E2ENet: Dynamic Sparse Feature Fusion for Accurate and Efficient 3D Medical Image Segmentation

Boqian Wu[1,2,*], **Qiao Xiao**[3,*], **Shiwei Liu**[4], **Lu Yin**[5], **Mykola Pechenizkiy**[3], **Decebal Constantin Mocanu**[2,3], **Maurice van Keulen**[1], **Elena Mocanu**[1]
[1] University of Twente, [2] University of Luxembourg, [3] Eindhoven University of Technology, [4] University of Oxford, [5] University of Surrey
{b.wu, m.vankeulen,e.mocanu}@utwente.nl
{q.xiao,m.pechenizkiy}@tue.nl, shiwei.liu@maths.ox.ac.uk,
l.yin@surrey.ac.uk, decebal.mocanu@uni.lu

Abstract

Deep neural networks have evolved as the leading approach in 3D medical image segmentation due to their outstanding performance. However, the ever-increasing model size and computational cost of deep neural networks have become the primary barriers to deploying them on real-world, resource-limited hardware. To achieve both segmentation accuracy and efficiency, we propose a 3D medical image segmentation model called Efficient to Efficient Network (E2ENet), which incorporates two parametrically and computationally efficient designs. i. Dynamic sparse feature fusion (DSFF) mechanism: it adaptively learns to fuse informative multi-scale features while reducing redundancy. ii. Restricted depth-shift in 3D convolution: it leverages the 3D spatial information while keeping the model and computational complexity as 2D-based methods. We conduct extensive experiments on AMOS, Brain Tumor Segmentation and BTCV Challenge, demonstrating that E2ENet consistently achieves a superior trade-off between accuracy and efficiency than prior arts across various resource constraints. E2ENet achieves comparable accuracy on the large-scale challenge AMOS-CT, while saving over 69% parameter count and 27% FLOPs in the inference phase, compared with the previous best-performing method. Our code has been made available at: https://github.com/boqian333/E2ENet-Medical.

1 Introduction

3D medical image segmentation plays an essential role in numerous clinical applications, including computer-aided diagnosis [Yu et al., 2020] and image-guided surgery systems [Ronneberger et al., 2015]. Over the past decade, the rapid development of deep neural networks has achieved tremendous breakthroughs and significantly boosted the progress in this area [Zhou et al., 2018, Huang et al., 2020, Isensee et al., 2021]. However, the model sizes and computational costs are also exploding, which deter their deployment in many real-world applications, especially for 3D models where resource consumption scales cubically [Hu et al., 2021, Valanarasu and Patel, 2022]. This naturally raises a research question: *Can we design a 3D medical image segmentation method that trades off accuracy and efficiency better, subjected to different resource availability?*

Accurately segmenting organs is a challenging task in 3D medical image segmentation due to the variability in size and shape even among the same type of organ, caused by factors such as patient anatomy and disease stage. This diversity amplifies the difficulty to accurately identify and distinguish

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

[*]Equal contribution.

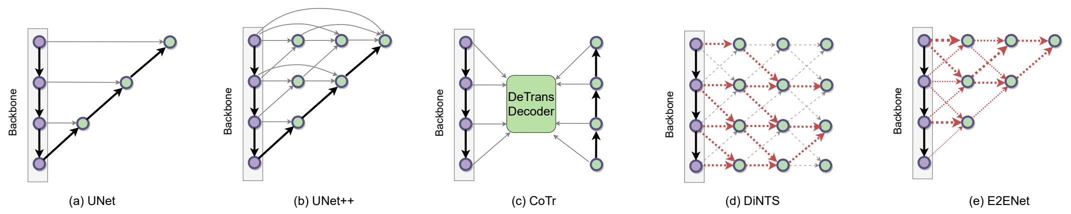

Figure 1: A comparison of feature fusion schemes. The purple nodes depict features extracted from the backbone, while the green nodes depict the fused features. In particular, in DiNTS (d), red lines indicate information flow paths determined through neural architecture search techniques. In E2ENet (e), the red lines with different widths represent sparse information flows determined by the DSFF mechanism, allowing for efficient feature fusion. E2ENet is capable of dynamically learning how many of the features to fuse are derived from the backbone.

the boundaries of different organs, leading to potential errors in segmentation. One of the main solutions for accurate medical image segmentation is to favorably leverage the multi-scale features extracted by the backbone network, but this remains a long-standing problem. Pioneering work UNet [Ronneberger et al., 2015] utilizes skip connection to propagate detailed information to high-level features. More recently, UNet++ [Zhou et al., 2018], CoTr [Xie et al., 2021], and DiNTS [He et al., 2021] have introduced more complex neural network architectures (e.g., dense skip connection and attention mechanism) and optimization techniques (e.g., neural architecture search (NAS) [Elsken et al., 2019]) for cross-scale feature fusion. NAS-based methods search for network topology and operators (e.g., 3x3 Convolution and Maxpool) and subsequently optimize model weights. However, these methods typically require substantial computational resources to explore network topologies and evaluate numerous candidate architectures, making them time-consuming. For instance, C2FNAS [Yu et al., 2020] requires nearly one GPU-year to discover a single 3D segmentation architecture, while DiNTS [He et al., 2021] improves search efficiency but still needs 5.8 GPU days to find a single architecture. In contrast to NAS approaches, our method does not require costly architecture search time and instead directly optimizes and searches for sparse topologies within the predefined architecture.

In this paper, we propose the **Efficient to Efficient Network (E2ENet)**, a model that efficiently incorporates both bottom-up and top-down features from the backbone network in a dynamically sparse pattern, achieving an improved accuracy-efficiency trade-off. As shown in Figure 1 (e), E2ENet incorporates multi-scale features from the backbone into the final output by gradually fusing adjacent features, allowing the network to fully utilize information across various scales. To prevent unnecessary information aggregation, we propose a **dynamic sparse feature fusion (DSFF)** mechanism and embed it in each fusion node. The DSFF mechanism adaptively integrates informative multi-scale features and filters out unnecessary ones during the course of the training process, significantly reducing the computational overhead without sacrificing performance. Additionally, to further improve efficiency, our E2ENet employs a **restricted depth-shift** strategy within 3D convolutions, derived from the temporal shift [Lin et al., 2019] and 3D-shift [Fan et al., 2020] strategies used in efficient video action recognition. This allows the 3D convolution operation with a kernel size of $(1, 3, 3)$ to capture 3D spatial relationships while maintaining the parameter complexity of 2D convolutions and reducing computational cost. To evaluate the performance of our proposed E2ENet, we conducted extensive experiments on the AMOS-CT [Ji et al., 2022], Brain Tumor Segmentation in the Medical Segmentation Decathlon (MSD) [Antonelli et al., 2022], and Multi-Atlas Labeling Beyond the Cranial Vault (BTCV) [Landman et al., 2015] challenges. We found that E2ENet can effectively trade-off between segmentation accuracy and efficiency compared to both convolution-based and transformer-based architectures. In particular, on the AMOS-CT challenge, E2ENet achieves competitive accuracy with a mDice of 90.0%, while being 69% smaller and using 27% fewer FLOPs during inference. With the DSFF mechanism filtering out 90% of feature connections, E2ENet further reduces resource costs without significantly sacrificing accuracy.

2 Methodology

In this section, we first present the overall architecture of E2ENet, which allows for the fusion of multi-scale features from three directions. Next, we describe the proposed DSFF mechanism, which adaptively selects informative features and filters out unnecessary ones during training. Lastly, to further increase efficiency, we introduce the use of restricted depth-shift in 3D convolution.

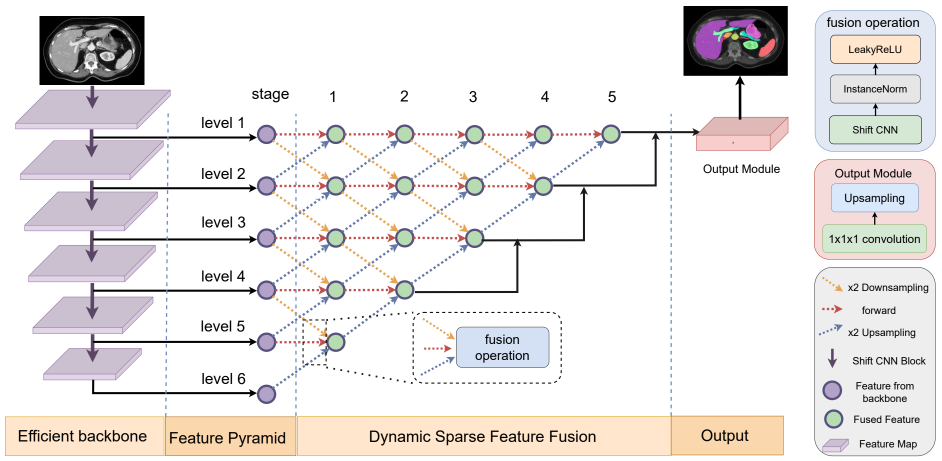

Figure 2: The overall architecture of the proposed E2ENet consists of a CNN backbone that extracts multiple levels of features. These features are then gradually aggregated through several stages, during which the multi-scale features are fused using a fusion operation.

2.1 The Architecture

Figure 2 provides an overview of the proposed architecture. Given the input 3D image $I_{in} \in \mathbb{R}^{D\times H\times W}$, the CNN backbone extracts feature maps at multiple scales, represented by $\mathbf{x}^0 = (\mathbf{x}^{0,1}, \mathbf{x}^{0,2}, \ldots, \mathbf{x}^{0,L})$, where L is the total number of feature scales. The feature at level i, $\mathbf{x}^{0,i}$ is a tensor with dimensions $d_i \times h_i \times w_i \times c_i$, where d_i, h_i, w_i and c_i represent the depth, height, width, and number of channels of the feature maps at level i, respectively. It is worth mentioning that the spatial resolution of the feature maps decreases as the level increases, while the number of channels increases up to a maximum of 320. To fully exploit the hierarchical features extracted by the CNN backbone, these features are aggregated across multiple stages, as depicted in Figure 2 (Dynamic Sparse Feature Fusion section). At stage j $(0 < j \leq L-1)$, the features at level i are obtained by fusing the adjacent features from the previous stage along three directions:

1. "*Downward flow*" (in yellow): The high-resolution feature $\mathbf{x}^{j-1,i-1}$, which provides richer visual details, is passed downward; 2. "*Upward flow*" (in blue): The low-resolution feature $\mathbf{x}^{j-1,i+1}$, which captures more global context, is passed upward; 3. "*Forward flow*" (in red): The features $x^{j-1,i}$, which maintain their spatial resolution, are passed forward for further information integration. The fused cross-scale feature maps at the i-th level of the j-th stage can be formulated as:

$$\mathbf{x}^{j,i} = \begin{cases} \mathcal{F}^{j,i}([\mathbf{x}^{j-1,1}, \mathcal{U}(\mathbf{x}^{j-1,2})]), & i = 1; \\ \mathcal{F}^{j,i}([\mathcal{D}(\mathbf{x}^{j-1,i-1}), \mathbf{x}^{j-1,i}, \mathcal{U}(\mathbf{x}^{j-1,i+1})]), & others, \end{cases} \tag{1}$$

where $\mathcal{F}^{j,i}(.)$ is a fusion operation, consisting of a convolution operation followed by Instance Normalization (IN) [Ulyanov et al., 2016] and a Leaky Rectified Linear Unit (LeakyReLU) [Maas et al., 2013]. $\mathcal{U}(.)$ and $\mathcal{D}(.)$ denote the up-sampling and down-sampling, respectively. $[.]$ denotes the concatenation operation. In the convolution operation, the input feature maps, which have a channel number of $C_{in}^{j,i}$ ($C_{in}^{j,i} = c_{i-1} + c_i + c_{i+1}$ or $c_i + c_{i+1}$), are fused and processed to produce output feature maps with a channel number of $C_{out}^{j,i}$ ($C_{out}^{j,i} = c_i$). Unlike the UNet++ model, which only considers bottom-up information flows for image segmentation, our proposed E2ENet architecture incorporates both bottom-up and top-down information flows. This allows E2ENet to **make use of both high-level contextual information and fine-grained details** in order to produce more accurate segmentation maps (experimental results can be found in Table 2).

2.2 Dynamic Sparse Feature Fusion

Such multi-stage cross-level feature propagation provides a more comprehensive understanding of the images but can also introduce redundant information, necessitating careful handling in feature fusion to ensure efficient interaction between scales or stages. Our proposed Dynamic Sparse Feature Fusion

(DSFF) mechanism addresses the issue of multi-scale feature fusion in an intuitive and effective way. It optimizes the process by enabling **selective and adaptive use of features from different levels during training**. This results in a more efficient feature fusion process with lower computational and memory overhead.

The DSFF mechanism is applied in each fusion operation, allowing the fusion operation $\mathcal{F}^{j,i}(\cdot)$ to select the informative feature maps from the input fused features. The feature map selection process is controlled by a binary mask $\mathbf{M}^{j,i}(\cdot) \in \{0,1\}^{C_{in}^{j,i} \times C_{out}^{j,i}}$, which are trained to filter out S unnecessary feature map connections by zeroing out the corresponding kernels in $\mathcal{F}^{j,i}(\cdot)$. With the DSFF mechanism, the output of the fusion operation is then computed as:

$$\mathbf{x}^{j,i}_{c_{out},:,:,:} = \sigma(\mathcal{I}(\sum_{c=0}^{C_{in}^{j,i}}(\tilde{\mathbf{x}}^{j,i}_{c,:,:,:} * (\mathbf{M}^{j,i}_{c,c_{out}} \cdot \theta^{j,i}_{c,c_{out},:,:,:})))), \tag{2}$$

where $\tilde{\mathbf{x}}^{j,i}_{c,:,:,:}$ is the input fused feature map at the c-th channel. $\theta^{j,i}_{c,c_{out},:,:,:}$ is the kernel (feature map connection, as in Figure 3) that connects the c-th input feature map to c_{out}-th output feature map, $*$ is a convolution operation, $\cdot$ is the product of a scalar (i.e., $\mathbf{M}^{j,i}_{c,c_{out}}$) and a matrix (i.e., $\theta^{j,i}_{c,c_{out},:,:,:}$) [2].

$\mathbf{M}^{j,i}_{c,c_{out}}$ denotes the existence or absence of a connection between the c-th input feature map and the c_{out}-th output feature map. $\mathcal{I}(.)$ and $\sigma(.)$ denote the Instance Normalization and LeakyReLU, respectively. Thus, the core of DSFF mechanism is the learning of binary masks during the training. At initialization, each binary mask is initialized randomly, with the number of non-zero entries $\|\mathbf{M}^{j,i}\|_0$ equals $(1-S) \times C_{in}^{j,i} \times C_{out}^{j,i}$. Here, S $(0 < S < 1)$ is a hyperparameter called the feature sparsity level, which determines the percentage of feature map connections that are inactivated. The activated connections can be updated throughout the course of training, while the feature sparsity level S remains constant. This enables efficiency in both testing and training, as the exploitation of connections remains sparse throughout the training process.

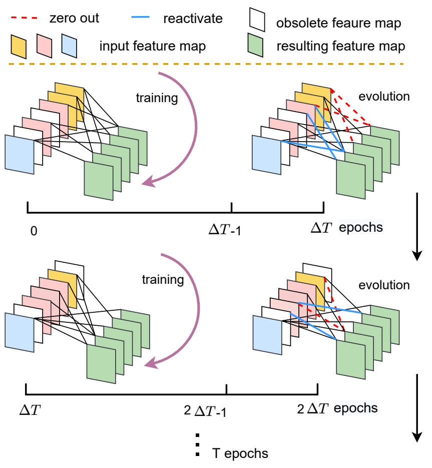

Figure 3: Illustration of our Dynamic Sparse Feature Fusion (DSFF) mechanism. The fusion operation starts from sparse feature connections and allows the connectivity to be evolved after training for ΔT epochs. During each evolution stage, a fraction of kernels with smaller L_1 norms will be zeroed out (red dotted line), while the same fraction of other inactivated connections will be reactivated randomly, keeping the feature sparsity S constant during training (blue solid line).

Every ΔT training epoch, the activated connections with lower importance are removed, while the same number of deactivated connections are randomly reactivated, as shown in Figure 3. The importance of activated connection is determined by the L_1 norm of the corresponding kernel. That is, for $\mathbf{M}^{j,i}$, the importance score of connection between c_{in}-th input feature and c_{out}-th output is $\|\theta^{j,i}_{c_{in},c_{out},:,:,:}\|_1$. Our intuition is simple: if a feature map connection is more important (i.e., has a larger effect on output accuracy) than others, the L_1 norm of the corresponding kernel should be larger. However, the importance score is suboptimal during training, as the kernels are gradually optimized. Randomly reactivating previous "obsolete" connections with re-initialization can avoid unrecoverable feature map abandonment and thoroughly explore the representative feature maps that contribute to the final performance.

The DSFF mechanism allows for the exploration and exploitation of multi-scaled sparse features through a sparse-to-sparse training method. Our method selects features at multiple levels, as opposed to relying solely on feature selection at the input layer [Sokar et al., 2022, Atashgahi et al., 2022]. Additionally, it is a plastic approach to fuse features that can adapt to changing conditions

[2]The computation cost of the masking operation is negligible. For instance, consider the feature map $\mathbf{x}^{j,i}_{c,:,:,:}$ with size $D \times H \times W$, the kernel $\theta^{j,i}_{c,c_{out},:,:,:}$ with size of $3 \times 3 \times 3$, and the number of kernels as $C_{in} \times C_{out}$. The computation cost of the masking operation alone is $C_{in} \times C_{out}$, whereas the combined cost of the masking operation and convolution operation is $D \times H \times W \times 3 \times 3 \times 3 \times C_{in} \times C_{out}$.

during training, as opposed to static methods that rely on one-shot feature selection, as shown in the comparison of dynamic and static sparsity in Table 2 of [Mostafa and Wang, 2019]. More details of the training process are elaborated in the Algorithm 1 in Appendix A.3.

2.3 Restricted Depth-Shift in 3D Convolution

In 3D medical image segmentation, the 2D-based methods (such as the 2D nnUNet [Isensee et al., 2021]), which apply the 2D convolution to each slice of the 3D image, are computationally efficient but can not fully capture the relationships between slices. To overcome this limitation, we take inspiration from the temporal-shift [Lin et al., 2019] and 3D-shift [Fan et al., 2020] in efficient video action recognition, and the axial-shift [Lian et al., 2021] in efficient MLP architecture for vision. Our proposed E2ENet incorporates a depth-shift strategy in 3D convolution operations, which facilitates inter-slice information exchange and **captures 3D spatial relationships while retaining the simplicity and computational efficiency of 2D-based method**. Temporal-shift [Lin et al., 2019] requires selecting a Shift Proportion (the proportion of channels to conduct temporal shift). Axial-shift [Lian et al., 2021] and 3D-Shift optimize the learnable 3D spatiotemporal shift. We have made refinements to the channel shifting technique by shifting along the depth dimension and incorporating constraints on the shift size. This adaptation is thoughtfully designed to align with the distinctive needs of sparse models employed in medical image segmentation.

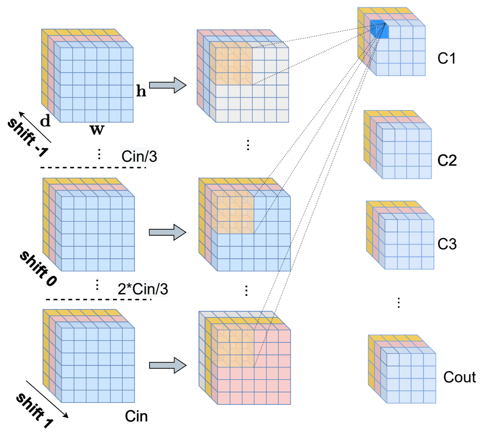

Figure 4: Illustration of restricted depth-shift in 3D Convolution of our E2ENet. The input features (left) are firstly split into 3 parts along the channel dimension, and then shifted by $\{-1, 0, 1\}$ units along the depth dimension respectively (middle). After that, 3D CNNs with kernel size $1\times3\times3$ are performed on the feature maps (middle) to generate the output features (right).

In our method, we employ a simple depth shift technique by shifting all channels while constraining the shift size to be either +1, 0 or -1, as shown in Figure 4. This choice is motivated using dynamic sparse feature fusion, where the feature maps contain sparse information. If the shift magnitude is too large, it can result in an insufficient representation of channels or an excessive representation of depth information, which can have a negative impact on the effectiveness of the shift operation (experimental results can be found in Table 3).

3 Experiments

In this section, we compare the performance of our E2ENet model to baseline methods and report results in terms of both segmentation quality and efficiency. In addition, we will perform ablation studies to investigate the behavior of each component in the E2ENet model. To further analyze the performance of our model, we will examine its generalizability and the effect of model capacity, and present qualitative results by visualizing the predicted segmentations on sample images. We will also visualize the feature fusion ratios to gain insights into which features play an important role in the segmentation process. The description of the dataset and experimental setup can be found in Appendix A.4 and A.5.

Table 1: Quantitative comparisons of segmentation performance on the validation set of AMOS-CT dataset. * denotes the results with postprocessing.

Methods	mDice (%) ↑	Params (M) ↓	FLOPs [1] (G) ↓	PT score ↑	mNSD (%) ↑
CoTr	77.1	41.87	1510.53	1.07	64.2
nnFormer	85.6	150.14	1343.65	1.12	74.2
UNETR	78.3	93.02	**391.03**	1.41	61.5
Swin UNETR	86.4	62.83	1562.99	1.14	75.3
VNet	82.0	45.65	1737.57	1.10	67.9
nnUNet	90.0	30.76	1067.89	1.30	82.1
nnUNet*	**90.5**	30.76	1067.89	1.31	**83.0**
E2ENet* (s=0.7)	90.3	11.23	969.32	1.54	82.7
E2ENet* (s=0.8)	90.3	9.44	778.74	1.65	82.5
E2ENet* (s=0.9)	89.9	**7.64**	492.29	**1.89**	81.8
E2ENet (s=0.7)	90.1	11.23	969.32	1.54	82.3
E2ENet (s=0.8)	90.0	9.44	778.74	1.65	82.3
E2ENet (s=0.9)	89.6	**7.64**	492.29	1.88	81.4

[1] The inference FLOPs are calculated based on the patch sizes of $1 \times 128 \times 128 \times 128$ without considering postprocessing cost.

3.1 Comparison with SOTA methods

AMOS-CT Challenge. To comprehensively validate our method, we compare it to several state-of-the-art CNN-based models (e.g. nnUNet [Isensee et al., 2021], and Vnet [Milletari et al., 2016]) and transformer-based models (e.g. CoTr [Xie et al., 2021], nnFormer [Zhou et al., 2021], UNETR [Hatamizadeh et al., 2022], and Swin UNETR [Tang et al., 2022]). We record the mDice (class-wise Dice can be found in Table 10 of Appendix), Params, inference FLOPs, and PT score [3] on the validation set [4] of the AMOS-CT challenge in Table 1. E2ENet with a feature sparsity (s) of 0.8 achieves comparable performance, with a mDice of 90.3%, while being 3× smaller in model size and requiring less computational cost during the inference phase compared to the top-performing lightweight model, nnUNet. As the feature sparsity of E2ENet increases, the number of model parameters and inference FLOPs can be further reduced without significantly compromising segmentation performance. This indicates that there is potential to trade-off performance and efficiency by adjusting the feature sparsity of E2ENet. We also report the results of mean normalized surface dice (mNSD), the official segmentation metric for the AMOS challenge, to provide supplementary information on boundary segmentation quality.

BraTS Challenge in MSD. E2ENet demonstrates superior performance in terms of mDice compared to other state-of-the-art methods (DiNTS [He et al., 2021], UNet++ [Zhou et al., 2018] and nnUNet [Isensee et al., 2021]). Additionally, it is a resource-efficient network that is competitive with the baselines, as evidenced by its small model size and low inference FLOPs. Specifically, E2ENet model a feature sparsity level of 90% has only 7.63 M parameters, which is significantly smaller than other models yet still outperforms them in terms of mDice. The results on the BraTS dataset, which are MRI images, further validate the effectiveness and efficiency of E2ENet across both CT and MRI medical image analysis.

Table 2: 5-fold cross-validation of segmentation performance on the BraTS Challenge training set in the MSD. Note: ED, ET, and NET denote edema, enhancing tumor, and non-enhancing tumor, respectively.

Methods	Dice (%) ↑			mDice	Params	FLOPs [1]	PT score
	ED	ET	NET	(%) ↑	(M) ↓	(G) ↓	↑
DiNTS	80.2	61.1	77.6	73.0	/	/	/
UNet++	80.5	<u>62.5</u>	79.2	74.1	58.38	3938.25	1.12
nnUNet	<u>81.0</u>	62.0	79.3	74.1	31.20	1076.62	1.35
E2ENet ($S = 0.7$)	**81.2**	**62.7**	**79.5**	**74.5**	11.24	1067.06	1.57
E2ENet ($S = 0.8$)	<u>81.0</u>	<u>62.5</u>	79.0	74.2	<u>9.44</u>	<u>780.97</u>	<u>1.72</u>
E2ENet ($S = 0.9$)	80.9	<u>62.5</u>	<u>79.4</u>	<u>74.3</u>	**7.63**	**494.52**	**2.00**

[1] The inference FLOPs are calculated based on patch sizes of $4 \times 128 \times 128 \times 128$. The number of parameters and inference FLOPs for DiNTS are not reported because calculating them is time-consuming. This is due to the fact that the architecture for DiNTS is not readily available for the dataset and must be found using Neural Architecture Search (NAS).

Due to the limited space and the similarity between the BTCV and AMOS-CT challenges, both of which involve multi-organ segmentation in CT images, the results of the **BTCV challenge** are provided in Appendix A.8.2.

3.2 Ablation Studies

In this section, we investigate the impact of two factors on the performance of E2ENet: (i) the DSFF mechanism and (ii) restricted depth-shift in 3D convolution. Specifically, we consider the following scenarios: (#1) w/ DSFF: DSFF mechanism is used to dynamically activate feature map, otherwise, all feature maps are activated during training; (#2) w/ shift: restricted depth/height/width shifting is applied on feature maps before the convolution operation, otherwise, the convolution is performed directly as a standard 3D convolution without shifting operation.

Table 3: Ablation study on the effects of the DSFF mechanism and restricted depth-shift in 3D convolution on the validation set of the AMOS-CT Challenge.

w/ DSFF	ΔT (# iters)	w/ shift	shift size	kernel size	mDice (%) ↑	Params (M) ↓	FLOPs (G) ↓	mNSD (%) ↑
✗	/	✓	$(-1, 0, 1)$	$(1 \times 3 \times 3)$	**90.2**	23.90	3069.55	82.6
✗	/	✗	/	$(1 \times 3 \times 3)$	88.6	23.90	3069.55	78.6
✓	1200	✓	$(-1, 0, 1)$	$(1 \times 3 \times 3)$	<u>90.1</u>	11.23	969.32	82.3
✓	1200	✗	/	$(1 \times 3 \times 3)$	88.2	11.23	969.32	79.4
✓	1200	✓	$(-2, 0, 2)$	$(1 \times 3 \times 3)$	89.8	11.23	969.32	82.0
✓	1200	✓	$(-3, 0, 3)$	$(1 \times 3 \times 3)$	89.7	11.23	969.32	81.6
✓	1200	✓	$(-7, 0, 7)$	$(1 \times 3 \times 3)$	87.6	11.23	969.32	77.5
✗	1200	✗	/	$(3 \times 3 \times 3)$	90.2	52.54	4511.57	82.5
✓	1200	✗	/	$(3 \times 3 \times 3)$	90.1	27.97	1778.55	82.1
✓	600	✓	$(-1, 0, 1)$	$(1 \times 3 \times 3)$	90.0	11.23	969.32	82.2
✓	1800	✓	$(-1, 0, 1)$	$(1 \times 3 \times 3)$	89.9	11.23	969.32	82.0

[3] The explanation of metrics can be found in Appendix A.7.

[4] The validation set includes 100 images and was the previous test set for the first stage of the challenge. In the training of E2ENet, we did not use the validation set and treated it as the test set.

Effect of DSFF Mechanism. Table 3 (rows 1 and 3) demonstrates that the E2ENet with the DSFF mechanism, achieved comparable performance while using three times fewer parameters and inference FLOPs. Table 4 shows that removing the DSFF mechanism caused the mDice score to drop from 74.5 % (row 4) to 74.1 % (row 2) on the BraTS challenge, while also increasing the number of parameters by more than 2× and the inference FLOPs by nearly 3×. These ablation study results highlight that dynamic sparse feature fusion (DSFF) helps reduce resource costs while maintaining segmentation performance.

To further investigate the impact of the DSFF mechanism, we compared E2ENet with other multi-scale medical image segmentation methods on the AMOS-CT validation dataset, as shown in Table 5. These methods include DeepLabv3 [Chen et al., 2017], CoTr [Xie et al., 2021], and MedFormer [Gao et al., 2022]. DeepLabv3 uses atrous convolution to capture multi-scale context, CoTr integrates multi-scale features using attention, and MedFormer employs all-to-all attention for comprehensive multi-scale fusion, addressing both semantic and spatial aspects.

Table 4: Ablation study on the effects of the DSFF mechanism and restricted depth-shift in 3D convolution, evaluated through 5-fold cross-validation of segmentation performance on the BraTS Challenge training set in the MSD.

w/ DSFF	w/ shift	kernel size	Dice (%) ↑ ED	ET	NET	mDice (%) ↑	Params (M) ↓	FLOPs (G) ↓
✗	✗	(1 × 3 × 3)	80.4	62.4	79.0	73.9	23.89	3071.78
✗	✓	(1 × 3 × 3)	81.0	62.3	79.0	74.1	23.89	3071.78
✓	✗	(1 × 3 × 3)	80.3	62.5	79.0	73.9	11.24	1067.06
✓	✓	(1 × 3 × 3)	**81.2**	**62.7**	**79.5**	**74.5**	11.24	1067.06
✗	✗	(3 × 3 × 3)	80.9	61.9	79.1	74.0	52.55	4519.26
✓	✗	(3 × 3 × 3)	81.0	62.2	79.1	74.1	28.02	2023.52
✓	✗	(3 × 1 × 3)	80.3	61.7	78.7	73.6	11.24	1067.06
✓	✗	(3 × 3 × 1)	80.5	61.9	78.7	73.7	11.24	1067.06
✓	✓	(3 × 1 × 3)	81.1	62.4	78.9	74.1	11.24	1067.06
✓	✓	(3 × 3 × 1)	81.0	62.3	79.4	74.2	11.24	1067.06

We observed that with a sparsity ratio of 0.9, E2ENet outperforms DeepLabv3 in terms of mDice while significantly reducing computational and memory costs by more than 3× and nearly 10×, respectively. Additionally, E2ENet with a sparsity ratio of 0.8 matches MedFormer's performance while significantly reducing computational and memory costs by 3× and 4×, respectively. This demonstrates the benefits of multi-scale feature aggregation for medical image segmentation, with E2ENet's DSFF mechanism being much more efficient than other multi-scale methods.

Moreover, as shown in Table 2, E2ENet with the DSFF mechanism outperforms other feature fusion architectures, such as UNet++ and DiNTS, on the BraTS challenge. Overall, the results from the BraTS and AMOS-CT challenges demonstrate that the DSFF mechanism provides an effective and efficient approach for feature fusion.

Finally, we also studied the impact of topology update frequency (ΔT) in the DSFF mechanism and observed that our algorithm is not sensitive to the hyperparameter ΔT.

Effect of Restricted Depth-Shift in 3D Convolution. We evaluated the effectiveness of our restricted shift strategy by comparing the performance of E2ENet and its variants. Without restricted depth-shift, the mDice decreased from 74.5% (row 4) to 73.9% (row 3) for the BraTS challenge as shown in Table 4, and decreased from 90.1% (row 4) to 88.2% (row 3) for AMOS-CT (as shown in Table 3). In Table 3, when comparing E2ENet with kernel sizes of 3x3x3 (rows 8 and 9) to E2ENet with kernel sizes of 1x3x3 combined with a depth shift (rows 1 and 3), their segmentation accuracy remains the same, whether DSFF is used or not. In Table 4, we find that E2ENet with kernel sizes of 1x3x3 combined with a depth shift (rows 2 and 4) even improves segmentation accuracy, whether without DSFF or with DSFF, when compared to kernel sizes of 3x3x3 (rows 5 and 6). This further demonstrates that our proposed efficient Restricted Depth-Shift 3D Convolutional layer, which utilizes a 1x3x3 kernel with restricted depth shift, is equivalent to a 3x3x3 kernel in terms of segmentation accuracy. Moreover, it offers significant savings in computational and memory resources. This demonstrates that the use of a 1x3x3 kernel with restricted depth shift is functionally equivalent to a 3x3x3 kernel in terms of segmentation accuracy, all the while offering savings in computational and memory resources.

Table 5: Comparison with other multi-scale medical image segmentation methods on the AMOS-CT validation dataset. The mDice score for MedFormer is sourced from [Gao et al., 2022].

method	mDice (%) ↑	Params (M) ↓	FLOPs (G) ↓
DeepLabv3	89.0	74.68	1546.25
CoTr	77.1	41.87	1510.53
MedFormer	**90.1**	39.59	2332.75
E2ENet (S=0.8)	90.0	9.44	778.74
E2ENet (S=0.9)	89.6	**7.64**	**492.29**

Rows 9 and 10 of Table 4 demonstrate that when restricted shift is applied to the height or width dimensions, the model's performance decreases compared to E2ENet (4th row) with restricted shift on the depth dimension, called as restricted depth-shift. We also observed that the performance of 3D convolution with kernel sizes of $1 \times 3 \times 3$ (row 3), $3 \times 1 \times 3$ (row 7), and $3 \times 3 \times 1$ (row 8) decreased significantly without restricted shift compared to their counterparts with restricted shift on the corresponding dimensions (rows 4, 9, and 10, respectively). These phenomena demonstrate the effectiveness of restricted shift, especially restricted depth-shift, for medical image segmentation.

Furthermore, we evaluated the impact of different shift sizes on the model's performance. As shown in Table 3, we compared the performance of E2ENet with shift sizes of (-1, 0, 1) to (-2, 0, 2), (-3, 0, 3) and (-7, 0, 7), and observed a decrease in performance from 90.1% to 89.8%, 89.7% and 87.6%, respectively, as the shift size increased. Increasing the shift size means considering more depth-wise information at the expense of channel-wise information, leading to an insufficient representation of channels, as discussed in Section 2.3. Additionally, a large shift size (equivalent to a large kernel size) leads to a loss of local spatial relationships, which are crucial for segmentation. This results in a blurring effect that reduces the precision of boundary alignment, particularly affecting metrics like mNSD, which rely heavily on accurate boundary information. Therefore, we use a shift size of (-1, 0, 1) in our restricted depth-shift strategy as the default setting in our experiments. Finally, we plot a critical distance diagram to show the statistical significance of our modules in Appendix A.8.3.

Table 6: Evaluating the generalizability of E2ENet on the AMOS-MRI dataset. CT $\rightarrow$ MRI: pretrained on the CT dataset and fine-tuned on the MRI dataset; MRI: trained solely on the MRI dataset; CT+MRI: trained on both the CT and MRI datasets. Results for all models, except E2ENet and nnUNet (CT $\rightarrow$ MRI), are sourced from the AMOS website.

method	mDice (%) ↑			mNSD (%) ↑		
	CT → MRI	MRI	CT + MRI	CT → MRI	MRI	CT + MRI
E2ENet	**87.8**	/	/	**83.9**	/	/
nnUNet	87.4	87.1	87.7	83.3	83.1	82.7
VNet	/	83.9	75.4	/	65.8	65.6
nnFormer	/	80.6	75.48	/	74.0	66.63
CoTr	/	77.5	73.46	/	78.0	65.35
Swin UNTER	/	75.7	77.52	/	65.3	69.10
UNTER	/	75.3	/	/	70.1	/

3.3 Generalizability Analysis

To evaluate the generalizability of the resulting architectures, we compared our model, pre-trained on AMOS-CT and fine-tuned on AMOS-MRI, with nnUNet, which was pre-trained on AMOS-CT and fine-tuned on AMOS-MRI, and both models are fine-tuned for 250 epochs. It is worth noting that the topology (weight connections) is determined through AMOS-CT pre-training and remains fixed during the fine-tuning phase. From Table 6, we discovered that the E2ENet architecture, initially designed for AMOS-CT, can be effectively applied to AMOS-MRI as well. This adaptation resulted in improved performance compared to nnUNet and other baselines. nnUNet and these baselines were either transferred from CT, trained solely on MRI data, or jointly on MRI and CT datasets.

Table 7: Quantitative comparisons of segmentation performance on AMOS-CT test dataset. ‡ and † denote the results with and without postprocessing that are reproduced by us. * indicates the results with postprocessing.

Methods	mDice (%) ↑	Params (M) ↓	FLOPs (G) ↓	PT score ↑	mNSD (%) ↑
CoTr	80.9	41.87	1510.53	1.11	66.3
nnFormer	85.6	150.14	1343.65	1.11	72.5
UNETR	79.4	93.02	**391.03**	1.41	60.8
Swin-UNETR	86.3	62.83	1562.99	1.13	73.8
VNet	82.9	45.65	1737.57	1.11	67.6
nnUNet†	90.6	30.76	1067.89	1.31	82.0
nnUNet‡	**91.0**	30.76	1067.89	1.31	82.6
E2ENet* (s=0.7)	90.7	11.23	969.32	1.54	82.2
E2ENet* (s=0.8)	90.7	9.44	778.74	1.65	82.1
E2ENet* (s=0.9)	90.4	**7.64**	492.29	**1.89**	81.4
E2ENet (s=0.7)	90.6	11.23	969.32	1.54	82.0
E2ENet (s=0.8)	90.4	9.44	778.74	1.65	81.8
E2ENet (s=0.9)	90.1	**7.64**	492.29	**1.89**	80.7

In the AMOS-CT dataset, there is a domain shift between the training and test datasets due to variations in the image acquisition scanners [Ji et al., 2022]. Thus, we also assess the generalizability of our approach by evaluating its performance on the AMOS-CT test dataset. As demonstrated in Table 7, E2ENet exhibits comparable performance even in the presence of domain shift when compared to nnUNet. This is achieved while maintaining lower computational and memory costs.

3.4 Model Capacity Analysis

To ensure a fair comparison, we scale down nnUNet by reducing the number of channels at level 1 from 32 to 27 and decreasing the total number of feature scales from 5 to 4, resulting in a modified version referred to as nnUNet (-). Additionally, we scale up E2ENet by increasing the number of channels at level 1 from 48 to 58, creating a modified version referred to as E2ENet (+). We evaluated the performance of these models, and the results can be found in Table 8. It is worth noting that scaling down nnUNet resulted in decreased performance in terms of mDice and mNSD, while scaling up E2ENet led to an increase in mDice and comparable performance in terms of mNSD. This indicates that the comparable performance of the memory and computation-efficient E2ENet is not attributed to the dataset's requirement for a small number of parameters and computations.

3.5 Qualitative Results

In this section, we compare the proposed E2ENet and nnUNet qualitatively on three challenges. To make the comparison easier, we highlight detailed segmentation results with red dashed boxes.

Table 8: A comparison under a similar FLOPs budget on the AMOS-CT challenge in two cases: when nnUNet is scaled down and when E2ENet is scaled up.

Method	mDice (%) $\uparrow$	Params (M) $\downarrow$	FLOPs (G) $\downarrow$	mNSD (%) $\uparrow$
E2ENet (S=0.8)	90.0	**9.43**	778.74	**82.3**
nnUNet	90.0	31.20	1067.89	82.1
nnUNet (-)	89.7	12.96	**755.79**	81.9
E2ENet (+)	**90.4**	10.37	1119.17	82.2

Figure 5 presents a qualitative comparison of our proposed E2ENet with nnUNet on the AMOS-CT challenge. As a widely used baseline, nnUNet has been evaluated on multiple medical image segmentation challenges. Our results demonstrate that, on certain samples, E2ENet can improve segmentation quality and overcome some of the challenges faced on the AMOS-CT challenge. For example, as shown in the first column, E2ENet more accurately distinguishes between the stomach and the esophagus, which is challenging due to their close proximity. In the second column, E2ENet better differentiates the duodenum from the background, while in the third column, E2ENet accurately identifies the precise boundaries of the liver, a structure that is prone to over-segmentation. These examples demonstrate the potential of our proposed method to improve the accuracy of medical image segmentation to some extent, especially in challenging cases such as distinguishing between closely located organs or accurately segmenting complex shapes. The Qualitative Results of BTCV and BraTS Challenge can be found in Appendix A.9.1.

3.6 Feature Fusion Visualization

In this section, we explore how the DSFF mechanism fuses features from three directions (upward, forward, and downward) during training. To shed light on this mechanism, we specifically analyze the proportions [5] of feature map connections from different directions to a specific fused feature node, considering a feature sparsity level of 0.8. After training, we observed in Figure 6 (d) that the DSFF module learned to assign greater importance to features from the "forward" directions for most fused feature nodes. For example, in the first feature level, the flow and processing of features can be observed as if they were passing through a fully convolutional neural network (FCN), which preserves the spatial dimensions of the input image. At the second feature level, the original image is downsampled by a factor of $1/2$ and flows through another FCN. Therefore, from this perspective, E2ENet can take advantage of FCN to effectively incorporate and preserve multi-scale information.

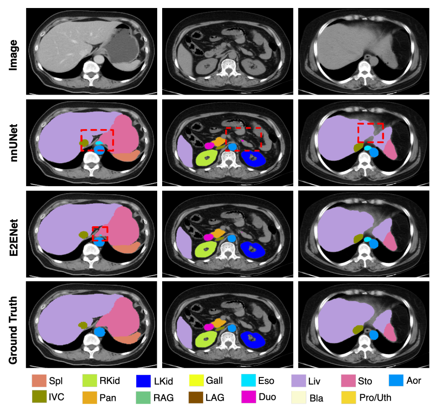

Figure 5: Qualitative comparison of E2ENet and nnUNet on the AMOS-CT challenges.

[5]We calculated these proportions by dividing the number of non-zero connections in a given direction by the total number of non-zero connections within each fused feature.

Simultaneously, the complementary feature flows from the 'upward' and 'downward' directions provide richer information. At the later fusion step of the lower feature levels, the 'upward' information becomes more dominant than the 'downward' information. This prioritization of upward feature flow is similar to the design of the decoders in UNet and UNet++. While E2ENet alleviates the semantic gap more effectively, the proportion of feature map connections in the 'downward' direction always has a certain ratio. This also allows the network to better preserve low-level information and integrate it with high-level information, leading to improved performance in capturing fine details. Interestingly, this trend becomes increasingly apparent during training, as illustrated in Figure 6 (a), (c) and (d).

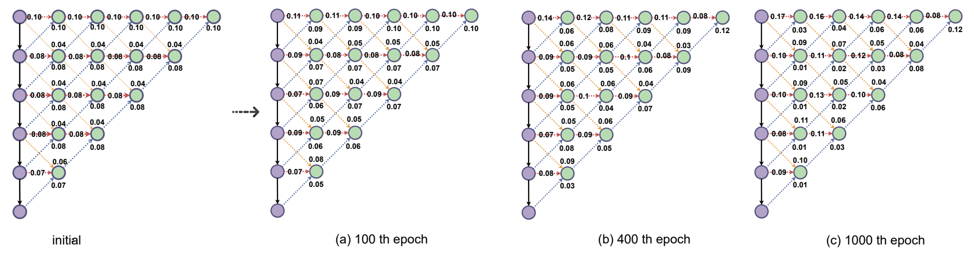

Figure 6: The proportions of feature connections during training with the DSFF mechanism at a feature sparsity level of 0.8 on the AMOS-CT challenge.

4 Conclusion

In this work, our aim is to address the challenge of designing a 3D medical image segmentation method that is both accurate and efficient. By proposing a dynamic sparse feature fusion mechanism and incorporating restricted depth-shift in 3D convolution, our E2ENet improves performance on 3D medical image segmentation tasks while significantly reducing computational and memory overhead. The dynamic sparse feature fusion mechanism demonstrates its ability to adaptively learn the importance of each feature map, zeroing out the less important ones. This results in a more efficient feature representation without compromising performance. Additionally, the experiments demonstrate that restricted depth-shift in 3D convolution enables the model to capture spatial information more effectively and efficiently. Extensive experiments on three benchmarks demonstrate that E2ENet consistently achieves a superior trade-off between accuracy and efficiency compared to previous state-of-the-art baselines. While E2ENet provides a promising solution for balancing accuracy and computational cost, future work could explore the potential of a learnable shift offset, which may further improve performance. Additionally, as plug-and-play components, the DSFF mechanism and restricted depth-shift could be interesting to apply to other 3D segmentation models to explore their potential in the future. Furthermore, future advancements in hardware support for sparse neural networks could fully unlock the potential of sparse training methods.

5 Acknowledgements

Qiao Xiao is supported by the research program 'MegaMind - Measuring, Gathering, Mining and Integrating Data for Self-management in the Edge of the Electricity System', (partly) financed by the Dutch Research Council (NWO) through the Perspectief program under number P19-25. Elena Mocanu is partly supported by the Modular Integrated Sustainable Datacenter (MISD) project funded by the Dutch Ministry of Economic Affairs and Climate under the European Important Projects of Common European Interest - Cloud Infrastructure and Services (IPCEI-CIS) program for 2024-2029. This research used the Dutch national e-infrastructure with the support of the SURF Cooperative, using grant no. EINF-7499.

References

Michela Antonelli, Annika Reinke, and Annette Kopp-Schneider Bennett A. Landman Geert Litjens Bjoern Menze Olaf Ronneberger Ronald M. Summers Bram van Ginneken Michel Bilello Patrick Bilic Patrick F. Christ Richard K. G. Do Marc J. Gollub Stephan H. Heckers Henkjan Huisman William R. Jarnagin Maureen K. McHugo Sandy Napel Jennifer S. Golia Pernicka Kawal Rhode Catalina Tobon-Gomez Eugene Vorontsov James A. Meakin Sebastien Ourselin Manuel Wiesenfarth Pablo Arbeláez Byeonguk Bae Sihong Chen Laura Daza Jianjiang Feng Baochun He Fabian Isensee Yuanfeng Ji Fucang Jia Ildoo Kim Klaus Maier-Hein Dorit Merhof Akshay Pai Beomhee Park Mathias Perslev Ramin Rezaiifar Oliver Rippel Ignacio Sarasua Wei Shen Jaemin Son Christian Wachinger Liansheng Wang Yan Wang Yingda Xia Daguang Xu Zhanwei Xu Yefeng Zheng Amber L. Simpson Lena Maier-Hein M. Jorge Cardoso Spyridon Bakas, Keyvan Farahani. The medical segmentation decathlon. *Nature Communications*, 2022.

Zahra Atashgahi, Ghada Sokar, Tim van der Lee, Elena Mocanu, Decebal Constantin Mocanu, Raymond Veldhuis, and Mykola Pechenizkiy. Quick and robust feature selection: the strength of energy-efficient sparse training for autoencoders. *Machine Learning*, 111(1):377–414, 2022.

Hu Cao, Yueyue Wang, Joy Chen, Dongsheng Jiang, Xiaopeng Zhang, Qi Tian, and Manning Wang. Swin-Unet: Unet-like pure transformer for medical image segmentation. *arXiv:2105.05537*, 2021.

Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017.

Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 11027–11036. Computer Vision Foundation / IEEE, 2020.

Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3D U-Net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention*, pages 424–432. Springer, 2016.

Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. CoAtNet: Marrying convolution and attention for all data sizes. *Advances in Neural Information Processing Systems*, 34:3965–3977, 2021.

Stéphane d'Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun. ConViT: Improving vision transformers with soft convolutional inductive biases. In *International Conference on Machine Learning*, pages 2286–2296. PMLR, 2021.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *International Conference on Machine Learning*, pages 2943–2952. PMLR, 2020.

Linxi Fan, Shyamal Buch, Guanzhi Wang, Ryan Cao, Yuke Zhu, Juan Carlos Niebles, and Li Fei-Fei. RubiksNet: Learnable 3D-shift for efficient video action recognition. In *European Conference on Computer Vision*, pages 505–521. Springer, 2020.

Y. Gao, M. Zhou, D. Liu, and D. Metaxas. A data-scalable transformer for medical image segmentation: Architecture, model efficiency, and benchmark. *arXiv preprint arXiv:2203.00131*, 2022.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Ali Hatamizadeh, Yucheng Tang, Vishwesh Nath, Dong Yang, Andriy Myronenko, Bennett Landman, Holger R Roth, and Daguang Xu. UNETR: Transformers for 3D medical image segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 574–584, 2022.

Yufan He, Dong Yang, Holger Roth, Can Zhao, and Daguang Xu. DiNTS: Differentiable neural network topology search for 3D medical image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5841–5850, 2021.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *J. Mach. Learn. Res.*, 22(241): 1–124, 2021.

Xiaowan Hu, Ruijun Ma, Zhihong Liu, Yuanhao Cai, Xiaole Zhao, Yulun Zhang, and Haoqian Wang. Pseudo 3D auto-correlation network for real image denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16175–16184, 2021.

Huimin Huang, Lanfen Lin, Ruofeng Tong, Hongjie Hu, Qiaowei Zhang, Yutaro Iwamoto, Xianhua Han, Yen-Wei Chen, and Jian Wu. 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)*, pages 1055–1059. IEEE, 2020.

Nabil Ibtehaz and M Sohel Rahman. MultiResUNet: Rethinking the U-Net architecture for multimodal biomedical image segmentation. *Neural Networks*, 121:74–87, 2020.

Fabian Isensee, Paul F Jaeger, Simon AA Kohl, Jens Petersen, and Klaus H Maier-Hein. nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature Methods*, 18(2):203–211, 2021.

Fabian Isensee, Constantin Ulrich, Tassilo Wald, and Klaus H Maier-Hein. Extending nnU-Net is all you need. *arXiv:2208.10791*, 2022.

Siddhant Jayakumar, Razvan Pascanu, Jack Rae, Simon Osindero, and Erich Elsen. Top-KAST: Top-K always sparse training. *Advances in Neural Information Processing Systems*, 33:20744–20754, 2020.

Debesh Jha, Pia H Smedsrud, Michael A Riegler, Dag Johansen, Thomas De Lange, Pål Halvorsen, and Håvard D Johansen. ResUNet++: An advanced architecture for medical image segmentation. In *2019 IEEE International Symposium on Multimedia (ISM)*, pages 225–2255. IEEE, 2019.

Yuanfeng Ji, Haotian Bai, Chongjian GE, Jie Yang, Ye Zhu, Ruimao Zhang, Zhen Li, Lingyan Zhanng, Wanling Ma, Xiang Wan, and Ping Luo. AMOS: A large-scale abdominal multi-organ benchmark for versatile medical image segmentation. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Davood Karimi, Serge Didenko Vasylechko, and Ali Gholipour. Convolution-free medical image segmentation using transformers. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 78–88. Springer, 2021.

B Landman, Z Xu, J Igelsias, M Styner, T Langerak, and A Klein. Miccai multi-atlas labeling beyond the cranial vault–workshop and challenge. *In Proc. MICCAI Multi-Atlas Labeling Beyond Cranial Vault—Workshop Challenge*, 2015.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. SNIP: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2018.

Chen Li, Yusong Tan, Wei Chen, Xin Luo, Yuanming Gao, Xiaogang Jia, and Zhiying Wang. Attention Unet++: A nested attention-aware U-Net for liver ct image segmentation. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 345–349. IEEE, 2020.

Dongze Lian, Zehao Yu, Xing Sun, and Shenghua Gao. AS-MLP: An axial shifted mlp architecture for vision. In *International Conference on Learning Representations*, 2021.

Ji Lin, Chuang Gan, and Song Han. TSM: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7083–7093, 2019.

Shiwei Liu and Zhangyang Wang. Ten lessons we have learned in the new" sparseland": A short handbook for sparse neural network researchers. *arXiv preprint arXiv:2302.02596*, 2023.

Shiwei Liu, Tianlong Chen, Xiaohan Chen, Zahra Atashgahi, Lu Yin, Huanyu Kou, Li Shen, Mykola Pechenizkiy, Zhangyang Wang, and Decebal Constantin Mocanu. Sparse training via boosting pruning plasticity with neuroregeneration. *Advances in Neural Information Processing Systems*, 34:9908–9922, 2021a.

Shiwei Liu, Decebal Constantin Mocanu, Amarsagar Reddy Ramapuram Matavalam, Yulong Pei, and Mykola Pechenizkiy. Sparse evolutionary deep learning with over one million artificial neurons on commodity hardware. *Neural Computing and Applications*, 33(7):2589–2604, 2021b.

Shiwei Liu, Lu Yin, Decebal Constantin Mocanu, and Mykola Pechenizkiy. Do we actually need dense over-parameterization? in-time over-parameterization in sparse training. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 6989–7000. PMLR, 2021c.

Shiwei Liu, Tianlong Chen, Xiaohan Chen, Xuxi Chen, Qiao Xiao, Boqian Wu, Mykola Pechenizkiy, Decebal Mocanu, and Zhangyang Wang. More ConvNets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. *arXiv:2207.03620*, 2022.

Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the 30th International Conference on Machine Learning*, number 1, page 3. Atlanta, Georgia, USA, 2013.

Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D vision (3DV)*, pages 565–571. IEEE, 2016.

Decebal Constantin Mocanu, Elena Mocanu, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. A topological insight into restricted boltzmann machines. *Machine Learning*, 104(2):243–270, 2016.

Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature Communications*, 9(1):1–12, 2018.

Hesham Mostafa and Xin Wang. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning*, pages 4646–4655. PMLR, 2019.

Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention U-Net: Learning where to look for the pancreas. In *Medical Imaging with Deep Learning*, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention*, pages 234–241. Springer, 2015.

Jiacheng Ruan and Suncheng Xiang. Vm-unet: Vision mamba unet for medical image segmentation. *arXiv preprint arXiv:2402.02491*, 2024.

Amber L Simpson, Michela Antonelli, Spyridon Bakas, Michel Bilello, Keyvan Farahani, Bram Van Ginneken, Annette Kopp-Schneider, Bennett A Landman, Geert Litjens, Bjoern Menze, et al. A large annotated medical image dataset for the development and evaluation of segmentation algorithms. *arXiv:1902.09063*, 2019.

Ghada Sokar, Zahra Atashgahi, Mykola Pechenizkiy, and Decebal Constantin Mocanu. Where to pay attention in sparse training for feature selection? In *Advances in Neural Information Processing Systems*, 2022.

Zhuo Su, Linpu Fang, Wenxiong Kang, Dewen Hu, Matti Pietikäinen, and Li Liu. Dynamic group convolution for accelerating convolutional neural networks. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part VI*, 2020.

Mingxing Tan, Ruoming Pang, and Quoc V Le. EfficientDet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10781–10790, 2020.

Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in Neural Information Processing Systems*, 33:6377–6389, 2020.

Yucheng Tang, Riqiang Gao, Ho Hin Lee, Shizhong Han, Yunqiang Chen, Dashan Gao, Vishwesh Nath, Camilo Bermudez, Michael R Savona, Richard G Abramson, et al. High-resolution 3D abdominal segmentation with random patch network fusion. *Medical Image Analysis*, 69:101894, 2021.

Yucheng Tang, Dong Yang, Wenqi Li, Holger R Roth, Bennett Landman, Daguang Xu, Vishwesh Nath, and Ali Hatamizadeh. Self-supervised pre-training of swin transformers for 3D medical image analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20730–20740, 2022.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv:1607.08022*, 2016.

Jeya Maria Jose Valanarasu and Vishal M Patel. UNeXt: MLP-based rapid medical image segmentation network. *arXiv:2203.04967*, 2022.

Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations*, 2019.

Haonan Wang, Peng Cao, Jiaqi Wang, and Osmar R Zaiane. UCTransNet: rethinking the skip connections in U-Net from a channel-wise perspective with transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2441–2449, 2022.

Wenxuan Wang, Chen Chen, Meng Ding, Hong Yu, Sen Zha, and Jiangyun Li. TransBTS: Multimodal brain tumor segmentation using transformer. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 109–119. Springer, 2021.

Ziyang Wang, Jian-Qing Zheng, Yichi Zhang, Ge Cui, and Lei Li. Mamba-unet: Unet-like pure visual mamba for medical image segmentation. *arXiv preprint arXiv:2402.05079*, 2024.

Yu Weng, Tianbao Zhou, Yujie Li, and Xiaoyu Qiu. NAS-Unet: Neural architecture search for medical image segmentation. *IEEE Access*, 7:44247–44257, 2019.

Qiao Xiao, Boqian Wu, Yu Zhang, Shiwei Liu, Mykola Pechenizkiy, Elena Mocanu, and Decebal Constantin Mocanu. Dynamic sparse network for time series classification: Learning what to "see". In *Advances in Neural Information Processing Systems*, 2022.

Yutong Xie, Jianpeng Zhang, Chunhua Shen, and Yong Xia. CoTr: Efficiently bridging CNN and transformer for 3D medical image segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention*, pages 171–180. Springer, 2021.

Zhaohu Xing, Tian Ye, Yijun Yang, Guang Liu, and Lei Zhu. Segmamba: Long-range sequential modeling mamba for 3d medical image segmentation. *arXiv preprint arXiv:2401.13560*, 2024.

Qihang Yu, Dong Yang, Holger Roth, Yutong Bai, Yixiao Zhang, Alan L. Yuille, and Daguang Xu. C2FNAS: coarse-to-fine neural architecture search for 3d medical image segmentation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 4125–4134. Computer Vision Foundation / IEEE, 2020.

Geng Yuan, Xiaolong Ma, Wei Niu, Zhengang Li, Zhenglun Kong, Ning Liu, Yifan Gong, Zheng Zhan, Chaoyang He, Qing Jin, et al. MEST: Accurate and fast memory-economic sparse training framework on the edge. *Advances in Neural Information Processing Systems*, 34:20838–20850, 2021.

Hong-Yu Zhou, Jiansen Guo, Yinghao Zhang, Lequan Yu, Liansheng Wang, and Yizhou Yu. nnFormer: Interleaved transformer for volumetric segmentation. *arXiv:2109.03201*, 2021.

Yuyin Zhou, Zhe Li, Song Bai, Chong Wang, Xinlei Chen, Mei Han, Elliot Fishman, and Alan L Yuille. Prior-aware neural network for partially-supervised multi-organ segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10672–10681, 2019.

Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. UNet++: A nested U-Net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 3–11. Springer, 2018.

A Appendix.

A.1 Impact Statement

In an era dominated by over-parameterized models, designing resource-aware AI models is becoming increasingly important, especially for time-consuming tasks like medical segmentation. Our insights into model efficiency training have the potential to broaden the application of deep neural networks in this area. Overall, this work advances our fundamental understanding of dynamic sparse training and offers future perspectives for scalable and efficient AI models. We do not anticipate any negative societal impacts resulting from this research.

A.2 Related Work

A.2.1 3D Medical Image Segmentation

Convolutional neural networks (CNNs) have become the dominant architecture for 3D medical image segmentation in recent years (e.g. 3D UNet [Çiçek et al., 2016], UNet++ [Zhou et al., 2018], UNet3+ [Huang et al., 2020], PaNN [Zhou et al., 2019] and nnUNet [Isensee et al., 2021]), due to their ability to capture local and weight-sharing dependencies [d'Ascoli et al., 2021, Dai et al., 2021]. However, some recent methods have attempted to incorporate transformer modules into CNNs (e.g. CoTr [Xie et al., 2021], TransBTS [Wang et al., 2021]), or use pure transformer architectures (e.g. ConvIt [Karimi et al., 2021], nnFormer [Zhou et al., 2021], Swin UNet [Cao et al., 2021]), in order to capture long-range dependencies. These transformer-based approaches often require large amounts of training data, longer training times, or specialized training techniques, and can also be computationally expensive. Most recently, a novel architecture called Mamba [Gu and Dao, 2023] has shown potential for computational efficiency as a State Space model in handling long sequences and has been applied to medical image segmentation tasks [Ruan and Xiang, 2024, Xing et al., 2024, Wang et al., 2024]. However, it has led to underwhelming performance compared to state-of-the-art convolutional models. In this paper, we propose an alternative method for efficiently incorporating 3D contextual information using a restricted depth-shift strategy in 3D convolutions, and further improving performance through adaptive multi-scale feature fusion.

A.2.2 Feature Fusion in Medical Image Segmentation

Multi-scale feature fusion is a crucial technique in medical image segmentation that allows a model to detect objects across a range of scales, while also recovering spatial information that is lost during pooling [Wang et al., 2022, Xie et al., 2021]. However, effectively representing and processing multi-scale hierarchy features can be challenging, and simply summing them up without distinction can lead to semantic gaps and degraded performance [Wang et al., 2022, Tan et al., 2020]. To address this issue, various approaches have been proposed, including adding learnable operations to reduce the gap with residuals [Ibtehaz and Rahman, 2020], attention blocks [Oktay et al., 2018]. More recently, UNet++ [Zhou et al., 2018] and its variants [Li et al., 2020, Huang et al., 2020, Jha et al., 2019] have adapted the gating signal to dense nesting levels, taking into account as many feature levels as possible. NAS-UNet [Weng et al., 2019] tries to automatically search for better feature fusion topology. While these methods have achieved better performance, they can also incur significant computational and information redundancy. Dynamic convolution [Su et al., 2020, Chen et al., 2020] utilizes coefficient prediction or attention modules to dynamically aggregate convolution kernels, thereby reducing computation costs. In our paper, we propose an intuitive approach to optimizing multi-scale feature fusion, which enables selective leveraging of sparse feature representations from fine-grained to semantic levels through the proposed dynamic sparse feature fusion mechanism.

A.2.3 Sparse Training

Recently, sparse training techniques have shown the possibility of training an efficient network with sparse connections that match (or even outperform) the performance of dense counterparts with lower computational cost [Mocanu et al., 2018, Liu et al., 2021b]. Beginning with [Mocanu et al., 2016], it has been demonstrated that initializing a static sparse network without optimizing its topology during training can also yield comparable performance in certain situations [Lee et al., 2018, Tanaka et al., 2020, Wang et al., 2019]. However, Dynamic Sparse Training (DST), also known as sparse training with dynamic sparsity [Mocanu et al., 2018], offers a different approach by jointly optimizing the

Algorithm 1 The Training Process of Dynamic Sparse Feature Fusion (DSFF)

```
Require: Dataset $\mathcal{X}$ with label $\mathcal{Y}$; feature sparsity $S$; backbone $f_\Theta(.)$; Output Module: $f_{out}$; Total training epochs: $T$;
    evolution period: $\Delta T$; connection updating number: $f_{decay}(\Delta T; \alpha, T) = \frac{\alpha}{2}\left(1 + \cos\left(\frac{\Delta T \pi}{T}\right)\right)$,
    $\alpha$ represents the number of updated connections during the initial topology update, which is set to $1/2$; Loss function: $\mathcal{L}(.)$; fusion operation: $\mathcal{F}^{j,i}(\cdot)$ with convolution kernels $\theta^{j,i}$, where the numbers of input and output channel are $C_{in}^{j,i}$, $C_{out}^{j,i}$.
 1: $\mathbf{M}^{j,i} \leftarrow$ random initialize masks for all levels and stages, satisfying that $\|\mathbf{M}^{j,i}\|_0$ equals $(1 - S) \times C_{in}^{j,i} \times C_{out}^{j,i}$
 2: for $t = 1$ to $T$ do
 3:     Sample a batch $I_t, Y_t \sim \mathcal{X}, \mathcal{Y}$
 4:     Generate multi-scaled features: $\left(\mathbf{x}^{0,1}, \mathbf{x}^{0,2}, \ldots, \mathbf{x}^{0,L}\right) = f_\Theta(I_t)$
 5:     for each stage $j = 1$ to $L - 1$ do
 6:         for each level $i = 1$ to $L - j$ do
 7:             if $i = 1$ then
 8:                 $\mathbf{x}^{j,i} = \mathcal{F}^{j,i}([\mathbf{x}^{j-1,1}, \mathcal{U}(\mathbf{x}^{j-1,2})])$
 9:             else
10:                 $\mathbf{x}^{j,i} = \mathcal{F}^{j,i}([\mathcal{D}(\mathbf{x}^{j-1,i-1}), \mathbf{x}^{j-1,i}, \mathcal{U}(\mathbf{x}^{j-1,i+1})])$
11:             end if
12:         end for
13:     end for
14:     $l_t = 4/7\mathcal{L}(f_{out}\left(\mathbf{x}^{L-1,1}\right), Y_i) + 2/7\mathcal{L}(f_{out}\left(\mathbf{x}^{L-2,2}\right), \mathcal{D}(Y_i)) + 1/7\mathcal{L}(f_{out}\left(\mathbf{x}^{L-3,3}\right), \mathcal{D}(\mathcal{D}(Y_i)))$
15:     if $(t \bmod \Delta T) == 0$ then
16:         for each stage $j = 1$ to $L - 1$ do
17:             for each level $i = 1$ to $L - j$ do
18:                 $u = (C_{in}^{j,i} \times C_{out}^{j,i}) f_{decay}(t; \alpha, T)(1 - S)$
19:                 $IS \leftarrow$ importance score ($L_1$ Norm of corresponding kernel) for activated each feature connection
20:                 $\mathbb{I}_{activate} = RandomK(\mathbb{I}_{inactivate}, u)$
21:                 $\mathbb{I}_{inactivate} = ArgTopK(-IS, u)$
22:                 $\mathbf{M}^{j,i} \leftarrow$ Update $\mathbf{M}^{j,i}$ using $\mathbb{I}_{inactivate}$ and $\mathbb{I}_{activate}$
23:             end for
24:         end for
25:     else
26:         Training the E2ENet using SGD optimizer
27:     end if
28: end for
```

sparse topology and weights during the training process starting from a sparse network [Liu et al., 2021a, 2022, Evci et al., 2020, Jayakumar et al., 2020, Mostafa and Wang, 2019, Yuan et al., 2021]. This allows the model's sparse connections to gradually evolve in a prune-and-grow scheme, leading to improved performance compared to naively training a static sparse network [Liu et al., 2021c, Xiao et al., 2022]. In contrast to prior methods that aim to find sparse networks that can match the performance of corresponding dense networks, we aim to leverage DST to adaptively fuse multi-scale features in a computationally efficient manner for 3D medical image segmentation.

A.3 Algorithm

A.4 Datasets and Experiment Setup

AMOS-CT: The Abdominal Multi-Organ Segmentation Challenge (AMOS) [Ji et al., 2022] task 1 consists of 500 computerized tomography (CT) cases, including 200 scans for training, 100 for validation, and 200 for testing. These cases have been collected from a diverse patient population and include annotations of 15 organs. The scans are from multiple centers, vendors, modalities, phases, and diseases.

BTCV: The Beyond the Cranial Vault (BTCV) abdomen challenge dataset [6] consists of 30 CT scan images for training and 20 for testing. These images have been annotated by interpreters under the supervision of radiologists, and include labels for 13 organs.

BraTS: The Brain Tumor Segmentation Challenge in the Medical Segmentation Decathlon (MSD) [Antonelli et al., 2022, Simpson et al., 2019] consists of 484 MRI images from 19 different institutions. These images contain three different tumor regions of interest (ROIs): edema (ED), non-enhancing tumor (NET) and enhancing tumor (ET). The goal of the challenge is to segment these ROIs in the images accurately.

A.5 Implementation Details

In our work, we utilized the PyTorch toolkit [Paszke et al., 2019] on an NVIDIA A100 GPU for all our experimental evaluations. We also used the nnUNet codebase [Isensee et al., 2021] to pre-process data before training our proposed E2ENet model. For the AMOS dataset, we used the nnUNet codebase as the benchmark implementation.

For training, we use the stochastic gradient descent (SGD) optimizer with an initial learning rate of 0.01, which is gradually decreased through a "poly" decay schedule. The optimizer is configured with a momentum of 0.99 and a weight decay of 3×10^{-5}. The maximum number of training epochs is 1000, with 250 iterations per epoch. For the loss function, we combine both cross-entropy loss and Dice loss as in [Isensee et al., 2021]. To improve performance, various data augmentation techniques such as random rotation, scaling, flipping, adding Gaussian noise, blurring, adjusting brightness and contrast, simulating low resolution, and Gamma transformation are used before training.

We employ a 5-fold cross-validation strategy on the training set for all experiments, selecting the final model from each fold and simply averaging their outputs for the final segmentation predictions. In the testing stage, we employ the sliding window strategy, where the window sizes are equal to the size of the training patches. Additionally, post-processing methods outlined in [Isensee et al., 2022] are applied for the AMOS-CT dataset during the testing phase.

A.6 The Architecture

The backbone generates a total of $L = 6$ multi-scale feature levels, each with a specified number of channels: [c1, c2, c3, c4, c5, c6] = [48, 96, 192, 320, 320, 320]. At each level of feature generation, there are two convolution layers with a kernel size of (1, 3, 3), followed by instance normalization and the application of leaky ReLU activation. The down-sampling ratios for each level are as follows: ((1, 2, 2), (2, 2, 2), (2, 2, 2), (2, 2, 2), (2, 2, 2), (2, 2, 2), (2, 2, 2)).

A.7 Evaluation Metrics

A.7.1 Mean Dice Similarity Coefficient

To assess the quality of the segmentation results, we use the mean Dice similarity coefficient (mDice), which is a widely used metric in medical image segmentation. The mDice is calculated as follows:

$$mDice = \frac{1}{N} \sum_{j=1}^{N} \frac{2|\mathbf{y}_j \cdot \hat{\mathbf{y}}_j|}{(|\mathbf{y}_j| + |\hat{\mathbf{y}}_j|)}, \tag{3}$$

where N is the number of classes, $\cdot$ is the pointwise multiplication, $\mathbf{y}_j$ and $\hat{\mathbf{y}}_j$ represent the ground truth and predicted masks of the j-th class, respectively, which are encoded in one-hot format. $\frac{2|\mathbf{y}_j \cdot \hat{\mathbf{y}}_j|}{(|\mathbf{y}_j| + |\hat{\mathbf{y}}_j|)}$ is the Dice of j-th class, which measures the overlap between the predicted and ground truth segmentation masks for that class.

A.7.2 Number of Parameters

The size of the network can be estimated by summing the number of non-zero parameters (Params), which includes the parameters of activated sparse feature connections (kernels) and parameters of the

[6] https://www.synapse.org/#!Synapse:syn3193805/wiki/89480

backbone. The calculation is given by the following equation:

$$Params = \|\Theta\|_0 + \sum_{j=1}^{L-1} \sum_{i=1}^{L-j} \sum_{c_{in}=1}^{C_{in}^{j,i}} \sum_{c_{out}=1}^{C_{out}^{j,i}} \mathbf{M}^{j,i}_{c_{in},c_{out}} \|\theta^{j,i}_{c_{in},c_{out}}\|_0. \tag{4}$$

Here, Θ is the parameter from backbone, L is the total number of feature levels, $\mathbf{M}^{j,i}$ is a matrix of size $C_{in}^{j,i} \times C_{out}^{j,i}$, and $\mathbf{M}^{j,i}_{c_{in},c_{out}}$ indicates whether the kernel $\theta^{j,i}_{c_{in},c_{out}}$ connecting the c_{in}-th input and c_{out}-th output feature map exist or not. The L_0 norm $\|\theta^{j,i}_{c_{in},c_{out}}\|_0$ provides the number of non-zero entries of $\theta^{j,i}_{c_{in},c_{out}}$.

A.7.3 Float Point Operations

Floating point operations (FLOPs) is a commonly used metric to compare the computational cost of a sparse model to that of a dense counterpart [Hoefler et al., 2021] [7]. In our comparison, it is calculated by counting the number of multiplications and additions performed in only one forward pass of the inference process without considering postprocessing. The inference FLOPs are estimated layer by layer and depend on the sparsity level of the network. For each convolution or transposed convolution layer, the inference FLOPs is calculated as follows:

$$FLOPs_{conv} = (2K_d K_h K_w C_{in}(1-S) + 1) \times C_{out} HWD, \tag{5}$$

where K_d, K_h and K_w are the kernel sizes in depth, height and width; S is the feature sparsity level, for layers that are not part of the DSFF mechanism, $S = 0$ is used; C_{in} and C_{out} are the numbers of input feature and output feature; H, W and D are the height, width and depth of output features. For each fully connected layer, the inference FLOPs is calculated as follows:

$$FLOPs_{fc} = (2C_{in}(1-S) + 1) \times C_{out}. \tag{6}$$

A.7.4 Performance Trade-Off Score

The accuracy-efficiency trade-offs could be further analyzed, from comparing resource requirements to describing holistic behaviours (including mDice, Params and inference FLOPs) for the 3D image segmentation methods. To quantify these trade-offs, we introduce the Performance Trade-Off (PT) score, which is defined as follows:

$$PT = \alpha_1 \frac{mDice}{mDice_{max}} + \alpha_2 (\frac{Params_{min}}{Params} + \frac{FLOPs_{min}}{FLOPs}), \tag{7}$$

where α_1 and α_2 are weighting factors, which control the trade-off between accuracy performance and resource requirements, and $mDice_{max}$, $Params_{min}$, and $FLOPs_{min}$ denote the highest mDice score, the smallest number of parameters, and the lowest inference FLOPs among the compared methods for a specific dataset, respectively. The term $\frac{mDice}{mDice_{max}}$ measures the segmentation accuracy, while $\frac{Params_{min}}{Params} + \frac{FLOPs_{min}}{FLOPs}$ measures the resource cost.

In most cases, we consider both segmentation accuracy and resource cost to be equally important, thus we set $\alpha_1 = 1$ and $\alpha_2 = 1/2$ in the following experiments. However, we also explore the impact of different choices of α_1 and α_2, as detailed in Section A.9. The PT score serves as a valuable metric for evaluating the trade-offs between segmentation accuracy and efficiency.

A.7.5 Discussion on Running Time

Given the relatively restricted support for sparse operations in current off-the-shelf commodity GPUs and TPUs without sparsity-aware accelerators, we did not attempt to achieve practical speedup during training. Instead, we chose to implement our models with binary masks in our work. The promising benefits of dynamic sparsity presented in this study have not yet translated into actual speedup. Accelerating training time will be a focus for our next work.

[7] This is because current sparse training methods often use masks on dense weights to stimulate sparsity. This is done because most deep learning hardware is optimized for dense matrix operations. As a result, using these prototypes doesn't accurately reflect the true memory and speed benefits of a truly sparse network [Hoefler et al., 2021].

Although not the focus of our current work, it would be interesting for future work to examine the speedup results of sparse operation during training, using such specialized hardware accelerators. For example, at high unstructured sparsity levels, XNNPACK [8] has already shown significant speedups over dense baselines on smartphone processors.

Although the support for unstructured sparsity on GPUs remains relatively limited, its practical relevance has been widely demonstrated on non-GPU hardware, such as CPUs and customized accelerators. For example, FPGA accelerators have demonstrated significant acceleration and energy efficiency for unstructured sparse RNNs, outperforming commercial CPUs and GPUs by effectively leveraging the FPGA's embedded resources. Additionally, DeepSparse [9] has shown impressive results in deploying large-scale, BERT-level sparse models on modern Intel CPUs. This approach achieved a 10× compression in model size with under a 1% accuracy loss, a 10× increase in CPU inference speed with less than a 2% accuracy reduction, and an impressive 29× CPU inference speedup with a maximum accuracy drop of 7.5% [Liu and Wang, 2023].

Table 9: Comparison of inference speeds between E2ENet and nnUNet.

Methods	nnUNet	E2ENet(s=0.8)	E2ENet(s=0.9)
Latency(ms)	10.07	8.02	7.28
Throughput(items/sec)	99.19	124.60	137.13
speedup	1.0×	1.26×	1.38×

Inspired by these advancements, we adopted an approach based on DeepSparse. As shown in Table 9, we conducted experiments with patches of images sized 32×32×32 as input, comparing the CPU wall-clock timings for online inference between our proposed E2ENet and nnUNet on an Intel Xeon Platinum 8360Y CPU with 18 cores. We acknowledge that, while our proposed models with sparsity do achieve speedups in practical inference, they are not as pronounced as those observed with BERT-level sparse models. This is primarily due to the nature of segmentation and 3D convolution operations. However, this presents a promising avenue for our future work.

A.8 More Experimental Results

A.8.1 Class-wise Dice of AMOS-CT

Table 10: Quantitative comparisons (class-wise Dice (%) ↑, mDice(%)↑, Params(M)↓, inference FLOPs(G)↓, PT score↑ and mNSD(%)↑) of segmentation performance on the validation set of AMOS-CT dataset. **Bold** indicates the best and underline indicates the second best. Note: Spl: spleen, RKid: right kidney, LKid: left kidney, Gall: gallbladder, Eso: esophagus, Liv: liver, Sto: stomach, Aor: aorta IVC: inferior vena cava, Pan: pancreas, RAG: right adrenal gland, LAG: left adrenal gland, Duo: duodenum, Bla: bladder, Pro/Uth: prostate/uterus. The class-wise Dice, mDice and mNSD results of baselines, except for nnUNet, are collected from the [Ji et al., 2022]. † indicates the results without postprocessing that are collected from the AMOS website. ‡ denotes the results with postprocessing that are reproduced by us. * indicates the results with postprocessing.

Methods	Spl	RKid	LKid	Gall	Eso	Liv	Sto	Aor	IVC	Pan	RAG	LAG	Duo	Bla	Pro/Uth	mDice	Params	FLOPs [3]	PT score	mNSD
CoTr	91.1	87.2	86.4	60.5	80.9	91.6	80.1	93.7	87.7	76.3	73.7	71.7	68.0	67.4	40.8	77.1	41.87	1510.53	1.07	64.2
nnFormer	95.9	93.5	94.8	78.5	81.1	95.9	89.4	94.2	88.2	85.0	75.0	75.9	78.5	83.9	74.6	85.6	150.14	1343.65	1.12	74.2
UNETR	92.7	88.5	90.6	66.5	73.3	94.1	78.7	91.4	84.0	74.5	68.2	65.3	62.4	77.4	67.5	78.3	93.02	**391.03**	1.41	61.5
Swin UNETR	95.5	93.8	94.5	77.3	83.0	96.0	88.9	94.7	89.6	84.9	77.2	78.3	78.6	85.8	77.4	86.4	62.83	1562.99	1.14	75.3
VNet	94.2	91.9	92.7	70.2	79.0	94.7	84.8	93.0	87.4	80.5	72.6	73.2	71.7	77.0	66.6	82.0	45.65	1737.57	1.10	67.9
nnUNet†	**97.1**	96.4	96.2	83.2	87.5	97.6	92.2	**96.0**	92.5	88.6	81.2	81.7	**85.0**	90.5	85.0	90.0	30.76	1067.89	1.30	82.1
nnUNet‡	**97.1**	**97.0**	**97.1**	**86.6**	**87.7**	**97.9**	**92.4**	**96.0**	**92.7**	88.8	**81.6**	82.1	**85.0**	**90.6**	**85.2**	**90.5**	30.76	1067.89	1.31	**83.0**
E2ENet* (s=0.7)	**97.1**	96.9	**97.1**	86.0	87.6	**97.9**	92.3	95.7	92.3	**89.0**	81.5	**82.4**	84.9	90.3	83.8	90.3	11.23	969.32	1.54	82.7
E2ENet* (s=0.8)	**97.1**	96.9	97.0	85.2	87.5	**97.9**	92.3	95.7	92.3	**89.0**	81.3	82.1	84.6	90.1	84.8	90.3	9.44	778.74	1.65	82.5
E2ENet* (s=0.9)	96.7	96.9	97.0	84.2	87.0	97.7	92.2	95.6	92.0	88.6	81.0	81.8	84.0	89.9	83.8	89.9	**7.64**	492.29	**1.89**	81.8
E2ENet (s=0.7)	**97.1**	96.6	96.5	83.4	87.6	97.5	92.3	95.8	92.3	**89.0**	81.4	82.3	84.9	90.3	83.8	90.1	11.23	969.32	1.54	82.3
E2ENet (s=0.8)	**97.1**	96.6	96.5	83.4	87.5	97.5	92.3	95.8	92.3	**89.0**	81.3	82.0	84.5	90.1	84.8	90.0	9.44	778.74	1.65	82.3
E2ENet (s=0.9)	96.7	95.4	96.4	82.6	86.9	97.4	92.2	95.6	92.0	88.6	80.9	81.7	84.0	89.9	83.8	89.6	**7.64**	492.29	1.88	81.4
E2ENet(static, s=0.9)	96.6	95.5	96.3	82.6	86.9	97.4	92.2	95.6	92.0	88.6	80.9	81.7	84.0	89.9	83.8	89.6	**7.64**	492.29	1.88	81.4

[3] The inference FLOPs are calculated based on the patch sizes of $1 \times 128 \times 128 \times 128$ without considering postprocessing cost.

A.8.2 BTCV Challenge

We compare the performance of our E2ENet model to several baselines (CoTr [Xie et al., 2021], RandomPatch [Tang et al., 2021], PaNN [Zhou et al., 2019], UNETR [Hatamizadeh et al., 2022], and nnUNet [Isensee et al., 2021]) on the test set of BTCV challenge, and report class-wise Dice, mDice, Params and inference FLOPs on the test set in Table 11. It is worth noting that nnUNet is a strong performer that uses an automatic model configuration strategy to select and ensemble

[8] https://github.com/google/XNNPACK.

[9] https://github.com/neuralmagic/deepsparse.

two best of multiple U-Net models (2D, 3D and 3D cascade) based on cross-validation results. In contrast, E2ENet is designed to be computationally and memory efficient, using a consistent 3D network configuration. Swin UNETR [Tang et al., 2022] is among the best on the leaderboard for this challenge. However, we do not include it in our comparison because it employs self-supervised learning with extra data. This falls outside of our goal of trading off training efficiency and accuracy without using extra data.

Our proposed E2ENet, a single 3D architecture without cascade, has achieved comparable performance to nnUNet, with mDice of 88.3%. Additionally, it has a significantly smaller number of parameters, 11.25 M, compared to other methods such as nnUNet (30.76 M), CoTr (41.87 M), and UNETR (92.78 M).

Table 11: Quantitative comparisons of segmentation performance on BTCV test set. Note: Spl: spleen, RKid: right kidney, LKid: left kidney, Gall: gallbladder, Eso: esophagus, Liv: liver, Sto: stomach, Aor: aorta IVC: inferior vena cava, Veins: portal and splenic veins, Pan: pancreas, AG: adrenal gland. The results (class-wise Dice and mDice) for these baselines are from [Hatamizadeh et al., 2022]. $^+$ denotes that the training of UNETR$^+$ is without using any extra data outside the challenge. The results of nnUNet‡, E2ENet and Hausdorff Distance (HD)$\downarrow$ of UNETR are from the standard leaderboard of BTCV challenge, while the results of nnUNet are from the free leaderboard.

Methods	Spl	RKid	LKid	Gall	Eso	Liv	Sto	Aor	IVC	Veins	Pan	AG	mDice	Params	FLOPs [1]	PT score	HD
CoTr	95.8	92.1	93.6	70.0	76.4	96.3	85.4	92.0	83.8	78.7	77.5	69.4	84.4	41.87	636.94	1.22	/
RandomPatch	96.3	91.2	92.1	74.9	76.0	96.2	87.0	88.9	84.6	78.6	76.2	71.2	84.4	/	/	/	/
PaNN	96.6	**92.7**	95.2	73.2	79.1	97.3	89.1	91.4	85.0	80.5	80.2	65.2	85.4	/	/	/	/
UNETR$^+$	<u>96.8</u>	<u>92.4</u>	94.1	75.0	76.6	97.1	91.3	89.0	84.7	78.8	76.7	74.1	85.6	92.79	**164.91**	<u>1.53</u>	23.4
nnUNet	**97.2**	91.8	**95.8**	75.3	<u>84.1</u>	**97.7**	**92.2**	**92.9**	**88.1**	<u>83.2</u>	**85.2**	<u>77.8</u>	**88.4**	<u>31.18</u>	<u>416.73</u>	1.38	15.6
nnUNet‡	96.5	91.7	**95.8**	**78.5**	84.2	97.4	<u>91.5</u>	<u>92.3</u>	<u>86.9</u>	<u>83.1</u>	<u>84.9</u>	77.5	88.0	<u>31.18</u>	<u>416.73</u>	1.38	16.9
E2ENet ($s = 0.7$)	96.5	91.3	<u>95.7</u>	<u>78.1</u>	**84.5**	<u>97.5</u>	<u>91.5</u>	<u>92.2</u>	86.7	**83.4**	84.8	**77.9**	<u>88.3</u>	**11.25**	449.00	**1.68**	16.1

[1] The inference FLOPs are calculated based on the patch sizes of $1 \times 96 \times 96 \times 96$. The codes for RandomPatch and PaNN are not publicly available, so it is not possible for us to determine their model size and inference FLOPs.

A.8.3 Statistical Significance of Designed Modules

To demonstrate the advantages of individual modules, we plot a critical distance diagram using the Nemenyi post-hoc test with a p-value of 0.05 to establish the statistical significance of our modules. In Figure 7, the top line represents the axis along which the methods' average ranks, and a lower value indicates better performance. Methods joined by thick horizontal black lines are considered not statistically different. From the diagram, we can clearly observe that E2ENet with depth shift significantly outperforms E2ENet without depth shift. Additionally, the incorporation of dynamic sparse feature fusion into E2ENet results in a substantial reduction in both the number of FLOPs (from 23.90M to 11.23M) and parameters (from 3069.55G to 969.32G) while maintaining comparable performance, without any significant performance degradation.

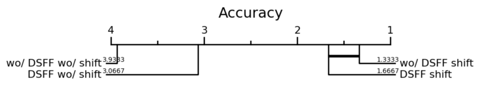

Figure 7: The critical distance diagram on the AMOS-CT validation dataset, with the evaluation metric being mDice.

A.9 The Impact of Weighting Factors α on PT Score

In this section, we investigate the impact of the weighting factors α_1 and α_2 on the Performance Trade-off Score for the AMOS-CT challenge, as defined in Equation 7. These factors are used to balance the trade-off between accuracy and resource cost. A higher value of α_1 prioritizes accuracy, while a lower value emphasizes resource cost. Figure 8 shows that as α_1 decreases, the gaps in the Performance Trade-off Score between E2ENet and other methods become larger, indicating that our method is more advantageous when prioritizing resource cost. However, even when α_1 is set to be 20 times greater than α_2, which prioritizes accuracy over resource cost, the Performance Trade-off Score of E2ENet remains superior to other baselines. This result indicates that our proposed E2ENet architecture is highly efficient in terms of computational cost and memory usage while achieving excellent segmentation performance on the AMOS-CT challenge.

A.9.1 Qualitative Results

BTCV Challenge In Figure 9 (b), we present a qualitative comparison of our proposed E2ENet method with nnUNet as a baseline model on the BTCV challenge. Our results demonstrate the

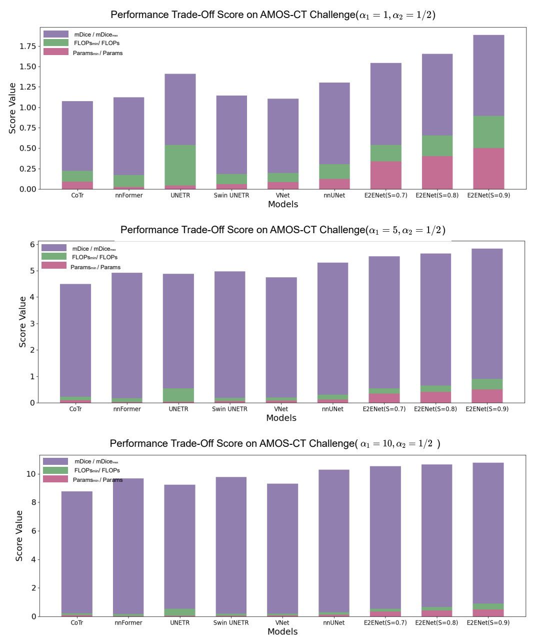

Figure 8: Comparison of Performance Trade-Off score between E2ENet and other models on AMOS-CT challenge with varying α_1 and α_2 values. E2ENet outperforms other baselines in achieving a better trade-off between accuracy and efficiency across different preferences.

effectiveness of our proposed method in addressing some of the challenges of medical image segmentation. For example, as shown in the first and third columns, our E2ENet method accurately distinguishes the stomach from the background without over- or under-segmentation, which can be difficult due to the low contrast in the image. In the second column, E2ENet performs well in differentiating the stomach from the spleen. These examples suggest that our DSFF module can effectively encode feature information for improved performance in medical image segmentation.

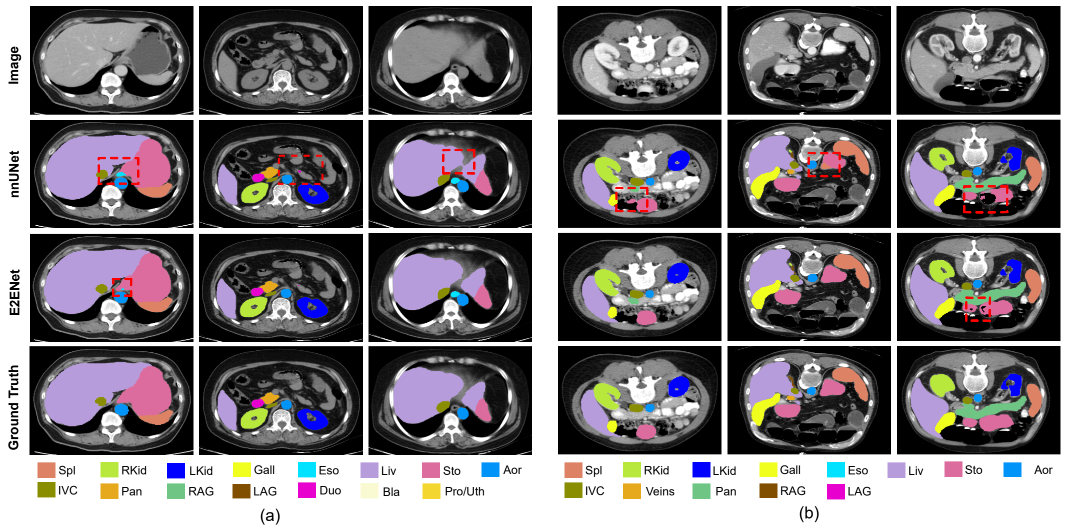

Figure 9: Qualitative comparison of the proposed E2ENet and nnUNet on AMOS-CT and BTCV challenges.

BraTS Challenge in MSD Figure 10 presents a qualitative comparison of our proposed E2ENet method with the nnUNet on the BraTS challenge with highly variable shapes of the segmentation

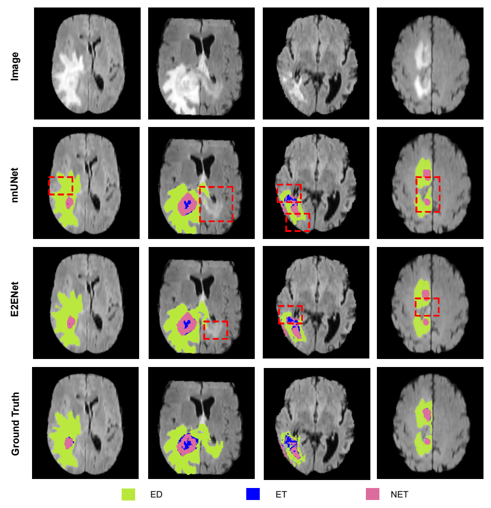

Figure 10: Qualitative comparison of the proposed E2ENet and nnUNet on BraTS Challenge in MSD.

targets. Based on the results of the baseline model, nnUNet, we observed that accurately distinguishing the edema (ED) from the background is difficult, as the edema tends to have less smooth boundaries. Our results suggest that E2ENet may have some potential to improve the distinguishability of the edema boundaries, as evidenced by the relatively better segmentation results in the first, second, and fourth columns. Moreover, E2ENet accurately differentiates the enhanced tumor (ET) from the edema, as shown in the third column, which is a challenging task due to the similarity in appearance between these two regions, and the dispersive distribution of ET. These findings suggest that E2ENet is a promising method for accurately segmenting brain tumors in challenging scenarios.

A.10 Convergence Analysis

In this section, we analyze the convergence behavior of E2ENet by examining the loss changes during topology updating (kernel activation/deactivation epochs), comparing it with the best-performing baseline nnUNet, and studying the impact of topology update frequency. From Figure 11, we observed that the activation/deactivation of weights initially led to an increase in training loss. However, over the long term, the training converged. Additionally, we compared the learning curve of E2ENet with that of nnUNet and found that E2ENet converged even faster than nnUNet, as shown in the subplot in Figure 12 (a). To account for the effect of the number of parameters, we scaled down nnUNet to have a similar number of parameters as E2ENet and observed that it converged even more slowly than the original nnUNet. We also studied the impact of topology update frequency. As shown in Figure 12 (b), when the topology updating frequency is increased, the convergence speed may decrease slightly, but the impact is not significant.

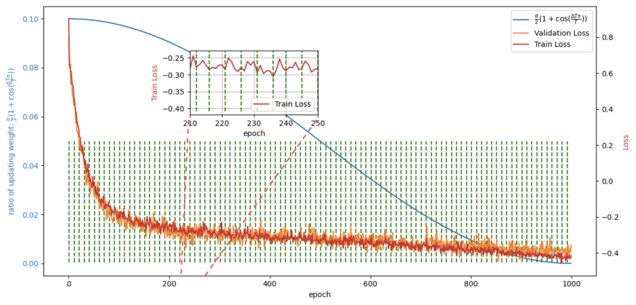

Figure 11: The learning curve of E2ENet on AMOS-CT, with green dotted vertical lines indicating the epochs of weight activation and deactivation. The blue line represents the ratio of weight deactivation/reactivation throughout the training process.

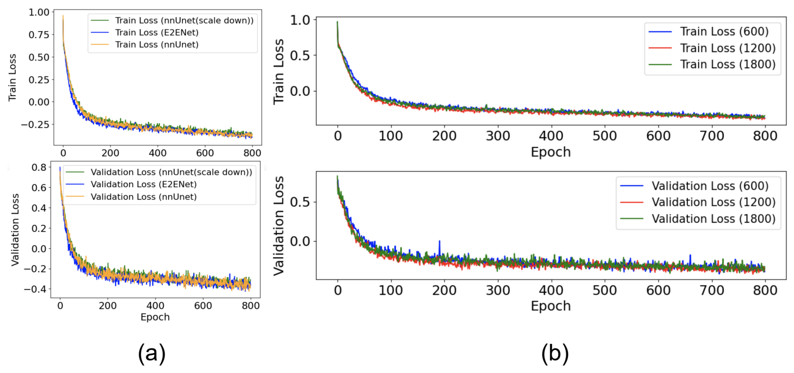

Figure 12: (a) Comparing the learning curve of E2ENet with that of nnUNet and scaled-down nnUNet (referred to as nnUNet (-)); (b) Comparing the learning curve of E2ENet with different topology update frequencies.

A.11 Organ Volume Statistics and Class-wise Results Visualization

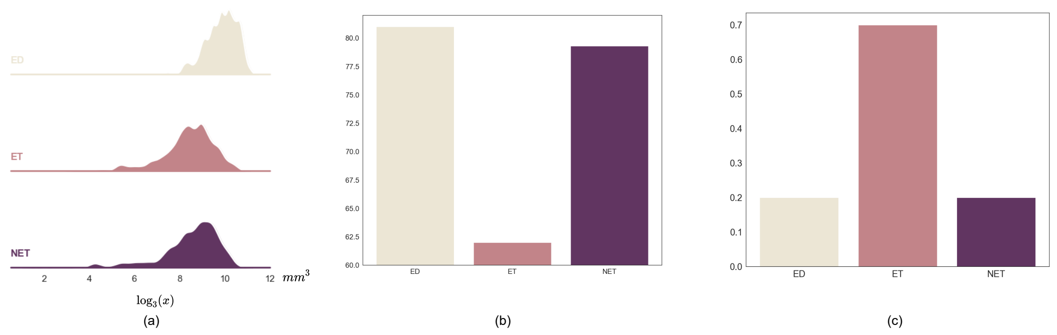

Figure 13: (a) The organ volume statistics of AMOS-CT training dataset. (b) Class-wise Dice of nnUNet without postprocessing (visualization of Table 1). (c) Class-wise Dice differences between E2ENet with feature sparsity 0.7 without postprocessing and nnUNet without postprocessing on AMOS-CT validation dataset. The positive value means that E2ENet outperforms nnUNet, and vice versa.)

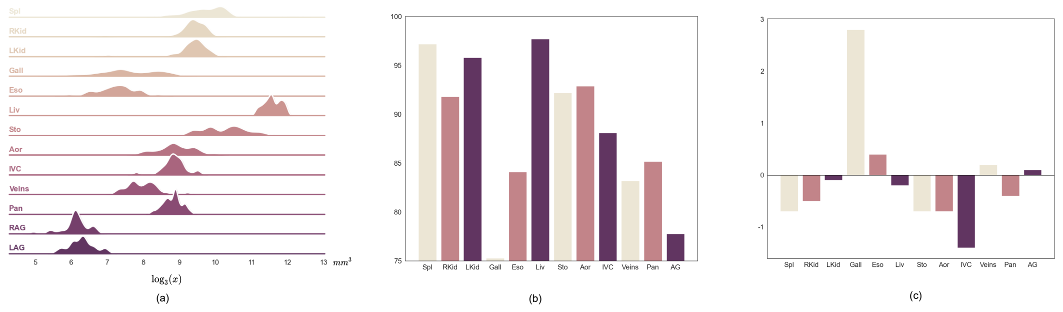

Figure 14: (a) The organ volume statistics of BTCV training dataset. (b) Class-wise Dice of nnUNet (visualization of Table 11). Note that AG denotes the average of the right and left adrenal glands (RAG and LAG). (c) Class-wise Dice differences between E2ENet with feature sparsity 0.7 and nnUNet on BTCV test dataset. The positive value means that E2ENet outperforms nnUNet, and vice versa.

In this section, we analyzed the relationship between organ volume and segmentation accuracy on the AMOS-CT, BTCV, and BraTS challenges. The results, depicted in Figures 13, 14 and 15, showed that small organs with relatively low segmentation accuracy. For the AMOS-CT challenge, RAG

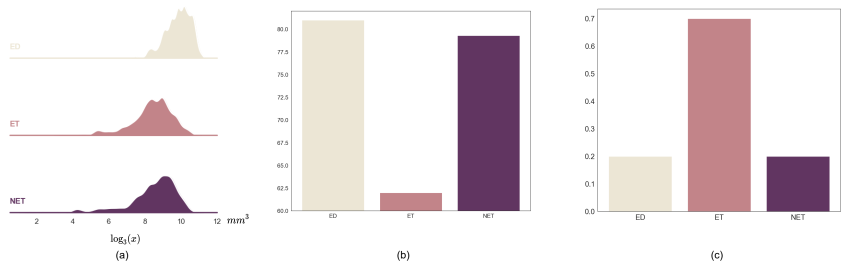

Figure 15: (a) The organ volume statistics of BraTS training dataset. (b) Class-wise Dice of nnUNet (visualization of Table 2) (c) Class-wise Dice differences between E2ENet with feature sparsity 0.7 and nnUNet on 5-fold cross-validation of the training dataset. The positive value means that E2ENet outperforms nnUNet, and vice versa.

(right adrenal gland), LAG (left adrenal gland), Gall (gallbladder), and Eso (esophagus) are more challenging to accurately segment. This may be due to the fact that smaller organ volumes provide less visual information for the segmentation algorithm to work with. However, our proposed method, E2ENet, also demonstrated comparable (or better) performance on these small organs, particularly for the organ "LAG", in which the Dice improved from 81.7% to 82.4%. On the BTCV challenge, the Dice of "Gall", which is considered to be the most challenging organ, improves from 75.3% to 78.1% when using E2ENet compared to nnUNet. For the BraTs challenge, E2ENet demonstrates the most significant improvement in the Dice score of the "ET" region, which is considered the most challenging class, with an increase of 0.7%.

These results indicate that by applying the DSFF mechanism, E2ENet is able to effectively utilize multi-scale information, potentially leading to improved performance in segmenting small organs. It is important to note that other factors, such as the quality and resolution of the medical images, as well as the complexity of the anatomy being imaged, may also impact the performance of the segmentation algorithms. Future work could focus on further exploring the potential impact of these factors on segmentation accuracy.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have claimed that our proposed Efficient to Efficient Network (E2ENet) in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The efficiency in this paper is theoretical, we still need future work for hard work support, which we claim in conclusion.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [No]

Justification: We don't include theoretical results for proof

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide implementation details in A.5, and the code and dataset links in the supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The dataset we use is accessible online, and the code is provided in the supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide implementation details in A.5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: In Section A.8.3, we plot a critical distance diagram using the Nemeny post-hoc test with a p-value of 0.05 to establish the statistical significance of our modules.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the type of hardware used for the implementation in Appendix A.5.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We include border impact in Section A.1.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

Justification: The paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The Licenses names for existing assets are CC-BY 4.0.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: Our paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Our paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.