Efficient Computation Sharing for Multi-Task Visual Scene Understanding
Supplementary Material

Sara Shoouri Mingyu Yang Zichen Fan Hun-Seok Kim
University of Michigan
{sshoouri,mingyuy,zcfan,hunseok}@umich.edu

1. Implementation Details

This supplementary section provides a detailed description of our model implementation, complementing the information presented in the paper.

Model optimization and training setting: We train the model for NYUD-v2 [11] dataset through four pixel-wise tasks: semantic segmentation, depth estimation, surface normal estimation, and edge detection. We utilize the cross-entropy loss function for discrete classification tasks such as semantic segmentation and edge detection. For the depth and normal estimation tasks, we employ the reverse Huber loss [6] and ℓ_1 loss, respectively. Similarly, for the PASCAL-Context [2] dataset, we consider five pixel-wise tasks: semantic segmentation, saliency detection, surface normal estimation, boundary detection, and human parsing. The task-specific models for these tasks are trained using the ℓ_1 loss for continuous regression tasks and the cross-entropy loss for discrete classification tasks.

As described in the paper, we employ a three-step training strategy to prune delta weight and delta activations. The first two steps are inherited from the Diff-pruning technique [4] to prune the delta weight by employing the ℓ_0 loss. The third step is required by our algorithm to prune the delta activation across the task and temporal domains by applying the ℓ_1 loss. One of the main advantages of the three-step training strategy is that the relaxed binary mask of delta weight can be fixed after the second step, ensuring that the sparsity ratio of the delta weight matrix will not be affected by activation pruning. We provide detailed training settings of the three-step strategy for the NYUD-v2 and PASCAL-Context datasets in Tables 1 and 2, respectively. Note that we gradually decrease the learning rate of each step to prevent the backward gradient fluctuation after fixing the location of non-zero elements in the delta weight matrix. This avoids performance degradation after step 1.

Data processing: We apply pre-processing steps specific to each dataset before training our model. For the PASCAL-Context dataset, we resize the images to 512×512 after zero-padding their borders. This is necessary because Swin transformer [7] requires images of similar height and width for patch merging. For the NYUD-v2 dataset, we resize all the images to 416×416 to reduce the memory requirements during the training of the delta task and temporal activation pruning. Our data augmentation pipeline involves random scaling, cropping, horizontal flipping, normalization, and color jittering with the following hyperparameters: *brightness=0.4*, *contrast=0.4*, *saturation=0.2*, *hue=0.1*, *p=0.5*. For video inputs in NYUD-v2, we first synchronize the training frames to produce a continuous video and then use a sampling rate of 30 frames per second for our experiments.

Task-specific network: As discussed in the paper, we employ the vision transformer [3, 7] as the encoder for each task, followed by a task-specific CNN head. We use the same architectures described in [3, 7] to build the Swin-B/ViT-B transformers. The architecture of each CNN head is designed based on the task-specific performance goals and FLOPs requirements. For depth estimation, we use a hybrid DPT-Small head [10] to decode layers $[3, 6, 9, 12]$ of the ViT-B transformer backbone into a dense feature map with dimensions of $[96, 192, 384, 768]$. These extracted features are combined utilizing RefineNet-based feature fusion [10] to create the final dense depth feature map, as described in [10]. Finally, we feed this map into 2D convolutional layers with dimensions of 128 to obtain the depth estimation output.

For other tasks, we adopt the ConvNeXt [8] architecture following the approach of [1]. After the transformer backbone, we first apply a linear projection to create new output tokens with an increased dimension of D. Then, we reshape these tokens to form a new feature map with a size of $(H/4) \times (W/4) \times (D/8)$. The

Table 1: Training setting of NYUD-v2 dataset.

Hyperparameters	Value
Optimizer	AdamW [9]
learning rate (step1)	$1e{-}4$
learning rate (step2)	$6e{-}5$
learning rate (step3)	$1e{-}5$
Weight decay	$1e{-}6$
Adam β	$(0.9, 0.95)$
Batch size	64
Learning rate sched.	Polynomial decay [13]
Training epochs	1500
Warmup learning rate	$1e{-}6$
Warmup epochs	40
λ_w	$[1e{-}8\,, 10e{-}8\,]$
λ_{a1}	$[1e{-}10\,, 10e{-}10\,]$
λ_{a2}	$[1e{-}10\,, 10e{-}10\,]$
Input resolution	416×416

Table 2: Training setting of PASCAL-context dataset.

Hyperparameters	Value
Optimizer	AdamW [9]
learning rate (step1)	$5e{-}5$
learning rate (step2)	$2e{-}5$
learning rate (step3)	$1e{-}5$
Weight decay	$1e{-}6$
Adam β	$(0.9, 0.95)$
Batch size	6
Learning rate sched.	Polynomial decay [13]
Training epochs	500
Warmup learning rate	$1e{-}6$
Warmup epochs	1
λ_w	$[1e{-}8\,, 10e{-}8\,]$
λ_{a1}	$[1e{-}10\,, 10e{-}10\,]$
Input resolution	512×512

Table 3: Utilized dimension for the ConvNeXt block in each task.

	Task	Dimension (D)
ViT-B	Semseg	3072
	Edge	3072
	Normal	1536
	Sal	3072
	Parsing	1536
Swin-B	Semseg	6144
	Edge	6144
	Normal	3072
	Sal	6144
	Parsing	3072

generated feature map is fed into two ConvNeXt blocks to create a dense task-specific feature map, where each ConvNeXt block contains one 7×7 depthwise convolution and two pointwise convolution layers, connected by a GELU [5] nonlinearity. Finally, we pass the final task-specific feature map through a 2D convolution layer and upsample it to full resolution using bilinear interpolation. Table 3 shows the dimensions utilized for each task in the ConvNeXt block for both ViT-B and Swin-B encoders. As the last block of the Swin-B (or ViT-B) transformer uses a patch size of 32×32 (or 16×16), we set the dimension of the projection in the ConvNeXt block for Swin-B to be 2 times larger than that of the ViT-B backbone to achieve similar FLOPs for the task-specific heads.

FLOPs computation: The FLOPs computation of each task-specific model includes the computations for the backbone and task-specific head. The calculation of the ViT-B/Swin-B backbone includes multi-head self-attention and feedforward network (FFN) layers. Also, the computation of the task-specific head comprises the 2D convolution, batch normalization, and fully-connected layers. The FLOPs counting formula for each operation type in each layer is presented in Table 4. As in the prior work [7], the computations of non-linear activation functions (e.g., GELU and Softmax) are ignored.

Table 4: The detail method for calculating FLOPs.

	Layer	FLOPs
Backbone	Linear Projection	$N \times D \times D$
	Multi-head Attention	$4N \times D \times D + 2N \times N \times D$
	LayerNorm	$D \times H \times W$
	Patch Merging (Swin-B)	$D \times H \times W + 2D \times D \times H \times W$
Head	2D Convolution	$C_o \times C_i \times k \times k \times \frac{H}{s} \times \frac{W}{s}$
	BatchNorm	$C_o \times H \times W$

N: number of patches, D: feature dimension
C_o, C_i, k, s: output, input channel, kernel size, stride
H, W: 2D input height, and width

2. Additional Experiment Results

Ablation study for Swin-B encoder: In the paper, we conducted an ablation study of ℓ_0 and ℓ_1 regularization coefficients on performance, computation reduction, and memory storage reduction for each sub-task. We conduct a similar hyperparameter ablation study for λ_w and λ_{a1} of the Swin-B encoder by sweeping λ_w in the range of $[1 \times 10^{-8}, 10 \times 10^{-8}]$ and λ_{a1} in the range of $[1 \times 10^{-10}, 10 \times 10^{-10}]$. Figure 1 shows the impact of different values of λ_w and λ_{a1} on the performance of the human parsing task. As expected, increasing λ_w leads to more parameter savings but less accurate performance. Also, increasing λ_{a1} saves more computations but leads to less accurate performance.

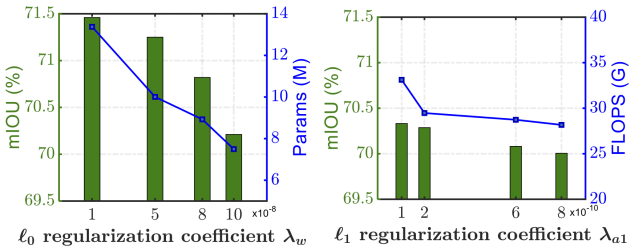

Figure 1: The impacts of λ_w and λ_{a1} on the human parsing task for the PASCAL-Context dataset using Swin-B as the backbone.

Delta weight and activation sparsity: As described in the paper, we conduct a quantitative analysis of the impact of weight and activation pruning by computing the average sparsity ratios (i.e., the ratio of zero-valued weights) of delta weights and delta activations in the backbone across all sub-tasks. We present the results for the PASCAL-Context dataset using the Swin-B encoder in Figure 2. It reveals that the overall sparsity ratios for both delta weight and delta activation are slightly lower than those obtained with ViT-B as the backbone. This may be due to the fact that Swin-B is a more compact model with fewer FLOPs than ViT-B, which makes increasing the sparsity ratios for both delta weights and delta activations more challenging.

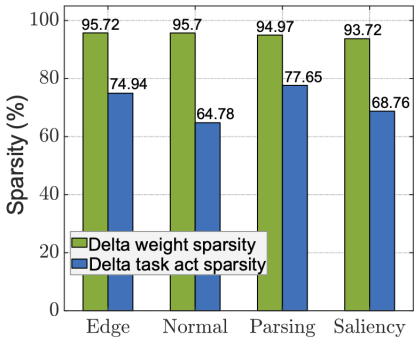

Figure 2: Overall delta weight and task activation sparsity for the PASCAL-Context dataset using Swin-B.

Additionally, we present the per-layer sparsity of the delta weights and delta activations for the NYUD-v2 (with ViT-B backbone) and PASCAL-Context datasets (with ViT-B or Swin-B backbone) in Figure 3. It shows that the delta task activation is more sparse in the initial layers, and the sparsity ratio decreases for later layers. It is expected as the model learns more common features in earlier layers and more task-specific features in later layers. Moreover, we observe that human parsing and edge detection tasks are more closely related to the base task (semantic segmentation), as their sparsity ratios are higher than the other sub-tasks. On the other hand, delta weight appears to be more sparse in the first and last few layers, while it is less sparse in the middle of the backbone, particularly for normal estimation and saliency detection.

Video Frame Evaluation: As explained in the paper, the NYUD-v2 dataset has sparsely annotated frames for ground-truth labels. Thus we evaluate the performance of our method by running it for all possible keyframe (start frame) time offsets (in terms of the frame index) within the range of $[0, 4]$ with respect to the ground-truth labeled frame. Table 5 presents the performance of each sub-task for each offset.

Table 5: Temporal Results for various keyframe interval offsets for NYUD-v2 dataset.

Interval offset	Semseg mIoU ↑	Depth RMSE ↓	Normal mErr ↓	Boundary odsF ↑
0	50.46	18.42	533.29	77.89
1	50.39	18.42	532.17	77.85
2	50.39	18.40	532.12	77.76
3	50.36	18.42	532.08	77.73
4	50.42	18.44	532.36	77.74

More Qualitative Results: We show per-pixel task visualization results of our proposed method and the SOTA method InvPT [12] on the challenging NYUD-V2 dataset in Figure 4. It is evident that our method generates better/comparable results than InvPT, especially on semantic segmentation and depth estimation. Furthermore, we provide additional visualization examples for the PASCAL-Context dataset using the ViT-B backbone in Figures 5 and 6. The figures demonstrate that our method yields significantly better results than InvPT, particularly on semantic segmentation and human parsing.

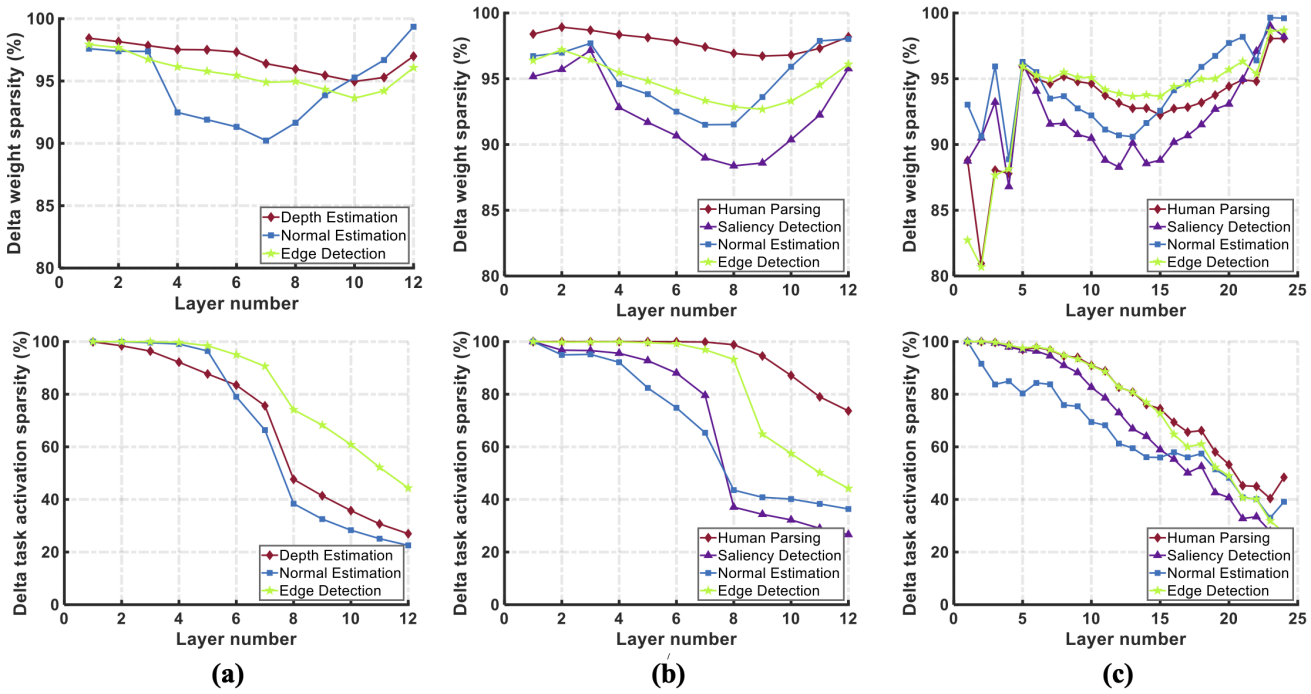

Figure 3: Delta weight and task activation sparsity ratios for each layer of backbone on (a) NYUD-v2 using ViT-B (b) PASCAL-Context using ViT-B (c) PASCAL-Context using Swin-B for single image input.

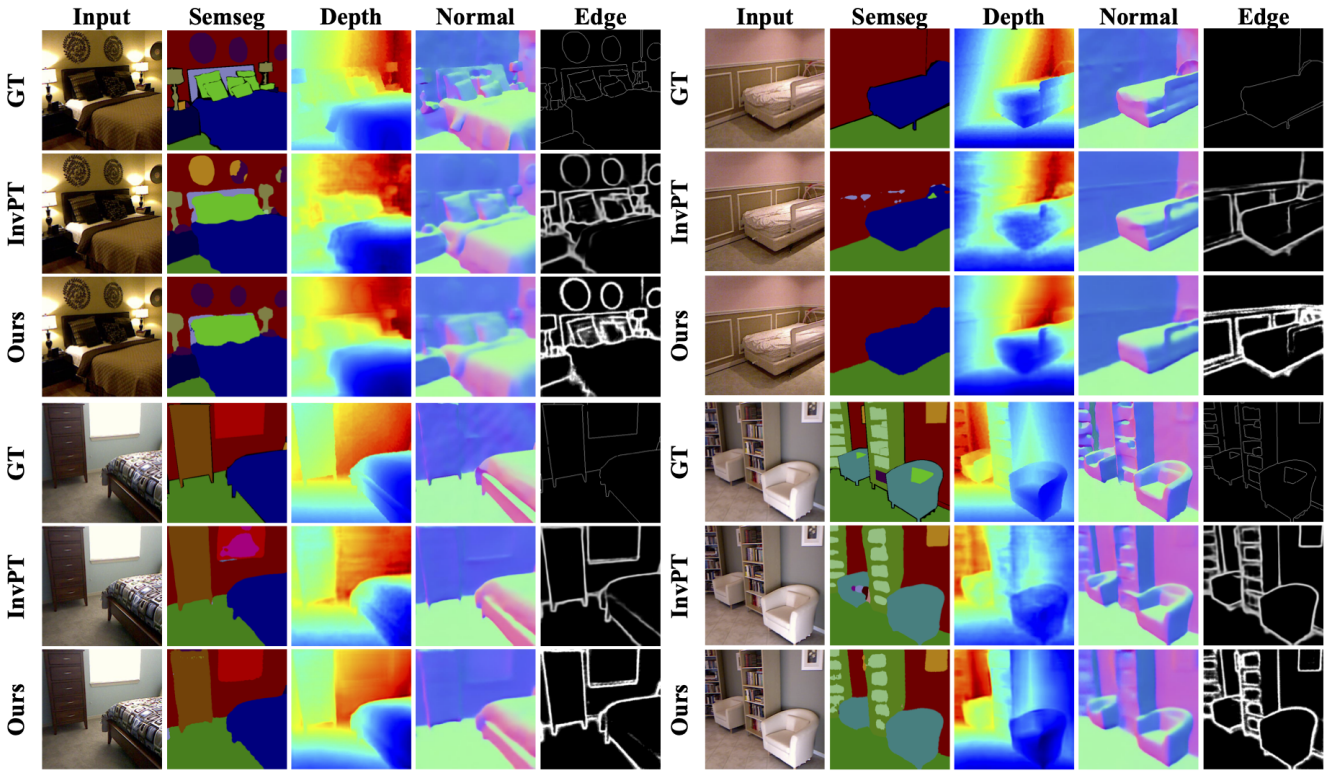

Figure 4: Qualitative comparison with the SOTA method InvPT on NYUD-v2 dataset. Our method generates better/comparable results than InvPT, especially on semantic segmentation and depth estimation.

References

[1] Roman Bachmann, David Mizrahi, Andrei Atanov, and Amir Zamir. Multimae: Multi-modal multi-task masked autoencoders. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVII*, pages 348–367. Springer, 2022. 1

[2] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1971–1978, 2014. 1

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1

[4] Demi Guo, Alexander Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. In *Annual Meeting of the Association for Computational Linguistics*, 2021. 1

[5] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 2

[6] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016. 1

[7] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 1, 2

[8] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022. 1

[9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. iclr, 2019, 2017. 2

[10] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12179–12188, 2021. 1

[11] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pages 746–760. Springer, 2012. 1

[12] Hanrong Ye and Dan Xu. Inverted pyramid multi-task transformer for dense scene understanding. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVII*, pages 514–530. Springer, 2022. 3

[13] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012. 2

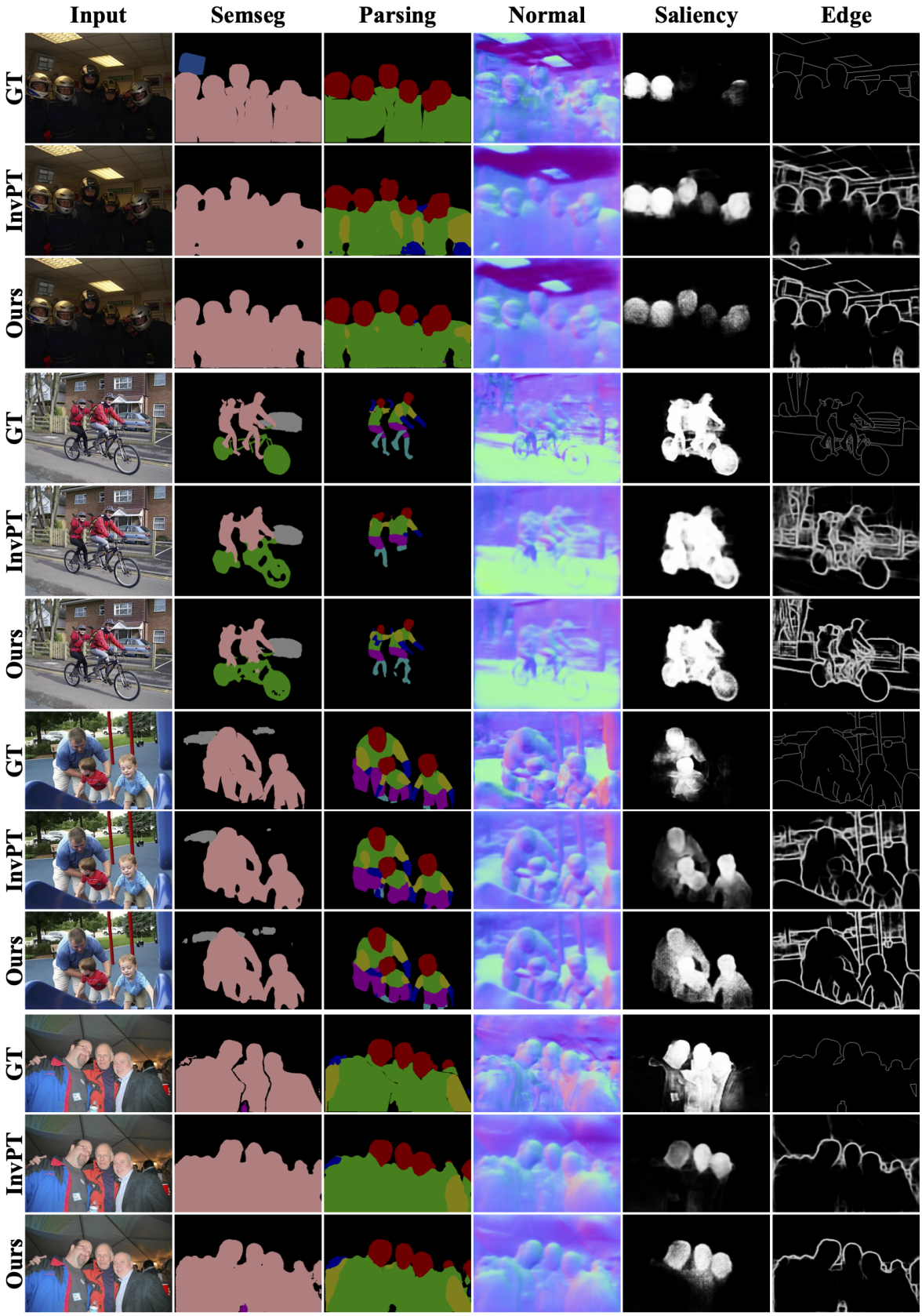

Figure 5: Qualitative comparison with the SOTA method InvPT on PASCAL-Context dataset. Our method produces significantly better results than InvPT, especially on semantic segmentation and human parsing.

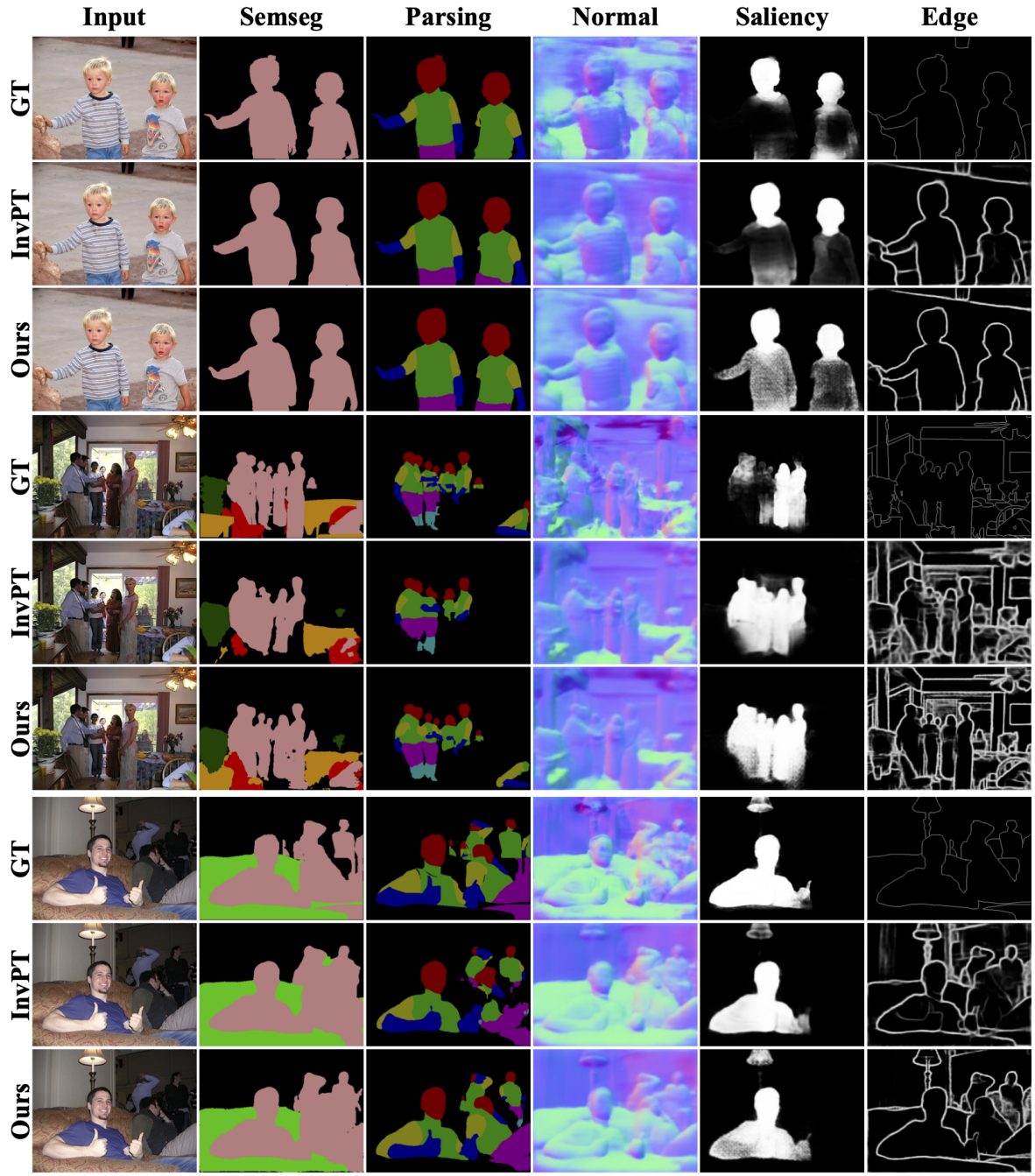

Figure 6: Qualitative comparison with the SOTA method InvPT on PASCAL-Context dataset. Our method produces significantly better results than InvPT, especially on semantic segmentation and human parsing.