

Appendix

1. Dataset Details

NYUv2 contains 1449 densely labelled RGB-depth images of indoor scenes. The raw dataset contains images with incomplete depth values; which are masked during training. The tasks associated with this dataset are 13-label semantic segmentation, depth estimation, and surface normals prediction. The dataset does not contain surface normal labels out-of-the-box, so following the literature [9], we used the pseudo ground surface normals data obtained from [4], which include some incomplete values at the same locations as the corresponding depth maps. The training and validation sets contain 795 and 654 images respectively, and the resolution of the images is 288×384 .

Cityscapes is a larger dataset containing 3475 outdoor urban street scenes with fine annotations taken from 50 cities over several months of the year. From the set of fine annotations, we have 2975 train and 500 validation images. The tasks associated with this dataset are 19-label semantic segmentation and depth estimation. The labels used are from their official documentation that group several labels into a void class, and specify 19 other labels that should be used during training. The resolution of the images is 128×256 .

PASCAL-Context [13] is an even larger dataset derived from the PASCAL VOC 2010 challenge [5], containing pixel-wise annotations for 10,103 images. These images cover a wide range of indoor and outdoor scenes with various objects. The dataset includes 4,998 training and 5,105 validation images. The tasks associated this dataset are 21-label semantic segmentation, human parts segmentation, edge detection, saliency, and surface normals. The resolution of the images varies, so they are padded and scaled to 512×512 .

2. Related Works

PAD-Net [19] is the first work to popularize the “task prediction distillation” framework. Their cross-task distillation module uses EM attention, which can capture local patterns intra- and inter-task, but lacks the ability to model long-range dependencies. PAP-Net [22] is another distillation algorithm that explicitly models feature similarities, known as “task affinity” using MM attention. Although they capture local and long-range dependencies intra-task, their simplistic cross-task diffusion mechanism inhibits inter-task pattern propagation. MTI-Net [15] extends distillation to multiple feature scales, which is known as “multi-scale task-prediction distillation”. The cross-task distillation algorithm they use for each scale is the same one used by PAD-Net (i.e., EM attention). The number of additional model parameters for generating initial task predictions and cross-task distillation modules at multiple scales makes this method very inefficient as the input image size and number of tasks increase. Also, this framework isn’t suited for most ViT-based backbones that output features as a single scale.

The aforementioned cross-task distillation algorithms are inspired by the attention mechanism [16]; which allows networks to place greater emphasis on certain parts of an input that are important for the downstream task. For dense vision tasks, it has been shown that attending to features in the spatial and/or channel dimensions leads to significant performance improvements [6, 18]. Consequently, these notions have been extended to the MTL domain, which explored different ways of modelling cross-task patterns using attention [9, 14, 15, 19, 22].

Other recent MTL works for dense scene predictions include ATRC [1], InvPt [20], and TaskPrompter [21]. ATRC applies a neural architecture search (NAS) to learn a branching structure that considers the global features, local features, source label, and target labels between every possible combination of task pairs. Although this study provides interesting insights into optimal task interactions, it is difficult to justify its use in a real-world setting because it takes an incredible amount of resources to train, and scales very poorly with more tasks. Hanrong Ye and Dan Xu [20, 21] create their own multitask network based on the Vision Transformer (ViT) [3]. The added model capacity allows them to explicitly model local and global relationships between tasks. Despite both being encoder-focused works, they compare their results to the

decoder-focused distillation algorithms using CNN backbones. Although they perform an unfair comparison, the broader consideration is that encoder- and decoder-focused algorithms are not mutually exclusive and can be used in a complimentary fashion. Additionally, these encoder-focused methods are not practical for real-world application because they require a handcrafted design for a given backbone; which are constantly evolving. Decoder-focused methods, like task-prediction distillation methods, are modular and can be used with an arbitrary pretrained backbone.

3. Tasks and Performance Metrics

Semantic Segmentation refers to the task of assigning a class label to each pixel in an image. During training, the objective is to minimize the depth-wise cross-entropy loss between the predicted labels \hat{y} , and the targets y , for all N pixels:

$$\mathcal{L}_{Semantic} = -\frac{1}{N} \sum_{n \in N} y_n \log(\hat{y}_n) \quad (1)$$

We also evaluate our models on mean intersection over union (mIoU) and absolute pixel accuracy. Given the true positives (TP), false positives (FP), and false negatives (FN) for each image, we compute mIoU as follows:

$$mIoU = \frac{1}{N} \sum_{n \in N} \frac{TP_n}{TP_n + FP_n + FN_n} \quad (2)$$

Human Parts Segmentation is defined and evaluated in the exact same way as the semantic segmentation task. The only difference between these tasks is the nature of the assigned labels. For human parts segmentation, pixels are assigned a label based on a human body part rather than labels of objects (i.e., car, road, building).

Depth Estimation involves predicting the depth values at each pixel. During training, we aim to minimize the absolute error (L_1 norm) of the predicted values \hat{d} , and the targets d :

$$\mathcal{L}_{Depth} = \sum_{n \in N} ||d_n - \hat{d}_n|| \quad (3)$$

We also report on the relative depth error:

$$Error_{rel} = \sum_{n \in N} \frac{||d_n - \hat{d}_n||}{d_n} \quad (4)$$

Surface Normals prediction involves estimating the direction perpendicular to the surface of objects in an image; making it useful for acquiring geometric and structural scene information. We train the model to minimize the element-wise dot product between the normalized predictions \hat{s} , and the targets s :

$$\mathcal{L}_{Normals} = -\frac{1}{N} \sum_{n \in N} s_n \cdot \hat{s}_n \quad (5)$$

For evaluating surface normals prediction performance, we also consider the mean angular distance between \hat{s} and s . Angular distance is the arccosine of the sum of the element-wise product of \hat{s} and s , as seen in Equation 6. We also report the proportion of predictions that fall within 11.25, 22.5, and 30.0 degrees of error.

$$D_\theta = \arccos\left(\sum_{n \in N} \hat{s}_n \cdot s_n\right) \quad (6)$$

Saliency detection involves identifying the most visually important regions in an image. The model is trained to predict a saliency map \hat{S} , highlighting areas that are likely to attract human attention. During training, we minimize the pixel-wise binary cross-entropy loss between the predicted saliency map \hat{S} and the ground truth saliency map S . We evaluate using the max F-measure, which evaluates the balance between precision and recall across different thresholds applied to the predicted saliency map. The formula for the F-measure is:

$$F_\beta = \frac{(1 + \beta^2) \cdot \text{Precision} \cdot \text{Recall}}{(\beta^2 \cdot \text{Precision}) + \text{Recall}} \quad (7)$$

For the max-F measure, you compute the F-measure across multiple thresholds (τ) and take the maximum:

$$\max\text{-}F_{\beta} = \max(F_{\beta}(\tau_1), F_{\beta}(\tau_2), \dots, F_{\beta}(\tau_n)) \quad (8)$$

Edge detection involves detecting boundaries between different regions in an image. The model is trained to predict binary edge maps \hat{E} , where pixels corresponding to edges are labeled as 1 and others as 0. We minimize the binary cross-entropy loss during training and evaluate using the validation loss.

MTL Gain [12] is an aggregate measure of the overall multitask improvement of method m with respect to a single task learning baseline b for all tasks $t \in [1, N]$, as seen in Equation 9.

$$\Delta_m = \frac{1}{N} \sum_t^N (-1)^{l_t} (M_{m,t} - M_{b,t}) / M_{b,t} \quad (9)$$

where $l_t = 1$ if a lower value of metric M is favorable, and 0 otherwise. We will treat Δ_m as a percentage in our evaluation. Although we use multiple metrics per task throughout our evaluations, we want to make sure that every task is weighed evenly when calculating Δ_m by selecting a single metric per task that best demonstrates generalization performance. Consequently, to compute Δ_m , we will use mIoU for segmentation, relative error for depth, mean angular distance for surface normals, max F-measure for saliency, and validation loss for edge detection. We also show that we still achieve superior MTL gain using other combinations of metrics in the Tab. 2 and Tab. 1. In our results, the metrics where larger values are favourable are denoted with (\uparrow) and smaller values with (\downarrow).

4. Results with Additional Metrics

In Tab. 2 and Tab. 1, we can see that in addition to the results in main paper, we also outperform all other models using other metrics for the Cityscapes and NYUv2 datasets. Therefore, using any combination of evaluation metrics to compute the multitask gain (Δ_m) will show we still achieve the best overall multitask performance.

	NYUv2 (CNN)								
	SemSeg		Depth		Normals				$\Delta_m \uparrow$
Model	mIoU \uparrow	pixAcc \uparrow	relErr \downarrow	mErr \downarrow	mErr \downarrow	11.25 \uparrow	22.5 \uparrow	30 \uparrow	
STL	49.23	72.83	0.1636	0.3853	23.15	35.18	62.50	73.48	+0.00
MTL	49.25	72.90	0.1658	0.3896	24.16	30.80	57.92	70.41	-1.89
PAD-Net	50.23	73.46	0.1622	0.3814	23.63	32.44	59.51	71.68	+0.27
PAP-Net	50.00	73.25	0.1615	0.3876	23.78	31.90	58.89	71.22	+0.04
CTAL _{SS}	51.59	74.14	0.1607	0.3808	22.84	35.14	62.06	73.40	+2.64
MTI-Net	51.51	74.50	0.1538	0.3650	23.50	34.16	60.85	72.31	+3.04
CTAL _{MS}	52.70	75.09	0.1529	0.3630	22.99	35.59	62.25	73.28	+4.76

Table 1. Validation set performance taken across all tasks on NYUv2 and Cityscapes using CNN backbones. Values in bold indicate the best value in a given column for multitask models in SS and MS configurations.

5. Implementation Details

All CNN models are equipped with a pre-trained HRNet18 [17] multiscale feature extractor backbone. The single-scale variants will use a fused version of the input features following the aforementioned CSF procedure. All transformer models are equipped with a pre-trained SwinV2-s [10] backbone. Since the output of the transformer backbone is an aggregated feature representation, only SS models are evaluated. The output heads for the initial predictions include two residual blocks [7] followed by an output convolution layer. The initial predictions used for task-prediction distillation are the outputs of the second residual block. The final output heads for the CNN models use the same architecture as the heads used for the initial predictions, but for Transformer models, we use a DeepLab [2] head to get the final predictions since it is still a popular choice for dense prediction tasks like segmentation. The implementation code for all baseline networks is taken from [15], except for PAP-Net which we carefully implemented ourselves since there was not a publicly available implementation.

Model	Sem. Seg.		Depth	
	mIoU \uparrow	pixAcc \uparrow	relErr \downarrow	mErr \downarrow
STL	48.89	90.87	29.91	1.296
MTL	49.78	91.07	31.80	1.155
PAP-Net	50.82	91.19	26.97	1.135
PAD-Net	50.67	91.24	27.37	1.136
CTAL _{SS}	51.36	91.34	23.84	1.119
MTI-Net	51.77	91.13	29.90	1.141
CTAL _{MS}	51.94	91.27	22.89	1.127

Table 2. Validation set performance taken across all tasks on NYUv2. Values in bold indicate the best value in a given column for multitask models in SS and MS configurations.

6. Hyperparameters

We train our models using an Adam [8] optimizer with a weight decay of 1×10^{-4} . The learning rates are 1×10^{-4} , 5×10^{-4} , and 2×10^{-5} for NYUv2, Cityscapes, and PASCAL-Context respectively. We performed a small learning rate search (within the range of $1e-2$ and $1e-5$) for each model to ensure that this configuration was favourable for all baselines. We also use a cosine annealing learning rate scheduler [11] for smooth convergence. Multi-scale models tend to converge early for Cityscapes, so for them, we used a cosine annealing learning rate scheduler with warm restarts [11] to promote exploration and escape local minima. For all datasets, we use a batch size of 8, a blending factor $\gamma = 0.05$ (like PAP-Net) and filter size $f = 3$ for all our models. The values for γ and f were not tuned for each dataset, and our models show little sensitivity to these parameters. We train for 200, 75, and 70 epochs on NYUv2, Cityscapes, and PASCAL-Context respectively using a single NVIDIA RTX A5000 GPU. The training time per run in this setup was approximately 4 hours for NYUv2, 1 hour for Cityscapes, and 9 hours for PASCAL-Context.

7. Hyperparameter Sensitivity

7.1. Blending Factor γ

Model	γ	Sem. Seg.	Depth	Normals	MTL Gain
		mIoU \uparrow	relErr \downarrow	mErr \downarrow	$\Delta_m \uparrow$
EMA-Net (SS)	0.025	50.68	0.1608	22.53	+2.45
	0.050	51.59	0.1607	22.84	+2.64
	0.100	52.49	0.1631	22.84	+2.76
EMA-Net (MS)	0.025	51.71	0.1526	23.01	+4.12
	0.050	52.70	0.1529	22.99	+4.76
	0.100	53.44	0.1557	23.06	+4.59

Table 3. Validation set performance taken across all tasks on NYUv2 for different values of blending factor γ .

As we can see from Table 3, there is noticeable variability in segmentation performance when using different blending factors (γ) for both SS and MS models. However, we can see that the performance of the other tasks compensates accordingly, such that the overall MTL gain does not change significantly. This is consistent with the expected competitive nature between tasks when training multitask systems.

Model	f	Sem. Seg. mIoU \uparrow	Depth relErr \downarrow	Normals mErr \downarrow	MTL Gain $\Delta_m \uparrow$
CTAL _{SS}	3	51.59	0.1607	22.84	+2.64
	5	52.19	0.1630	22.89	+2.50
	7	51.58	0.1640	22.79	+2.03
CTAL _{MS}	3	52.70	0.1529	22.99	+4.76
	5	53.21	0.1547	23.06	+4.64
	7	52.70	0.1557	23.14	+3.97

Table 4. Validation set performance taken across all tasks on NYUv2 for different filter sizes f .

7.2. Filter Size f

In Table 4, we can see that using different filter sizes (f) for cross-task pattern modelling, we do not see a significant drop in performance between $f = 3$ and $f = 5$. However, using too large of a filter size, i.e., $f = 7$, we can expect a drop in performance.

8. Results With Standard Deviation

Tables 5 and 6 contain identical results presented in the main paper, but with the addition of the standard deviation across all runs. This is to provide a notion of statistical confidence for our results. The formula used to compute the standard deviation is as follows:

$$\sigma = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2} \quad (10)$$

Model	NYUv2			
	Sem. Seg. mIoU (σ) \uparrow	Depth relErr (σ) \downarrow	Normals mErr (σ) \downarrow	$\Delta_m \uparrow$
STL	49.23 (0.29)	0.1636 (0.0024)	23.15 (0.09)	+0.00
MTL	49.25 (0.43)	0.1658 (0.0028)	24.16 (0.05)	-1.89
PAP-Net	50.00 (0.49)	0.1615 (0.0043)	23.78 (0.07)	+0.04
PAD-Net	50.23 (0.41)	0.1622 (0.0016)	23.63 (0.06)	+0.27
CTAL _{SS}	51.59 (0.33)	0.1607 (0.0008)	22.84 (0.06)	+2.64
MTI-Net	51.51 (0.63)	0.1538 (0.0011)	23.50 (0.04)	+2.64
CTAL _{MS}	52.70 (0.34)	0.1529 (0.0027)	22.99 (0.06)	+4.76

Table 5. Average validation set performance taken across all tasks on NYUv2 for 3 runs. Values in bold indicate the best value in a given column for multitask models in SS and MS configurations. Values in brackets indicate the standard deviation across three runs.

Model	Cityscapes		
	Sem. Seg. mIoU (σ) \uparrow	Depth relErr (σ) \downarrow	Δ_m \uparrow
STL	48.89 (0.74)	29.91 (0.88)	+0.00
MTL	49.78 (0.36)	31.80 (0.48)	-2.25
PAP-Net	50.82 (0.72)	26.97 (0.67)	+6.89
PAD-Net	50.67 (0.44)	27.37 (0.52)	+6.07
CTAL _{SS}	51.36 (0.64)	23.84 (0.58)	+12.67
MTI-Net	51.77 (0.84)	29.90 (0.48)	+2.96
CTAL _{MS}	51.94 (0.26)	22.89 (0.48)	+14.85

Table 6. Average validation set performance taken across all tasks on Cityscapes for 3 runs. Values in bold indicate the best value in a given column for multitask models in SS and MS configurations. Values in brackets indicate the standard deviation across three runs.

References

- [1] David Brüggenmann, Menelaos Kanakis, Anton Obukhov, Stamatis Georgoulis, and Luc Van Gool. Exploring relational context for multi-task dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15869–15878, 2021.
- [2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.
- [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. *arXiv preprint arXiv:2010.11929*, 2020.
- [4] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015.
- [5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. <http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html>.
- [6] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3146–3154, 2019.
- [7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [9] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1871–1880, 2019.
- [10] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022.
- [11] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.
- [12] Kevis-Kokitsi Maninis, Ilija Radosavovic, and Iasonas Kokkinos. Attentive single-tasking of multiple tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1851–1860, 2019.
- [13] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [14] Dimitrios Sinodinos and Narges Armanfard. Attentive task interaction network for multi-task learning. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 2885–2891. IEEE, 2022.
- [15] Simon Vandenhende, Stamatis Georgoulis, and Luc Van Gool. Mti-net: Multi-scale task interaction networks for multi-task learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 527–543. Springer, 2020.
- [16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

- [17] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3349–3364, 2020.
- [18] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.
- [19] 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 Conference on Computer Vision and Pattern Recognition*, pages 675–684, 2018.
- [20] Hanrong Ye and Dan Xu. Inverted pyramid multi-task transformer for dense scene understanding. In *ECCV*, 2022.
- [21] Hanrong Ye and Dan Xu. Taskprompter: Spatial-channel multi-task prompting for dense scene understanding. In *ICLR*, 2023.
- [22] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Yan Yan, Nicu Sebe, and Jian Yang. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4106–4115, 2019.