Vision Transformer Adapters for Generalizable Multitask Learning (Supplementary)

Deblina Bhattacharjee, Sabine Süsstrunk and Mathieu Salzmann
School of Computer and Communication Sciences, EPFL, Switzerland
{deblina.bhattacharjee, sabine.susstrunk, mathieu.salzmann}@epfl.ch

This supplementary is organized as follows:

- Section 1: Additional Quantitative Results
- Section 2: Additional Experimental Details
- Subsection 2.1: Datasets
- Subsection 2.2: Baselines
- Subsection 2.3: Metrics
- Subsection 2.4: Training details
- Section 3: Visualizing task-adapted attention (TAA)
- Section 4: Ablation study
- Section 5: Additional qualitative results

Detailed Taxonomy We discuss a detailed taxonomy of multi-task learning approaches in Table 1 extending those provided in Table 1 of the main paper.

1. Additional Quantitative Results

1.1. Multitask Learning: Effect of Backbones

We report our MTL experiments on the NYUDv2 [17] dataset in Table 2, where we also integrate the best-performing models with the same vision transformer backbones such as Swin [12], ViT [8], Pyramid Transformer (PVTv2-B5) [31], and Focal Transformer (Focal-B) [33] for a fair comparison. Further, in Table 4 we evaluate the methods on additional datasets such as Synthia [23] and Vkitti2 [3]. Our method outperforms all the baselines, showing the benefit of leveraging task-adapted attention instead of using attention from a single task, as done in the second-best performing model of MulT [1]. This corroborates the trend seen across Taskonomy [39] and Cityscapes [7] in Table 2 of the main paper. Furthermore, we show that the task performances consistently improve with the addition of more tasks, signifying the benefit of injecting additional geometrical cues to help the other tasks. Note that all the methods are initialized with pre-trained ImageNet-22K weights.

Our method with the Swin backbone [12] shows the best performance. We, thus, choose to report the results with the Swin backbone in the main paper. Note that Swin is the most widely used model for dense prediction and its architecture compares with the hierarchical architecture of CNN-based baselines in Tables 2 and 4.

1.2. Zero-Shot Task Transfer

Although we have shown experiments on dense tasks throughout our paper, note that our model is not restricted to just dense tasks. In Table 3, we report our model's performance for the zero-shot image captioning task (IC) on 'noCaps out-of-domain' benchmark. Following the typical zero-shot task transfer setting, our model is trained with segmentation and captions from Coco and applied to noCaps for zero-shot IC. For training, we follow the GIT [30] text decoder configuration. During training on Coco, we enforce the highest similarity between the TAA token and the text decoder output token. On noCaps, we achieve comparable IC performance to GIT using a quarter #params.

1.3. Unsupervised Domain Adaptation

In Figure 1, we illustrate the architecture employed for our UDA setting that makes use of the adversarial learning scheme in [24]. We align the source and the target domains by applying a task-head discriminator. The alignment is done at the final output-levels for both segmentation and depth in order to preserve the architecture of our original model. In Table 5, we report additional UDA results with the source domain as Synthia [23] and the target domain as Cityscapes [7]. Following the same trend as seen in Table 4 of the main paper, we outperform both the ResNet-50 (CNN) baselines as well as the Swin-B V2 transformer baselines.

1.4. Generalization

We study the generalizability of our method to the comics domain [18] when the network is trained on MS-

	Methods	Architecture				Task-affinity generalization			
		Encoder-focused	Decoder-focused	Attention	Task-loss	MTL	Task-transfer	UDA	Novel domain
CNN-based	MTL-baseline [28]	✓	✗	✗	✗	✓	✗	✗	✗
	Consistency [38]	✗	✓	✗	✓	✓	✓	✗	✗
	XTAM [13]	✗	✗	✓	✗	✓	✗	✓	✗
	TAWT [5]	✓	✗	✗	✗	✓	✓	✗	✗
	Cross-stitch [15]	✓	✗	✗	✗	✓	✗	✗	✗
	MTAN [11]	✓	✗	✗	✗	✓	✗	✗	✗
	MTI-Net [29]	✗	✓	✓	✗	✓	✗	✗	✗
	TSwitch [27]	✗	✓	✗	✓	✓	✗	✗	✗
	Grad-norm [6]	✗	✗	✗	✓	✓	✗	✗	✗
	PCGrad [37]	✗	✗	✗	✓	✓	✗	✗	✗
	TTNet [19]	✗	✗	✗	✓	✓	✓	✗	✗
	PAD-Net [32]	✗	✓	✓	✗	✓	✗	✗	✗
	Taskonomy [39]	✗	✓	✗	✓	✓	✓	✗	✗
	Taskgrouping [26]	✗	✓	✗	✓	✓	✗	✗	✗
	ATRC [2]	✗	✓	✗	✓	✓	✓	✗	✗
Vision Transformer-based	IPT [4]	✗	✓	✓	✓	✓	✗	✗	✗
	ST-MTL [16]	✗	✓	✓	✓	✓	✗	✗	✗
	Vid-MTL [25]	✗	✓	✓	✓	✓	✗	✗	✗
	UniT [9]	✗	✓	✓	✓	✓	✗	✗	✗
	InvPT [35]	✗	✓	✓	✓	✓	✗	✗	✗
	Taskprompter [36]	✓	✓	✓	✗	✓	✓	✗	✗
	MulT [1]	✗	✓	✓	✓	✓	✗	✓	✗
	Our	✓	✗	✓	✓	✓	✓	✓	✓

Table 1: A detailed taxonomy of MTL methods (c.f. Table 1 main paper).

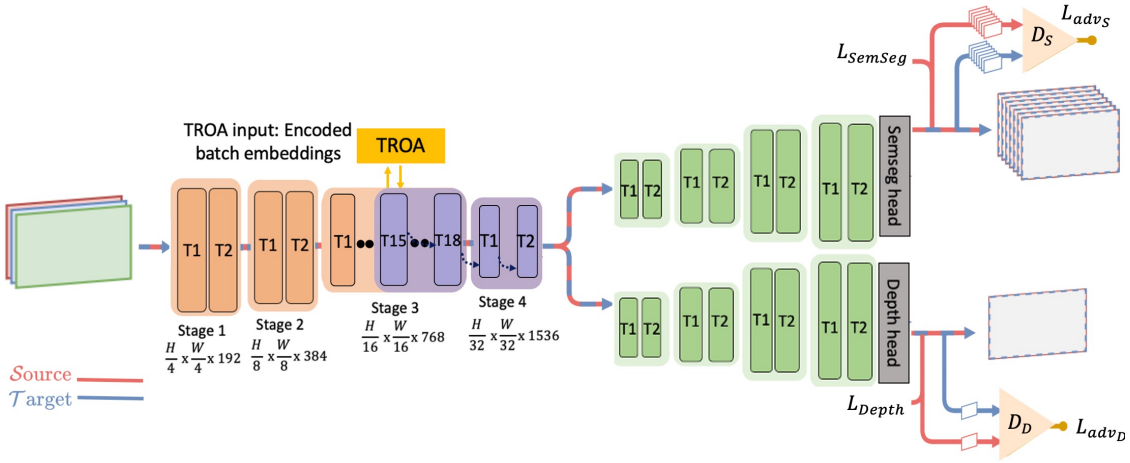


Figure 1: **UDA architecture for our method** with output-level adversarial learning. Arrows indicating data flows are drawn in either red (source), blue (target), or a mix (both). Domain Discriminators (shown as yellow triangles) are jointly trained with our multitask model.

Coco [10] dataset and is no fine-tuned to the DCM comics dataset. Shown in Table 6, our model outperforms both the ResNet-50 (CNN) baselines as well as the Swin-B V2 transformer baselines.

2. Additional Experimental Details

2.1. Datasets

We evaluate our method on the following datasets.

Taskonomy [39] comprises 4 million real images of indoor scenes with multi-task annotations for each image. The experiments were performed using the following 4 tasks from the dataset: semantic segmentation, depth (zbuffer), surface normals, and 2D (Sobel) texture edges. The tasks were selected to cover both geometric and semantic-based cues and have sensor-based/semantic ground truth. We report results on the official test set.

NYUDv2 [17] consists of sequences of RGB images, depth recorded by a Kinect camera, and dense labeling for semantic segmentation covering 894 classes. Officially, 249 scenes are reserved for training (for a total amount of 240k frames) and 215 scenes are reserved for testing. We use the official train-test split for our evaluations.

Cityscapes [7] consists of 5000 images with semantic annotations for 30 classes, grouped into 8 categories. For depth, we use depth from semi-global matching as depth labels. We estimate surface normal labels from the depth maps following [34].

Vkitti2 [3] contains 50 high-resolution monocular videos (21,260 frames) generated from five different virtual worlds. These photo-realistic synthetic videos are densely, exactly, and fully annotated with semantic segmentation (14 classes) and depth labels.

Synthia [23] contains synthetic images of 9400 multi-viewpoint photo-realistic frames rendered from a virtual city. We use pixel-level semantic annotations for 16 classes

Quantitative results on NYUDv2 [17]

	Methods	*'S-D'* SemSeg mIoU%↑	*'S-D'* Depth RMSE↓	*'S-D-N'* SemSeg mIoU%↑	*'S-D-N'* Depth RMSE↓	*'S-D-N'* Normal mErr.↓	*'S-D-N-E'* SemSeg mIoU%↑	*'S-D-N-E'* Depth RMSE↓	*'S-D-N-E'* Normal mErr. ↓	*'S-D-N-E'* Edges F1%↑
ResNet-50 (CNN) backbone	MTL-baseline [28]	44.40	0.5870	44.61	0.5790	28.34	45.26	0.5407	25.79	76.07
	Cross-stitch [15]	44.20	0.5900	44.40	0.5850	28.57	45.04	0.5500	26.11	76.00
	MTAN [11]	45.00	0.5840	45.04	0.5490	27.85	45.50	0.5263	25.62	76.18
	TTNet [19]	45.12	0.5730	45.16	0.5400	25.80	46.00	0.5217	24.58	76.20
	Taskonomy [39]	44.33	0.5890	44.52	0.5820	28.47	45.10	0.5433	25.94	76.12
	TSwitch [27]	47.04	0.5581	47.55	0.5270	24.02	47.99	0.5151	25.40	76.34
	Consistency [38]	45.38	0.5627	46.14	0.5360	24.73	46.79	0.5204	25.58	76.30
	ATRC [2]	46.77	0.5436	47.18	0.5195	23.30	47.56	0.5167	22.11	76.58
	XTAM [13]	46.90	0.5372	47.24	0.5177	22.73	48.80	0.5150	22.39	76.88
	TAWT [5]	47.02	0.5330	47.29	0.5152	22.69	48.87	0.5146	22.30	76.90
HRNet-48 (CNN) backbone	MTI-Net [29]	49.00	0.5290	49.52	0.5050	20.24	49.88	0.4940	20.13	76.95
Swin-B V2 [12] transformer backbone	ATRC [2]	49.11	0.5273	49.55	0.5034	20.36	50.40	0.4978	20.06	77.11
	XTAM [13]	49.25	0.5251	50.13	0.5008	20.19	50.55	0.4977	19.50	77.30
	TAWT [5]	49.33	0.5242	50.20	0.5001	20.10	50.71	0.4955	19.28	77.35
	MTI-Net [29]	49.33	0.5180	49.81	0.4990	20.15	50.38	0.4933	19.08	77.95
	ST-MTL [16]	49.84	0.5178	52.68	0.4975	19.82	53.72	0.4924	18.19	78.10
	InvPT [35]	51.59	0.5166	52.94	0.4960	19.15	54.37	0.4906	18.08	78.61
	Taskprompter [36]	53.27	0.5150	54.04	0.4951	18.88	55.34	0.4888	18.00	78.71
	MulT [1]	53.48	0.5130	54.17	0.4937	18.72	55.89	0.4885	17.97	78.65
	Our	**53.61**	**0.5111**	**54.80**	**0.4922**	**18.63**	**56.13**	**0.4861**	**17.50**	**80.03**
ViT-B [8] transformer backbone	ATRC [2]	48.49	0.5285	49.38	0.5050	20.52	50.25	0.4991	20.20	76.91
	XTAM [13]	49.11	0.5265	49.94	0.5024	20.29	50.39	0.4987	19.61	77.17
	TAWT [5]	49.15	0.5255	50.00	0.5022	20.19	50.54	0.4968	19.40	77.22
	MTI-Net [29]	49.24	0.5192	49.70	0.4997	20.21	50.25	0.4938	19.21	77.80
	ST-MTL [16]	49.72	0.5187	52.54	0.4988	19.96	53.57	0.4936	18.25	77.97
	InvPT [35]	51.49	0.5177	52.83	0.4974	19.33	54.23	0.4920	18.21	78.44
	Taskprompter [36]	53.22	0.5164	53.92	0.4966	19.00	55.30	0.4910	18.19	78.56
	MulT [1]	53.33	0.5148	54.00	0.4955	18.91	55.81	0.4901	18.13	78.50
	Our	**53.47**	**0.5123**	**54.68**	**0.4937**	**18.75**	**56.00**	**0.4872**	**17.57**	**79.83**
PVTv2-B5 [31] transfomer backbone	ATRC [2]	49.00	0.5280	49.47	0.5043	20.46	50.33	0.4986	20.14	77.00
	XTAM [13]	49.17	0.5258	50.03	0.5019	20.25	50.45	0.4983	19.58	77.20
	TAWT [5]	49.22	0.5250	50.11	0.5017	20.13	50.60	0.4963	19.37	77.27
	MTI-Net [29]	49.26	0.5189	49.72	0.4995	20.20	50.29	0.4938	19.18	77.83
	ST-MTL [16]	49.77	0.5185	52.60	0.4983	19.91	53.62	0.4934	18.22	78.02
	InvPT [35]	51.53	0.5172	52.88	0.4969	19.27	54.29	0.4915	18.18	78.50
	Taskprompter [36]	53.25	0.5161	54.00	0.4960	18.96	55.33	0.4904	18.14	78.60
	MulT [1]	53.37	0.5144	54.04	0.4950	18.87	55.85	0.4898	18.11	78.55
	Our	**53.53**	**0.5117**	**54.72**	**0.4934**	**18.71**	**56.05**	**0.4869**	**17.55**	**79.88**
Focal-B [33] transformer backbone	ATRC [2]	49.09	0.5277	49.50	0.5038	20.42	50.39	0.4982	20.10	77.07
	XTAM [13]	49.20	0.5253	50.07	0.5010	20.22	50.51	0.4981	19.53	77.26
	TAWT [5]	49.28	0.5247	50.15	0.5011	20.11	50.66	0.4960	19.32	77.31
	MTI-Net [29]	49.29	0.5186	49.77	0.4992	20.19	50.33	0.4938	19.15	77.88
	ST-MTL [16]	49.80	0.5181	52.64	0.4980	19.87	53.66	0.4930	18.26	78.06
	InvPT [35]	51.56	0.5168	52.90	0.4965	19.22	54.33	0.4913	18.15	78.56
	Taskprompter [36]	53.26	0.5158	54.02	0.4955	18.93	55.36	0.4892	18.07	78.73
	MulT [1]	53.40	0.5137	54.11	0.4944	18.80	55.90	0.4890	18.03	78.68
	Our	**53.57**	**0.5115**	**54.75**	**0.4933**	**18.68**	**56.07**	**0.4867**	**17.52**	**79.91**

Table 2: **Multitask learning results** on the NYUDv2 [17] benchmark for different multitask settings of *'S-D'*, *'S-D-N'*, and *'S-D-N-E'*. Our model consistently outperforms both the CNN-based and vision transformer-based baselines. Adding more tasks improves their respective performances based on their task affinities. Bold and underlined values show the best and second-best results, respectively.

Method	Pretraining	#Params	METEOR↑	CIDEr↑	SPICE↑
GIT	800M image & text	700M	**30.45**	**122.04**	**15.70**
Ours	ImageNet-22K only	**163M**	29.82	119.93	13.13

Table 3: **Zero-Shot Task Transfer results for Image Captioning** on the noCaps out-of-domain benchmark. Our model is comparable in performance to GIT [30] while using a quarter number of parameters.

and depth labels from Synthia's RAND-Cityscapes-CVPR-2016 benchmark as used in [13].

MS-Coco [10] comprises 164k training images that span over 80 categories with semantic segmentation annotations. For depth, normal, and edge we use pseudo-labels from [21, 34, 20], respectively. We use the MS-Coco dataset to evaluate if the models generalize to novel domains like comics which comprise data categories like 'faces' and 'animals'.

DCM [18] is a comics dataset comprising 772 full-page images with multiple comics panel images within. We use these images as test images from a novel domain for our additional experiments in the *generalization to novel domains*

setting.

2.2. Baselines

The baselines for our evaluation are described below. To prevent confounding factors, all baselines in the main paper (Tables 2-6) were implemented using the training procedure and the best model configurations as outlined in their respective works. Additionally, as shown in Table 2, we report the best-performing CNN-based baselines on the same transformer backbone architectures.

2.2.1 CNN-based Methods

MTL-baseline [28]: This is a naive multi-task learning network with one shared encoder and multiple task-specific decoders based on a ResNet-50 backbone.
Cross-stitch [15]: introduced soft-parameter sharing in deep MTL architectures. Being a ResNet-50 encoder-focused method that can achieve task transfer learning, we use this baseline for comparison.
MTAN [11]: used an attention mechanism to share a general feature pool amongst the task-specific networks. Being an encoder-focused method, we use this baseline for comparison.
TSwitch [27]: We use this as a baseline for comparison, as it uses a task embedding network to learn task-specific conditioning parameters that encourages constructive interaction between tasks in a pairwise manner.
TTNet [19]: presents a meta-learning algorithm that regresses model parameters for novel tasks for which no ground truth is available (zero-shot tasks).
Taskonomy [39]: We use this as a baseline as Taskonomy studies the relationships between multiple visual tasks for task transfer learning.
MTI-Net [29]: is a multiscale distillation procedure to explicitly model the unique task interactions that happen at each individual scale. We use MTI-Net with HRNet-48 as a baseline. Additionally, we compare MTI-Net with the the different transformer backbones.
Consistency [38]: This work presents a data-driven framework for augmenting standard multi-task learning with a cross-task consistency constraint, which is learned over a graph of arbitrary tasks.
TAWT [5]: This method uses gradient-loss to find optimal task representations to perform multi-task learning. TAWT shows that learning task representations in the encoder benefits multi-task learning. Being an encoder-focused method that can achieve task transfer learning, we use this baseline for comparison.
XTAM [13]: exploits correlation-guided attention between task pairs to enhance the average representation learning for all tasks. We use this baseline for comparison as it investigates the problem of MTL and UDA.
Adaptive Task-Relational Context (ATRC) [2]: leverages pairwise task similarities to create attention gates for global cross-task message passing.

Quantitative results on Synthia [23]

	Methods	*'S-D'* SemSeg mIoU%↑	*'S-D'* Depth RMSE↓	*'S-D-N'* SemSeg mIoU%↑	*'S-D-N'* Depth RMSE↓	*'S-D-N'* Normal mErr.↓
ResNet-50 backbone	MTL-baseline [28]	69.83	5.166	72.27	4.949	19.28
	Cross-stitch [15]	69.00	5.228	71.80	5.085	21.05
	MTAN [11]	77.42	4.285	77.90	4.298	17.48
	TTNet [19]	77.51	4.270	78.00	4.266	17.54
	Taskonomy [39]	69.40	5.209	72.16	4.974	20.09
	TSwitch [27]	78.01	4.255	78.42	4.200	17.05
	Consistency [38]	77.95	4.263	78.37	4.209	17.28
	XTAM [13]	80.53	4.222	82.99	4.088	14.46
	TAWT [5]	80.87	4.161	83.03	4.056	14.30
Swin-B V2 backbone	XTAM [13]	81.70	4.199	83.40	4.040	14.00
	TAWT [5]	81.91	4.118	83.75	4.000	13.66
	ST-MTL [16]	82.48	4.001	85.02	3.808	13.49
	MulT [1]	83.04	3.883	86.90	3.662	13.27
	Our	**85.13**	**3.695**	**88.50**	**3.476**	**13.10**
ViT-B backbone	XTAM [13]	81.50	4.207	83.32	4.049	14.08
	TAWT [5]	81.82	4.133	83.66	4.012	13.80
	ST-MTL [16]	82.37	4.009	84.00	3.851	13.61
	MulT [1]	82.90	3.892	86.82	3.689	13.35
	Our	**85.00**	**3.707**	**88.35**	**3.487**	**13.22**

Quantitative results on VKITTI2 [3]

ResNet-50 backbone	MTL-baseline [28]	87.75	5.511	88.86	5.312	22.27
	Cross-stitch [15]	86.11	5.719	87.50	5.505	23.03
	MTAN [11]	89.00	4.425	90.00	4.197	20.66
	TTNet [19]	89.13	4.440	90.11	4.188	20.52
	Taskonomy [39]	87.52	5.517	88.61	5.400	22.70
	TSwitch [27]	89.63	4.399	92.13	4.155	19.00
	Consistency [38]	89.25	4.461	90.75	4.180	19.37
	XTAM [13]	93.20	4.274	95.44	4.020	17.00
	TAWT [5]	93.41	4.202	95.96	3.991	16.76
Swin-B V2 backbone	XTAM [13]	96.93	3.425	97.58	3.092	14.49
	TAWT [5]	97.52	3.385	97.92	3.061	14.41
	ST-MTL [16]	97.91	3.365	98.22	3.040	14.08
	MulT [1]	98.03	3.341	98.75	3.015	13.95
	Our	**98.51**	**3.297**	**99.00**	**2.881**	**13.02**
ViT-B backbone	XTAM [13]	96.80	3.433	97.41	3.099	14.57
	TAWT [5]	97.40	3.391	97.81	3.065	14.55
	ST-MTL [16]	97.80	3.372	98.13	3.049	14.16
	MulT [1]	98.00	3.349	98.66	3.024	14.05
	Our	**98.43**	**3.303**	**98.89**	**2.890**	**13.13**

Table 4: **Multitask learning results** on the Synthia [23] (Top) and Vkitti2 [3] (Bottom) benchmark, respectively. Our method consistently outperforms all the ResNet-50 backbone-based MTL methods and the Swin-B V2 backbone-based methods. We also alternate the ResNet-50 backbone and the Swin-B V2 backbone with the ViT-B backbone for the best-performing methods. Bold and underlined values show the best and second-best results, respectively.

2.2.2 Transformer-based Methods

ST-MTL [16]: Leveraging vision transformers, this method achieves dense predictions in an encoder-decoder setup.
InvPT [35]: performs simultaneous modeling of spatial positions and multiple dense prediction tasks in a unified transformer framework.
Taskprompter [36]: focuses on the representation learning capability of the multitask networks by using a set of task prompts.
MulT [1]: Based on the Swin backbone, MulT uses a shared attention mechanism from a reference task that mod-

els the dependencies across the tasks in an end-to-end transformer framework.

Vanilla MTL Swin [12]: The Vanilla MTL Swin is based on the vanilla Swin-B V2 network with a single encoder and four shared decoders and task-specific heads.

1-task Swin [12]: We compare our performance against single-task learning networks using the baseline Swin-B V2 backbone, where each task is predicted separately by a dedicated Swin-B V2 network. This baseline is used as an Oracle in our *Unsupervised Domain Adaptation (UDA)* setting.

2.3. Metrics

We report the performances of all the models by using four task-specific metrics as follows:

Semantic segmentation uses *mIoU* as the average of the per-class Intersection over Union (%) between the ground-truth segmentation and predicted map.

Depth uses the Root Mean Square Error *(RMSE)* computed between the depth label and the predicted depth map, where the RMSE metric is reported in meters over the evaluated set of images.

Normal estimation uses the absolute angle error in degrees *(mErr)* between the normal ground-truth label and normal estimation map.

Edge estimation uses the *F1-score* between the ground-truth edges and the predicted edge maps.

2.4. Training Details

We train all the multi-tasking models with the Adam optimizer [14] with $\beta_1 = 0.9$ and $\beta_2 = 0.98$; learning rate of $5.0e-5$ and a warm-up cosine learning rate schedule. The number of warmup epoch is 5 out of the total 30 training epochs. We report the average over 3 runs. We use 4 A100 40 GB GPUs for training our MTL model.

Synthia [23]⟶Cityscapes [7]			'S-D'	
	Methods	MTL	SemSeg mIoU%↑	Depth RMSE↓
CNN	MTL-baseline-UDA [28]	✓	17.26	14.85
	Consistency-UDA [38]	✓	34.19	12.84
	XTAM-UDA [13]	✓	37.93	11.66
Transformer	1-task Swin-UDA [22]	✗	39.00	11.03
	MulT-UDA [1]	✓	<u>42.12</u>	<u>09.55</u>
	Our-UDA	✓	**50.03**	**06.99**
	1-task Swin-target (Oracle) [12]	✗	75.97	06.65

Table 5: **Unsupervised Domain Adaptation (UDA)** results for Synthia [23]→Cityscapes [7]. Our model outperforms all the baselines. Bold and underlined values show the best and second-best results, respectively.

3. Visualizing Task-adapted Attention (TAA)

We visualize the task-adapted attention for each tasks in the 'S-D-N-E' setting and show that it differs from the existing self-attention mechanism in Figure 2. TAA is more task-specific compared with self-attention, thanks to its task conditioning from TROA. In Figure 3, we demonstrate the effect of TAA that learns the task affinities and improves the prediction for each task. For instance, TAA improves the semantic segmentation performance where the bed mask is correctly classified in our predictions as in the ground truth. Without TAA the bed is segmented as a table.

	Methods	MTL	SemSeg mIoU%↑	Depth RMSE↓
CNN	Consistency [38]	✓	18.22	1.884
	XTAM [13]	✓	18.63	1.795
Transformer	1-task Swin [12]	✗	20.76	1.508
	ST-MTL	✓	22.49	1.446
	MulT [1]	✓	<u>24.02</u>	<u>1.300</u>
	Our	✓	**27.11**	**1.182**

Table 6: **Generalization results** of our model trained on MS-Coco [10] and applied to DCM Comics [18]. Our method outperforms all the baselines. Bold and underlined values show the best and second-best results, respectively.

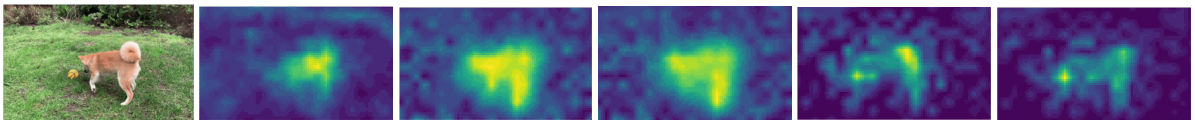
4. Ablation Study

4.1. Effect of Different Modules of Our Network

Model Changes	SemSeg mIoU%↑	Depth RMSE↓	Normal mErr. ↓	Edges F1%↑	#Parameters (Millions)
Vanilla MTL Swin [12]	48.13	0.4956	24.53	54.88	348.0
+ TAA	59.42	0.4111	18.55	69.91	408.0
+ bottleneck	59.93	0.4066	18.08	70.32	104.0
+ TSN **(Our)**	**60.80**	**0.3903**	**17.13**	**71.09**	**105.7**

Table 7: **Ablation study of the different components of our network** on the Taskonomy benchmark [39]. We show from left to right, the performances of each added module on multiple tasks. Our TAA and TSN components improves the performance consistently across all the tasks while the bottleneck reduces the number of parameters.

In Table 7, we present the results of an ablation study to determine which component of our method has the largest positive gain on the different task predictions. Starting from a Swin baseline that employs the Swin encoder and task-specific decoders as is — initialized with the pre-trained ImageNet 22k weights — and trained using random task sampling, we find that the task learning interferes with each other in the absence of task-adapted attention (TAA). Note that in this setup, the trainable encoder layers and decoder layers are jointly trained with just the Vanilla Swin self-attention (SA) as in [12], therefore lacking in task-adapted attention (TAA). We then add our model's components, one by one, starting with TAA conditioned on the task affinity weights from TROA. However, in this part, we do not add the adapter bottleneck, i.e., FFup and FFdown in Figure 3 of the main paper. We then add the bottleneck and finally add the Task-Scaled Norm (TSN). We report both the performances and parameters required for each added compo-



(a) Image (b) Self-attention (c) SemSeg TAA (d) Depth TAA (e) Normal TAA (f) Edge TAA

Figure 2: **Visualizing TAA versus the self-attention** of the Swin-B V2 encoder layer T18. We show that TAA has more task-specific attention compared to self-attention in the encoder. Here, our model that is used for visualization is trained on MS-Coco [10] with depth, surface normal, and edge labels from [21, 34, 20], respectively.

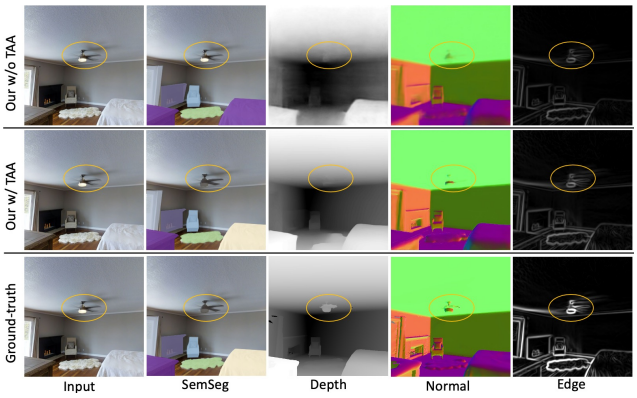


Figure 3: **Effect of TAA on our model.** The yellow-circled region shows how our model with TAA improves, for instance, the semantic segmentation performance, where the fan mask is correctly classified in our predictions. However, our model without TAA fails to segment the fan. Best viewed on screen and when zoomed in.

nent. Not only does each module lift the task performances but the introduction of the adapter bottleneck significantly reduces the number of parameters.

Note that TAA variants with operations like Matmul or concatenation between the $A'(.)$ matrix and $q.k^T$ matrix are extremely computationally expensive, scaling non-linearly with an increase in the number of tasks. Hence, we do not report them. Also with a TROA variant where 'w= constant' for all tasks, the model fails to leverage the task interdependencies [1, 26] and defaults to self-attention that is shifted by a constant. Failing to account for the task relationships, is *not* a typical multitask setting [1, 26, 38]. We, therefore, do not report this TROA variant.

Furthermore, in Table 8, we study the effect of different Swin V2 [12] backbones such as Swin-B and Swin-L, different pre-trained initializations, i.e. Imagenet-1K and Imagenet-22K, various hidden feed-forward network (FFN) dimensions with 48 or 96 hidden dimensions, and different bottleneck sizes for the FF down and FF up in our vision transformer adapters. We observe that the configuration of Swin-B backbone initialized with ImageNet-22K, an FFN with a hidden dimension of 96 and a bottleneck dimension of 12 achieves an improved performance across all tasks. Other configurations with a larger Swin network, a larger FFN dimension, and a larger bottleneck size give slight performance gains but they are parametrically costly.

Effect of Adapter Placement and Number of Adapters. In Table 9, we study the effect of varying the placement of the vision adapters, as well as varying the overall number of adapters in the different stages. We append adapters to *every* transformer layer for a given stage. We show that our adapters are more efficient when located later in the encoder stages (i.e. stage 3 or 4), thereby leveraging the richer semantics. Drawing motivation from Table 9, we study the effect of applying the vision adapters to fewer transformer layers as opposed to all of them, in Stage 3. In Table 10, we apply the adapters to the later layers of stage 3, guided by the principle of extracting richer semantics. This configuration achieves comparable performance on all four tasks while significantly reducing the number of parameters. Therefore, we use this layout as our model, where the adapters are applied to transformer layers 15-18 in Stage 3 and layers 1-2 in Stage 4, respectively.

5. Additional Qualitative Results

We compare the best-performing methods in the UDA setting with the source domain as Synthia [23] and the target domain as Cityscapes [7]. Our method outperforms all the baselines as shown in Figure 4. We qualitatively compare the best-generalizing methods to a novel domain of comics for segmentation in Figure 5 and depth in Figure 6, respectively.

Additionally, we qualitatively compare our model for the multi-task learning setting with the best-performing baselines that utilize the same Swin-B V2 backbone. The results in Figure 7, Figure 8, and Figure 9 show the performance of the different networks across multiple vision tasks on the NYUDv2 [17], Synthia [23], and Cityscapes [7], respectively. Our model yields higher-quality predictions than all the multitask baselines.

Acknowledgement. This work was supported in part by the Swiss National Science Foundation via the Sinergia

Backbone Size	Pre-trained Initialization	FFN Dimension	Bottleneck Size	SemSeg mIoU%↑	Depth RMSE↓	Normal mErr. ↓	Edges F1%↑	Parameter (in millions)
Swin-B v2 transformer	ImageNet 1K	48	8	50.52	0.4511	22.13	62.16	103.0
			12	56.55	0.4230	20.39	67.03	103.6
			24	56.85	0.4191	20.22	67.66	104.4
		96	8	52.18	0.4410	22.04	63.60	105.2
			12	58.10	0.4105	19.15	69.07	105.7
			24	58.81	0.4095	19.08	69.19	109.4
	ImageNet 22K	48	8	52.11	0.4392	20.88	64.29	103.0
			12	58.72	0.4051	18.62	69.10	103.6
			24	58.87	0.3998	18.20	69.90	104.4
		96	8	54.19	0.4220	20.03	65.65	105.2
			12	60.80	0.3903	17.13	71.09	105.7
			24	60.83	0.3892	17.09	71.13	109.4
Swin-L v2 transformer	ImageNet 22K	48	8	52.20	0.4385	20.81	64.38	360.0
			12	59.01	0.4042	18.50	69.22	364.2
			24	58.92	0.3990	18.11	69.98	367.8
		96	8	54.31	0.4150	19.91	65.88	380.8
			12	60.89	0.3892	17.02	71.15	383.4
			24	60.95	0.3880	16.90	71.33	388.0

Table 8: **Ablation study for the network sizes** on the Taskonomy [39] benchmark for *'S-D-N-E'* task set. We study the effect of different Swin backbone network sizes, different pre-trained initializations, 2 different feed-forward network (FFN) sizes, and 3 different bottleneck sizes on our model, respectively. As shown, in the grey row, we select the model which outperforms the baselines while being parameter efficient. Note Swin-L does not have pre-trained models with ImageNet 1K.

Adapter Placement				'S-D-N-E'				
Stage 1	Stage 2	Stage 3	Stage 4	SemSeg mIoU%↑	Depth RMSE↓	Normals mErr. ↓	Edges F1%↑	Parameter (in millions)
✓				51.05	0.4913	28.11	50.34	97.20
	✓			51.11	0.4899	27.02	54.99	94.00
		✓		58.85	0.4001	18.23	69.88	159.4
			✓	54.07	0.4412	20.95	65.77	92.60
	✓		✓	57.81	0.4121	20.03	66.13	119.0
		✓	✓	60.85	0.3888	17.07	71.21	163.0
	✓	✓	✓	60.89	0.3885	17.01	71.28	188.0
✓	✓	✓	✓	60.92	0.3884	16.98	71.31	227.0

Table 9: **Ablation study for varying the placement of our adapters** as well as varying the overall number of adapters across the different Swin encoder stages. In this setting, our adapters are placed at *every* transformer layer for a given stage if that stage is marked with a (✓). The **grey** row is *not* our model. The upper part shows our vision transformer adapters perform better at later stages in the encoder (i.e. stages 3 and 4). Applying adapters to more Swin encoder stages leads to a small boost at the cost of more parameters.

Adapter Placement		'S-D-N-E'				
Stage 3	Stage 4	SemSeg mIoU%↑	Depth RMSE↓	Normals mErr. ↓	Edges F1%↑	Parameter (in millions)
Layers 15-18	Layers 1-2 (all)	<u>60.80</u>	<u>0.3903</u>	<u>17.13</u>	<u>71.09</u>	**105.7**
Layers 1-18 (all)	Layers 1-2 (all)	**60.85**	**0.3888**	**17.07**	**71.21**	<u>163.0</u>

Table 10: **Ablation study for the Swin encoder stages that applies our vision adapters.** We study the effect of appending adapters to the later Swin encoder layers as opposed to all the Swin encoder layers in Stage 3. Following this setting, we report the performances and number of parameters for the *'S-D-N-E'* setting on the Taskonomy [39] benchmark. The upper row **(our model)** shows our vision transformer adapters are more parameter efficient when located in the later transformer layers while giving a comparable performance with those reported in the bottom row. Bold and underlined values show the best and second-best results, respectively.

grant CRSII5−180359.

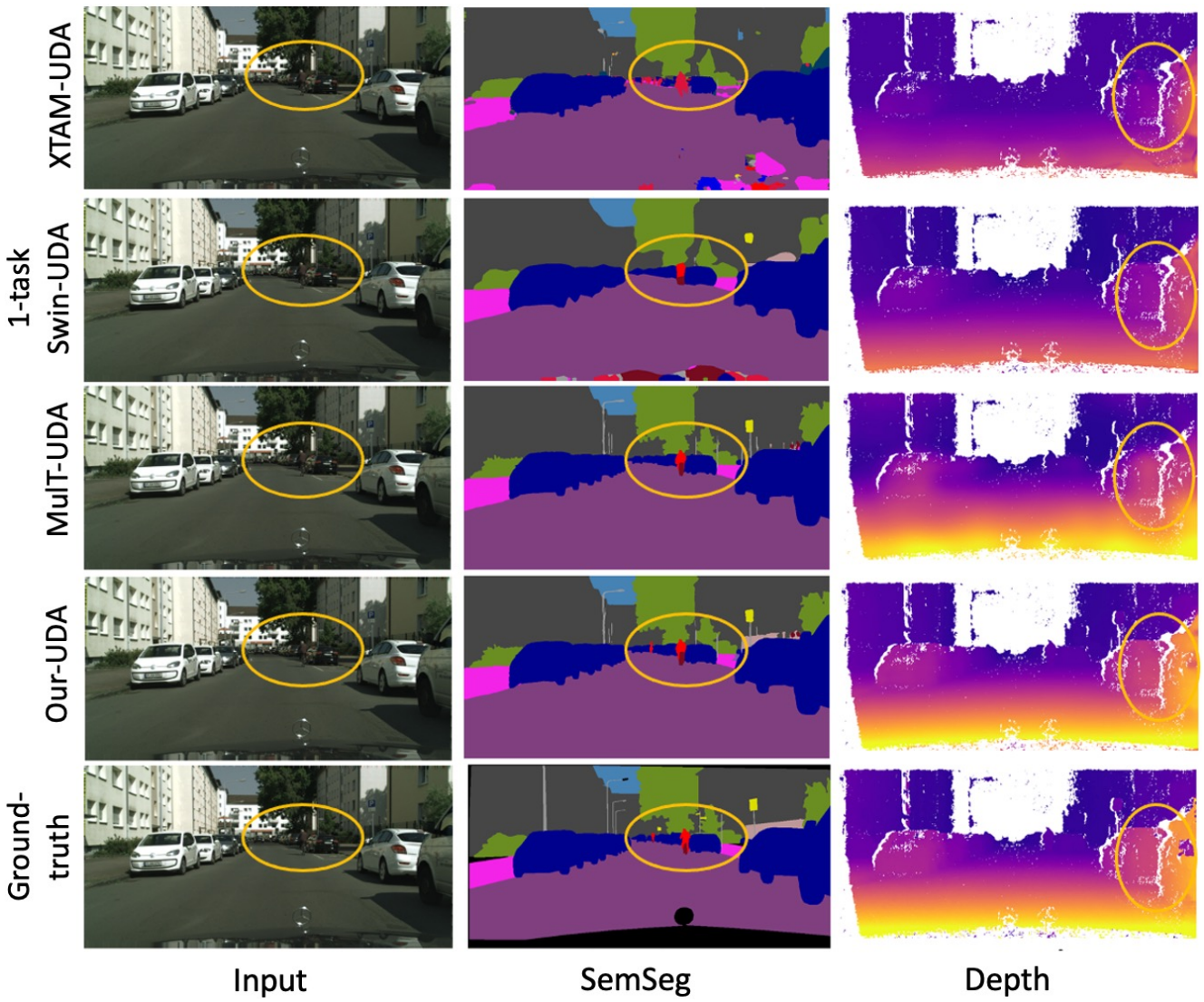


Figure 4: **Unsupervised Domain Adaptation (UDA)** results of the best-performing methods in Table 5 on Synthia [23]→Cityscapes [7]. Our model outperforms the CNN-based baseline (XTAM-UDA [13]) and the Swin-B V2-based baselines (1-task Swin-UDA [12], MulT-UDA [1]), respectively. For instance, our method can predict the depth of the car tail light, unlike the baselines. Best seen on screen and zoomed within the yellow circled region.

Figure 5: **Generalization of our model trained on MS-Coco [10] and applied to DCM comics [18] for segmentation.** Our method outperforms both the 1-task Swin and the MTL models [16, 1], respectively. For instance, the airplane is more accurately segmented than the one in the baselines. All the methods are based on the same Swin-B V2 backbone. We show the best-performing methods in Table 6. Best viewed on screen and when zoomed in.

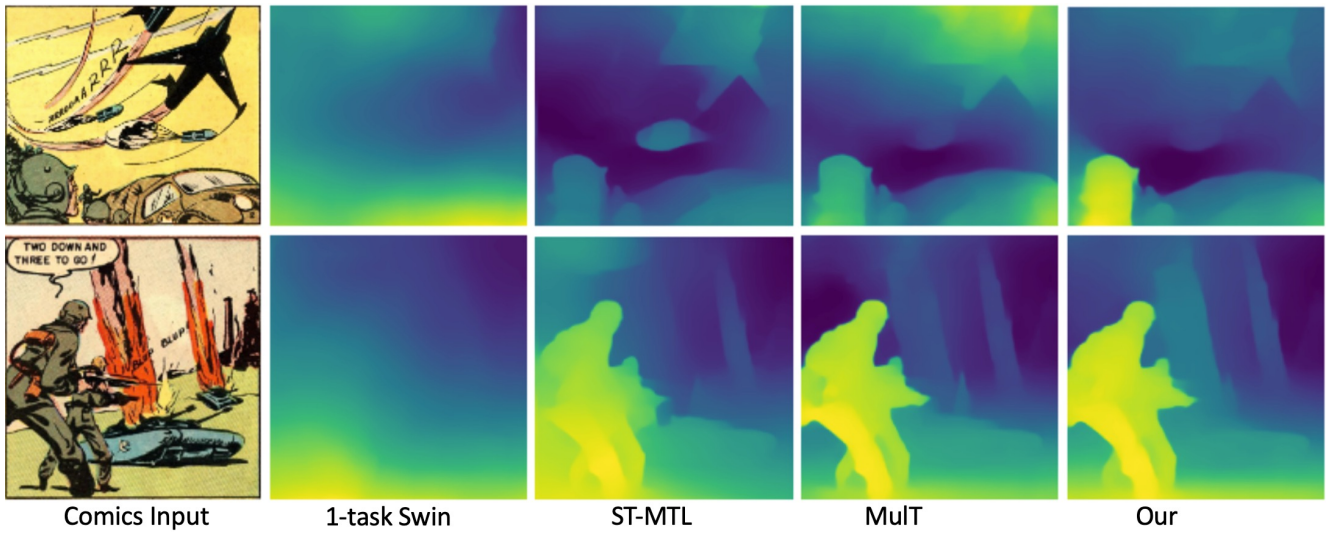


Figure 6: **Generalization of our model trained on MS-Coco [10] and applied to DCM comics [18] for depth.** Our method outperforms both the 1-task Swin [12] and the MTL baselines [16, 1], respectively. For instance, our method correctly separates the foreground depth plane from the background, unlike the baselines. All the methods are based on the same Swin-B V2 backbone. We show the best-performing methods in Table 6. Best viewed on screen and when zoomed in.

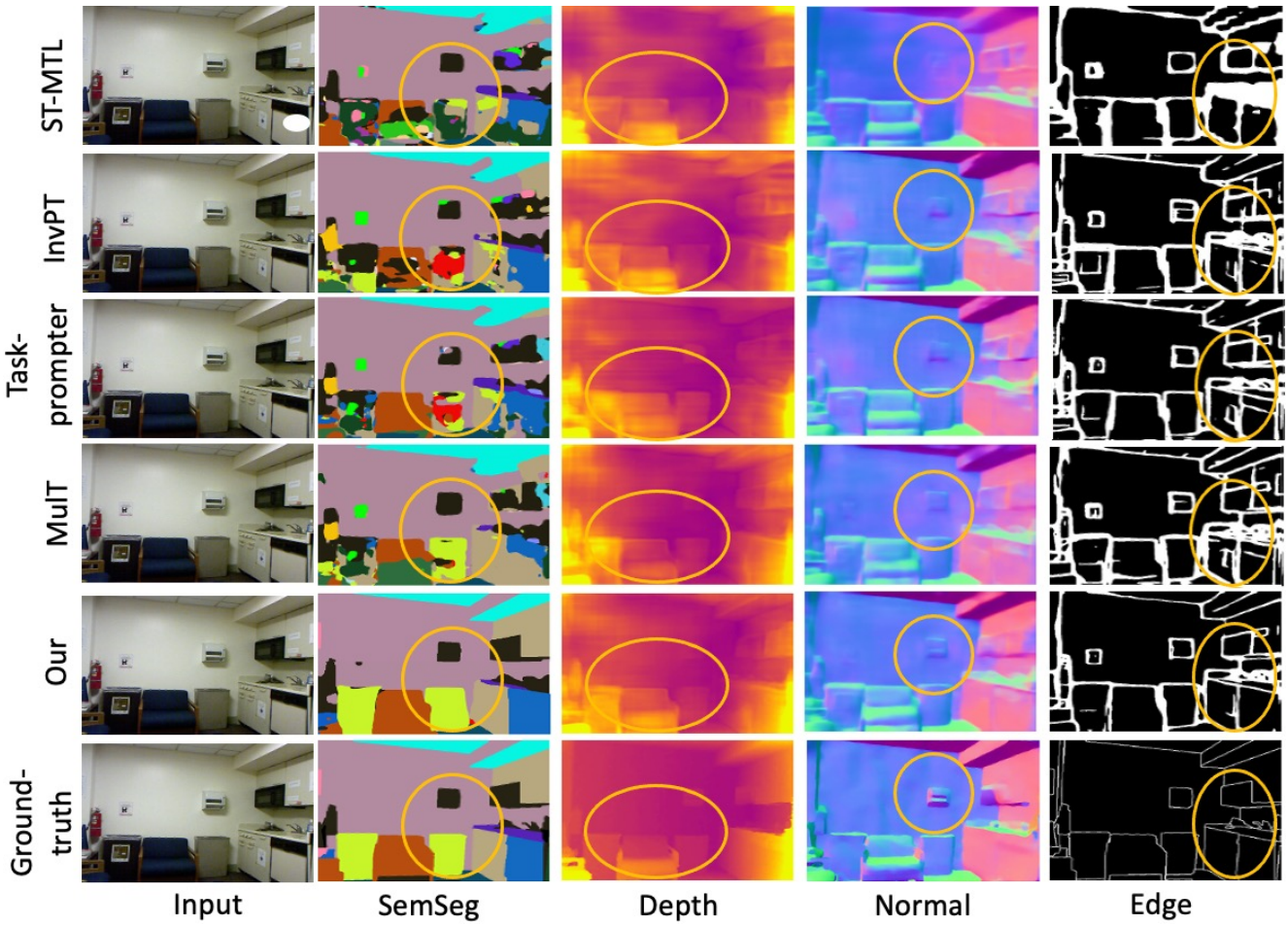


Figure 7: **Multitask Learning comparison on NYUDv2** [17] benchmark in the *'S-D-N-E'* setting. Our model outperforms all the multitask baselines, i.e. ST-MTL [16], InvPT [35], Taskprompter [36], and MulT [1], respectively. For instance, our model correctly segments and predicts the surface normal of the elements within the yellow-circled region, unlike the baseline. All the methods are based on the same Swin-B V2 backbone. We show the best-performing methods in Table 2. Best seen on screen and zoomed in.

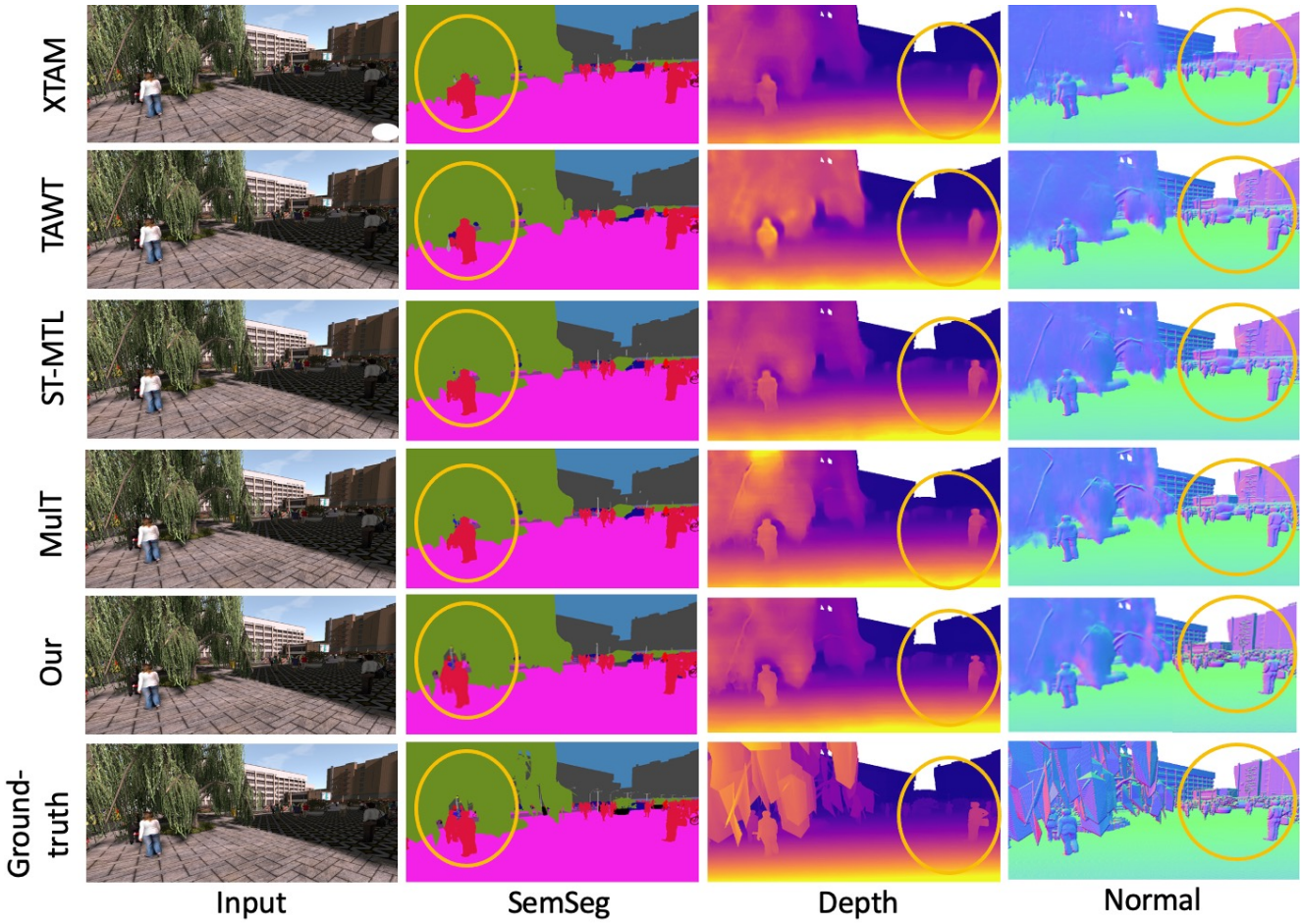


Figure 8: **Multitask Learning comparison on Synthia** [23] benchmark in the *'S-D-N'* setting. Our model outperforms all the multitask baselines. For instance, our method correctly segments the people, unlike the baselines. All the methods are based on the same Swin-B V2 backbone. We show the best-performing methods in Table 4, i.e. XTAM [13], TAWT [5], ST-MTL [16], and MulT [1], respectively. Best seen on screen and zoomed within the yellow circled regions.

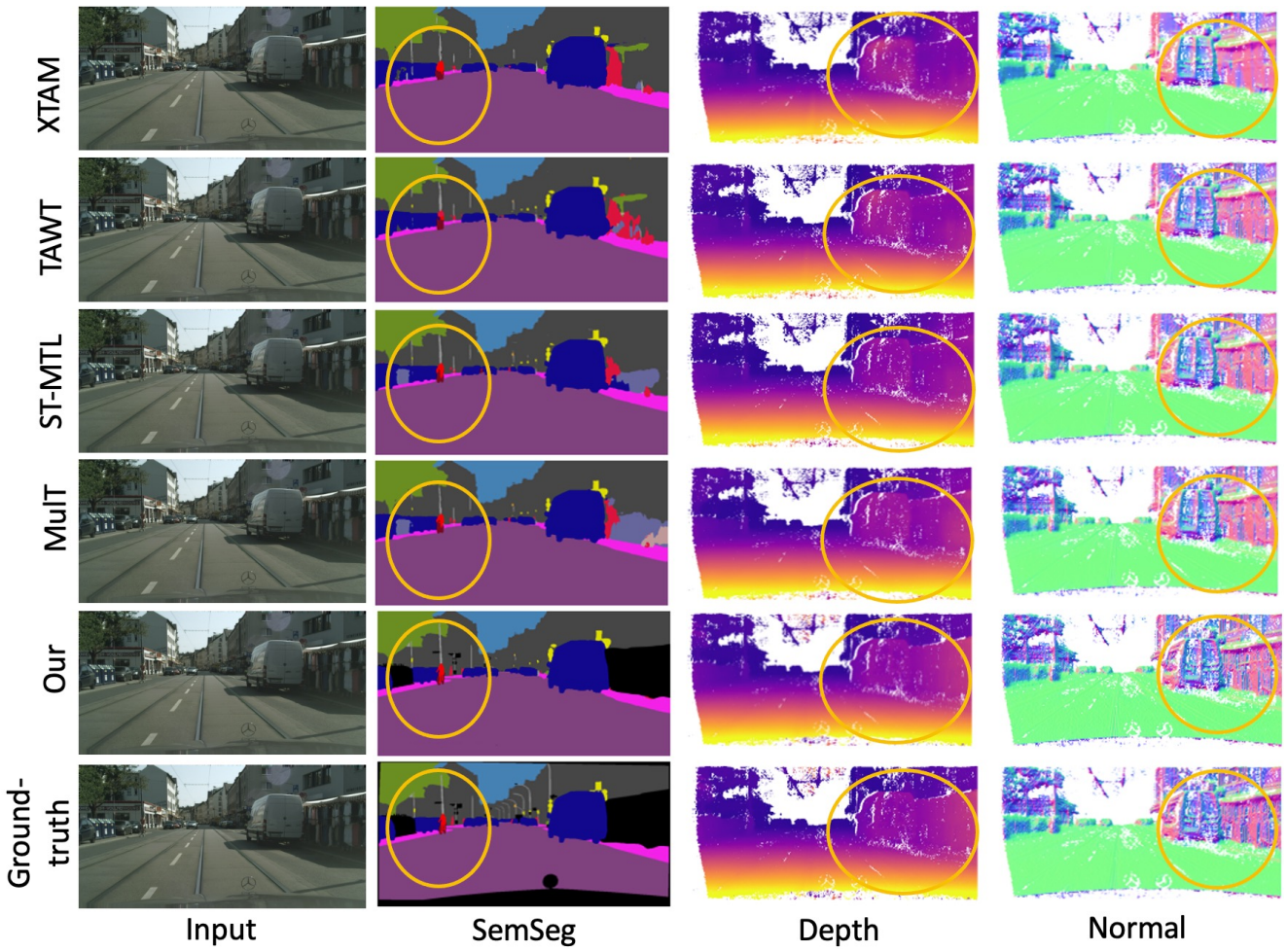


Figure 9: **Multitask Learning comparison on Cityscapes** [7] benchmark in the *'S-D-N'* setting. Our model outperforms all the multitask baselines. For instance, our method correctly segments the elements within the yellow-circled region, unlike the baselines. We show the best-performing methods in Table 2 of the main paper, i.e. XTAM [13], TAWT [5], ST-MTL [16], and MulT [1], respectively. Best seen on screen and zoomed in.

References

[1] Deblina Bhattacharjee, Tong Zhang, Sabine Süsstrunk, and Mathieu Salzmann. Mult: An end-to-end multitask learning transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12031–12041, June 2022. 1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12

[2] David Bruggemann, Menelaos Kanakis, Anton Obukhov, Stamatios Georgoulis, and Luc Van Gool. Exploring relational context for multi-task dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 2, 3, 4

[3] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv: 2001.10773*, 2020. 1, 2, 4

[4] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. *arXiv: 2012.00364, cs.CV*, 2021. 2

[5] Shuxiao Chen, Koby Crammer, Hangfeng He, Dan Roth, and Weijie J Su. Weighted training for cross-task learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. 2, 3, 4, 11, 12

[6] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 794–803, 2018. 2

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2, 5, 6, 8, 12

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. 1, 3

[9] Ronghang Hu and Amanpreet Singh. Unit: Multimodal multitask learning with a unified transformer. *arXiv: 2102.10772, cs.CV*, 2021. 2

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the IEEE/CVF European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014. 2, 3, 5, 6, 9

[11] Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. *CoRR*, abs/1803.10704, 2018. 2, 3, 4

[12] 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. *arXiv: 2103.14030, cs.CV*, 2021. 1, 3, 5, 6, 8, 9

[13] Ivan Lopes, Tuan-Hung Vu, and Raoul de Charette. Cross-task attention mechanism for dense multi-task learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023. 2, 3, 4, 5, 8, 11, 12

[14] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *Openreview*, 2018. 5

[15] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3994–4003, 2016. 2, 3, 4

[16] Eslam Mohamed and Ahmed El-Sallab. Spatio-temporal multi-task learning transformer for joint moving object detection and segmentation. *arXiv: 2106.11401, cs.CV*, 2021. 2, 3, 4, 9, 10, 11, 12

[17] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Proceedings of the IEEE/CVF European Conference on Computer Vision (ECCV)*, 2012. 1, 2, 3, 6, 10

[18] Nhu-Van Nguyen, Christophe Rigaud, and Jean-Christophe Burie. Digital comics image indexing based on deep learning. *Journal of Imaging*, 4(7), 2018. 1, 3, 5, 9

[19] Arghya Pal and Vineeth N Balasubramanian. Zero-shot task transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2, 3, 4

[20] Mengyang Pu, Yaping Huang, Qingji Guan, and Haibin Ling. Rindnet: Edge detection for discontinuity in reflectance, illumination, normal and depth. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6879–6888, October 2021. 3, 6

[21] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 3, 6

[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015. 5

[23] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3234–3243, 2016. 1, 2, 4, 5, 6, 8, 11

[24] Suman Saha, Anton Obukhov, Danda Pani Paudel, Menelaos Kanakis, Yuhua Chen, Stamatios Georgoulis, and Luc Van Gool. Learning to relate depth and semantics for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8197–8207, 2021. 1

[25] Hongje Seong, Junhyuk Hyun, and Euntai Kim. Video multitask transformer network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshop (ICCVW)*, pages 1553–1561, 2019. 2

[26] Trevor Standley, Amir R. Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? *arXiv: 1905.07553, cs.CV*, 2019. 2, 6

[27] Guolei Sun, Thomas Probst, Danda Pani Paudel, Nikola Popović, Menelaos Kanakis, Jagruti Patel, Dengxin Dai, and Luc Van Gool. Task switching network for multi-task learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8291–8300, 2021. 2, 3, 4

[28] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. 2, 3, 4, 5

[29] Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Mti-net: Multi-scale task interaction networks for multi-task learning. *Proceedings of the IEEE/CVF European Conference on Computer Vision (ECCV)*, 2020. 2, 3, 4

[30] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*, 2022. 1, 3

[31] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvtv2: Improved baselines with pyramid vision transformer. *arXiv:2106.13797, cs.CV*, 2021. 1, 3

[32] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[33] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. *arXiv: 2107.00641, cs.CV*, 2021. 1, 3

[34] Zhenheng Yang, Peng Wang, Wei Xu, Liang Zhao, and Ramakant Nevatia. Unsupervised learning of geometry with edge-aware depth-normal consistency. *arXiv preprint arXiv:1711.03665*, 2017. 2, 3, 6

[35] Hanrong Ye and Dan Xu. Inverted pyramid multi-task transformer for dense scene understanding. In *Proceedings of the IEEE/CVF European Conference on Computer Vision (ECCV)*, 2022. 2, 3, 4, 10

[36] Hanrong Ye and Dan Xu. Taskprompter: Spatial-channel multi-task prompting for dense scene understanding. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023. 2, 3, 4, 10

[37] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, volume 33, pages 5824–5836, 2020. 2

[38] Amir R Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J Guibas. Robust learning through cross-task consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11197–11206, 2020. 2, 3, 4, 5, 6

[39] Amir R. Zamir, Alexander Sax, William B. Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 3, 4, 5, 7