Patcher: Patch Transformers with Mixture of Experts for Precise Medical Image Segmentation

Yanglan Ou[1], Ye Yuan[2], Xiaolei Huang[1], Stephen T.C. Wong[3], John Volpi[4], James Z. Wang[1], Kelvin Wong[3]

[1] The Pennsylvania State University, University Park, Pennsylvania, USA
[2] Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
[3] TT and WF Chao Center for BRAIN & Houston Methodist Cancer Center, Houston Methodist Hospital, Houston, Texas, USA
[4] Eddy Scurlock Comprehensive Stroke Center, Department of Neurology, Houston Methodist Hospital, Houston, Texas, USA

Abstract. We present a new encoder-decoder Vision Transformer architecture, Patcher, for medical image segmentation. Unlike standard Vision Transformers, it employs Patcher blocks that segment an image into large patches, each of which is further divided into small patches. Transformers are applied to the small patches within a large patch, which constrains the receptive field of each pixel. We intentionally make the large patches overlap to enhance intra-patch communication. The encoder employs a cascade of Patcher blocks with increasing receptive fields to extract features from local to global levels. This design allows Patcher to benefit from both the coarse-to-fine feature extraction common in CNNs and the superior spatial relationship modeling of Transformers. We also propose a new mixture-of-experts (MoE) based decoder, which treats the feature maps from the encoder as experts and selects a suitable set of expert features to predict the label for each pixel. The use of MoE enables better specializations of the expert features and reduces interference between them during inference. Extensive experiments demonstrate that Patcher outperforms state-of-the-art Transformer- and CNN-based approaches significantly on stroke lesion segmentation and polyp segmentation. Code for Patcher is released to facilitate related research.[5]

Keywords: Medical Image Segmentation · Vision Transformers · Mixture of Experts

1 Introduction

Deep learning-based medical image segmentation has many important applications in computer-aided diagnosis and treatment. Until recently, the field of medical image segmentation has mainly been dominated by convolutional neural networks (CNNs). U-Net and its variants [15,3,21,12,8,2] are a representative class of CNN-based models, which are often the preferred networks for image segmentation. These models mainly adopt an encoder-decoder architecture, where

[5] Code: `https://github.com/YanglanOu/patcher.git`

an encoder uses a cascade of CNN layers with increasing receptive fields to capture both local and global features while a decoder leverages skip-connections and deconvolutional layers to effectively combine the local and global features into the final prediction. Despite the tremendous success of CNN-based models, their drawbacks start to become apparent as their performance saturates. First, CNNs are suboptimal at modeling global context. While increasing the depth of CNN-based models can enlarge the receptive fields, it also leads to problems such as diminishing feature reuse [16], where low-level features are diluted by consecutive multiplications. Second, the translation invariance of CNNs is a double-edged sword – it allows CNNs to generalize better, but also severely constrains their ability to reason about the spatial relationships between pixels.

Originally designed for natural language processing tasks, Transformers [17] have recently become popular in computer vision and image analysis domains since the invention of ViT [4]. By design, Vision Transformers have the ability to address the two aforementioned drawbacks of CNNs: (1) They can effectively model the global context by segmenting an image into patches and applying self-attention to them; (2) The use of positional encodings makes Transformers be able to model spatial relationships between patches. Due to these advantages, many Vision Transformer models have been proposed for image segmentation. For instance, SETR [20] employs a ViT-based encoder in an encoder-decoder architecture to attain superior performance over CNNs. Recently, Swin Transformer [10] adopts a hierarchical design to improve the efficiency of Vision Transformers and achieves SOTA results on various vision tasks including image segmentation. The success of Vision Transformers has attracted significant attention in the domain of medical image segmentation. For example, Swin-Unet [1], LambdaUNet [13], and U-NetR [5] replace convolutional layers with Transformers in a U-Net-like architecture. Other models like TransUNet [2], U-Net Transformer [14], and TransFuse [19] adopt a hybrid approach where they use Transformers to capture global context and convolutional layers to extract local features. So far, most prior works utilize Transformers mainly to extract patch-level features instead of fine pixel-level features. Given Transformers' strong abilities to model spatial relationships, we believe there is an opportunity to fully leverage Transformers for extracting fine-grained pixel-level features without delegating the extraction task to convolutional layers.

To this end, we propose a new encoder-decoder Vision Transformer architecture, Patcher, that uses Transformers for extracting fine-grained local features in addition to global features. Its key component is the Patcher block, which segments an image into large patches (*e.g.,* 32×32), each of which is further divided into small patches (*e.g.,* 2×2). Transformers are applied to the small patches within each large patch to extract pixel-level features. Each large patch constrains the receptive fields of the pixels inside, and we intentionally make the large patches overlap to enhance intra-patch communication. The encoder employs a cascade of Patcher blocks with increasing receptive fields to output a sequence of feature maps extracted from local to global levels. This design allows Patcher to combine the best of both worlds: it shares the classic coarse-to-

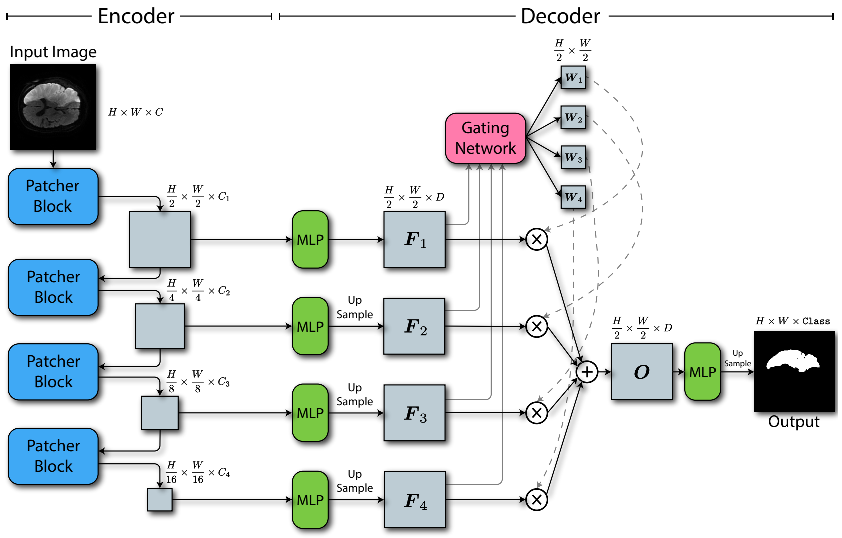
Encoder
Decoder
Input Image
$H \times W \times C$
Patcher Block
Patcher Block
Patcher Block
Patcher Block
$\frac{H}{2} \times \frac{W}{2} \times C_1$
$\frac{H}{4} \times \frac{W}{4} \times C_2$
$\frac{H}{8} \times \frac{W}{8} \times C_3$
$\frac{H}{16} \times \frac{W}{16} \times C_4$
MLP
Up Sample
$\frac{H}{2} \times \frac{W}{2} \times D$
F_1
F_2
F_3
F_4
Gating Network
$\frac{H}{2} \times \frac{W}{2}$
W_1
W_2
W_3
W_4
O
$H \times W \times$ Class
Output

Fig. 1. Model overview of Patcher. The encoder uses a cascade of Patcher blocks to extract expert features from local to global levels. The MoE-based decoder uses a gating network to select a suitable set of expert features for the prediction of each pixel.

fine feature extraction common in CNNs and also enjoys Transformers' superior spatial relationship modeling power to capture low-level details. Moreover, we observe that image segmentation models mainly require local features for some pixels (*e.g.,* edge pixels) while relying more on global features for other pixels (*e.g.,* pixels inside a global shape). This motivates us to further propose a new mixture-of-experts (MoE) based decoder. It treats the feature maps from the encoder as experts and learns a gating network to select a suitable set of expert features to predict the label for each pixel. By using MoE, the model can learn more specialized and disentangled expert feature maps and reduce interference between them during inference.

The *contributions* of this paper are as follows. First, we propose a new Vision Transformer architecture that can effectively make use of large and small patches to focus on global and local context. Second, we propose a new MoE-based decoder that enables better specialization and disentanglement of feature maps, which substantially improves performance. Lastly, extensive experiments demonstrate that our method outperforms SOTA Transformer- and CNN-based approaches significantly on stroke lesion segmentation and polyp segmentation.

2 Method

2.1 Overall Architecture

The network architecture of Patcher is outlined in Fig. 1. Given an input image of size $H \times W \times C$, Patcher first uses an encoder to extract features from the input

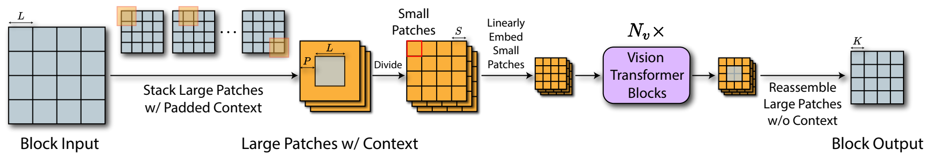
L
Stack Large Patches w/ Padded Context
P
L
Divide
Small Patches
S
Linearly Embed Small Patches
$N_v \times$
Vision Transformer Blocks
Reassemble Large Patches w/o Context
K
Block Input
Large Patches w/ Context
Block Output

Fig. 2. Overview of the Patcher block. The input is segmented into large patches with overlapping context, each of which is further divided into small patches. The small patches are processed by a sequence of Vision Transformer blocks to extract fine-grained features. The final output is produced by reassembling the large patches.

image. The encoder contains a cascade of Transformer-based Patcher blocks (detailed in Sec. 2.2), which produce a sequence of feature maps capturing visual features from local to global levels with increasing receptive fields. These feature maps are then input to a decoder with a mixture-of-experts (MoE) design, where each of the feature maps from the encoder serves as an expert. A four-layer gating network in the decoder outputs weight maps for the expert feature maps and uses the weights to obtain a combined feature map. A multi-layer perceptron (MLP) and an up-sampling layer are then used to process the combined feature map into the final segmentation output. The MoE-based design increases the specialization of different levels of features while reducing the interference between them. It allows the network to make predictions for each pixel by choosing a suitable set of expert features. For example, the network may need global features for pixels inside a particular global shape, while it may require local features to capture fine details at segmentation boundaries. Finally, we use the standard binary cross-entropy (BCE) loss for image segmentation to train Patcher.

2.2 Patcher Block

The key component inside the Patcher encoder is the Patcher block, which is a versatile Transformer-based module that can extract visual features from the block input at different spatial levels. The overview of the Patcher block is outlined in Fig. 2. We first divide the block input of spatial dimensions $H \times W$ into an $N_h \times N_w$ grid of large patches. Each of the large patches is of size $L \times L$, where $N_h = H/L$ and $N_w = W/L$. We further pad each large patch with P pixels from neighboring patches on each side, forming a patch of size $(L+2P) \times (L+2P)$. We stack the large patches with padded context along the batch dimension (batch size $B = B_0 N_h N_w$ where B_0 is the original batch size), so different large patches will not interfere with each other in the subsequent operations. Each large patch defines a receptive field similar to the kernel in CNNs, with the difference that all pixels inside the large patch share the same receptive field. Therefore, it is crucial to have the padded context since it enlarges the receptive field of pixels, which is especially important for pixels close to the patch boundaries. Next, we partition the stacked large patches further into an $M_h \times M_w$ grid of small patches, each of size $S \times S$, which have a limited receptive field as defined by the large patch, therefore focusing on local context modeling. Similar to prior work, we linearly embed all pixels inside each small patch into a token, and the tokens of all small patches form a sequence. We then use N_v Vision Transformer blocks [4] to process the sequence to model the relationship between patches and extract useful visual features. Inspired by SegFormer [18], we do not use positional encodings but mix convolutional layers inside the MLPs of the Transformer to capture spatial relationships. We also use the efficient self-attention in SegFormer to further reduce the computational cost. We provide more details of the Vision Transformer blocks in the supplementary materials. The output feature maps of the Transformer blocks have a spatial dimension of $M_h \times M_w$ with batch size B. We take the center $K \times K$ area from the feature maps where $K = L/S$, which excludes the padded context and corresponds to the actual large patches. We then reassemble the large patches based on their locations in the original image to form the final output with spatial dimensions $\frac{H}{S} \times \frac{W}{S}$. There are two important hyperparameters of the Patcher block: (1) large patch size L, which defines the receptive field and allows feature extraction at local or global levels; (2) padded context size P, which controls how much information from neighboring large patches are used.

2.3 Patcher Encoder

As shown in Fig. 1, the Patcher encoder employs a cascade of four Patcher blocks to produce four feature maps with decreasing spatial dimensions and increasing receptive fields. The small patch size S is set to 2 for all the blocks, which means the spatial dimensions are halved after each block. The large patch size L and padded context size P are set to 32 and 8, respectively. By setting L and P the same for all the blocks, we allow deeper Patcher blocks to have a larger receptive field, therefore gradually shifting the focus of the blocks from capturing local features to global features. This encoder design mirrors the behavior of its CNN counterparts such as U-Net [15], which has been proven effective. Therefore, the Patcher encoder combines the best of both worlds – it not only benefits from the superior spatial relationship modeling of Transformers but also enjoys the effective coarse-to-fine feature extraction of CNNs.

2.4 Mixture of Experts Decoder

The decoder follows an MoE design, where it treats the four feature maps from the encoder as experts. As illustrated in Fig. 1, the decoder first uses pixel-wise MLPs to process each of the feature maps and then upsamples them to the size of the first feature maps, *i.e.,* $\frac{H}{2} \times \frac{W}{2} \times D$, where D is the number of channels after the MLPs. We use $[\boldsymbol{F}_1, \boldsymbol{F}_2, \boldsymbol{F}_3, \boldsymbol{F}_4]$ to denote the upsampled features, which are also called expert features. Next, a gating network takes the expert features as input and produces the weight maps $[\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4]$ for the expert feature maps, where each weight map is of size $\frac{H}{2} \times \frac{W}{2}$. The weight maps sum to 1 per pixel, *i.e.,* $\boldsymbol{W}_1 + \boldsymbol{W}_2 + \boldsymbol{W}_3 + \boldsymbol{W}_4 = \mathbf{1}$. The gating network first concatenates all the expert feature maps along channels and uses several convolutional layers and a final softmax layer to process the concatenated features into the weight maps. We then use the weight maps to produce the combined feature map $\boldsymbol{O}$:

$$\boldsymbol{O} = \sum_{i=1}^{4} \boldsymbol{W}_i \odot \boldsymbol{F}_i \,, \tag{1}$$

where $\odot$ denotes pixel-wise multiplication. The combined feature map $\boldsymbol{O}$ then passes through another MLP to predict the segmentation logits before being upsampled to the original image size. The MoE design of the decoder allows the network to learn more specialized feature maps and reduce the interference between them. For the prediction of each pixel, the gating function chooses a suitable set of features by weighing the importance of global *vs.* local features.

3 Experiments

Datasets. We perform experiments on two important medical image segmentation tasks. (1) **Stroke lesion segmentation**: We collect 99 acute ischemic stroke cases for this research with help from Houston Methodist hospital. The dataset contains 99 cases (2,451 images) in total. Each image has two channels, the eADC and DWI, from ischemic stroke patients. We use 67 cases (1,652 images) for training, 20 cases (499 images) for validation, and 12 cases (300 images) for testing. The test data has a high diversity of lesion sizes, locations, and stroke types. (2) **Polyp segmentation**: We further conduct experiments on a public dataset of gastrointestinal polyp images, Kvasir-SEG [7]. The dataset contains 1,000 RGB images and corresponding segmentation masks. We randomly split the dataset with a ratio of 8:1:1 for training, validation, and testing.
Implementation Details. We train our models on two NVIDIA RTX 6000 GPUs using a batch size of 8, which takes around 12 hours. For both stroke lesion and polyp segmentation, the input image is scaled to 256×256. During training, we randomly scale the image with a ratio from 0.7 to 2.0 and crop the image to 256×256 again. For the encoder's four Patcher blocks, we use $[64, 128, 320, 512]$ for the Transformer embedding dimensions and $[3, 6, 40, 3]$ for the number of Transformer blocks N_v. For the decoder, all the MLPs have hidden dimensions $[256, 256]$ with ReLU activations. The expert feature maps $[\boldsymbol{F}_1, \boldsymbol{F}_2, \boldsymbol{F}_3, \boldsymbol{F}_4]$ have $D = 256$ channels. The gating network uses four 3×3 convolutional layers with channels $[256, 256, 256, 4]$ and ReLU activations. For stroke lesion segmentation, we use the AdamW [11] optimizer with an initial learning rate of $6e-5$. For polyp segmentation, we use the Adam [9] optimizer with an initial learning rate of $1e-4$. We adopt a polynomial-decay learning rate schedule for both datasets. We also use the Intersection-over-Union (IoU) loss in addition to the BCE loss for polyp segmentation, which improves model training.

3.1 Comparison with State-of-the-Art Methods

We compare our model, Patcher, with SOTA Transformer- and CNN-based segmentation models – SETR [20], SegFormer [18], Swin Transformer [10], U-Net [15], AttnUNet [12], and TransUNet [2] – using their released code. We use

Table 1. Quantitative comparison for stroke lesion segmentation.

Method	DSC	IoU
UNet [15]	84.54	82.07
TransUNet [2]	87.37	83.14
AttnUNet [12]	85.30	83.28
SETR [20]	82.88	77.40
SegFormer [18]	81.45	79.56
Swin Transformer [10]	84.74	80.73
Patcher (Ours)	**88.32**	**83.88**

Table 2. Quantitative comparison for polyp segmentation on Kvasir-SEG [7].

Method	DSC	IoU
UNet [15]	78.89	67.81
TransUNet [2]	82.80	70.86
AttnUNet [12]	77.08	61.48
SETR [20]	75.30	61.60
SegFormer [18]	87.39	78.81
Swin Transformer [10]	85.58	77.64
Patcher (Ours)	**90.67**	**84.31**

two common metrics for image segmentation – dice score coefficient (DSC) and IoU.

Quantitative Results. The results for stroke lesion segmentation and polyp segmentation are shown in Tables 1 and 2, respectively. We can observe that Patcher outperforms the Transformer- and CNN-based baselines significantly. It is worth noting that the two segmentation tasks evaluate different aspects of the models. Stroke lesion segmentation requires the model to capture local details, and CNN-based models such as TransUnet [2] and AttnUNet [12] perform better than Transformer-based baselines. However, these CNN-based models perform much worse than Transformer-based models for polyp segmentation, *e.g.,* 82.80 (TransUNet) *vs.* 90.26 (Patcher), which is because polyp segmentation relies on better modeling of the global context where Transformers-based models excel. We notice that some papers report higher scores on Kvasir-SEG [6,19]. However, [6,19] use different train/test splits and do not have a validation split. In contrast, we use a validation split to select the best model to avoid overfitting to test data. On both tasks, Patcher consistently outperforms the baselines, which shows its superior ability to model both local details and global context.

Visual Results. In Fig. 3, we provide a visual comparison of the segmentation maps from various models, where we paint correct predictions in green, false positives (FPs) in red, and false negatives (FNs) in white. We can observe that the baselines often have FPs and FNs, do not predict the segmentation boundaries well, and often miss small lesions. In contrast, Patcher has much fewer FPs and FNs, and can predict the segmentation boundaries accurately, even for very small lesions (Fig. 3, rows 2 and 4). This is due to Patcher's effective use of Transformers for capturing fine-grained pixel-level details. We also provide a visual comparison for polyp segmentation in the supplementary materials.

Visualization of MoE weight maps. To better understand the MoE decoder, in Fig. 4, we visualize the MoE weight maps $[\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4]$ for the expert features for stroke lesion segmentation. We can see that MoE allows different weight maps to focus on different areas: $\boldsymbol{W}_1$ focuses on local details, $\boldsymbol{W}_2$ focuses on the global context inside the brain, $\boldsymbol{W}_3$ focuses on the boundaries of the brain, and $\boldsymbol{W}_4$ focuses on areas of the brain that are complementary to the lesions.

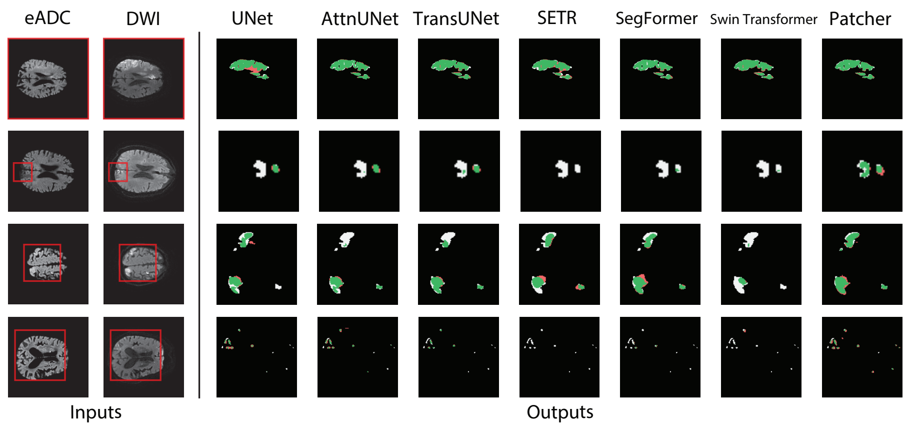

Fig. 3. Visualization of stroke lesion segmentation. The outputs correspond to the red box. We highlight correct predictions (green), false positives (red), and false negatives (white). Patcher outputs more accurate segmentation especially for small lesions.

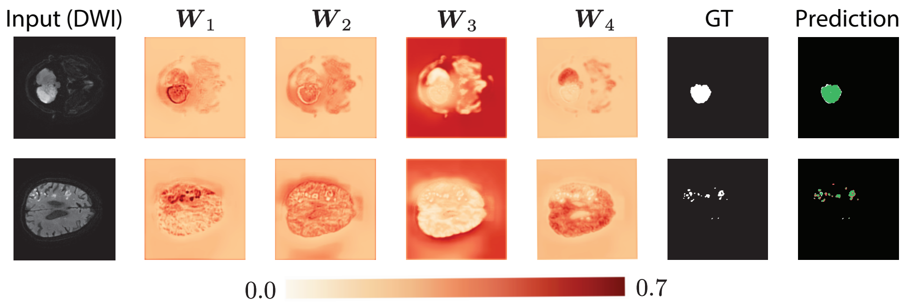

Fig. 4. Visualization of four MoE weight maps $[\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4]$. Different weight maps focus on different areas (local details, global context, brain boundaries, *etc.*)

3.2 Ablation Study

We first perform ablation studies to evaluate the importance of the Patcher encoder and MoE decoder by replacing them with popular encoder or decoder designs. As shown in Table 3, when replacing the Patcher encoder with SETR [20] or Swing Transformer [10], the performance decreases significantly. Similarly, when replacing the MoE decoder with SETR-PUP [20] and U-Net [15], the performance also drops considerably. This validates the importance of both designs.

We also conduct experiments to study the effect of varying the large patch size L and padded context size P. As shown in Table 4, both L and P need to be carefully selected as they need to have enough context (not too small) while also focusing mainly on local features (not too large) to attain better performance.

Table 3. Ablation studies of the Patcher encoder and MoE decoder.

Encoder	Decoder	DSC	IoU
SETR	MoE (Ours)	77.23	72.69
Swin Transformer	MoE (Ours)	85.29	81.47
Patcher (Ours)	SETR-PUP	85.84	79.79
Patcher (Ours)	U-Net	87.27	83.06
Patcher (Ours)	MoE (Ours)	**88.32**	**83.88**

Table 4. Effect of padded context size P and large patch size L for the 4 Patcher blocks.

L = [32,32,32,32]			P = 8		
Context P	DSC	IoU	Large Patch L	DSC	IoU
0	86.85	83.40	[64,64,64,32]	85.50	83.55
4	85.12	83.36	[64,64,32,32]	87.70	83.78
8	**88.32**	**83.88**	[32,32,32,32]	**88.32**	**83.88**
16	86.88	83.31	[32,16,16,16]	87.15	83.53

4 Conclusions

In this paper, we proposed Patcher, a new Vision Transformer architecture for medical image segmentation that can extract fine-grained pixel-level features with Transformers only. By stacking a cascade of encoder blocks with increasing receptive fields, Patcher can extract both local and global features effectively. We also proposed a new MoE-based decoder that uses a gating network to select a suitable set of expert features from the encoder to output the prediction for each pixel. The use of MoE enables better specializations of the expert features and reduces interference between them. Extensive experiments on stroke lesion and polyp segmentation indicate that our method can be applied to various medical image data. We hope the use of MoE in our work can provide a new perspective on Transformer-based architecture for medical imaging. We also look forward to extending Patcher to other medical diagnosis tasks, such as classification and landmark detection.

References

1. Cao, H., Wang, Y., Chen, J., Jiang, D., Zhang, X., Tian, Q., Wang, M.: Swin-Unet: Unet-like pure transformer for medical image segmentation. arXiv preprint arXiv:2105.05537 (2021)
2. Chen, J., Lu, Y., Yu, Q., Luo, X., Adeli, E., Wang, Y., Lu, L., Yuille, A.L., Zhou, Y.: TransUNet: Transformers make strong encoders for medical image segmentation. arXiv preprint arXiv:2102.04306 (2021)
3. Çiçek, Ö., Abdulkadir, A., Lienkamp, S.S., Brox, T., Ronneberger, O.: 3D U-Net: learning dense volumetric segmentation from sparse annotation. In: Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 424–432. Springer (2016)
4. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16×16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
5. Hatamizadeh, A., Tang, Y., Nath, V., Yang, D., Myronenko, A., Landman, B., Roth, H.R., Xu, D.: UNETR: Transformers for 3D medical image segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 574–584 (2022)

6. Huang, C.H., Wu, H.Y., Lin, Y.L.: HarDNet-MSEG: a simple encoder-decoder polyp segmentation neural network that achieves over 0.9 mean dice and 86 FPS. arXiv preprint arXiv:2101.07172 (2021)
7. Jha, D., Smedsrud, P.H., Riegler, M.A., Halvorsen, P., de Lange, T., Johansen, D., Johansen, H.D.: Kvasir-SEG: A segmented polyp dataset. In: Proceedings of the International Conference on Multimedia Modeling. pp. 451–462. Springer (2020)
8. Jha, D., Smedsrud, P.H., Riegler, M.A., Johansen, D., De Lange, T., Halvorsen, P., Johansen, H.D.: ResUNet++: An advanced architecture for medical image segmentation. In: Proceedings of the IEEE International Symposium on Multimedia (ISM). pp. 225–2255. IEEE (2019)
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
10. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin Transformer: Hierarchical vision transformer using shifted windows. Proceedings of the International Conference on Computer Vision (ICCV) (2021)
11. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
12. Oktay, O., Schlemper, J., Folgoc, L.L., Lee, M., Heinrich, M., Misawa, K., Mori, K., McDonagh, S., Hammerla, N.Y., Kainz, B., Glocker, B., Rueckert, D.: Attention U-Net: Learning where to look for the pancreas. arXiv preprint arXiv:1804.03999 (2018)
13. Ou, Y., Yuan, Y., Huang, X., Wong, K., Volpi, J., Wang, J.Z., Wong, S.T.: LambdaUNet: 2.5D stroke lesion segmentation of diffusion-weighted MR images. In: Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 731–741. Springer (2021)
14. Petit, O., Thome, N., Rambour, C., Themyr, L., Collins, T., Soler, L.: U-Net Transformer: Self and cross attention for medical image segmentation. In: International Workshop on Machine Learning in Medical Imaging. pp. 267–276. Springer (2021)
15. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 234–241. Springer (2015)
16. Srivastava, R.K., Greff, K., Schmidhuber, J.: Highway networks. arXiv preprint arXiv:1505.00387 (2015)
17. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in Neural Information Processing Systems **30** (2017)
18. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: SegFormer: Simple and efficient design for semantic segmentation with transformers. arXiv preprint arXiv:2105.15203 (2021)
19. Zhang, Y., Liu, H., Hu, Q.: Transfuse: Fusing transformers and cnns for medical image segmentation. In: Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 14–24. Springer (2021)
20. Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P.H., Zhang, L.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6881–6890 (2021)
21. Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., Liang, J.: UNet++: A nested U-Net architecture for medical image segmentation. In: Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, pp. 3–11. Springer (2018)

Supplementary Material for Patcher: Patch Transformers with Mixture of Experts for Precise Medical Image Segmentation

Yanglan Ou, Ye Yuan, Xiaolei Huang, Stephen T.C. Wong, John Volpi, James Z. Wang, Kelvin Wong

1 Details of the Transformer Blocks

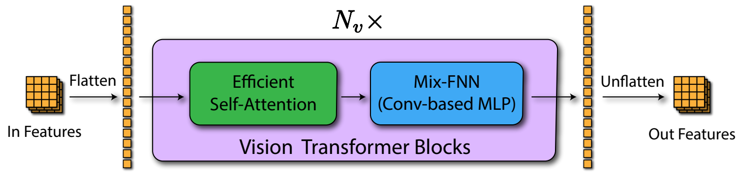

Fig. 1. Details of the Transformer blocks used in the Patcher block (Fig. 2 of the main paper). Efficient self-attention and Mix-FNN are adopted from SegFormer.

2 Visualization for Polyp Segmentation

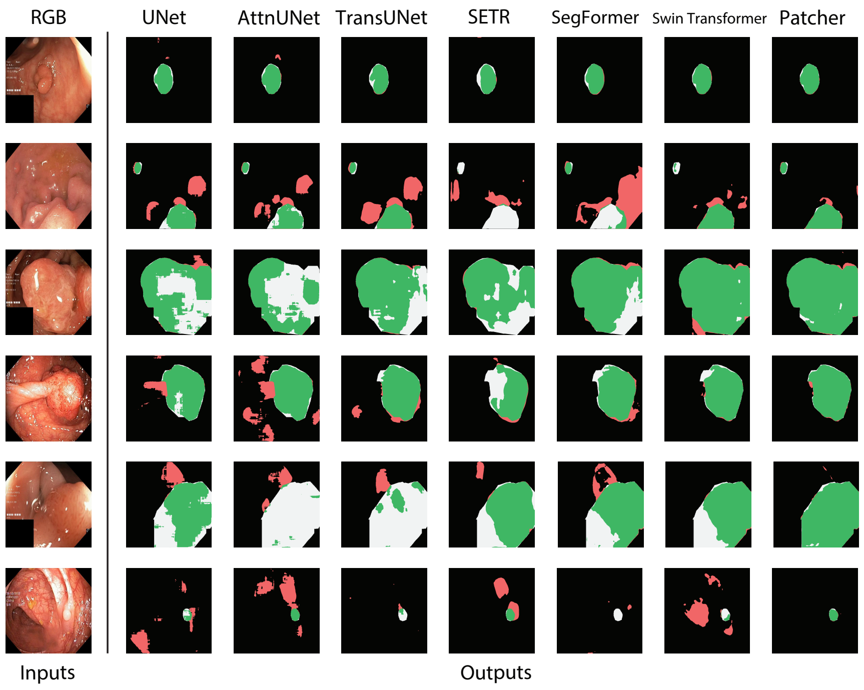

Fig. 2. Visualization of polyp segmentation. We highlight correct predictions (green), false positives (red), and false negatives (white) in the red box of the input.

3 Additional Visualization for Stroke Lesion Segmentation

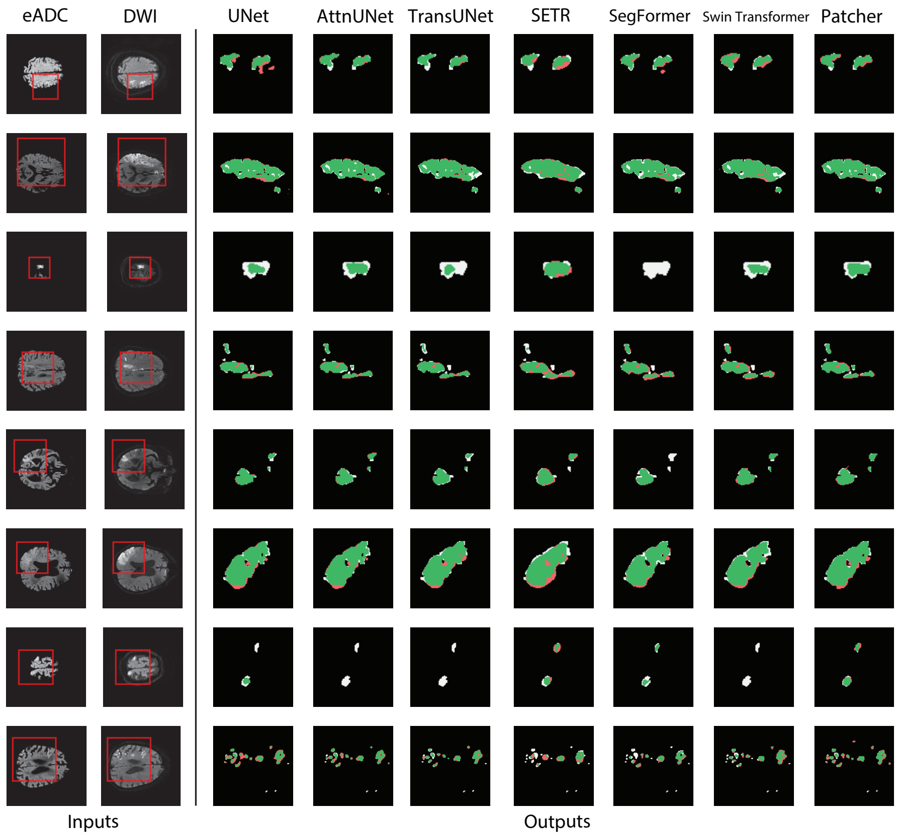

Fig. 3. Additional visualization of stroke lesion segmentation. We highlight correct predictions (green), false positives (red), and false negatives (white) in the red box.

4 Additional Visualization of the MoE Weights

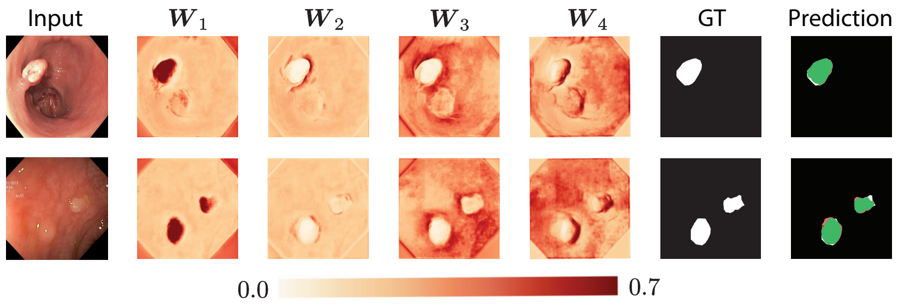

Fig. 4. Additional visualization of four MoE weight maps $[W_1, W_2, W_3, W_4]$ for polyp segmentation. We can observe that different weight maps focus on different areas.