Neuroverse3D: Developing 3D In-Context Learning Universal Model in Neuroimaging

Supplementary Material

A. Method

A.1. Gradient Equivalence

Here, we prove that the expected value of the gradient using Adaptive Parallel-Sequential Processing (APSP) is the same as the full-context gradient when no mini-contexts are discarded. The full-context gradient, exemplified by the i-th stage output $\bar{c}^i_{\text{dec},j}$, with respect to parameters θ is defined as:

$$\nabla_\theta \mathcal{L}^i_{\text{full}} = \nabla_\theta \sum_{j=1}^{n} \frac{\text{len}(s_j)}{L} \bar{c}^i_{\text{dec},j}, \tag{8}$$

where $L = \sum_{j=1}^{n} \text{len}(s_j)$ ensures equal weighting of image-label pairs across mini-contexts. For brevity, we focus on the i-th stage, with the understanding that the analysis applies analogously to other stages.

In APSP, mini-contexts are randomly shuffled, and we select the last mini-context to compute the gradient. Numerically, this is equivalent to randomly selecting a mini-context with index π from $\{1, 2, ..., n\}$ with a uniform distribution, denoted as $\pi \sim \mathcal{U}\{1, ..., n\}$. The APSP gradient using only the π-th mini-context is:

$$\nabla_\theta \mathcal{L}^i_{\text{APSP},\pi} = \nabla_\theta \frac{\text{len}(s_\pi)}{L} \bar{c}^i_{\text{dec},\pi}. \tag{9}$$

To ensure the expected APSP gradient matches the full-context gradient, during gradient computation we scale $\bar{c}^i_{\text{dec},\pi}$ by a factor of n, making $\nabla_\theta \mathcal{L}^i_{\text{APSP-scaled},\pi} = \nabla_\theta \frac{\text{len}(s_\pi)}{L} n \bar{c}^i_{\text{dec},\pi}$ the scaled gradient for the π-th mini-context. It is important to note that the scaling factor of n is exclusively applied during the gradient computation phase and is omitted during the forward computation, thus having no impact on the forward computation of the model. The expected APSP gradient with scaling is:

$$\begin{aligned}
\mathbb{E}_\pi \left[\nabla_\theta \mathcal{L}^i_{\text{APSP-scaled},\pi} \right] &= \mathbb{E}_\pi \left[\nabla_\theta \frac{\text{len}(s_\pi)}{L} n \bar{c}^i_{\text{dec},\pi} \right] \\
&= \sum_{j=1}^{n} P(\pi = j) \left[n \nabla_\theta \frac{\text{len}(s_j)}{L} \bar{c}^i_{\text{dec},j} \right] \\
&= \sum_{j=1}^{n} \frac{1}{n} \left[n \nabla_\theta \frac{\text{len}(s_j)}{L} \bar{c}^i_{\text{dec},j} \right] \\
&= \sum_{j=1}^{n} \nabla_\theta \frac{\text{len}(s_j)}{L} \bar{c}^i_{\text{dec},j} \\
&= \nabla_\theta \sum_{j=1}^{n} \frac{\text{len}(s_j)}{L} \bar{c}^i_{\text{dec},j} \\
&= \nabla_\theta \mathcal{L}^i_{\text{full}}.
\end{aligned} \tag{10}$$

Equation (10) shows that the expected APSP gradient equals the full-context gradient $\nabla_\theta \mathcal{L}^i_{\text{full}}$. This analysis, exemplified for the i-th stage, extends to all network stages and parameters.

Thus, by employing the gradient of only the last mini-context and scaling by n, APSP preserves the full-context gradient expectation while enabling memory-efficient computation.

A.2. U-Shape Fusion Strategy Explanation

In Figure 8, (a) illustrates our proposed U-Shape fusion strategy, while (b) shows a fusion strategy with alternating feature transmission, similar to those presented in [12] and [14] for comparison in the following analysis. The red semi-transparent arrows indicate the order of computation for feature maps in the network.

In case (b), there is a problematic trade-off between memory usage and computational cost. For example, when computing layer 2 of the target branch, it is necessary to calculate the feature maps for all image-label pairs in the context branch at layer 2, which are then passed to the target branch. After this, two options arise:

1. Retaining the context branch layer 2 features for subsequent computations requires storing feature maps for all image-label pairs across all mini-contexts during sequential processing, which significantly increases memory usage.
2. Discarding the context branch layer 2 features saves memory but requires recomputing them for subsequent layers, which substantially increases computational cost.

Both options severely hinder efficient sequential processing. In contrast, strategy (a) avoids these issues. For each image-label pair, all required context representations are obtained in a single pass, allowing computed feature maps to be discarded during sequential processing, thereby ensuring computational efficiency.

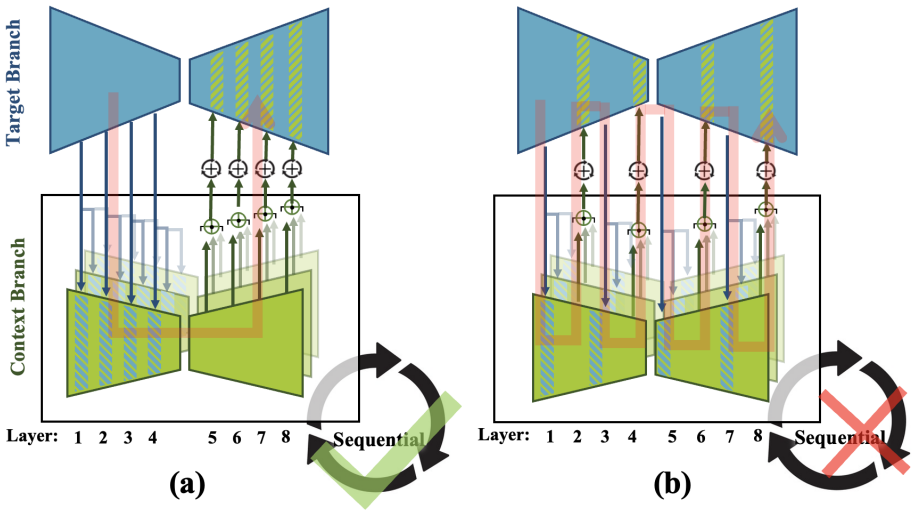

Figure 8. Illustration of different fusion strategies. (a) The proposed U-Shape fusion strategy. (b) The alternating fusion strategy. The red semi-transparent arrows indicate the computation order of the feature maps in the network.

B. Data

B.1. Domain Shift Between Training and Test Data

We trained a discriminator to distinguish between images from the training set and those from the held-out set, achieving accuracy, precision, recall, and F1 scores of 0.934, 0.932, 0.918, and 0.925, respectively. These results serve as proxy evidence for the presence of domain shift.

B.2. Data Sampling

During training, tasks are selected based on a predefined sampling rate, followed by the random selection of a dataset within the chosen task with equal probability. Given a context size L, we randomly sample $L + 1$ image-label pairs from the training set of the selected dataset. One of these samples is alternately designated as the target image and ground truth, while the remaining L samples serve as the context set for the model. This process generates $L + 1$ unique target image and corresponding context set, significantly reducing the time consumption associated with data I/O.

Since the model is designed for binary classification, for multi-class segmentation datasets, we iteratively select each class as the foreground and the remaining classes as the background during training.

In generation tasks, we simulate various scenarios. For bias field correction, 3D bias fields are generated using Legendre polynomials with random coefficients. Gaussian noise removal involves simulating noise with a mean of 0 and a standard deviation randomly selected within the range 0.15 to 0.25. For salt-and-pepper noise removal, noise is applied with random probabilities equal to 0.04, where salt noise (value of 1) and pepper noise (value of 0) are added separately. For the inpainting task, binary masks are created using random 3D Perlin noise to occlude specific regions of the input image. The 2D-to-3D task requires the model to reconstruct a complete 3D brain volume from only three central brain slices, with this task restricted

Type for use	Dataset	Task	# Scans	# Masks	Modality
Training and Validation Set	TopCow[59]	Seg., Gen.	90	90	MRA
	CAS2023[4]	Seg., Gen.	100	100	MRA
	ISLES2022[24]	Gen., Mod.	750	0	DWI, ADC, FLAIR
	ATLAS[39]	Seg., Gen.	655	655	T1w
	IXI[3]	Gen., Mod.	2268	0	T1, T2, MRA, PD
	ICH Unlabeled[16]	Gen.	2000	0	CT
	ADHD[1]	Gen.	950	0	T1
	ADNI[35]	Gen., Mod.	9923	0	T1
	CMI[5]	Gen.	5146	0	T1
	GSP[26]	Gen.	2616	0	T1
	HAB[18]	Seg., Gen.	460	460	T1
	NIMH[49]	Seg., Gen.	248	248	T1
	OASIS[46]	Gen.	3916	828	T1
	UKBiobank[52]	Seg., Gen.	4000	2000	T1, T2
	BraTS[48]	Seg., Gen., Mod.	5004	1251	FLAIR, T1, T1CE, T2
Held-out Set	WMH[37]	Mod.	120	0	T1, FLAIR
	CCNP[43]	Gen.	1580	0	T1
	FCON1000[2]	Seg., Gen.	1096	1096	T1
	PPMI[47]	Gen.	2752	0	T1
	Total	Seg., Gen., Mod.	43674	6728	T1, T2, FLAIR, MRA, DWI, ADC, PD, CT

Table 2. Summary of Datasets. Seg., Gen., and Mod. represent segmentation, generation (excluding modality transformation), and modality transformation tasks, respectively.

to images in MNI space. In the super-resolution task, images are downsampled by a factor of 2. For skull stripping, input images include the skull, while ground truth images, with the skull removed, are generated using FreeSurfer [15]. Lastly, the modality transformation task uses registered pairs of different imaging modalities as input and output.

Figure 9 illustrates the target image, ground truth, and context set (with two image-label pairs shown as examples) for all tasks. The model takes the target image and context set as input, using the context set to infer the required task.

Task	Sampled Rate	Weight
Segmentation	2	50
Bias Remove	1	1
Gaussian Noise Remove	1	1
Salt & Pepper Noise Remove	1	1
2D to 3D Generation	1	0.5
Inpainting	1	1
Super-Resolution	1	1
Skull Stripping	1	1
Modality Transform	1	1

Table 3. Sampling rate and weight assigned to each task when training.

B.3. Image and Task Augmentation

Image and task augmentation are performed after obtaining data sampling. For image augmentation, we applied the following transformations: random affine transformations (p=0.05), elastic deformations (p=0.05), flips (p=0.05), and rotations

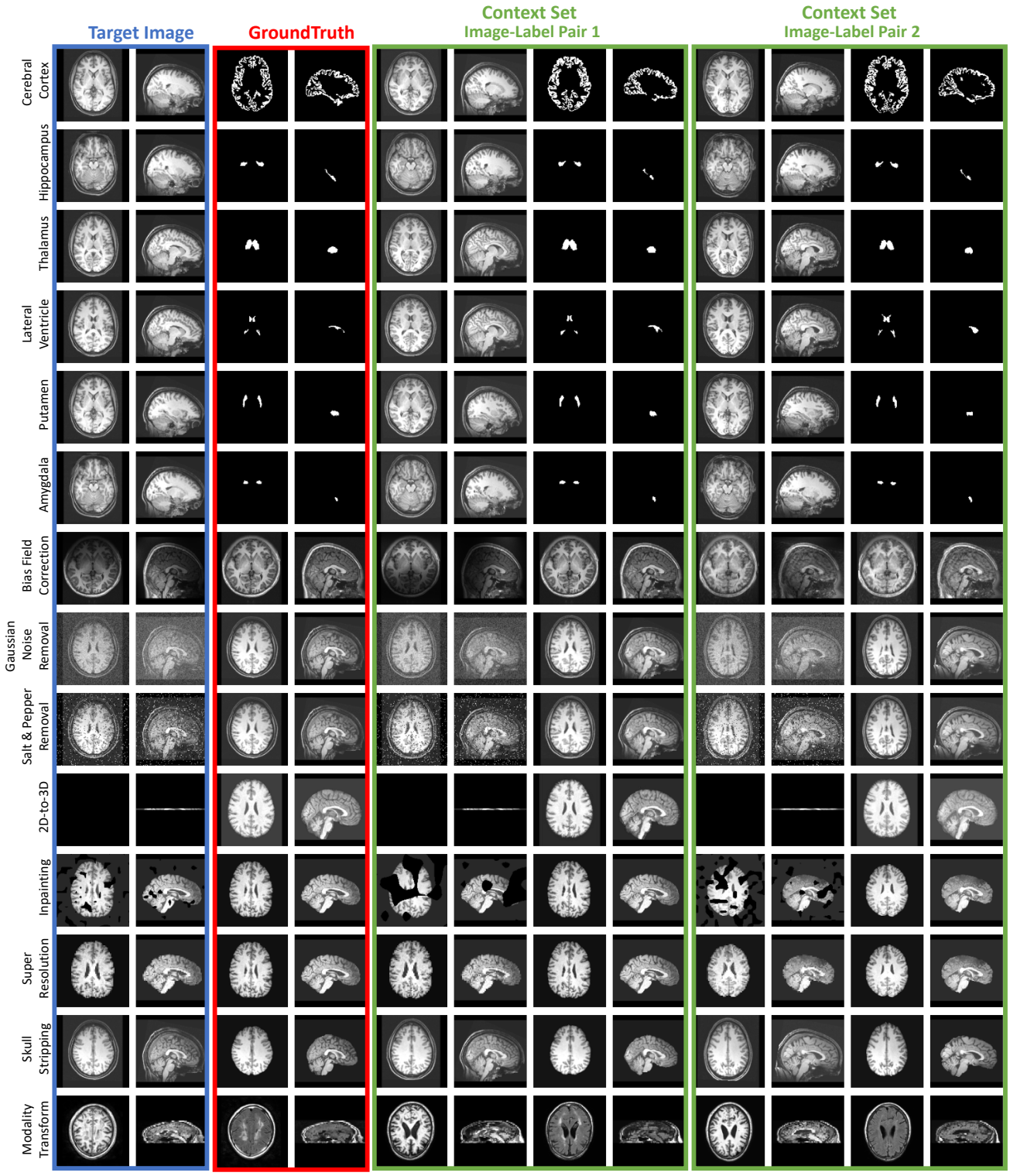

Figure 9. Visualization of the target image, ground truth, and image-label pairs in the context set.

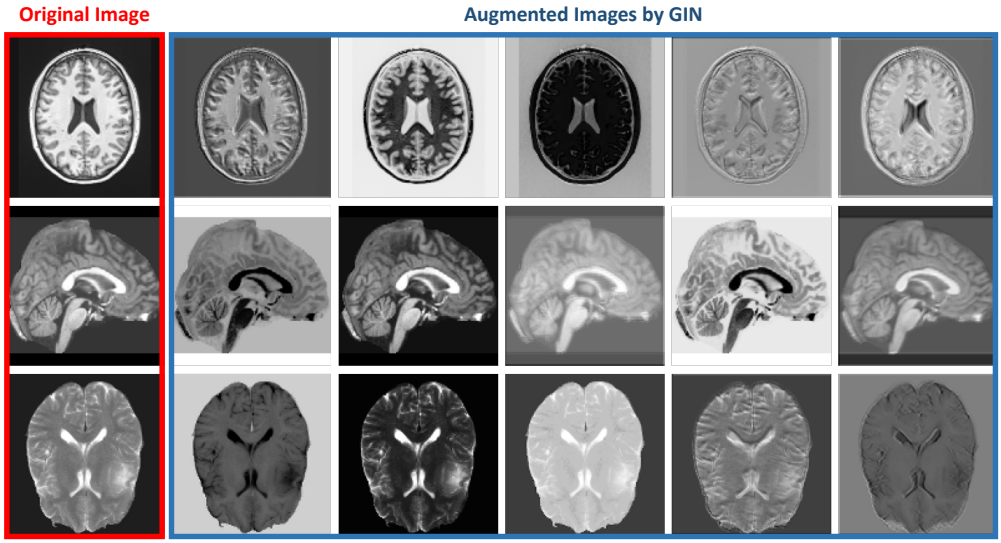

Figure 10. Visualization of the images augmented by GIN [51].

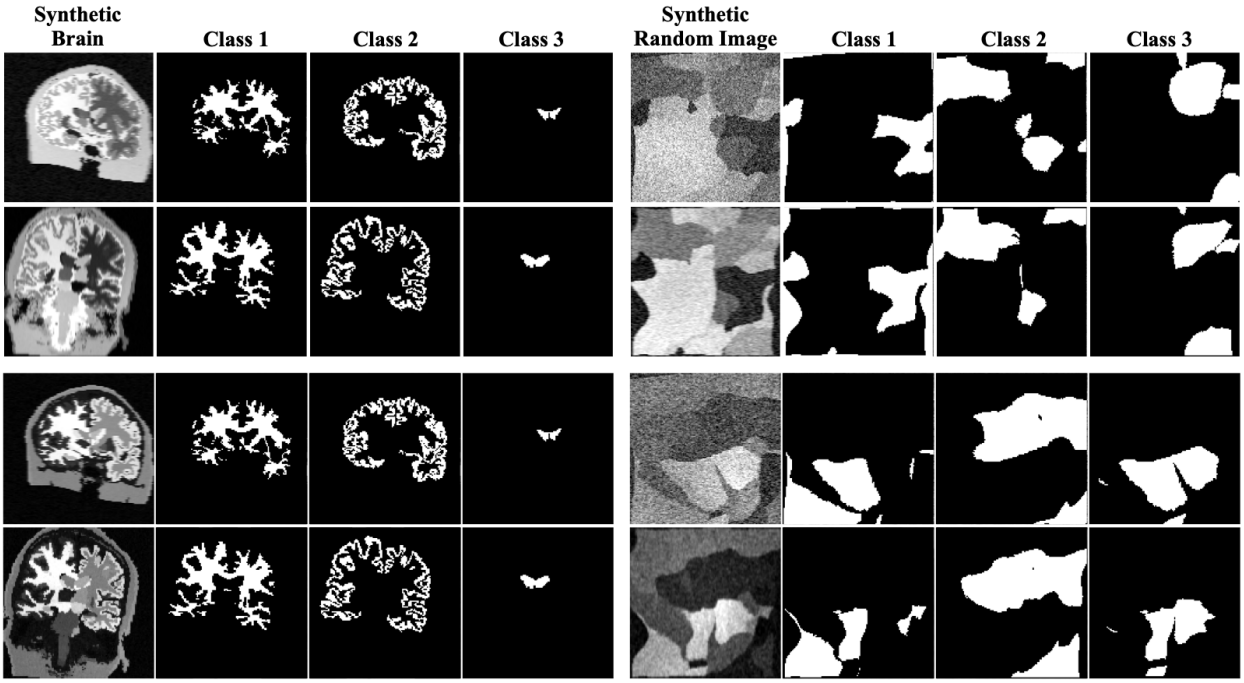

Figure 11. Visualization of the synthetic images [12, 25] for the segmentation task and corresponding segmentation masks.

(p=0.05). To extend the diversity of input images, we introduced random intensity shifts (p=0.2), intensity scaling (p=0.2), Gaussian noise (p=0.1), intensity inversion (p=0.05), and contrast enhancement using a random convolutional network called GIN [51] (p=0.05). Figure 10 showcases examples of applying GIN to process images, resulting in the generation of random and diverse modalities.

For task augmentation, we employed several methods as follows:

- **Random Task Overlapping**: To enhance the model's ability to perform multiple tasks in a single run, we randomly overlay tasks during training: bias field correction, Gaussian noise removal, salt and pepper noise removal, inpainting, and super-resolution, each with a probability of 0.05. These overlays are independent.

- **Random Modalities** (p=0.05): In modality transformation tasks, we also applied GIN [51] to the target and ground truth images. The same parameters were used for both context and target images.
- **Random Foreground** (p=0.5): In multi-class segmentation datasets, multiple classes were randomly grouped into a single foreground class, with remaining classes assigned as background. This approach increases the variety of segmentation tasks. Specifically, we randomly sampled k classes in the dataset (capped at 10) to form the foreground.
- **Sobel Filter** (p=0.05): In segmentation tasks, the Sobel filter was applied to the labels, with the model tasked to predict the filtered output.
- **Mask Inversion** (p=0.05): In segmentation tasks, the foreground and background masks were swapped to introduce variability.
- **Random Dilation** (p=0.05): The segmentation masks of the target and context are dilated by 1 voxel.
- **Random Erosion** (p=0.05): The segmentation masks of the target and context are eroded by 1 voxel.

Each image and task augmentation was applied independently according to its probability, and these augmentations are mutually compatible.

Synthetic Data. To enhance generalization, we incorporated synthetic data, following [12, 25], as shown in Figure 11. This added 100 segmentation datasets with varying contrasts, each containing 100 3D image samples.

C. Experiments

C.1. Training

Neuroverse3D was trained on eight NVIDIA V100 GPUs, with a batch size of 1 per GPU, using the ADAM optimizer. Training ran for 120K steps with an initial learning rate of 10^{-4}. Validation loss was evaluated every 1.2K steps, and the learning rate was halved if no improvement was observed over 20 evaluations. To improve training efficiency, the context size and mini-context size were fixed at 3 for the first 100K steps, requiring the model to process only one mini-context. Then, the context size was uniformly selected between 1 and 8 for the final 20K steps. Total training took approximately 8 days, with the model achieving the lowest validation loss selected. All task-specific models were trained on a single GPU for 36K steps, due to the much smaller training set and faster convergence compared to the multi-task ICL model.

C.2. Evaluation

Performance was assessed using the Dice coefficient for segmentation tasks and the Peak Signal-to-Noise Ratio (PSNR) for generation tasks on held-out datasets. Each task and setting were evaluated 10 times with randomly selected context sets to compute the average and standard deviation, ensuring robust performance measurements.

C.3. Qualitative Results

Figure 12 present the comparisons between our model and other ICL models. In segmentation tasks, other ICL models often produce false positive results on background slices, while our Neuroverse3D effectively captures global information, eliminating this issue. For generation tasks, Neuroverse3D demonstrates improved slice-to-slice consistency compared to 2D ICL models, underscoring the critical importance of leveraging a 3D model.

Moreover, Painter, an ICL model trained on natural images, struggles to handle these specialized medical imaging tasks despite being exposed to extensive data and having a large number of parameters. This highlights that for ICL models, merely providing context might be insufficient for adaptation to the medical domain without incorporating domain-specific knowledge.

Figure 13 presents the comparison results between our model and task-specific models. Overall, in 3D scenarios, with a limited number of samples and well designed data augmentation, few-shot task-specific models can achieve impressive results, which aligns with findings from previous studies [6]. For 3D segmentation tasks, the performance of Neuroverse3D is on par with both few-shot and fully supervised models.

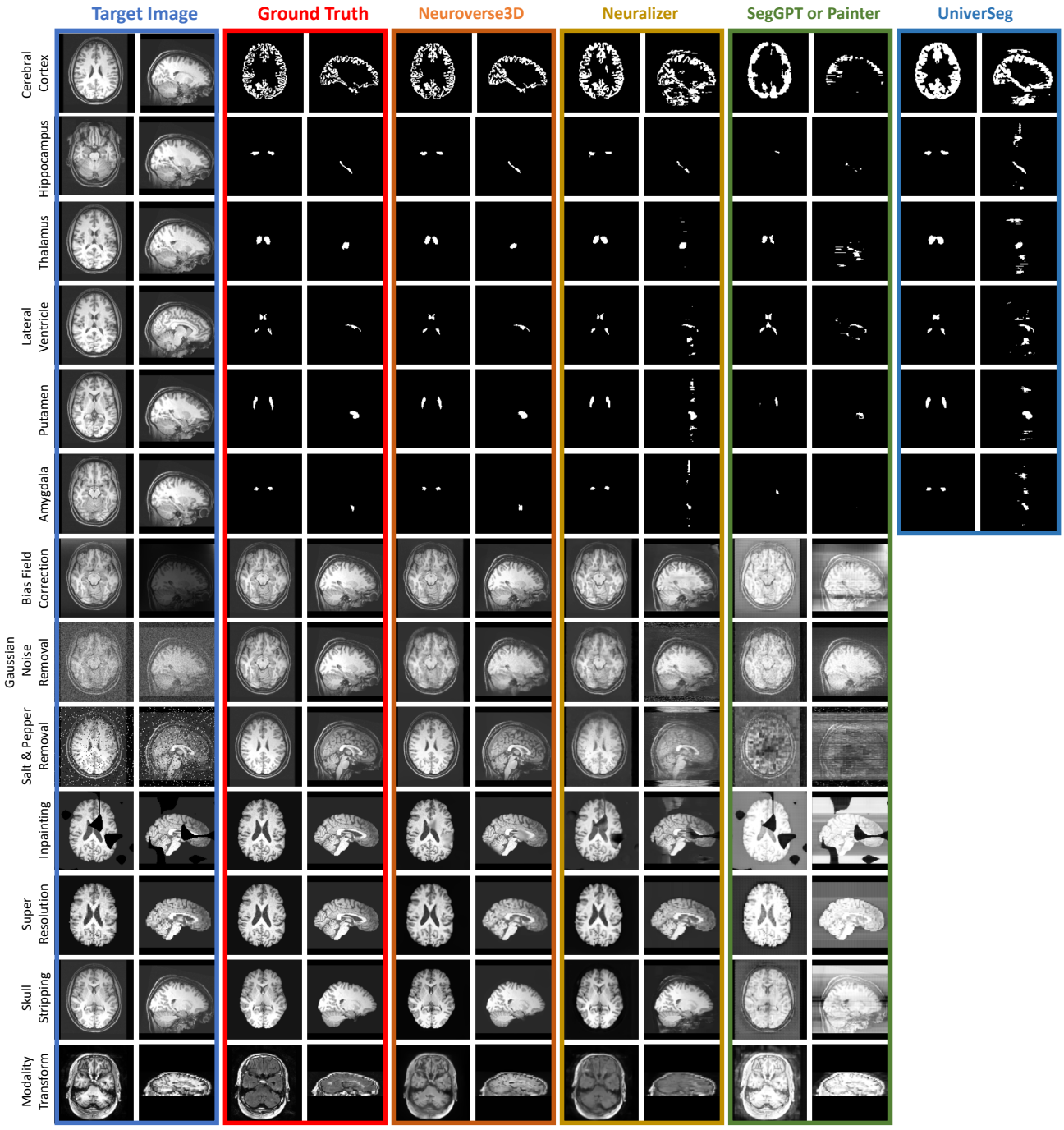

Figure 12. Qualitative results comparing of ICL models. In the SegGPT or Painter column, the results for segmentation tasks are from SegGPT, and the results for generation tasks are from Painter.

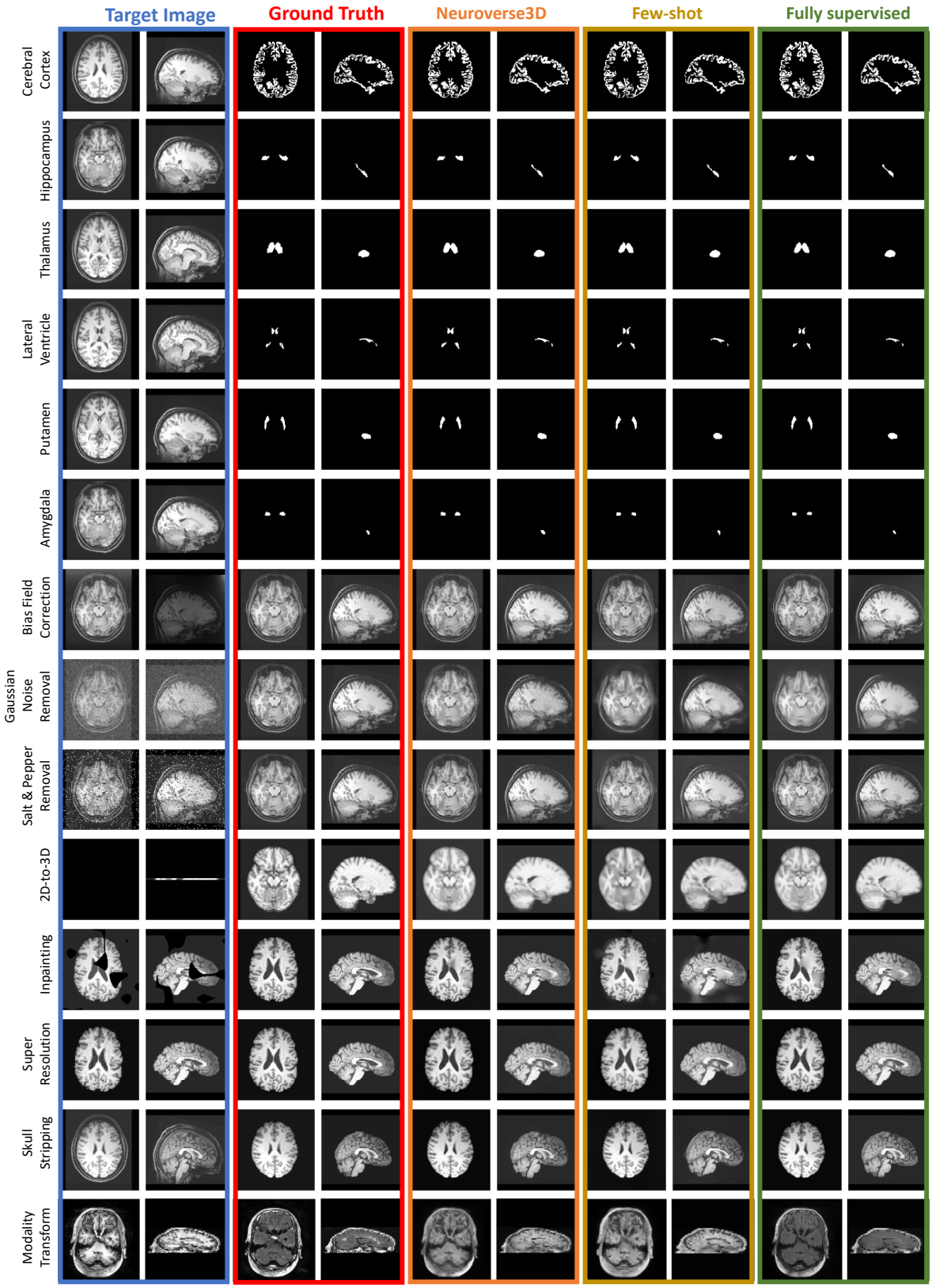

Figure 13. Qualitative comparison with task-specific models.

Smodified L1	Gradient Loss	Cerebral Cortex	Hippocampus	Thalamus	Lateral Ventricle	Putamen	Amygdala	Average
	✓	0.8847	0.7618	0.8586	0.8150	0.8220	0.6439	0.7977
✓		0.8724	0.7791	0.8648	0.8307	0.8167	0.6446	0.8014
✓	✓	0.8898	0.7917	0.8726	0.8290	0.8398	0.6812	0.8173

Table 4. Detailed ablation study of segmentation tasks.

Smodified L1	Gradient Loss	Bias Field Correction	Gaussian Noise Removal	Salt-and-Pepper Noise Removal	2D-to-3D	Inpainting	Super Resolution	Skull Stripping	Modality Transformation	Average
	✓	27.63	25.57	35.85	25.50	31.49	27.71	27.99	23.79	28.19
✓		24.16	24.93	30.82	25.55	29.19	27.48	27.44	23.81	26.67
✓	✓	27.68	25.89	35.72	26.08	31.21	28.08	28.61	23.74	28.38

Table 5. Detailed ablation study of generation tasks.

C.4. Inference Cost and Model Size

Table 6 summarizes the FLOPs and parameter counts for the models during inference. The task-specific model is implemented as a 5-stage U-Net with channel sizes of (32, 64, 128, 256, 512). Similarly, both the target and context branches of Neuroverse3D utilize a 5-stage U-Net with the same channel configuration.

Model	Parameters (M)	Inference TFLOPs
Task-specific	35.02	1.71
Neuroverse3D ($L = 1$)	70.85	4.35
Neuroverse3D ($L = 8$)	70.85	16.8

Table 6. Model parameters and inference FLOPs.

	Inference Time (s)	Context (pair)	Parameters (M)
Neuroverse3D	1.01	8 3D	70.85
Neuralizer [14]	4.96	32 2D	1.27
UniverSeg [12]	8.36	64 2D	1.18
Painter [54]	31.35	1 2D	307.72
SegGPT [55]	184.89	8 2D	307.72

Table 7. Inference time for a single 3D image (128 2D slices) and the corresponding model settings on a V100 GPU. For other comparison methods, we adopt the optimal context settings reported in their respective papers, consistent with the settings in Fig. 4.

C.5. Different Fusion Stage Configurations

Fusion Stage(s)	5	5–4	5–3	5–2	5–1 (Ours)
Segmentation (Dice)	0.8019	0.8118	0.8274	0.8347	**0.8446**
PET Segmentation (Dice)	0.2870	0.3162	0.3714	0.4945	**0.5314**
Generation (PSNR)	25.82	26.36	27.41	27.94	**28.87**

Table 8. Performance under different fusion stage configurations. Stage 5 corresponds to the deepest stage.

Tab. 8 summarizes the results of the ablation study on fusion stages. Denser connections improved the model's performance, especially on PET segmentation—an unseen modality.

Context Modality	Bias Removal	Gaussian	Salt & Pepper	Inpainting	Super-res.	2D to 3D
T1	**28.32**	**26.31**	**35.98**	**31.52**	**28.29**	**26.63**
T2	24.93	18.88	34.86	29.18	25.34	14.15
CT	24.95	23.32	33.97	28.67	24.62	12.27

Table 9. Generation performance (PSNR) under different context modalities, with T1 as the target modality.

C.6. Impact of Different Context Modalities

Tab. 9 presents the impact of different context modalities. For all tasks, performance is optimal when the context modality matches the target modality. The performance drop in salt-and-pepper noise removal is relatively small, potentially because this task is less dependent on contextual information. In contrast, Gaussian noise removal and 2D-to-3D transformation show more pronounced performance degradation, likely due to their greater reliance on context to convey task-relevant knowledge. These findings underscore the importance of modality alignment for the effective performance of ICL models.

C.7. Additional Baseline Comparisons

In Tab. 10, we compare a fine-tuned version of SegGPT (SegGPT*), representative segmentation models (nnUNet [34], Swin-UNETR [21]) trained on our training dataset, and the prompt-based model SAM-Med3D [53]. Our model consistently achieves the best overall performance. While nnUNet performs well on the held-out dataset, its performance degrades significantly on the PET modality, highlighting the strength of ICL models in generalizing across substantially different domains.

	nnUNet	Swin-UNETR	SegGPT*	SAM-Med3D (1 Point)	Ours
Held-out Set	0.8429	0.8036	0.523 (↑ 0.154)	0.1526	**0.8446**
PET (Unseen modality)	0.3380	0.4321	0.200 (↑ 0.036)	0.1466	**0.5314**

Table 10. Segmentation performance (Dice coefficient). SegGPT* denotes the SegGPT model fine-tuned on our training set. The brain anatomical segmentation task is conducted on the PET modality from the ADNI dataset.