Multi-Resolution based Compute-cheap Gated Global Attention for airway segmentation

Wu Guoqing and Wu Yonghuang

School of Information Science and Technology, Fudan University, Shanghai 200433, China

Abstract. Airway segmentation is a key step in the analysis of lung diseases. Effectively displaying the state of airway can reveal the patients with chronic obstructive pulmonary disease (COPD). However, it is a great difficulty for neural network to accurately identify the difference between the atmospheric tube and the tiny trachea. We propose a multi-resolution network and implement a three-axis fusion, computationally inexpensive self-attention mechanism, which enables longer dependencies to be explored and ultimately achieves excellent performance.

Keywords: Airway Segmentation, Multi-domain Multi-site, Self Attention.

1 Preprocessing

The preprocessing consists of two parts: intensity normalization and volume of interest (VOI) extraction. The purpose of intensity normalization is to normalize the image intensity to [0,1]. Specifically, the image intensity is first limited to the [min_value, max_value] interval through hard threshold comparison, and then the image intensity is normalized to [0,1] by the following formula:

$$I_{norm} = (I - min\_value)/(max\_value - min\_value) \quad (1)$$

I and I_{norm} represent the pixel values before and after normalization, respectively. When the minimum value of the read-in image intensity is less than or equal to -1000, [min_value, max_value] is set to [-1024, -512]. When the minimum value of the read-in image grayscale is greater than or equal to 0, [min_value, max_value] is set to [1, 9000].

Observing the segmentation data, it is found that the airway is only distributed in part of the volume, so we first extract the VOI of the airway from the original volume, and then train and test based on the VOI. After getting the segmentation result of the VOI, we put it back into the original volume to get the final overall segmentation result.

VOI extraction includes the following steps. First determine the layers used to extract VOI: the starting layer ($0.28 \times l$), the end layer ($0.75 \times l$) and the layer extraction step size (20), and l represents the number of layers of the volume. Then we determine the ROI area of each layer separately through the three steps shown in **Fig 1**. Next, we merge the ROI regions of each layer on one layer (take the union of the regions), and

determine the center of the merged region (x_c, y_c). Finally, taking (x_c, y_c) as the center, we extract the VOI of size [320, 200, l-start_slice], where start_slice $(0.1311 \times l)$ is the starting layer.

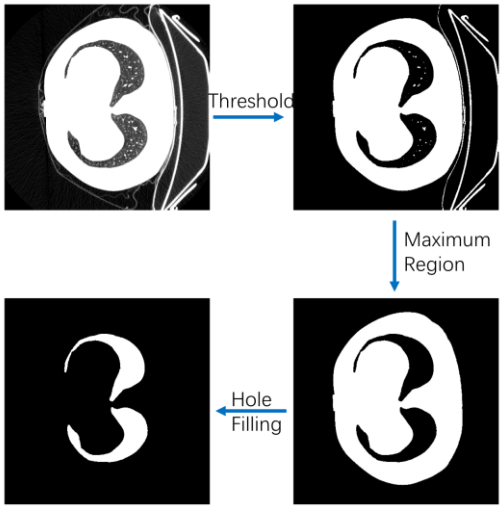

Fig. 1. VOI extraction.

2 Method

2.1 Overview

The traditional CNN networks have a very intuitive advantage in mining local features, and due to the mechanism of weight sharing, the pure CNN methods can achieve high performance while maintaining relatively controllable computational consumption. As Transformers expand into the field of computer vision, the shortcomings of CNNS in capturing global dependencies have been paid more and more attention by researchers. Through Query-Key Matching, the network can calculate the correlation between any feature on the feature map, which enhances the performance of the neural network, but also greatly increases the computational consumption. To alleviate this problem, some researchers have proposed local window attention or interacted it with convolution computation. Based on previous research, we propose compute-cheap gated Global Attention, which implements attention globally in a way that saves computational resources. Based on the design of Compute-cheap Gated Global Attention, we propose the complete multi-resolution network, and **Fig 2** is the overview of our model.

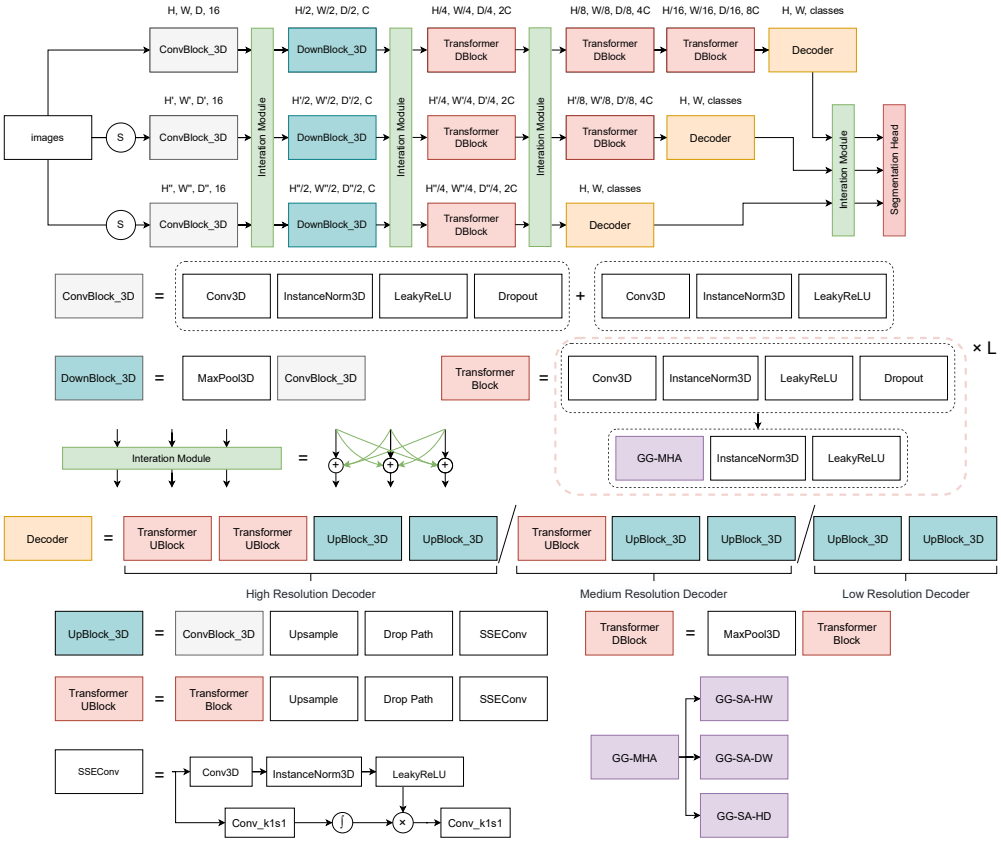

Fig. 2. Overview of our proposed structure. Round mark with S denotes $Interpolate$ operation which generate multi-resolution inputs for multi-branch network. $Decoder$ for different branch has different decoding layer depth as shown in figure.

2.2 Compute-cheap Global Attention

To use the traditional attention mechanism in the image data, first we need to generate three types of matrices, namely Query, Key and Value matrices. By matching the Query and Key, we can obtain the attention map. The attention map contains the correlation between all tokens and realizes the interaction between tokens, so that the model can establish theoretically arbitrarily long dependencies. This standard scale-dot attention can be expressed as:

$$Attention(Q, K, V) = softmax(QK^T)V \tag{2}$$

$Q, K, V \in \mathbb{R}^{n \times c}$, $softmax$ normalizes the channel dimension of the attention map. Because of the $softmax$ operation in the above expression and the complexity of QK^T is $\mathcal{O}(n^2)$, the calculation process is not optimal.

If the calculation of self-attention is carried out on the global feature map, the calculation efficiency will be reduced due to the large number of tokens. Some works, such as Swin Transformer, build relationships between tokens locally on the feature map and compensate for the lack of local attention by sliding the feature map operations, which is a trade-off strategy. We design a scheme to carry out the attention mechanism globally.

First, we generate Query, Key and Value according to the following formula,

$$Q, K, V = \phi_q(XW_q), \phi_k(XW_k), \phi_v(XW_v) \tag{3}$$

where $X \in \mathbb{R}^{n \times c}$, n is the number of tokens, c is the feature dimension, $W_q, W_k, W_v \in \mathbb{R}^{c \times c}$, ϕ is activation function, such as LeakyReLU.

The implementation of regular self-attention incurs unavoidable computational cost. As mentioned above, the complexity of $QK^T$is $\mathcal{O}(n^2)$, while the complexity of $K^T V$ is only $\mathcal{O}(c^2)$.

If we could replace the $softmax$ function with compute-cheap operations, we would get better results simply by changing the order in which the matrix is computed. To keep the calculations as similar as possible, we expanded the process of generating the attention map once to twice,

$$Attention(Q, K, V) = softmax\left(Q\widehat{K}^T\right) \cdot softmax\left(\widehat{Q}K^T\right) \cdot V \tag{4}$$

$\widehat{Q} \in \mathbb{R}^{m \times c}, \widehat{K} \in \mathbb{R}^{m \times c}$ are the new representations obtained by aggregating the original Q, K, respectively. In theory, we treat Q, K as n c-dimensional vectors and use Kmeans to aggregate them, and then obtain $\widehat{Q}, \widehat{K}$ represented by cluster centers. However, although this method is more interpretable, it needs the forward and backward propagation process of the reverse neural network in the calculation of Kmeans, so we will introduce a more concise implementation in the following.

Since the tokens are obtained from the image data through the $flatten$ operation, these tokens naturally have ability to recover into a feature map with spatial information. Therefore, we use the $trans$ operation to transform $X \in \mathbb{R}^{n \times c}$, which contains a sequence of tokens, into $Z \in \mathbb{R}^{c \times h \times w}$ ($reverse\_trans$ is also defined as the inverse operation), where h, w correspond to the aspect ratio of the input image, and the product is equal to n. We obtain $\widehat{Q}, \widehat{K}$ by the following equation.

$$\widehat{Q} = reverse\_trans\left(Pool(trans(Q))\right) \tag{5}$$

$$\widehat{K} = reverse\_trans\left(Pool(trans(K))\right) \tag{6}$$

$Pool(\cdot)$ is an average pooling function. Thus, we can not only avoid the problem of gradient tracking, but also make efficient use of the spatial information between tokens, which makes up for the deficiency that the previous self-attention can only focus on the temporal information between tokens.

2.3 Gated Attention

GLU(Gated Linear Unit) is a module based on MLP to achieve stronger performance by adding a gating mechanism. Each token representation (V_i) will be controlled by the corresponding U_i. It is common to express GLU as the following formula, where $\odot$ denotes the Hadamard product.

$$O = (U \odot V)W_o, U = \phi_u(XW_u), V = \phi_v(XW_v) \tag{7}$$

Although the gating implemented by GLU method is used in various works, it only controls a certain token, and the interaction between tokens is not completed in this process, which will undoubtedly affect the actual gain brought by the gating mechanism. In fact, we can achieve the union of Self-Attention and Gated Unit by enhancing the expression of V. We use the attention map generated by the Compute-cheap Global Attention method introduced above to enhance the expression of the features of V, and this process can be described as the following equation.

$$O = (U \odot AV)W_o \tag{8}$$

$$A = softmax(Q\hat{K}^T) \cdot softmax(\hat{Q}K^T) \tag{9}$$

2.4 Apply to 3D Image

In order to make better use of the spatial information of trachea and use the correlation between adjacent pixels to strengthen the network's ability to understand the segmented region, we input CT scans as data in 3D form. In order to avoid that the number of tokens increases dramatically with the increase of data dimension, and lead to higher computational complexity, we enhance the above Compute-cheap Gated Global Attention from 2D form to 2.5D. We implement the attention mechanism in three axes and fuse the output of the three axes to obtain global dependencies on the 3D data. In a real network implementation, we show this module as the following structure, where GG-SA module is a Compute-cheap Gated Global Attention in one axis.

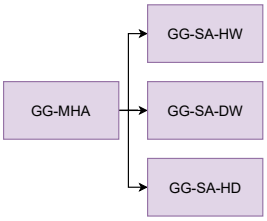

Fig. 3. Enhance Compute-cheap Gated Global Attention from 2D form to 2.5D instead of 3D.

Given an input $Z \in \mathbb{R}^{b\times c\times h\times w\times d}$ with batch symbol b, we can obtain $Z_h \in \mathbb{R}^{bh\times c\times w\times d}$, $Z_w \in \mathbb{R}^{bw\times c\times h\times d}$, $Z_d \in \mathbb{R}^{bd\times c\times h\times w}$ by transformation (we define the inverse of this operation as $Rearrange$). With the $trans$ operation and the respective Query, Key, and Value, we can fuse the output of the three axes using the following equation.

$$Out = \sum_i^{[h,w,d]} r\,earrange(O_i) = \sum_i^{[h,w,d]} r\,earrange\left(\left(U_i \odot \left(softmax\left(Q_i \widehat{K_i}^T\right) \cdot softmax\left(\widehat{Q_i} K_i^T\right)\right) V_i\right) W_o\right) \tag{10}$$

2.5 Compute-cheap Interaction

In order to enhance the multi-scale mining ability of the model and adapt to the segmentation task of different objects, we apply a multi-resolution network. Using the interpolation algorithm, we resize the original input to different resolution sizes and feed them into subnetworks whose weights are not shared. The number of layers in the three subnetworks are different, and depths are 4, 3 and 2, which correspond to high-resolution, medium-resolution and low-resolution inputs, respectively. In order to avoid parameter redundancy and feature redundancy brought by the three sub-networks and complicate the training process, we adopt the up-sampling layer and down-sampling layer without learning parameters, and conduct feature interaction in layers of the three networks to enhance the robustness of the network to different target sizes.

Given three input features X_s, X_m, X_l with different resolutions, we use an interpolation algorithm to align the features of the other two resolutions to the current feature for the interaction between three subnetworks. $Interpolate\left(X_j, X_i\right)$ indicates that the interpolation algorithm would be used to align X_j to the same resolution as X_i.

$$X_i = X_i + \sum_{j \in \{s,m,l\}/\{i\}} Interpolate\left(X_j, X_i\right),\ i = s, m, l \tag{11}$$

3 Experiments

We divide the initially provided 300 cases of data into a training set (290 cases included) and a validation set (10 cases included). Based on the VOI regions obtained by preprocessing, we randomly extracted patches of size 320*200*32 from the VOI area of each sample in the training set, and used the final 5362 image patches without transformation. We test the validation set every epoch of training, and calculate the Dice Similarity Coefficient, Sensitivity and Specificity to select the best model for test phase. The final model receive inputs with shape of 320*192*32 on the test set and post procession is applied to obtain the officially required NII file size.

The loss function in the initial 5 epochs is standard Cross Entropy loss, then we optimize the pixel-wise weighted CE loss which weight map generated from categorical information distribution. Given a mask $y \in R^{D \times H \times W}$ corresponding to input $x \in R^{C \times D \times H \times W}$, a map based on local category information statistics can be obtained.

$$\mathcal{W}(i,j,k) = \frac{1}{N} \sum_{a,b,c} y(a,b,c)$$

$$i - l < a < i + 1, j - l < b < j + l, k - l < c < k + l$$

A position with a pixel value equal to 1 means that the field is completely filled with objects, for which we don't need to pay much attention, just set the weight to 1. For the field represented by the pixel value of 0, it means that there is no target in this region.

However, in order to avoid the insufficient discrimination ability of the model for the background region in the inference process, we still set the weight of this region as 1. For other non-0 and non-1 pixels, we set the maximum weight to M and subtract the pixel value from W. Finally, the smaller the object contained in the field represented by the pixel, the more weight we give, while the larger the object, the less weight is given, but with a minimum value of 1. When all the weights are 1, the loss function degenerates into the standard unweighted cross entropy loss .

Our model was deployed through Pytorch and trained on a computer equipped with an Intel(R) Xeon(R) Gold 6230 CPU and an NVIDIA Tesla V100 (32 GB). The adaptive moment estimation (Adam) optimizer with a learning rate of 0.0005 was used for training. Batch size was set to 2. Total epoch was set to 50.