Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(1): 128–144.
DOI: 10.3934/mbe.2023007
Received: 19 June 2022
Revised: 05 September 2022
Accepted: 09 September 2022
Published: 29 September 2022

Research article

ConvWin-UNet: UNet-like hierarchical vision Transformer combined with convolution for medical image segmentation

Xiaomeng Feng[1,*,†] **Taiping Wang**[2,3,†], **Xiaohang Yang**[2,†], **Minfei Zhang**[2], **Wanpeng Guo**[2] **and Weina Wang**[1]

[1] Department of Mathematics, School of Sciences, Hangzhou Dianzi University, Hangzhou 310018, China
[2] Hangzhou Medipath Intelligent Technology Co., Ltd, Hangzhou, China
[3] School of Business, Macau University of Science and Technology, Macau, China

† Xiaomeng Feng, Taiping Wang and Xiaohang Yang are the co-first authors.

* **Correspondence:** Email: xmfeng@hdu.edu.cn; Tel: +8618258466306.

Abstract: Convolutional Neural Network (CNN) plays a vital role in the development of computer vision applications. The depth neural network composed of U-shaped structures and jump connections is widely used in various medical image tasks. Recently, based on the self-attention mechanism, the Transformer structure has made great progress and tends to replace CNN, and it has great advantages in understanding global information. In this paper, the ConvWin Transformer structure is proposed, which refers to the W-MSA structure in Swin and combines with the convolution. It can not only accelerate the convergence speed, but also enrich the information exchange between patches and improve the understanding of local information. Then, it is integrated with UNet, a U-shaped architecture commonly used in medical image segmentation, to form a structure called ConvWin-UNet. Meanwhile, this paper improves the patch expanding layer to perform the upsampling operation. The experimental results on the Hubmap datasets and synapse multi-organ segmentation dataset indicate that the proposed ConvWin-UNet structure achieves excellent results. Partial code and models of this work are available at https://github.com/xmFeng-hdu/ConvWin-UNet.

Keywords: ConvWin-UNet; vision transformer; convolution; UNet; segmentation

1. Introduction

Along with the development of deep learning, image segmentation algorithms, such as UNet [1–4], have achieved great success in medical applications, and the algorithms are significant for medical auxiliary diagnosis.

The classical UNet is a full convolution neural network (FCN) that mainly depends on the U-shaped structure [1,5,6]. The early UNet and its variants, such as U-Net++ [2], UNet3+ [3], U^2-Net [4], 3D U-Net [7], Res UNet [8], etc. are constructed based on CNN. At present, there are still much excellent research emerging in the UNet based on CNN, such as KUB-UNet [9]. Recently, the self-attention [10] structure, represented by Transformer [11], has swept the field of computer vision. Meanwhile, a series of UNet models based on Transformer have emerged, such as TransUNet [12], SwinUNet [13], TransClaw U-Net [14], and MT-UNet [15]. The performance of UNet has been further improved by the emergence of these networks.

In recent studies, the Transformer structure is feted. And driven by the success of the Transformer structure, the pioneering vision Transformer (ViT) is introduced in [11], which achieves an impressive speed-accuracy trade-off on image recognition tasks. To alleviate the difficulty that the training of ViT requires large-scale training datasets, Deit [16] introduces several training strategies that allow ViT to be effectively trained on ImageNet. Recently, the Swin Transformer [17] architecture achieves the best performance, and it shows a good speed-accuracy trade-off on various vision tasks including image classification. But the Transformer structure is rarely combined with CNN structure.

This paper focuses on the integration of the Transformer and CNN, and combines this structure with UNet to apply it to medical image segmentation. It can not only use the Transformer to obtain global features, but also use CNN to fuse local features, and realize information exchange between patches. The rest of this paper is organized as follows. In Section 2, the related work of CNN and Transformer is introduced briefly. The ConvWin Transformer structure and the integration of this structure into the U-shaped network in UNet are elaborated. Also, the improvements based on the patch expanding layer and various loss functions are described. In Section 3, the experiments and results are discussed. Finally, Section 4 concludes this paper.

2. Materials and methods

2.1. Datasets

Hubmap - Hacking the kidney identify glomeruli in human kidney tissue images [18]: The Hubmap dataset includes 11 fresh frozen and 9 Formalin-Fixed Paraffin-Embedded (FFPE) PAS kidney images. Among 15 labeled samples publicly, there are Glomeruli FTU annotations for all 20 tissue samples. In this paper, to verify the effectiveness of the algorithm, only the 15 labeled samples are used, where 10 samples are divided into the training set and 5 samples into the testing set. Each picture is cut into small pieces, and the effective area size of each piece is 1024×1024. See Figure 1A for the specific data pretreatment process. After this, there are 5867 pieces in the training set and 2599 pieces in the test set. The average Dice-Similarity coefficient (DSC) is used to evaluate our method on this dataset.

Synapse multi-organ segmentation dataset [19]: This dataset consists of 8 abdominal organs (aorta, gallbladder, spleen, left kidney, right kidney, liver, pancreas, spleen, stomach). In this paper, the use of the training or testing images is the same as that in TransUNet [12]. They have used 30 abdominal CT scans in the MICCAI 2015 Multi-Atlas Abdomen Labeling Challenge, with 3779 axial contrast-enhanced abdominal clinical CT images in total. Each CT volume consists of $85 \sim 198$ slices of 512×512 pixels, with a voxel spatial resolution of $([0.54 \sim 0.54] \times [0.98 \sim 0.98] \times [2.5 \sim 5.0])$ mm^3. See Figure 1B for the specific data pretreatment process. Similarly, the average DSC and average

Hausdorff Distance (HD) are used to evaluate our method on this dataset.

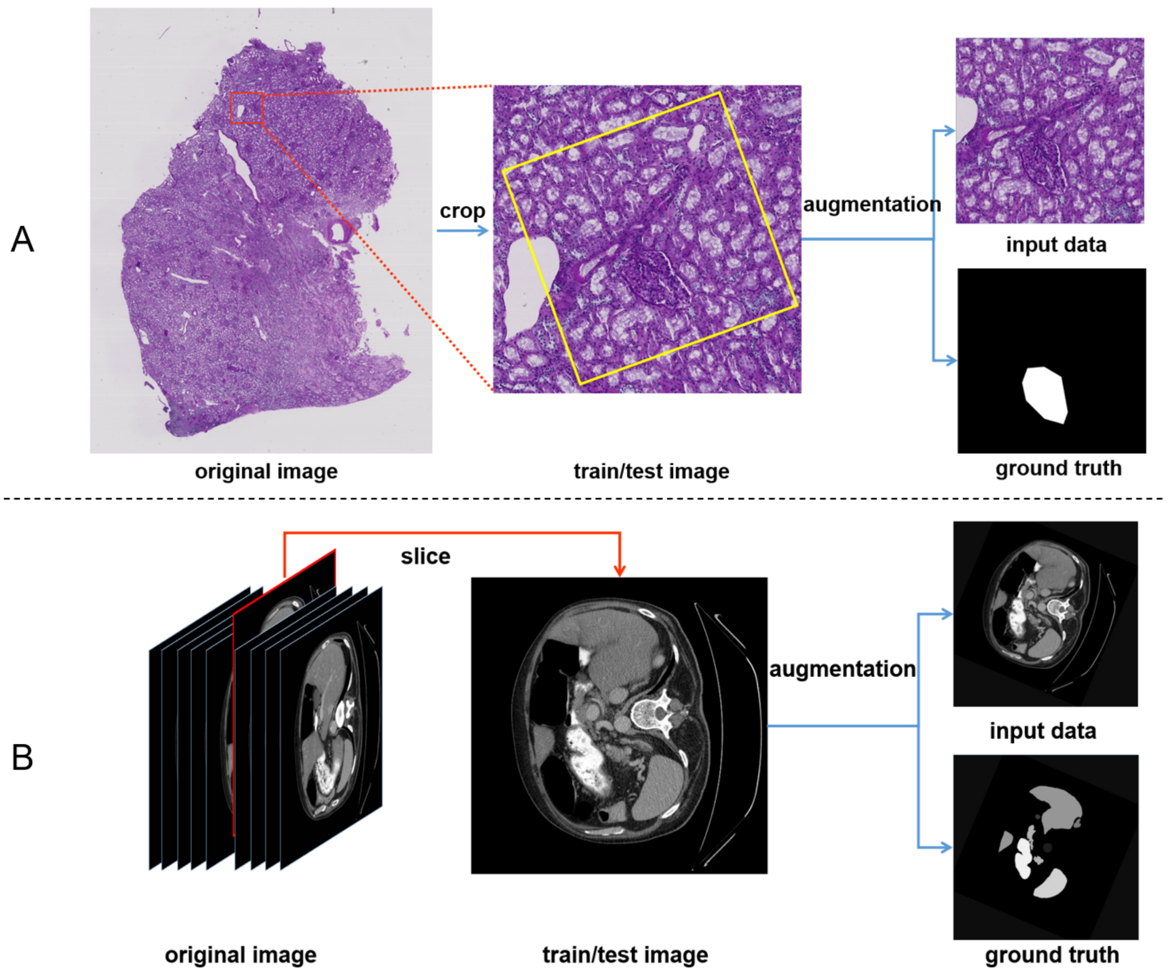

Figure 1. The specific data pretreatment process: A is for Hubmap dataset; B is for Synapse dataset.

2.2. *Related work*

CNN-based UNet: CNN is the first automatic learning algorithm that can successfully train the multi-layer network model in a real sense. The network structure of CNN is mainly designed for image pretreatment. With the development of deep CNN, AlexNet [20] is proposed for image classification. Since then, deeper and more effective convolutional neural architectures have been proposed, such as VGG [21], GoogleNet [22], ResNet [23], DenseNet [24], and EfficientNet [25]. At the same time, UNet, which has been widely used in medical images, comes into being. The original UNet creatively combines the U-shaped structure and convolution network based on FCN to realize the semantic segmentation of medical images at the pixel level. Subsequently, UNet++, UNet3+ add the information interaction between encoder and decoder based on its structure. UNet++ introduces the dense connection into the U-shaped structure, while UNet3+ constructs full scale skip connections to realize the multi-scale information interaction between encoder and decoder. In addition, Res-UNet introduces the resblock structure in ResNet into UNet to make it have stronger feature understanding ability. U^{2}Net creatively replaces each block in UNet with a U-shaped structure. At present, UNet based on CNN still has important applications and research in different fields. For example, the latest KUB-UNet [9] applies UNet to the segmentation of the organs of urinary system and achieves the best results.

Transformer-based UNet: Recently, Transformer has been introduced into the field of medical image segmentation in many studies, and the Transformer structure was combined with UNet [12–15]. Among them, Trans Unet [12] uses CNN to extract shallow features and uses Transformer to extract depth features. MT UNet [14] further constructs the mixed Transformer module based on Transformer structure and takes it as the encoder and decoder with deep features. TransClaw U-Net [15] also uses the Transformer structure to extract deep-seated features. The difference is that the algorithm additionally performs the same level of upsampling on the features extracted by the Transformer structure and combines the results with the features of the original encoder and decoder to use concatenation. Finally, SwinUNet [13] is based on Swin Transformer [17], and it uses pure Transformer structure for feature extraction. At the same time, it constructs a patch expanding layer that do not depend on convolution to realize upsampling based on Transformer.

2.3. Methods

Our research focuses on the integration of Transformer and CNN. First, the ConvWin Transformer structure is constructed in this paper, which refers to the W-MSA structure in Swin and combines with the convolution. The convolution is added after the window attention structure. Then, the ConvWin Transformer structure is integrated as the basic module into the U-shaped structure that is commonly used for medical image segmentation, and the new network is called ConvWin-UNet. Extensive experiments demonstrate that the Transformer structure can be integrated well with CNN, and the use of group convolution is more effective than the use of traditional CNN. Meanwhile, the order of LayerNorm and the convolution has an important impact on the experimental results. Besides, the convergence can be faster if the linear layer in the upsampling of the decoder is replaced with the convolution layer, but the convergence is sometimes not ideal.

2.3.1. Architecture overview & ConvWin Transformer block

The overall architecture of the proposed ConvWin-UNet is presented in Figure 2A, which mainly includes the encoding part and the decoding part. As shown in Figure 2A, the encoding part and the decoding part are divided into four stages. The first stage of the encoding part is composed of a patch embedding layer [11] and two ConvWin Transformer blocks. In the other stages of the encoding part, the patch embedding layer is replaced with the patch merging layer. Every stage of the decoding part consists of two ConvWin Transformer blocks and a patch expanding layer. Between every stage of the encoding part and the decoding part, the low-level features are connected through skip connections. As shown in Figure 2B, the skip connections are composed of 1-layer LayerNorm (LN) and 1-layer Linear, which is different from CNN.

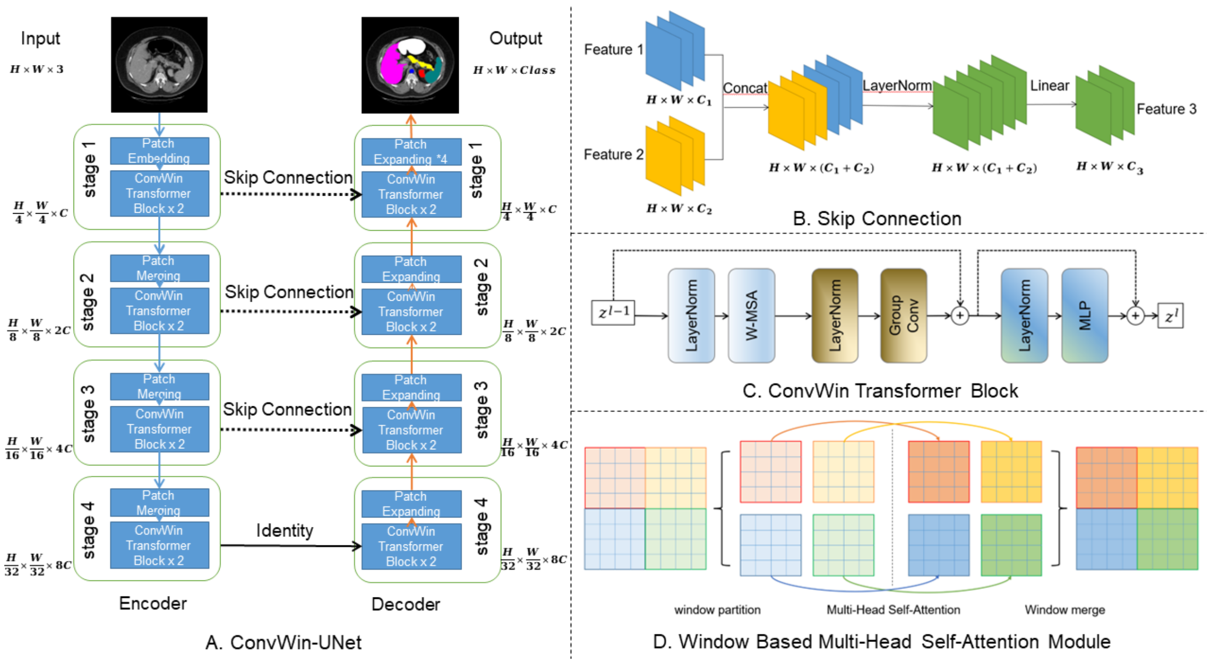

Figure 2. The architecture of ConvWin-UNet: A is the complete network structure of ConvWin-UNet; B is the specific structure of the skip connection; C shows the structure of ConvWin Transformer Block; D shows the structure of window based multi-head self-attention module.

The basic unit of ConvWin-UNet is the ConvWin Transformer block (as shown in Figure 2C), which is mainly constructed by convolution and window based multi-head self-attention module (W-MSA). Here, the W-MSA divides the input feature map into $M \times M$ windows, then uses MSA for each window, and finally reassembles the results obtained from each window into a complete output feature map. Figure 2D shows the specific implementation form of this structure (taking 8×8 feature map and 2×2 windows as examples). Further, the ConvWin Transformer block is composed of an LN layer, a window based multi-head self-attention module (W-MSA), a group convolution, a residual connection, and a MLP block which contains two linear layers and uses GELU as the activation function. Different from Swin Transformer, this paper uses a convolution layer (without activation function and BN) after each multi-head self-attention layer and realizes the interaction between adjacent patches without the shift window in Swin Transformer. And through the experiments, it is found that the group convolution (GC) has a better effect than the ordinary convolution, and adding a LayerNorm module before the convolution further improves the effectiveness of the model. Similarly, ConvWin Transformer blocks can be formulated as:

$$\hat{z}^l = GC(LN(W\_MSA(LN(z^{l-1})))) + z^{l-1}, \tag{2.1}$$

$$z^l = MLP(LN(\hat{z}^l)) + \hat{z}^l, \tag{2.2}$$

where $\hat{z}^l$ and z^l represent the outputs of the W-MSA module and the MLP module of the l^{th} block, respectively. The self-attention is calculated as follows, which is the same as the previous works

[10,11]:

$$Attention(Q, K, V) = SoftMax(\frac{QK^T}{\sqrt{d}} + B)V, \tag{2.3}$$

where $Q, K, V \in \mathbb{R}^{M^2 \times d}$ are the query, key, and value matrices; M^2 is the number of patches in a window, and d is the query or key dimension. The matrix B is taken from the bias matrix $\hat{B} \in \mathbb{R}^{(2M-1)\times(2M-1)}$.

2.3.2. Encoder

In this paper, the encoder part is divided into four stages. The first stage is composed of a patch embedding layer and 2 ConvWin Transformer blocks, and the other stages consist of a patch merging layer and 2 ConvWin Transformer blocks. Among them, the patch embedding layer is composed of a patch partition layer and linear embedding layer, which is the same as that in Swin Transformer. It concatenates the features of each group of 4×4 neighboring patches and then passes the 48 dimensional concatenated features through a linear layer to project them to an arbitrary dimension (denoted by C). The patch merging layer is similar to the patch embedding layer, and it concatenates the features of each group of 2×2 neighboring patches. In this way, the number of patchs reduces to 1/4 of the original, and the output dimension is set to $2C$.

For an input image with a dimension of $H \times W \times 3$, the dimension is reduced to $\frac{H}{4} \times \frac{W}{4} \times C$ after the first stage. In this paper, $C = 96$. Each of the remaining stage performs 2× downsampling. Thus, the feature dimensions of the output are respectively $\frac{H}{8} \times \frac{W}{8} \times 2C$, $\frac{H}{16} \times \frac{W}{16} \times 4C$, and $\frac{H}{32} \times \frac{W}{32} \times 8C$ after the other stages. Finally, the feature vector with a dimension of $\frac{H}{32} \times \frac{W}{32} \times 8C$ is obtained through the encoder.

2.3.3. Decoder

Correspondingly, the decoder part is also divided into four stages in this paper. Each stage is composed of a patch expanding layer and 2 ConvWin Transformer blocks. Similar to that described in the previous paragraph, the features of each stage are fused with its corresponding encoder features through skip connections. For the patch expanding layer, several comparison structures have been constructed in this paper, as shown in Table 1. The main differences between these structures are 1) the order of LayerNorm operations; 2) whether using linear or convolution to expand feature dimensions; 3) whether using realrange or pixelshuffle for downsampling. The specific effects of these structures will be compared in Section 3.3. It should be noted that the patch expanding layer of the last stage uses 4× upsampling, the others use 2× upsampling.

Therefore, the dimension of the feature vector obtained by the encoder changes from $\frac{H}{32} \times \frac{W}{32} \times 8C$ to $\frac{H}{16} \times \frac{W}{16} \times 4C$, $\frac{H}{8} \times \frac{W}{8} \times 2C$, and $\frac{H}{4} \times \frac{W}{4} \times C$ respectively after the first three stages of the decoder. After the last stage, a feature vector with a dimension of $H \times W \times N$ is obtained, where N is the number of classes.

2.3.4. Loss functions & multi-stage outputs

In this paper, some experiments are conducted on multi-stage outputs. Based on the structure shown in Figure 2, by using the same patch expanding layer, the output of multiple stages in the decoder is directly upsampled as the output of the network structure. The ground truth is denoted as P, and the output of each layer is denoted as Q^i, where i indicates the i-th stage.

Table 1. Five structures of the patch expanding layer.

	A	B	C	D	E
input	H/2×W/2×2C	H/2×W/2×2C	H/2×W/2×2C	H/2×W/2×2C	H/2×W/2×2C
step1	**Linear** H/2×W/2×4C	**LayerNorm** H/2×W/2×2C	**LayerNorm** H/2×W/2×2C	**LayerNorm** H/2×W/2×2C	**LayerNorm** H/2×W/2×2C
step2	**Rearrange** H×W×C	**Linear** H/2×W/2×4C	**Linear** H/2×W/2×4C	**Conv3×3** H/2×W/2×4C	**GroupConv3×3** H/2×W/2×4C
step3	**LayerNorm** H×W×C	**Rearrange** H×W×C	**Pixelshuffle** H×W×C	**Rearrange** H×W×C	**Rearrange** H×W×C
output	H×W×C	H×W×C	H×W×C	H×W×C	H×W×C

For the loss function, a combination of various loss functions is used for experiments, including CrossEntropy Loss [26], Dice Loss [27], MSSIM (Mean Structure Similarity Index Measure) [28,29].

For image x and image y, denote $l(x,y)$ as the mean value to estimate brightness, $c(x,y)$ as the variance to estimate the contrast, and $s(x,y)$ as the covariance to estimate the structural similarity. Specifically,

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \tag{2.4}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \tag{2.5}$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}, \tag{2.6}$$

where C_1, C_2, C_3 are positive constants. $C_1 = (K_1L)^2$, $C_2 = (K_2L)^2$, and $C_3 = C_2/2$, where the default values of K_1, K_2, and L are used, i.e., $K_1 = 0.01$, $K_2 = 0.03$, $L = 2^B - 1$ (B is the bit depth, and $B = 256$). Denoting CrossEntropy Loss as L_{ce}, Dice Loss as L_{dice} and MSSIM as L_{mssim}, we have:

$$L_{ce}(Q^i,P) = \frac{1}{H\times W}\cdot\sum_{m=0}^{H-1}\sum_{n=0}^{W-1}\left(-\sum_{k=0}^{D-1}P_{mnk}\cdot\ln\left(\frac{\exp(Q^i_{mnk})}{\sum_{t=0}^{D-1}\exp(Q^i_{mnt})}\right)\right), \tag{2.7}$$

$$L_{dice}(Q^i,P) = 1 - \frac{2|Q^i\cap P| + \varepsilon}{|Q^i| + |P| + \varepsilon}, \quad fix\,\varepsilon = 1, \tag{2.8}$$

$$L_{mssim}(Q^i,P) = [l_M(Q^i,P)]^{\alpha_M}\cdot\prod_{j=1}^{M}[c_j(Q^i,P)]^{\beta_j}\cdot[s_j(Q^i,P)]^{\gamma_j}, \tag{2.9}$$

where H, W, D is the height, width and depth (the output channel size) of the image, $\alpha_1 = \beta_1 = 0.0448$, $\alpha_2 = \beta_2 = 0.2856$, $\alpha_3 = \beta_3 = 0.3001$, $\alpha_4 = \beta_4 = 0.2363$, $\alpha_5 = \beta_5 = \gamma_5 = 0.1333$. Thus, the loss function of each stage is:

$$loss(Q^i, P) = w_{ce} \times L_{ce}(Q^i, P) + w_{dice} \times L_{dice}(Q^i, P) + w_{mssim} \times L_{mssim}(Q^i, P), \tag{2.10}$$

where w_{ce}, w_{dice}, w_{mssim} are weights for the loss functions, and they are set to 1 by default. If a loss is not used, the corresponding weight is set to 0. Then, the total loss function is

$$Loss = \sum_{i=1}^{n} w_i \times loss(Q^i, P), \tag{2.11}$$

where w_i is the weight of the i-th stage, and it is a super parameter; n is the number of the output stages. If multi-stage outputs are not used, $n = 1$.

3. Experiments and results

3.1. Implementation details

In the experiment on the Hubmap dataset, more than 50% of the data is cut around the effective slice to prevent the black edge caused by rotations in data expansion. During the test, only the 1024×1024 effective area in the middle of the data is taken. Also, a comparative experiment is conducted on the use of data augmentation for each algorithm. The size of the input image is resized to 512×512 when it is input to the network. All experimental model parameters do not use any pretraining model but only use the same initialization method. During the training period, the batch size is set to 16, and the Adam optimizer with a base learning rate of 0.001 and a weight decay of 1e-4 is used to optimize our model for backpropagation. The default number of training epochs is 200. On this dataset, four pieces of Nvidia RTX2080Ti GPU are used to train each model.

In the experiment on the Synapse dataset, to keep the working environment the same, the input image size of 224×224 is used. Meanwhile, all models are trained with the SGD optimizer with a learning rate of 0.05, the momentum of 0.9, and the weight decay of 1e-4. The default batch size is 24, and the default number of training epochs is 150. Besides, only one piece of Nvidia RTX2080Ti GPU is used to train the models.

In the experiments, this paper compares the effects of whether using group convolution on the models in encoder and decoder respectively. At the same time, the performance differences between linear layer, convolution layer and group convolution layer are compared in the selection of upper sampling layer. Here, the structures of the comparison models in the experiments are as follows:

- 1. Encoder:
  - unet: Original UNet encoder structure,
  - swin: Original Swin Transformer structure,
  - cnvwin: Convwin Transformer structure constructed in this paper,
  - group_convwin: Group convlution used in the Convwin Transformer structure.

- 2. Decoder:
  - unet: Original UNet encoder structure,
  - cnvwin: Convwin Transformer structure constructed in this paper,
  - group_convwin: Group convlution used in the Convwin Transformer structure.
- 3. Upsampling in decoder
  - linear up: Linear layer used in upsampling (Table 1B),
  - conv up: convlution layer used in upsampling (Table 1D),
  - group conv up: group convlution layer used in upsampling (Table 1E).

3.2. Experiment results on the hubmap dataset

In this section, experiments are conducted on the combinations of the encoder, decoder, and loss function. It should be noted that there are 2 groups of comparison models construct, namely the base model and the plus model. The network layer used in the encoder stages of the base model is $[3, 4, 6, 3]$, while that of the plus model is $[4, 8, 8, 4]$. Moreover, in the decoder, the output channel of each stage is set to 64, and the network layer is set to 1.

In Figure 3, several combinations of the encoder and decoder are compared. The ConvWin structure constructed in this paper has certain advantages. Meanwhile, in the comparison of encoders (Table 2), the performance differences brought by different encoders with the same decoder structure are investigated. The results show that the ConvWin structure has advantages over UNet and Swin. From Table 2, it can also be seen that when the decoder is further replaced with ConvWin structure, the performance is more significantly improved. In addition, it can be seen from Figure 4 that ConvWin structure is more sensitive to data augmentation. When using data augmentation, the performance improvement of ConvWin structure is more obvious than that of the original UNet. It can also be seen here that the structure constructed in this paper needs additional data augmentation to improve the spatial understanding ability (such as rotation invariance).

Table 2. The experimental results of different encoders.

Encoder	Decoder	Loss function	Aug	Max DSC	Mean DSC
unet	unet	CE	yes	90.83	90.25
swin	unet	CE	yes	89.66	88.65
convwin	unet	CE	yes	91.14	90.29
group_convwin-base	unet	CE	yes	91.11	90.43
group_convwin-plus	unet	CE	yes	91.11	90.40
convwin	group_convwin+conv up	CE	yes	91.24	90.40
group_convwin-base	group_convwin+conv up	CE	yes	91.46	90.70
group_convwin-plus	group_convwin+conv up	CE	yes	91.65	90.73

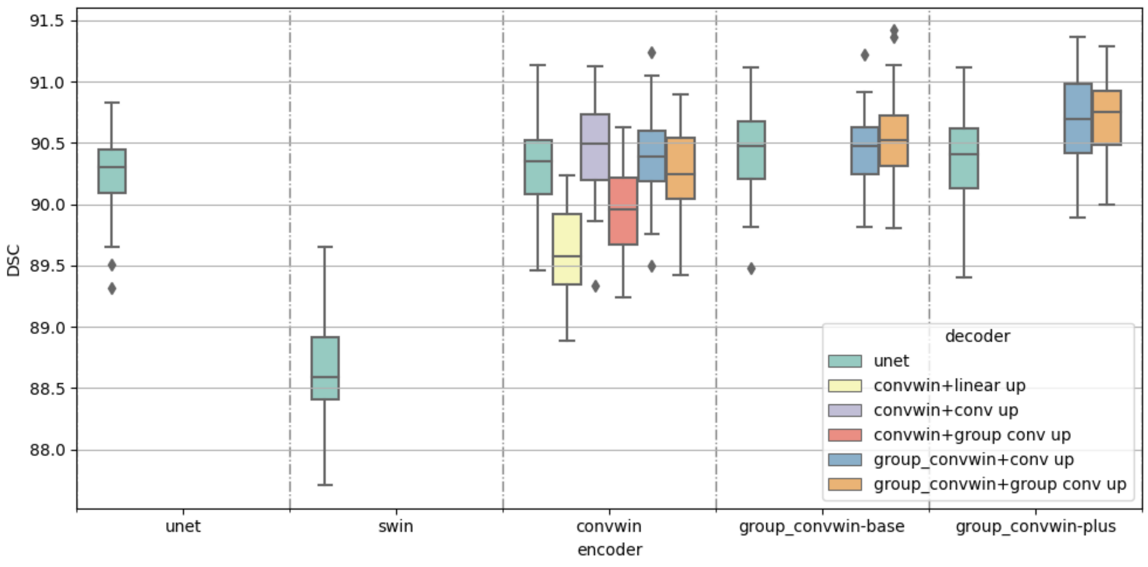

Figure 3. The boxplot of the model results: the abscissa represents the encoder, and the ordinate represents the score. Different colors represent different decoders. The encoder structures including unet, swin, conwin, and group convwin are considered. The decoder structures including unet, convwin + linear up, convwin + conv up, convwin + group conv up, group convwin + conv up, and group convwin + group conv up are considered.

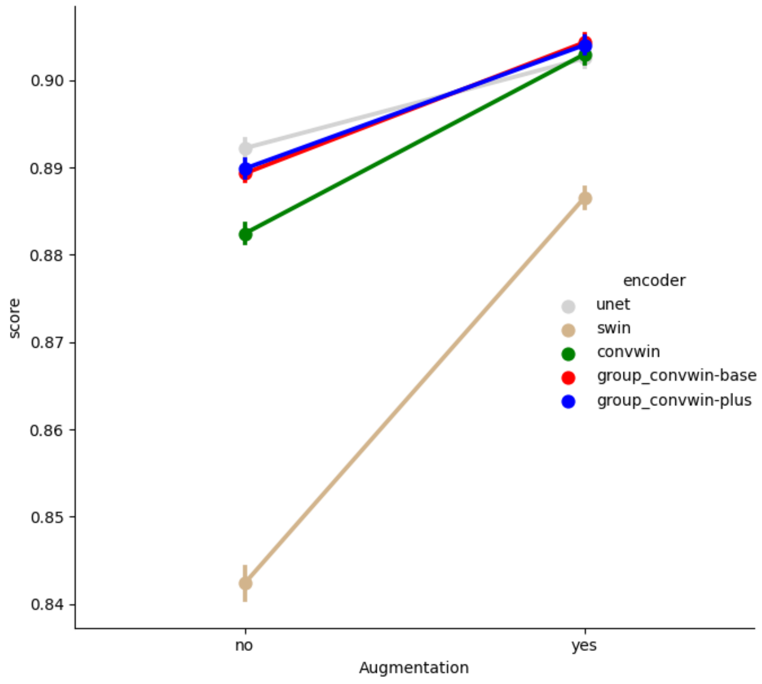

Figure 4. The line segments go up with the results from no data augmentation to data augmentation, under different encoders and the same decoder (original UNet structure). The slopes of the line segments reflect the sensitivity of different encoders to data augmentation.

Besides, the use of different decoders leads to different effects. As shown in Table 3, the effects of using different decoders and the same ConvWin structure as the encoder are investigated. The results show that using the (Group) ConvWin structure as the decoder has advantages, especially compared with the original UNet structure, the ConvWin structure constructed in this paper has been improved to different degrees under different encoders. In addition, it is not good to use group ConvWin in all places. Especially in the patch expanding layer, group ConvWin is not a better choice than ConvWin. As can be further seen from Figure 5, there are differences in using and not using the group convolution in the patch expanding structure of the decoder. The expanding layer with the group convolution performs slightly better than that without the group convolution in the case of using the data expansion, but the result is the opposite without the data expansion. However, it can be seen from the experimental results that there is no significant difference between using and not using the group convolution in expanding. In addition, the experimental results of expanding on the Synapse dataset will be further analyzed.

Table 3. The experimental results of different decoders.

Encoder	Decoder	Loss function	Augmentation	Max DSC	Mean DSC
convwin	unet	CE	yes	91.14	90.29
convwin	convwin+conv up	CE	yes	91.12	90.46
convwin	group_convwin+conv up	CE	yes	91.24	90.40
group_convwin-base	unet	CE	yes	91.11	90.43
group_convwin-base	group_convwin+conv up	CE	yes	91.46	90.70
group_convwin-plus	unet	CE	yes	91.11	90.40
group_convwin-plus	group_convwin+conv up	CE	yes	91.65	90.73
convwin	convwin+linear up	CE	yes	90.23	89.59
convwin	convwin+group conv up	CE	yes	90.62	89.93
convwin	convwin+conv up	CE	yes	91.12	90.46

Furthermore, on the Hubmap dataset, an experiment is performed to compare the loss functions and the pre-trainings of ImageNet. As shown in Figure 6A, the use of MSSIM improves the performance slightly but the use of Dice significantly reduces the performance of the model. Meanwhile, the use of multi-stage outputs ($n = 4$) improves the model performance slightly, and the model performance is also more stable. In this paper, the models pre-trained for 20 epochs on the Imagenet-1K dataset are used as pre-training models to train the corresponding segmentation model from scratch. The results are shown in Figure 6B. Pre-training does not significantly improve the performance of the model. However, the training curve in Figure 6C shows that the pre-training model is helpful for convergence.

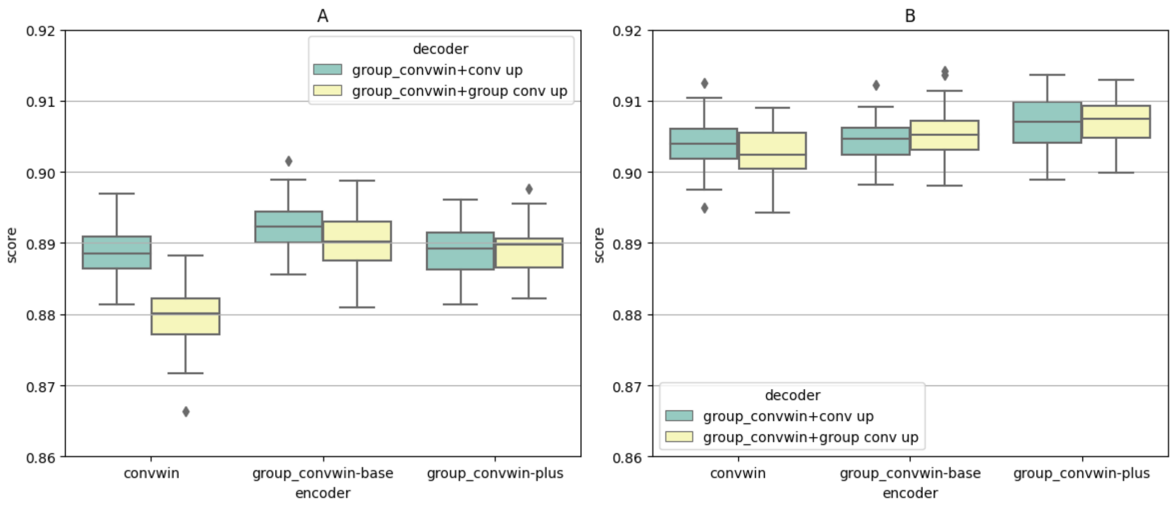

Figure 5. The boxplot of the results under different patch expanding blocks: A presents the results without data augmentation; B presents the results with data augmentation. The green color indicates that group convolution is used in the patch expanding block; the yellow color indicates that the convolution is used without group.

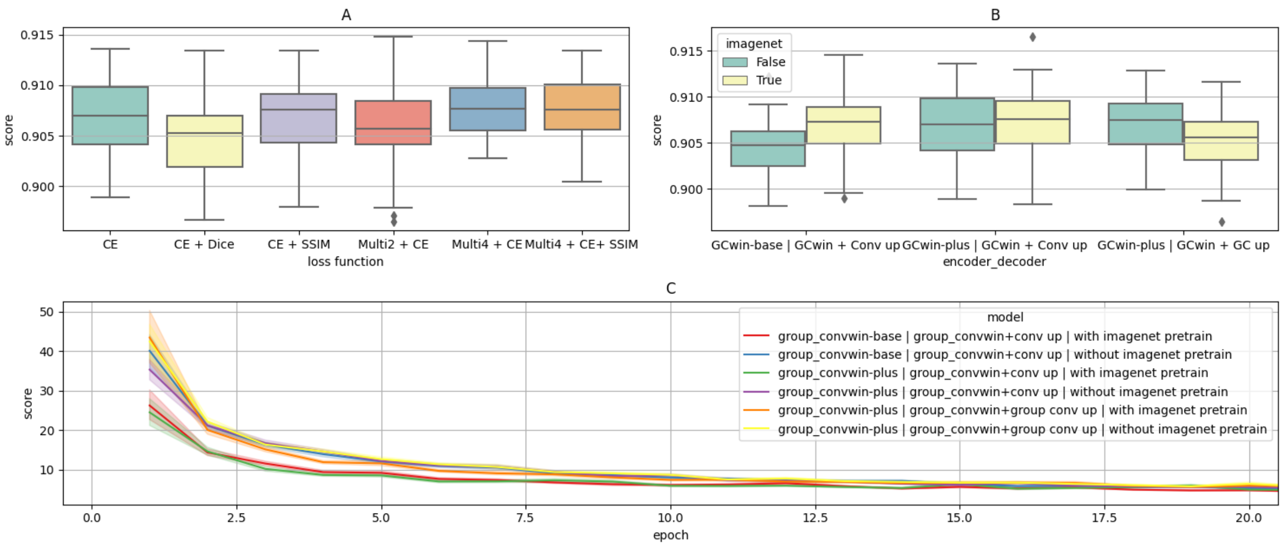

Figure 6. A. The results of using different loss functions; B. The results of using and not using the pre-trained model (GCwin means group ConvWin); C. The training loss curves of using and not using the pre-trained model.

3.3. *Experiment results on the synapse dataset*

To make a fair comparison of similar methods, the experiments are also conducted on the Synapse dataset. The network structure described in Section 2 is used here, and the loss function is Dice + CE (keep up with TransUNet). As shown in Table 4, our method achieves the best performance among the algorithms taken for comparison. The segmentation results of different methods on the Synapse dataset are shown in Figure 7. Our method handles the details more carefully. This is attributed to the added convolution module, which increases the perception of local features.

Table 4. The segmentation accuracy of different methods on the Synapse dataset.

Methods	DSC↑	HD↓	Aor.	Gal.	Kid. (L)	Kid. (R)	Liv.	Pan.	Spl.	Sto.
V-Net [27]	68.81	-	75.34	51.87	77.10	80.75	87.84	40.05	80.56	56.98
DARR [30]	69.77	-	74.74	53.77	72.31	73.24	94.08	54.18	89.90	45.96
R50 U-Net [12]	74.68	36.87	87.74	63.66	80.60	78.19	93.74	56.90	85.87	74.16
U-Net [1]	76.85	39.70	89.07	69.72	77.77	68.60	93.43	53.98	86.67	75.58
R50 Att-UNet [12]	75.57	36.97	55.92	63.91	79.20	72.71	93.56	49.37	87.19	74.95
Att-UNet [31]	77.77	36.02	89.55	68.88	77.98	71.11	93.57	58.04	87.30	75.75
R50 ViT [12]	71.29	32.87	73.73	55.13	75.80	72.20	91.51	45.99	81.99	73.95
TransUNet [12]	77.48	31.69	87.23	63.13	81.87	77.02	94.08	55.86	85.08	75.62
SwinUNet [13]	79.13	21.55	85.47	66.53	83.28	79.61	94.29	56.58	90.66	76.60
TransClaw [14]	78.09	26.38	85.87	61.38	84.83	79.36	94.28	57.65	87.74	73.55
MT-UNet [15]	78.59	26.59	87.92	64.99	81.47	77.29	93.06	59.46	87.75	76.81
ConvWin UNet	79.39	21.39	84.37	69.89	84.85	82.08	93.69	58.84	87.32	74.04

Furthermore, further experiments are conducted on the expanding structure (see Table 1 specifically). As shown in Table 5, compared with method A, method B can significantly improve the performance by placing Layernorm before Linear in the expanding structure. However, the use of convolution in the expanding structure reduces the performance, which is quite different from the results obtained on the Hubmap dataset. However, the training curve in Figure 8 indicates that the use of convolution in the expanding structure makes the model converge faster, and the loss value is lower (It may be easier to fit in the training set after adding the convolution).

4. Discussion

This paper focuses on the integration of Transformer and CNN and combines this structrue with UNet to apply it to medical image segmentation. Based on this, this paper constructs a ConvWin block, and a variety of patch expanding blocks to realize upsampling. It can not only use Transformer to obtain global features, but also use CNN to fuse local features, and realize information exchange between patches. Finally, a new loss function is constructed with MSSIM. At the same time, experiments are carried out on several datasets, and good results are obtained.

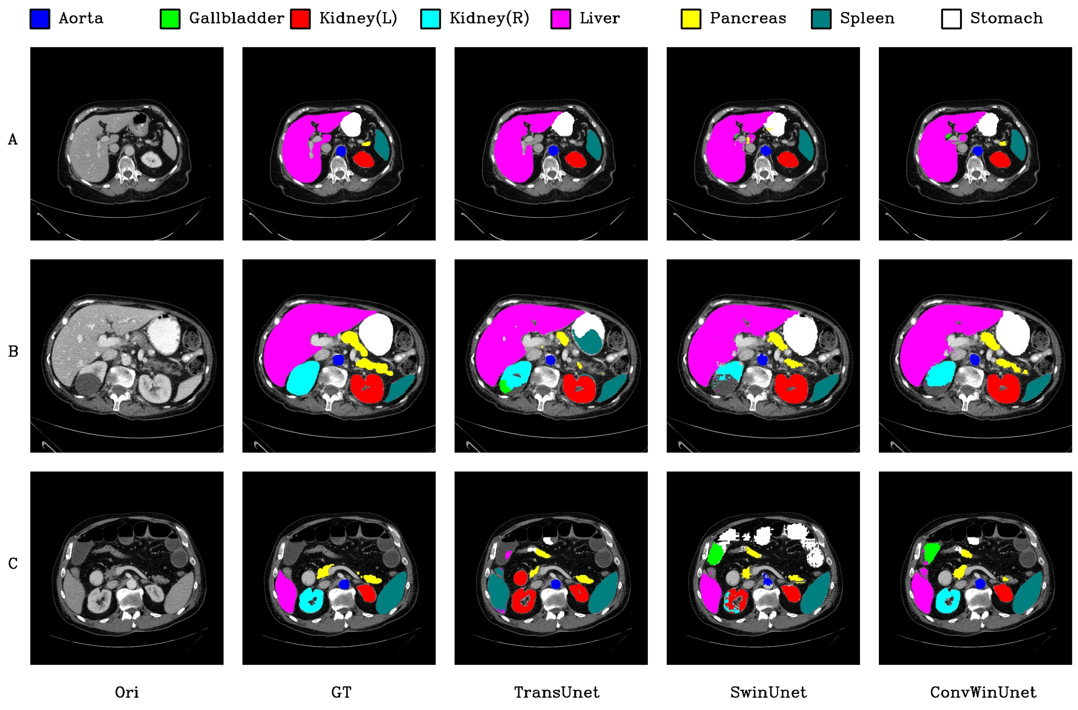

Figure 7. The segmentation results of different methods on the Synapse dataset.

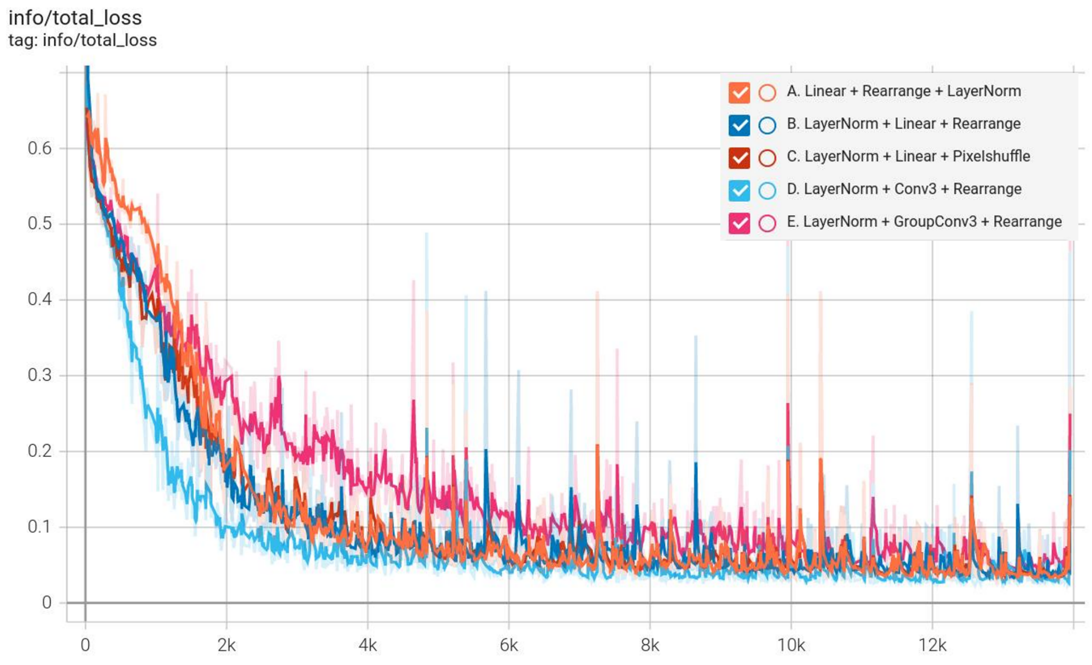

Figure 8. The training loss curve of different patch expanding blocks on the Synapse dataset.

Table 5. The segmentation accuracy of different methods on the Synapse dataset.

Methods	DSC↑	HD↓	Aor.	Gal.	Kid. (L)	Kid. (R)	Liv.	Pan.	Spl.	Sto.
A	77.24	27.91	83.06	67.47	84.46	80.13	93.15	50.82	88.02	70.81
B	79.38	21.39	84.37	69.89	84.85	82.08	93.69	58.84	87.32	74.04
C	77.40	23.27	82.99	66.43	83.14	80.34	93.27	55.57	87.81	69.66
D	75.96	27.42	82.37	70.79	81.45	73.33	93.07	46.19	88.10	72.36
E	62.81	58.85	71.01	59.53	67.53	60.35	89.28	33.25	72.41	49.12

[A] Linear + Rearrange + LayerNorm
[B] LayerNorm + Linear + Rearrange
[C] LayerNorm + Linear + Pixelshuffle
[D] LayerNorm + Conv3+ Rearrange
[E] LayerNorm + GroupConv3+ Rearrange

In addition, this paper also has some shortcomings and can be further studied. 1) For different datasets, the effect of the patch expanding blocks constructed in this paper are different, and more stable and universal patch expanding block can be further explored in future research. 2) It seems that the understanding of the global information needs to be further improved. Although SSIM is added as part of the loss, the local details need to be further refined and further optimization needs to be made for SSIM. 3) The initialization weight can also be further studied. In the subsequent research, the Self-Supervised Learning algorithm can be further tried to improve the robustness of initialization parameters.

Acknowledgments

The work is support by Fundamental Research Funds for the Provincial Universities of Zhejiang: GK219909299001-025; Scientific Research Foundation of Hangzhou Dianzi University: KYS075621028; Natural Science Foundation of China: 12001144; Zhejiang Provincial Natural Science Foundation of China: LQ20A010007. We would like to thank the referees for their careful reading and helpful comments and suggestions which improved the presentation of the paper.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. 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), 234–241. Available from: https://link.springer.com/chapter/10.1007/978-3-319-24574-4_28.

2. Z. Zhou, M. Siddiquee, N. Tajbakhsh, J. Liang, UNet++: redesigning skip connections to exploit multiscale features in image segmentation, *IEEE Trans. Med. Imaging*, **39** (2020), 1856–1867. https://doi.org/10.1109/TMI.2019.2959609

3. 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 (ICASSP)*, (2020), 1055–1059. https://doi.org/10.1109/ICASSP40776.2020.9053405

4. X. Qin, Z. Zhang, C. Huang, M. Dehghan, O. R. Zaiane, M. Jagersand, U^2-Net: going deeper with nested U-structure for salient object detection, *Pattern Recognit.*, **106** (2020), 107404. https://doi.org/10.1016/j.patcog.2020.107404

5. F. Isensee, P. F. Jaeger, S. A. Kohl, J. Petersen, K. H. Maier-Hein, nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation, *Nat. Methods*, **18** (2021), 203–211. Available from: https://www.nature.com/articles/s41592-020-01008-z.

6. Q. Jin, Z. Meng, C. Sun, H. Cui, R. Su, RA-UNet: a hybrid deep attention-aware network to extra liver and tumor in ct scans, preprint, arXiv:1811.01328.

7. Ö, Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, O. Ronneberger, 3D U-Net: learning dense volumetric segmentation from sparse annotation, in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, (2016), 424–432. Available from: https://link.springer.com/chapter/10.1007/978-3-319-46723-8_49.

8. X. Xiao, S. Lian, Z. Luo, S. Li, Weighted res-unet for high-quality retina vessel segmentation, in *2018 9th International Conference on Information Technology in Medicine and Education (ITME)*, (2018), 327–331. https://doi.org/10.1109/ITME.2018.00080

9. G. Rani, P. Thakkar, A. Verma, V. Mehta, R. Chavan, V. S. Dhaka, et al., KUB-UNet: segmentation of organs of urinary system from a KUB X-ray image, *Comput. Methods Programs Biomed.*, **224** (2022), 107031. https://doi.org/10.1016/j.cmpb.2022.107031

10. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., Attention is all you need, in *Advances in Neural Information Processing Systems*, (2017), 5998–6008. https://doi.org/10.48550/arXiv.1706.03762

11. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, et al., An image is worth 16x16 words: transformers for image recognition at scale, preprint, arXiv:2010.11929.

12. J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, et al., Transunet: transformers make strong encoders for medical image segmentation, preprint, arXiv:2102.04306.

13. H. Cao, Y. Wang, J. Chen, D. Jiang, X. Zhang, Q. Tian, et al., Swin-Unet: Unet-like pure transformer for medical image segmentation, preprint, arXiv:2105.05537.

14. C. Yao, M. Hu, G. Zhai, X. P. Zhang, Transclaw U-Net: claw U-Net with transformers for medical image segmentation, preprint, arXiv:2107.05188.

15. H. Wang, S. Xie, L. Lin, Y. Iwamoto, X. Han, Y. Chen, et al., Mixed transformer u-net for medical image segmentation, in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (2022), 2390–2394. Available from: https://ieeexplore.ieee.org/abstract/document/9746172.

16. H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, H. Jégou, Training data-efficient image transformers & distillation through attention, preprint, arXiv:2012.12877.

17. Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, et al., Swin transformer: hierarchical vision transformer using shifted windows, preprint, arXiv:2103.14030.
18. Synapse multi-organ segmentation dataset. Available from: https://www.synapse.org/#!Synapse:syn3193805/wiki/217789.
19. HuBMAP - Hacking the Kidney Identify glomeruli in human kidney tissue images. Available from: https://www.kaggle.com/c/ hubmap-kidney-segmentation/data.
20. A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Commun. ACM*, **60** (2017), 84–90. https://doi.org/10.1145/3065386
21. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, preprint, arXiv:1409.1556
22. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., Going deeper with convolutions, in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2015), 1–9. https://doi.org/10.1109/CVPR.2015.7298594
23. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2016), 770–778. https://doi.org/10.1109/CVPR.2016.90
24. G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger, Densely connected convolutional networks, preprint, arXiv:1608.06993.
25. M. Tan, Q. Le, Efficientnet: rethinking model scaling for convolutional neural networks, preprint, arXiv:1905.11946.
26. P. T. De Boer, D. P. Kroese, S. Mannor, R. Y. Rubinstein, A tutorial on the cross-entropy method, *Ann. Oper. Res.*, **134** (2005), 19–67. Available from: https://link.springer.com/article/10.1007/s10479-005-5724-z.
27. F. Milletari, N. Navab, S. A. Ahmadi, V-net: fully convolutional neural networks for volumetric medical image segmentation, in *2016 Fourth International Conference on 3D Vision (3DV)*, (2016), 565–571. https://doi.org/10.1109/3DV.2016.79
28. Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, Image quality assessment: from error visibility to structural similarity, *IEEE Trans. Image Process.*, **13** (2004), 600–612. https://doi.org/10.1109/TIP.2003.819861
29. Z. Wang, E. P. Simoncelli, A. C. Bovik, Multiscale structural similarity for image quality assessment, in *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, (2003), 1398–1402. https://doi.org/10.1109/ACSSC.2003.1292216
30. S. Fu, Y. Lu, Y. Wang, Y. Zhou, W. Shen, E. Fishman, et al., Domain adaptive relational reasoning for 3d multi-organ segmentation, preprint, arXiv:2005.09120.
31. O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, et al., Attention u-net: learning where to look for the pancreas, preprint, arXiv:1804.03999v3.

AIMS Press

© 2023 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0)