EffiDec3D: An Optimized Decoder for High-Performance and Efficient 3D Medical Image Segmentation

Supplementary Material

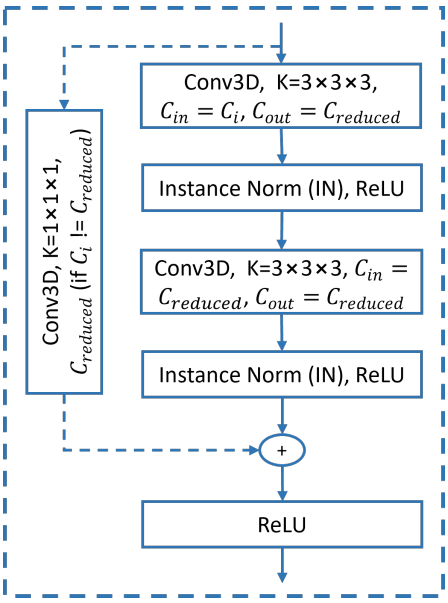


Figure S1. Diagram of Channel Reduction Residual Block described in Section 3.2.

7. Channel Reduction Residual Block Diagram

Figure S1 presents the block diagram of our $ChannelReductionResidualBlock$ described in Section 3.2 of our main paper. As shown, $ChannelReductionResidualBlock$ consists of two consecutive Conv3D layers, each followed by an Instance Norm (IN) and ReLU activation. A residual connection is also used to enable direct information flow from the given input to the output of this layer.

8. Experimental Details

This section extends our Section 4 in the original paper by describing the datasets, implementation details, evaluation metrics, additional experimental results, and qualitative results, followed by comparisons of different computational metrics (#Params, #FLOPs, Inference Time, Throughput).

8.1. Datasets

To evaluate the performance of our EffiDec3D decoder, we carry out experimental analyses across 12 datasets, as described next:

- **FeTA 2021**: The MICCAI 2021 FeTA Challenge dataset (FeTA2021) [17] comprises 80 T2-weighted infant brain MRIs acquired using 1.5T and 3T clinical whole-body scanners at the University Children's Hospital. The dataset focuses on fetal infant brain tissue segmentation, with detailed annotations for seven distinct tissue types (main Table 1). We randomly divide the dataset into 80% training (64 scans), 10% validation (8 scans), and 10% test (8 scans) sets. More details including pre-processing steps are provided in Supplementary Table S1.
- **BTCV:** The Beyond the Cranial Vault (BTCV) abdominal dataset[2] [11] includes 30 abdominal CT scans with annotations for 13 organs, performed by interpreters under the guidance of radiologists from Vanderbilt University Medical Center. Each scan was acquired during the portal venous contrast phase, containing between 80 and 225 slices with a resolution of 512×512 pixels and slice thickness varying from 1 to 6 mm. Following TransUNet [3], we use the same 18 scans for training and 12 scans for validation. Our multi-organ segmentation task involves both 13 (main Table 2) and 8 classes (supplementary Table S3). The pre-processing and other details are in the supplementary Table S1.
- **MSD**: The Medical Segmentation Decathlon (MSD) dataset [1] consists of 10 segmentation tasks (i.e., Brain Tumor, Heart, Liver Tumor, Hippocampus, Prostate, Lung, Pancreas, Hepatic Vessel, Spleen, and Colon segmentation) involving various organs and imaging modalities (i.e., CT and MRI). These tasks are specifically designed to encompass a range of challenges in medical imaging, including limited training data, class imbalance, multi-modality inputs, and the segmentation of small structures. Consequently, the MSD dataset provides a thorough benchmark for assessing the generalizability of medical image segmentation techniques. Specific details with pre-processing pipeline used for these 10 datasets are provided in the supplementary Table S1.

8.2. Implementation Details

We optimize three architectures (3D UX-Net, SwinUNETR, and SwinUNETRv2) incorporating our decoder by replacing their original decoder. All architectures employ a four-stage encoder design, with 3D UX-Net utilizing progressively increasing channel sizes (48, 96, 192, 384), while SwinUNETR and SwinUNETRv2 use the initial feature size of 48 and increase exponentially with the #stages. We set the reduced number of decoder channels $C_{reduced} = 48$ which is the minimum number of channels in the original

[2] https://www.synapse.org/#!Synapse:syn3193805/wiki/217789

Dataset	Anatomical Region	Modality	#Samples	Data Splits	Pre-processing
FeTA 2021 [17]	Infant Brain	MRI	80	Train: 64 Validation: 8 Test: 8	1. Interpolate scans to isotropic voxel spacing of $[0.43\,\mathrm{mm} - 0.70\,\mathrm{mm}]$ 2. Clip brain volumes to [0, 1000] and normalize to $[0, 1]$ 3. Random sampling of $96 \times 96 \times 96$ voxels (1:1 positive-to-negative ratio) 4. Augmentations: random intensity shifting (0.5), affine transformations with scale factor (0.1) and rotation range (0,0,π/15) 5. Labels: External Cerebrospinal Fluid (ECF), Grey Matter (GM), White Matter (WM), Ventricles, Cerebellum, Deep Grey Matter (DGM), Brainstem
BTCV [11]	Abdomen	CT	30	Train: 18 Validation: 12	1. Interpolate scans to isotropic voxel spacing of $[1.5 \times 1.5 \times 2.0]$ mm 2. Clip intensities using soft tissue window [-125, 275], normalize to $[0, 1]$ 3. Random sampling of $96 \times 96 \times 96$ voxels (1:1 positive-to-negative ratio) 4. Augmentations: random flip (0.1), rotation (0.1), intensity shifting (0.5), affine transformations with scale factor (0.1) and rotation range (0,0,π/30) 5. Labels: Spleen, Right kidney, Left kidney, Gallbladder, Esophagus, Liver, Stomach, Aorta, IVC, Veins, Pancreas, Right adrenal glands, Left adrenal glands
Task01 [1]	Brain Tumor	MRI	484	Train: 388 Validation: 96	1. Use all 4 MRI modalities as 4 input channels 2. Normalize non-zero MRI intensities channel-wise 3. Crop sub-volumes of $96 \times 96 \times 96$ 4. Augmentations: random flips (0.5), intensity scaling (0.1), shifting (0.1) 5. Labels: Tumor Core (TC), Whole Tumor (WT), Non-enhancing Tumor (NET)
Task02 [1]	Heart	MRI	20	Train: 16 Validation: 4	1. Interpolate to isotropic voxel spacing $1.0\,\mathrm{mm}$ 2. Normalize non-zero intensities channel-wise 3. Sample $96 \times 96 \times 96$ training sub-volumes (2:1 positive-to-negative ratio) 4. Augmentations: random flip (0.5), rotation (0.1), intensity scaling (0.2), shifting (0.5), affine w/ scale factor (0.1) and rotation range (0,0,π/15) 5. Labels: Left Artium
Task03 [1]	Liver	CT	131	Train: 105 Validation: 26	1. Interpolate scans to isotropic voxel spacing of $1.0\,\mathrm{mm}$ 2. Scale intensities to $[-21, 189]$, normalize to $[0, 1]$ 3. Extract $96 \times 96 \times 96$ patches with 1:1 positive-to-negative sampling ratio 4. Augmentations: random flip (0.2), rotation (0.2), intensity scaling (0.1), shifting (0.1), affine w/ scale factor (0.1) and rotation range (0,0,π/30) 5. Labels: Liver, Cancer
Task04 [1]	Hippocampus	MRI	260	Train: 208 Validation: 52	1. Interpolate voxel spacing to $0.2\,\mathrm{mm} \times 0.2\,\mathrm{mm} \times 0.2\,\mathrm{mm}$ 2. Random spatial sampling of $96 \times 96 \times 96$ (1:1 positive-to-negative ratio) 3. Normalize non-zero intensities channel-wise 4. Augmentations: random flip (0.1), rotation (0.1), intensity scaling (0.1), shifting (0.1), affine w/ scale factor (0.1) and rotation range (0,0,π/15) 5. Labels: Anterior, Posterior
Task05 [1]	Prostate	MRI	32	Train: 26 Validation: 6	1. Use both MRI modalities as two input channels 2. Resample voxel spacing to $0.5\,\mathrm{mm}$, normalize non-zero intensities channel-wise 3. Pad or crop spatial samples of $96 \times 96 \times 96$ (1:1 positive-to-negative ratio) 4. Augmentations: random flip (0.5), rotation (0.5), intensity scaling (0.5), shifting (0.5), affine w/ scale factor $(-0.3, 0.3)$, rotation range $(0, 0, \pi)$ in each axis 5. Labels: Peripheral Zone (PZ), Transition Zone (TZ)
Task06 [1]	Lung	CT	63	Train: 51 Validation: 12	1. Interpolate each scan to isotropic voxel spacing $1.0\,\mathrm{mm}$ 2. Normalize intensities to $[0, 1]$ after clipping HU range $[-1000, 1000]$ 3. Extract $96 \times 96 \times 96$ patches (2:1 positive-to-negative sampling ratio) 4. Data augmentation: random flip (0.5), rotation (0.3), intensity scaling (0.1), shifting (0.1), affine w/ scale factor (0.1), rotation range $(0, 0, \pi/15)$ 5. Labels: Cancer
Task07 [1]	Pancreas	CT	281	Train: 225 Validation: 56	1. Clip intensities to $[-87, 199]$, normalize to $[0, 1]$ 2. Sample patches of $96 \times 96 \times 96$ w/ 1:1 positive-to-negative ratio 3. Augmentations: random flip (0.5), rotation (0.25), intensity scaling, shifting (0.5) 4. Labels: Pancreas, Cancer
Task08 [1]	Hepatic Vessel	CT	303	Train: 243 Validation: 60	1. Clip intensities to $[0, 230]$ and normalize to $[0, 1]$ 2. Sample patches of $96 \times 96 \times 96$ w/ 1:1 positive-to-negative ratio 3. Augmentations: random flip (0.5), rotation (0.25), intensity scaling, shifting (0.5) 4. Labels: Vessel, Tumor
Task09 [1]	Spleen	CT	41	Train: 33 Validation: 8	1. Interpolate voxel spacing to $1.0\,\mathrm{mm}$ on all axes 2. Clip intensities to HU range $[-125, 275]$, normalize to $[0, 1]$ 3. Sample patches of $96 \times 96 \times 96$ w/ 1:1 positive-to-negative ratio 4. Augmentations: random flip (0.1), rotation (0.1), intensity shifting (0.5), affine w/ scale factor (0.1), rotation range $(0, 0, \pi/30)$ 5. Labels: Spleen
Task10 [1]	Colon	CT	126	Train: 101 Validation: 25	1. Interpolate voxel spacing to $1.0\,\mathrm{mm}$ on all axes 2. Clip intensities to $[-57, 175]$ and normalize to $[0, 1]$ 3. Sample training sub-volumes with positive-to-negative ratio of 1:1 4. Augmentations: random flip (0.5), rotation (0.25), intensity scaling (0.2), shifting (0.5), affine w/ scale factor (0.1), rotation range $(0, 0, \pi/15)$ 5. Labels: Colon cancer primaries

Table S1. Details of each dataset in our experiments, including BTCV [11], FeTA 2021 [17], and 10 MSD [1] datasets. Center crop of the foreground is applied in all datasets.

Hyperparameters	3D UX-Net	SwinUNETR	SwinUNETRv2
Encoder Stage	4	4	4
Layer-wise #Channel / Feature Size	48, 96, 192, 384	48	48
Patch Size	$96 \times 96 \times 96$	$96 \times 96 \times 96$	$96 \times 96 \times 96$
No. of Sub-volumes Cropped	2	2	2
Training Steps^{+}	45000	45000	45000
Batch Size	1	1	1
AdamW ϵ	1×10^{-8}	1×10^{-8}	1×10^{-8}
AdamW β	(0.9, 0.999)	(0.9, 0.999)	(0.9, 0.999)
Learning Rate*	1×10^{-4}	1×10^{-4}	1×10^{-4}
Weight Decay	0.08	0.08	0.08
Loss Function	DICE + CrossEntropy	DICE + CrossEntropy	DICE + CrossEntropy

Table S2. Hyperparameters for different architectures in our experiments. *We use a learning rate of 1×10^{-3} for BTCV, Task01_Heart, Task05_Prostate, and Task06_Lung datasets based on dataset-specific optimization for all architectures; in addition, we use 1×10^{-3} learning rate for Task04_HippoCampus and Task07_Pancreas with 3D UX-Net architecture. $^{+}$ For Task01_BrainTumour dataset, we use training steps of 60000. **We use the same hyperparameters for the optimized architectures incorporating our EffiDec3D.**

decoder across stages. We also remove the high-resolution stage (D, H, W) and output at $(D/2, H/2, W/2)$ resolution from our decoder for all three optimized architectures. We apply *Softmax* activation followed by *Argmax* to the output from the decoder, except the Task01_BrainTumor segmentation (where we use *Sigmoid* activation followed by a 0.5 threshold for multi-level classification).

The input patch size is set to $96 \times 96 \times 96$, with 2 cropped sub-volumes from each dataset except BTCV. All our models are trained 45,000 steps for most tasks, with an extended 60,000 steps used for Task01_BrainTumour due to its complexity. We saved the best model based on validation DICE scores. A learning rate of 1×10^{-4} is used for most datasets except BTCV, Task01_Heart, Task05_Prostate, and Task06_Lung (learning rate of 1×10^{-3}). The AdamW [14] optimizer with $\epsilon = 1 \times 10^{-8}$, $(\beta_1, \beta_2) = (0.9, 0.999)$, and a weight decay of 0.08 is utilized for optimization. For loss calculation, a combination of DICE and CrossEntropy functions (Eq. 10) is utilized to balance segmentation overlap and accuracy:

$$\mathcal{L} = \mathcal{L}_{\text{DICE}} + \mathcal{L}_{\text{CE}} \tag{10}$$

where $\mathcal{L}_{\text{DICE}}$ is used to optimize the segmentation overlap, and $\mathcal{L}_{\text{CE}}$ is used to handle the pixel-wise classification.

Detailed training hyperparameters can be found in the supplementary Table S2. Notably, these hyperparameters are consistent across both original and EffiDec3D-optimized decoders. We have implemented our models in PyTorch and MONAI[3]. All models are trained on a NVIDIA RTX 6000 (Ada generation) or A6000 server with 48GB of GPU memory.

8.3. Evaluation metrics

To evaluate the segmentation performance across all datasets, we primarily utilize the DICE score, while incorporating the 95% Hausdorff Distance (HD95) as an additional metric for the BTCV 8-organ segmentation. The DICE score, denoted as $DICE(Y, \hat{Y})$, quantifies the overlap between the predicted segmentation $\hat{Y}$ and the ground truth Y, whereas HD95 measures the boundary alignment by computing the 95th percentile of distances between the points on the predicted and ground truth boundaries.

The DICE score $DICE(Y, \hat{Y})$, and HD95 distance $HD(Y, \hat{Y})$ are defined in Eqs. 11 and 12, respectively:

$$DICE(Y, \hat{Y}) = \frac{2 \times |Y \cap \hat{Y}|}{|Y| + |\hat{Y}|} \times 100 \tag{11}$$

$$HD(Y, \hat{Y}) = \max \left\{ \max_{y \in Y} \min_{\hat{y} \in \hat{Y}} d(y, \hat{y}), \max_{\hat{y} \in \hat{Y}} \min_{y \in Y} d(y, \hat{y}) \right\} \tag{12}$$

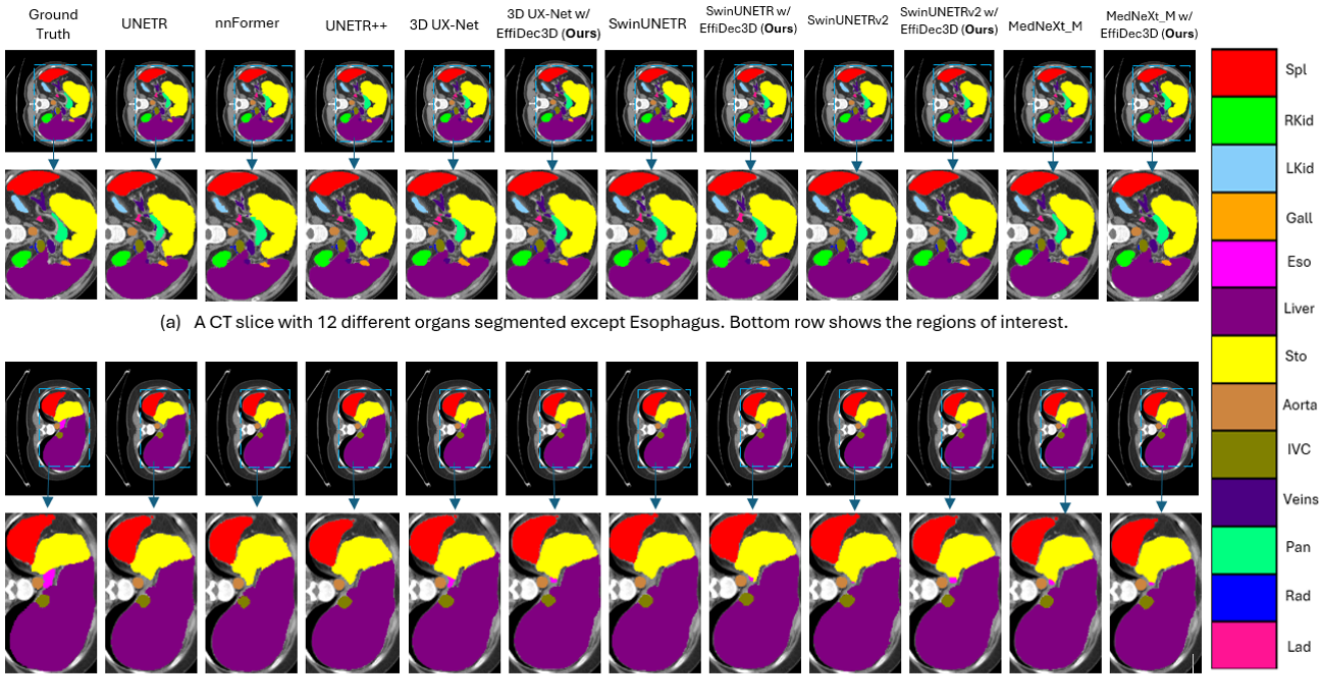
where, Y and $\hat{Y}$ denote the ground truth and predicted segmentation maps, respectively, and $d(y, \hat{y})$ represents the Euclidean distance between the boundary points. The DICE score provides a measure of overlap, while HD95 captures the accuracy of boundary alignment, thus offering complementary insights into the segmentation performance.

8.4. Qualitative Results

Figure S2 presents qualitative segmentation results on two different CT slices from the BTCV dataset for various methods, including our optimized EffiDec3D decoders. Figure S2a shows an example where 12 different organs are segmented, excluding the esophagus, while Figure S2b contains an example where 6 organs including the esophagus are segmented.

In general, UNETR, nnFormer, and TransBTS exhibit comparatively worse performance, missing critical regions and showing less accurate segmentation boundaries. For instance, in Figure S2a, most methods, including these three, fail to accurately segment parts of the veins, which are particularly challenging due to their small size and heterogeneous presence. Furthermore, UNETR, SwinUNETR, and

[3] https://monai.io/



(a) A CT slice with 12 different organs segmented except Esophagus. Bottom row shows the regions of interest.

(b) A CT slice with 6 different organs segmented including Esophagus. Bottom row shows the regions of interest.

Figure S2. Qualitative results of 13-organ segmentation on BTCV dataset, (a) 12 different organs segmented except Esophagus, (b) 6 different organs segmented including Esophagus. The dashed rectangular blue box highlights the region of interest having different organs. As shown, our optimized EffiDec3D-based models produce competitive segmentation results to the original decoders with substantially lower #Params and #FLOPs. For example, SwinUNETR completely misses the Esophagus, whereas our SwinUNETR with EffiDec3D successfully segments parts of it. **Note**: Spl: spleen, RKid: right kidney, LKid: left kidney, Gall: gallbladder, Eso: esophagus, Sto: stomach, IVC: inferior vena cava, Veins: portal and splenic veins, Pan: pancreas, Lad: left adrenal glands, Rad: right adrenal glands.

optimized SwinUNETR demonstrate mis-segmentation for the right adrenal gland (Rad) in the same figure.

In Figure S2b, SwinUNETR completely misses the esophagus, a small, rarely seen (in the whole volume), and difficult-to-detect organ, whereas our EffiDec3D optimized SwinUNETR successfully segments parts of it. This improvement highlights the advantages of EffiDec3D's optimization in preserving segmentation accuracy for challenging anatomical structures.

Overall, our EffiDec3D decoder-based optimized models consistently outperform the original models (Figure S2) and other methods in capturing both large and small structures, even with significant computational savings. This demonstrates the capability of EffiDec3D to improve segmentation performance while maintaining efficiency across diverse and challenging clinical tasks, even with substantial reductions in #Params and #FLOPs.

8.5. BTCV 8-organ Segmentation Results

Table S3 presents a comprehensive evaluation of DICE scores (%) across various 2D and 3D architectures for the BTCV 8-organ segmentation task. The methods in the top six rows represent 2D architectures, such as UNet and AttUNet, which process slice-by-slice inputs at resolutions of 224×224. Among these, PVT-EMCAD-B2 achieves the highest average DICE score (83.63%), using transformer-based feature extraction. However, due to their inability to capture volumetric context, 2D models generally underperform compared to 3D models, especially for organs with complex 3D structures like the pancreas and gallbladder.

The remaining rows in Table S3 showcase 3D architectures, which use volumetric inputs ($96 \times 96 \times 96$) to capture the spatial context more effectively. Among the original 3D models, MedNeXt_M_K3 and 3D UX-Net achieve average DICE scores of 86.22% and 85.33%, respectively. However, the computational cost of MedNeXt_M_K3 (110.65 GFLOPs and 17.55M Params) and 3D UX-Net (631.97 GFLOPs and 53M Params) is extremely high, which makes them impractical for resource-constrained settings. Similarly, SwinUNETR and SwinUNETRv2 demonstrate the strengths of self-attention mechanisms by achieving competitive DICE scores of 84.64% and 85.67%, respectively, with substantial computational requirements.

Our proposed EffiDec3D decoder applied to these 3D architectures offers substantial computational savings with minimal performance loss. For instance, the integration of EffiDec3D with 3D UX-Net reduces #Params and #FLOPs by 94.0% and 91.9%, respectively, while only slightly re-

Architecture	Inp. Res.	MParams↓	GFLOPs↓	Spl.	RKid	LKid	Gall	Pan.	Liver	Sto.	Aorta	Avg (%)↑	HD95↓
UNet [23]	224×224	34.53	65.53	81.48	62.64	72.41	56.70	48.73	86.98	67.96	84.00	70.11	44.69
AttUNet [15]	224×224	34.88	66.64	80.67	70.42	76.07	61.94	46.70	87.54	67.66	82.61	71.70	34.47
TransUNet [3]	224×224	105.32	38.52	87.06	78.53	80.54	60.43	58.47	94.33	75.00	86.56	77.61	26.90
SwinUNet [2]	224×224	27.17	6.20	88.04	79.22	82.32	65.95	53.81	93.73	75.79	81.76	77.58	27.32
PVT-CASCADE [19]	224×224	34.12	7.62	90.10	80.37	82.23	70.59	64.43	94.08	83.69	83.01	81.06	20.23
PVT-EMCAD-B2 [22]	224×224	26.76	5.60	92.17	84.10	**88.08**	68.87	68.51	95.26	**83.92**	88.14	83.63	15.68
SlimUNETR [16]	$96 \times 96 \times 96$	**1.79**	20.17	85.90	82.85	86.15	73.33	58.38	95.68	73.79	87.31	80.42	10.12
SegFormer3D [18]	$96 \times 96 \times 96$	4.50	**5.03**	89.28	83.43	85.48	70.12	64.10	94.76	77.50	88.63	81.66	9.23
UNETR [8]	$96 \times 96 \times 96$	92.78	82.60	89.80	83.12	86.68	71.58	62.33	95.80	77.23	89.25	81.97	10.03
nnUNet [7]	$96 \times 96 \times 96$	31.78	417.96	91.13	84.35	84.67	70.76	63.73	96.09	74.85	91.32	82.11	11.49
nnFormer [31]	$96 \times 96 \times 96$	149.25	213.60	**93.49**	85.01	85.71	76.43	66.85	95.67	82.87	90.45	84.56	9.08
TransBTS [27]	$96 \times 96 \times 96$	31.58	110.34	90.38	84.94	86.10	71.53	68.91	96.53	80.90	90.05	83.67	9.77
UNETR++ [25]	$96 \times 96 \times 96$	42.62	53.99	85.26	84.88	86.31	76.08	68.19	95.87	72.57	90.06	82.40	8.82
MedNeXt_M_K3 [24]	$96 \times 96 \times 96$	17.55	110.65	91.26	85.14	87.11	75.59	**78.31**	**96.81**	83.61	91.96	**86.22**	6.62
MedNeXt_M_K3 w/ **EffiDec3D (Ours)**	$96 \times 96 \times 96$	5.77	49.73	89.80	86.34	87.21	79.25	73.64	96.18	82.89	**92.10**	85.93	7.83
3D UXNet [12]	$96 \times 96 \times 96$	53.00	631.97	90.34	85.48	86.60	**79.74**	73.46	96.55	78.88	91.62	85.33	8.82
3D UXNet w/ **EffiDec3D (Ours)**	$96 \times 96 \times 96$	3.16	51.47	87.32	85.57	86.71	75.87	71.07	96.02	79.48	91.31	84.17	9.42
SwinUNETR [26]	$96 \times 96 \times 96$	69.19	337.61	87.53	85.37	85.98	77.49	73.36	95.72	78.57	91.33	84.42	9.93
SwinUNETR w/ **EffiDec3D (Ours)**	$96 \times 96 \times 96$	11.21	57.29	90.79	84.94	86.91	75.72	71.50	96.45	79.08	91.75	84.64	9.21
SwinUNETRv2 [9]	$96 \times 96 \times 96$	83.19	353.61	89.65	**86.80**	86.51	78.06	74.51	96.58	81.57	91.71	85.67	8.39
SwinUNETRv2 w/ **EffiDec3D (Ours)**	$96 \times 96 \times 96$	18.21	63.29	88.47	85.97	87.60	78.70	71.88	96.35	80.89	91.51	85.17	**5.93**

Table S3. BTCV 8-organ segmentation DICE scores (%) and HD95 distances using TransUNet's 60:40 train-validation split. The #FLOPs are reported for a one-channel input of resolution 224×224 (2D) and $96 \times 96 \times 96$ (3D) and 9 output classes. The original MedNeXt_M_K3, 3D UX-Net, SwinUNETR, and SwinUNETRv2 are compared side by side with our optimized versions. We report the results of different 2D methods such as UNet, AttUNet, TransUNet, SwinUNet, PVT-CASCADE, and PVT-EMCAD-B2 from the EMCAD [22] paper. We reproduce the results of different 3D methods such as UNETR, nnUNet, nnFormer, TransBTS, UNETR++, SlimUNETR, SegFormer3D, MedNeXt_M_K3 (kernel=3), 3D UX-Net, SwinUNETR, and SwinUNETRv2. **Bold** and underlined are the best and second-best values in each column. **Note**: Inp. Res.: Input Resolution, Spl: spleen, RKid: right kidney, LKid: left kidney, Gall: gallbladder, Sto: stomach, Pan: pancreas. Only DICE scores (%) are reported for individual organs.

Architecture	Params↓	FLOPs↓	Infer. Time (ms)↓		M. (GB)↓	Thrgh. (/s)↑		Avg. DICE (%)↑					
	(M)	(G)	GPU	CPU	GPU	GPU	CPU	BTCV13	BTCV8	FeTA	BraT	Lung	Heart
SlimUNETR [16]	**1.79**	20.17	8.90	184.14	**0.115**	112.37	5.43	72.56	80.42	82.70	72.66	67.66	90.42
SegFormer3D [18]	4.50	**5.03**	**3.93**	**107.85**	0.165	**254.42**	**9.27**	74.34	81.66	86.57	73.85	55.07	91.64
UNETR	92.78	82.70	11.56	515.65	0.772	86.49	1.94	74.96	81.97	84.19	75.69	65.38	91.42
nnUNet[4][7]	31.78	417.96	23.97	829.09	1.290	41.72	1.21	77.82	82.11	84.57	74.37	53.14	91.89
nnFormer	149.32	273.41	20.17	723.63	0.935	49.58	1.38	78.28	84.56	87.03	74.79	69.79	92.21
TransBTS	31.58	110.66	14.49	687.19	0.514	69.01	1.46	78.87	83.67	87.18	77.82	63.57	90.12
UNETR++ [25]	42.62	53.99	26.69	592.42	0.499	37.47	1.69	80.49	82.40	86.72	77.16	76.09	92.39
MedNeXt_M_K3 [24]	17.55	110.65	49.07	3417.44	1.250	20.38	0.29	**82.98**	**86.22**	87.74	78.71	74.38	**92.82**
MedNeXt_M_K3 w/ **EffiDec3D (Ours)**	5.77	49.73	20.32	1629.59	0.763	49.21	0.62	82.55	85.93	87.54	77.81	**76.12**	92.57
3D UX-Net	53.01	632.25	48.11	1643.39	1.441	20.79	0.61	79.74	85.33	87.28	78.58	71.46	92.03
3D UX-Net w/ **EffiDec3D (Ours)**	3.16	51.53	10.72	205.10	0.215	93.25	4.88	79.25	84.17	**87.97**	78.06	74.14	92.63
SwinUNETR	62.19	329.20	47.98	1985.09	1.512	20.84	0.50	80.13	84.42	86.71	**79.06**	65.12	91.92
SwinUNETR w/ **EffiDec3D (Ours)**	11.21	57.88	24.77	1092.01	1.323	40.37	0.92	79.82	84.64	87.50	78.79	73.19	91.98
SwinUNETRv2	80.73	356.74	49.60	2116.74	1.585	20.16	0.47	81.26	85.67	87.29	78.78	73.52	91.96
SwinUNETRv2 w/ **EffiDec3D (Ours)**	21.79	83.70	26.07	1170.65	1.362	38.36	0.85	80.52	85.17	87.91	78.55	75.19	92.29

Table S4. Computational complexity comparison of SOTA methods with performance: *Params* (M), FLOPs (G), *Inference Time* (Infer. Time (ms)), GPU *Memory* (M. (GB)), and *Throughput* (Thrgh. (/s)) on BTCV13, BTCV8, FeTA, MSD Task01 Brain Tumour (BraT), MSD Task06 Lung (Lung), and MSD Task02 Heart (Heart) datasets. All the results are produced following the experimental setup in 3D UX-Net for fair comparison. The *Infer. Time* and *Thrgh.* are reported for only the forward pass of a 96^3 input averaging over 200 iterations on a NVIDIA RTX 6000 (Ada) GPU with 48GB memory and an AMD EPYC 9554 (Zen 4) CPU with 128 cores (256 threads), 1.5–3.76 GHz, and 512 MB total L3 cache. *M.* is the allocated peak GPU memory by Pytorch during the forward pass. Best and second-best values are in **bold** and underlined, respectively.

ducing the DICE score from 85.33% to 84.17%. Similarly, EffiDec3D reduces SwinUNETRv2's #Params and #FLOPs by 78.1% and 82.1%, respectively, however, achieves a competitive average DICE score of 85.17% with minimal reduction (0.5%). Notably, the optimized models exhibit improved boundary precision, as reflected by lower Hausdorff distances (e.g., SwinUNETRv2 w/ EffiDec3D achieves the best Avg. HD of 5.93), which further underscores the effectiveness of EffiDec3D.

The results highlight the trade-offs between computational efficiency and segmentation performance. While EffiDec3D maintains comparable performance across most organs, slight reductions are observed in high-resolution-

[4]The performance of nnUNet [7] is reproduced with similar experimental setups as ours (avoiding pre- and post-processing used in nnUNet framework) for fair comparisons of architectures.

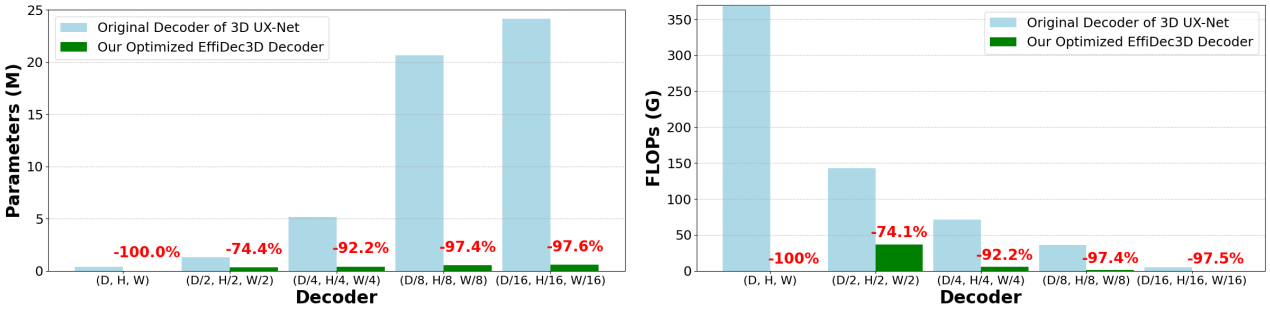


Figure S3. Resolution-wise computational complexity comparison between original and optimized decoders (as described in main paper Section 3.2) for the 3D UX-Net [12] architecture. #Params (left) and #FLOPs (right) for different resolution outputs. As shown, our optimized decoder significantly reduces #FLOPs and #Params for each output resolution.

dependent structures such as the pancreas and gallbladder. Nevertheless, these losses are offset by superior boundary precision and significant computational savings which make EffiDec3D a robust and efficient solution for deployment in resource-constrained environments.

8.6. Computational Complexity Comparison

We show detailed comparisons of GPU and CPU inference time, throughput, and GPU memory usage in Table S4. Notably, when incorporating EffiDec3D into 3D UX-Net, the GPU inference time is reduced from 48.11 ms to 10.72 ms (4.5× faster), and the CPU inference time decreases from 1643.39 ms to 205.10 ms (8× faster). The significant reduction in inference time and increase in throughput can also be observed in the case of our EffiDec3D-optimized MedNeXt_M, SwinUNETR, and SwinUNETRv2.

9. Additional Ablation Study

This section further elaborates on Section 5 by detailing four additional ablation studies related to our architectural design and experimental setup.

9.1. Resolution-wise Complexity Comparison

Figure S3 shows the resolution-wise computational complexity reductions achieved by our optimized decoder for different output resolutions in the 3D UX-Net architecture. The left plot shows the dramatic reduction in #Params, with reductions ranging between 92.2% and 100% across most resolutions. At the highest resolution (D, H, W), our optimized decoder eliminates the high-resolution layers, thus achieving a 100% reduction in #Params compared to the original decoder. Even at intermediate resolutions $(D/8, H/8, W/8$ and $D/16, H/16, W/16)$, the reduction remains substantial (exceeding 97%) due to using a reduced number of decoder channels.

Similarly, the right plot highlights the #FLOPs reduction, following a comparable trend. Reductions range predominantly between 92.2% and 100%, with the optimized decoder eliminating the highest computational costs associated with full-resolution operations. Notably, only at resolution $(D/2, H/2, W/2)$, the reduction is slightly lower (74.1%), but this remains an outlier compared to the significant reductions achieved at other resolutions due to using a reduced number of decoder channels.

These results underscore the profound impact of our optimization strategy on computational efficiency. The consistent removal of high-resolution layers and reduction of decoder channel counts enables significant savings in computational complexity, with most reductions exceeding 90%. This efficiency makes the optimized decoder eminently suitable for deployment in resource-constrained environments, such as those with limited GPU memory or computational power, without requiring high-resolution operations that contribute minimally to segmentation quality. The trends highlighted in Figure S3 strongly support the practicality and scalability of the proposed optimization strategy for efficient 3D medical image segmentation tasks.

9.2. Effect of Skip Aggregation

Table S5 evaluates the impact of skip aggregation strategies, namely Addition and Concatenation, on computational efficiency and segmentation performance across three architectures (3D UX-Net with EffiDec3D, SwinUNETR with EffiDec3D, and SwinUNETRv2 with EffiDec3D) on four tasks: BTCV 13-organ, BTCV 8-organ, FeTA, and MSD Task01 Brain_Tumour.

While Addition aggregation consistently achieves lower computational complexity, with reductions in both #Params and #FLOPs (e.g., from 51.47G to 43.04G #FLOPs in 3D UX-Net), Concatenation aggregation generally delivers better segmentation accuracy. For example, in optimized SwinUNETRv2, Concatenation aggregation achieves a DICE score of 85.17% on BTCV 8-organ segmentation, slightly outperforming Addition (85.12%).

Similarly, for MSD Task01 Brain_Tumour segmentation with optimized 3D UX-Net, Concatenation aggregation improves the DICE score from 77.65% (Addition aggregation)

Architecture	Aggregation	#Params (M) ↓	#FLOPs (G) ↓	BTCV13	BTCV8	FeTA	Task01_BrainTumour
3D UX-Net w/ EffiDec3D	Addition	**2.95**	**43.04**	**79.25**	**84.17**	87.91	77.65
3D UX-Net w/ EffiDec3D	Concatenation	3.15	51.47	79.10	83.93	87.97	78.06
SwinUNETR w/ EffiDec3D	Addition	**10.73**	**48.84**	79.28	84.02	**87.50**	78.60
SwinUNETR w/ EffiDec3D	Concatenation	10.99	57.29	**79.82**	**84.52**	87.38	**78.79**
SwinUNETRv2 w/ EffiDec3D	Addition	**21.32**	**74.84**	80.36	85.12	**87.93**	78.50
SwinUNETRv2 w/ EffiDec3D	Concatenation	21.59	83.26	**80.52**	**85.17**	87.91	**78.55**

Table S5. Comparison of Skip Aggregation (Addition vs. Concatenation) on BTCV 13-organ, BTCV 8-organ, FeTA and MSD Task01 Brain_Tumour segmentation tasks. The average DICE scores (%) are reported for each segmentation task. The #FLOPs are reported for a one-channel input of resolution $96 \times 96 \times 96$ and 14 output classes. **Bold** are the best values for an aggregation in each architecture.

Configurations	CR	HRR	#Params (M)	#FLOPs (G)	BTCV13	FeTA
3D UX-Net (Original decoder)	No	No	51.69	624.16	79.74	87.28
No CR but with HRR	No	Yes	51.28 (-0.7%)	255.45 (-59.1%)	79.40 (-0.34)	87.72 (+0.44)
With CR but No HRR	Yes	No	3.55 (-93.1%)	404.40 (-35.2%)	79.65 (-0.09)	87.91 (+0.63)
3D UX-Net w/ EffiDec3D (**Ours**)	Yes	Yes	**1.84 (-96.4%)**	**43.66 (-93.0%)**	79.25 (-0.49)	87.97 (+0.69)

Table S6. Effect of Channel Reduction (CR) and High-resolution Removal (HRR) strategy on #Params, #FLOPs, and DICE scores on BTCV 13-organ and FeTA segmentation. We use the 3D UX-Net as our base network and report only the decoder #Params and #FLOPs. The percentage (%) reduction (-) in #Params and #FLOPs compared to the original decoder of 3D UX-Net is presented inside parentheses. The reduction (-) and increase (+) in DICE scores is reported also inside parentheses. **Bold** highlights the lowest #Params and #FLOPs.

Decoder	MParams ↓	GFLOPs ↓
SwinUNETR Decoder [26]	62.09	316.78
EffiDec3D (SwinUNETR)	**3.15 (-94.9%)**	**43.74 (-86.2%)**
SwinUNETRv2 Decoder [9]	62.09	316.78
EffiDec3D (SwinUNETRv2)	**3.15 (-94.9%)**	**43.74 (-86.2%)**
3D UX-Net Decoder [12]	51.69	624.16
EffiDec3D (3D UX-Net)	**1.84 (-96.4%)**	**43.66 (-93.0%)**

Table S7. Comparison of decoder computational complexity (reductions in red) between original and our optimized models. **Bold** highlights results of our EffiDec3D optimized models.

to 78.06%. Although the accuracy gains with Concatenation are modest, they underscore the importance of prioritizing segmentation performance, especially in clinical applications where accuracy is critical. These results suggest that Concatenation aggregation may be preferred when computational resources are available, as it delivers consistent improvements in segmentation accuracy, albeit at the expense of comparatively higher computational complexity.

9.3. Effect of Optimization Strategies on #Params, #FLOPs, and Segmentation Performance

Table S6 highlights the effects of our two optimization strategies (Channel Reduction (CR) and High-Resolution Removal (HRR)) on the computational complexity and segmentation accuracy of the 3D UX-Net decoder across the BTCV 13-organ and FeTA datasets. The original decoder, with neither CR nor HRR, exhibits the highest computational demands, requiring 51.69M #Params and 624.16G #FLOPs, with DICE scores of 79.74% and 87.28% for BTCV 13-organ and FeTA, respectively.

Incorporating HRR alone reduces #FLOPs by 59.1% (to 255.45G) and #Params by only 0.7% (to 51.28M), with minor accuracy trade-offs (-0.34% on BTCV and +0.44% on FeTA). When CR is applied without HRR, #Params are reduced significantly by 93.1% (to 3.55M), and #FLOPs drop by 35.2% (to 404.4G), while maintaining competitive DICE scores (-0.09% for BTCV and +0.63% for FeTA).

Finally, applying both CR and HRR in the EffiDec3D decoder achieves the largest reductions, with #Params reduced by 96.4% (to 1.84M) and #FLOPs by 93.0% (to 43.66G). Despite these dramatic reductions, EffiDec3D maintains comparable segmentation performance (-0.49% on BTCV and +0.69% on FeTA), which demonstrates its effectiveness in significantly lowering the computational demands while sustaining high accuracy.

9.4. Decoder Complexity Comparisons

Table S7 compares the computational complexity of the original and EffiDec3D-optimized decoders for SwinUNETR, SwinUNETRv2, and 3D UX-Net. EffiDec3D achieves substantial reductions across all models: a 94.9% decrease in parameters (from 62.09M to 3.15M) and an 86.2% drop in FLOPs (from 316.78 to 43.74 GFLOPs) for SwinUNETR and SwinUNETRv2.

For 3D UX-Net, #Params are reduced by 96.4% (from 51.69M to 1.84M) and #FLOPs by 93.0% (from 624.16 to 43.66 GFLOPs). These reductions enhance model efficiency, making these architectures more suitable for real-time and resource-constrained settings.