01).'

Data availability

Data will be made available on request.

References

[1] R. Lalonde, Z. Xu, I. Irmakci, S. Jain, U. Bagci, Capsules for biomedical image segmentation, Med. Image Anal. 68 (2) (2021), 101889, https://doi.org/10.1016/j.media.2020.101889.

[2] S. Pang, A. Du, Z. Yu, M. Orgun, 2D medical image segmentation via learning multi-scale contextual dependencies, Methods 202 (6) (2022) 40–53, https://doi.org/10.1016/j.ymeth.2021.05.015.

[3] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440, doi: 10.1109/CVPR.2015.7298965.

[4] 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, 2015, pp. 234–241, doi: 10.1007/978-3-319-24574-4_28.

[5] Z. Zhou, M.M.R. Siddiquee, N. Tajbakhsh, et al., Unet++: redesigning skip connections to exploit multiscale features in image segmentation, IEEE Trans. Med. Imaging 39 (6) (2019) 1856–1867, https://doi.org/10.1109/TMI.2019.2959609.

[6] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, et al., Unet 3+: a full-scale connected unet for medical image segmentation, in: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing, 2020, pp. 1055–1059, doi: 10.1109/ICASSP40776.2020.9053405.

[7] J.M.J. Valanarasu, V.M. Patel, Unext: mlp-based rapid medical image segmentation network, in: Medical Image Computing and Computer Assisted Intervention, 2022, pp. 23–33, doi: 10.1007/978-3-031-16443-9_3.

[8] A. Farshad, Y. Yeganeh, P. Gehlbach, et al., Y-Net: a spatiospectral dual-encoder network for medical image segmentation, in: Medical Image Computing and Computer Assisted Intervention, 2022, pp. 582–592, doi: 10.1007/978-3-031-16434-7_56.

[9] N. Dhingra, G. Chogovadze, A. Kunz, Border-seggcn: improving semantic segmentation by refining the border outline using graph convolutional network, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 865–875, doi: 10.1109/ICCVW54120.2021.00102.

[10] C. Yu, J. Wang, C. Gao, G. Yu, C. Shen, N. Sang, Context prior for scene segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 12416–12425, doi: 10.1109/CVPR42600.2020.01243.

[11] D. Jha, M.A. Riegler, D. Johansen, P. Halvorsen, H.D. Johansen, Doubleu-net: a deep convolutional neural network for medical image segmentation, in: 2020 IEEE 33rd International Symposium on Computer-based Medical Systems, 2020, pp. 558–564, doi: 10.1109/CBMS49503.2020.00111.

[12] L. Yang, H. Wang, Q. Zeng, Y. Liu, G. Bian, A hybrid deep segmentation network for fundus vessels via deep-learning framework, Neurocomputing 448 (30) (2021) 168–178, https://doi.org/10.1016/j.neucom.2021.03.085.

[13] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, et al., Transunet: transformers make strong encoders for medical image segmentation, arXiv preprint arXiv:2102.04306, 2021, doi: 10.48550/arXiv.2102.04306.

[14] A. Lin, B. Chen, J. Xu, Z. Zhang, G. Lu, D. Zhang, Ds-transunet: dual swin transformer u-net for medical image segmentation, IEEE Trans. Instrum. Meas. 71 (1) (2022) 1–15, https://doi.org/10.1109/TIM.2022.3178991.

[15] H. Wang, P. Cao, J. Wang, et al., Uctransnet: rethinking the skip connections in u-net from a channel-wise perspective with transformer, Proc. AAAI Conf. Artif. Intell. 36 (3) (2022) 2441–2449, doi: 10.1609/aaai.v36i3.20144.

[16] S.Y. Shin, S. Lee, I.D. Yun, K.M. Lee, Deep vessel segmentation by learning graphical connectivity, Med. Image Anal. 58 (8) (2019), 101556, https://doi.org/10.1016/j.media.2019.101556.

[17] Y. Lu, Y. Chen, D. Zhao, B. Liu, Z. Lai, J. Chen, Cnn-g: convolutional neural network combined with graph for image segmentation with theoretical analysis, IEEE Trans. Cognitive Dev. Syst. 13 (3) (2020) 631–644, https://doi.org/10.1109/TCDS.2020.2998497.

[18] R.D. Soberanis-mukul, N. Navab, S. Albarqouni, Uncertainty-based graph convolutional networks for organ segmentation refinement, Med. Imaging Deep Learning (2020) 755–769.

[19] Z. Tian, X. Li, Y. Zheng, Z. Chen, Z. Shi, L. Liu, et al., Graph-convolutional-network-based interactive prostate segmentation in MR images, Med. Phys. 47 (9) (2020) 4164–4176, https://doi.org/10.1002/mp.14327.

[20] S. Liu, S. Liang, X. Huang, et al., Graph-enhanced u-net for semi-supervised segmentation of pancreas from abdomen ct scan, Phys. Med. Biol. 67 (15) (2022), 155017, https://doi.org/10.1088/1361-6560/ac80e4.

[21] R. Xu, Y. Li, C. Wang, et al., Instance segmentation of biological images using graph convolutional network, Eng. Appl. Artif. Intell. 110 (4) (2022), 104739, https://doi.org/10.1016/j.engappai.2022.104739.

[22] X. Chen, D. Qi, J. Shen, Boundary-aware network for fast and high-accuracy portrait segmentation, arXiv preprint arXiv:1901.03814, 2019, doi: 10.48550/arXiv.1901.03814.

[23] H. Hu, J. Cui, H. Zha, Boundary-aware graph convolution for semantic segmentation, in: 2020 25th International Conference on Pattern Recognition, 2021, pp. 1828–1835, doi: 10.1109/ICPR48806.2021.9412034.

[24] K. Wang, X. Zhang, Y. Lu, et al., Cgrnet: contour-guided graph reasoning network for ambiguous biomedical image segmentation, Biomed. Signal Process. Control 75 (7) (2022), 103621, https://doi.org/10.1016/j.bspc.2022.103621.

[25] Y. Li, Y. Zhang, W. Cui, B. Lei, X. Kuang, T. Zhang, Dual encoder-based dynamic-channel graph convolutional network with edge enhancement for retinal vessel segmentation, IEEE Trans. Med. Imaging 41 (8) (2022) 1975–1989, https://doi.org/10.1109/TMI.2022.3151666.

[26] L. Mou, Y. Zhao, H. Fu, Y. Liu, J. Cheng, Y. Zhang, et al., Cs2-net: deep learning segmentation of curvilinear structures in medical imaging, Med. Image Anal. 67 (1) (2021), 101874, https://doi.org/10.1016/j.media.2020.101874.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, et al., Attention is all you need, in: Annual Conference on Neural Information Processing Systems, 2017, pp. 5998–6008.

[28] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv:1609.02907, 2016, doi: 10.48550/arXiv.1609.02907.

[29] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: Annual Conference on Neural Information Processing Systems, 2016, pp. 3837–3845.

[30] 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, doi: 10.1109/ICCV.2019.00936.

[31] X. Zhu, S. Zhang, Y. Zhu, P. Zhu, Y. Gao, Unsupervised spectral feature selection with dynamic hyper-graph learning, IEEE Trans. Knowl. Data Eng. 34 (6) (2020) 3016–3028, https://doi.org/10.1109/TKDE.2020.3017250.

[32] F. Ma, F. Gao, J. Sun, H. Zhou, A. Hussain, Attention graph convolution network for image segmentation in big sar imagery data, Remote Sens. (Basel) 11 (21) (2019) 2586, https://doi.org/10.3390/rs11212586.

[33] Y. Zhu, J. Ma, C. Yuan, X. Zhu, Interpretable learning based dynamic graph convolutional networks for Alzheimer's disease analysis, Information Fusion 77 (1) (2022) 53–61, https://doi.org/10.1016/j.inffus.2021.07.013.

[34] J.C. Caicedo, A. Goodman, K.W. Karhohs, B.A. Cimini, J. Ackerman, M. Haghighi, et al., Nucleus segmentation across imaging experiments: the 2018 data science bowl, Nat. Methods 16 (12) (2019) 1247–1253, https://doi.org/10.1038/s41592-019-0612-7.

[35] W. Al-Dhabyani, M. Gomaa, H. Khaled, A. Fahmy, Dataset of breast ultrasound images, Data Brief 28 (1) (2020), 104863, https://doi.org/10.1016/j.dib.2019.104863.

[36] O. Oktay, J. Schlemper, L. Folgoc, M. Lee, M. Heinrich, K. Misawa, et al., Attention u-net: learning where to look for the pancreas, arXiv:1804.03999, 2018, doi: 10.48550/arXiv.1804.03999.

[37] Z. Gu, J. Cheng, H. Fu, K. Zhou, H. Hao, Y. Zhao, et al., Ce-net: context encoder network for 2d medical image segmentation, IEEE Trans. Med. Imaging 38 (10) (2019) 2281–2292, https://doi.org/10.1109/TMI.2019.2903562.