Supplementary Material: Weakly-Supervised Domain Adaptive Semantic Segmentation with Prototypical Contrastive Learning

Anurag Das[1], Yongqin Xian[2]*, Dengxin Dai[1], Bernt Schiele[1]
[1]MPI for Informatics, Saarland Informatics Campus, [2]ETH Zurich
{andas,ddai,schiele}@mpi-inf.mpg.de, yongqin.xian@gmail.com

In this supplementary material, we provide additional experimental results and technical details that were omitted in the main paper due to space constraints.

- Section 1: Additional details
  - Classwise result on GTA5→Cityscapes and Synthia→Cityscapes, Tab. 1 main paper
  - Annotation cost vs performance plot for Synthia→Cityscapes, Fig. 3 main paper
  - Boundary loss ($\mathcal{L}^2_{boundary}$, Sec. 3.1, main paper)
  - Image loss ($\mathcal{L}^2_{image}$, Sec. 3.4, main paper)
  - Hyerparameters for our framework
  - Algorithm for our framework
  - Visualisation of features
- Section 2: More qualitative results

1. Additional details

1.1. Classwise result on GTA5→Cityscapes and Synthia→Cityscapes

In this experiment, we perform two comparisons. First, we compare classwise mIoU for our framework for each weak label with prior works(WeakSegDA [6], coarse-to-fine [1]). Second, we compare our framework's performance with UDASS SoTA methods(CorDA [10], ProDA [11], DASS [4]). We report the results for GTA5→Cityscapes in Tab. 1 and for Synthia→Cityscapes in Tab. 2.

For the first comparison, our framework consistently outperforms prior works for most of the classes for both GTA5→Cityscapes and Synthia→cityscapes. Notably, for tail distribution classes like train, truck, bus and traffic light we see significant improvement. For e.g., we see an improvement of 26.4, 40 and 16.8 mIoU for train class for image, point and coarse labels respectively, in GTA5→Cityscapes setting. Similarly for traffic light, we see an improvement of 25.2, 11.6 and 12.1 mIoU for image, point and coarse labels respectively for Synthia→Cityscapes setting. This improvement for tail classes shows the advantage of using our framework for tail distribution classes, that are difficult to predict.

For second comparison, overall our framework with additional weak labels outperforms UDASS for most of classes. We specifically point out the performance of class 'train'. For UDASS, due to big domain gap between GTA5 train and Cityscapes 'train', the performance is bad, eg. 1.0 mIoU ProDA [11] and 4.4 for DASS [4]. With additional weak labels from target domain, we significantly improve over UDASS by achieving 60.8 mIoU with just image label, 63.2 with point label and 68.3 mIoU for coarse labels. We see similar improvements for fence in Synthia→Cityscapes setting. These results show the importance of using additional cheaper weak labels for improving the performance over UDASS task.

Annotation cost vs performance plot for Synthia→Cityscapes In this experiment, we extend the results from Sec. 4.2 from the main paper by showing additional results from Synthia→Cityscapes. We show the cost effectiveness of different weak labels for WDASS task. We first compare the performance within different weak labels(eg. image, point and coarse labels). Similar to GTA5→Cityscapes setting, we observe that Ours-Point outperform other weak labels suggesting it to be more suited for lower budget settings. Next, we compare our weak label performance with supervised baselines (only source, supervised). For all weak labels, our framework outperforms supervised baselines. Further, with only 8% budget (347 vs 4463 hrs) our framework with coarse annotation bridges the gap with supervised learning with a gap of only 2.9% mIoU. Overall, we show that weak labels are cost-effective alternative that achieve competitive performances compared to fine labels.

Boundary loss, Sec.3.1 main paper Coarse labels do not have labeled boundaries. This becomes severe with point

*Currently with Google. This work was done at ETH Zürich.

GTA5 → Cityscapes

	Method	road	sidewalk	building	wall	fence	pole	tra. light	tra. sign	vege.	terrain	sky	person	rider	car	truck	bus	train	motor.	bicycle	mIoU	gap
	Source	75.8	16.8	77.2	12.5	21.0	25.5	30.1	20.1	81.3	24.6	70.3	53.8	26.4	49.9	17.2	25.9	6.5	25.3	36.0	36.6	+30.8
UDA	CorDA [10]	94.7	63.1	87.6	30.7	40.6	40.2	47.8	51.6	87.6	47.0	89.7	66.7	35.9	90.2	48.9	57.5	0.0	39.8	56.0	56.6	+10.8
	ProDA [11]	87.8	56.0	79.7	46.3	44.8	45.6	53.5	53.5	88.6	45.2	82.1	70.7	39.2	88.8	45.5	59.4	1.0	48.9	56.4	57.5	+9.9
	DASS [4]	93.5	60.2	88.1	31.1	37.0	41.9	54.7	37.8	89.9	45.5	89.9	72.7	38.2	90.7	34.3	53.2	4.4	47.2	58.5	57.1	+10.3
image	baseline	82.9	35.9	80.9	30.5	32.2	39.9	48.2	45	86.6	41	84	66.3	21.8	86.8	45.9	48.4	10.8	43.7	45.7	51.4	+16.0
	WeakSegDA [6]	89.5	**54.1**	83.2	31.7	34.2	37.1	43.2	39.1	85.1	39.6	85.9	61.3	34.1	82.3	42.3	51.9	34.4	33.1	45.4	53.0	+14.4
	Ours	**91.9**	51.2	**87.6**	**41.2**	**41**	**47.1**	**55.7**	**47.8**	**89.6**	**42.7**	**89.2**	**70.8**	**35.5**	**90.1**	**59.8**	**71.9**	**60.8**	**46.8**	**47.3**	**61.5**	**+5.9**
point	baseline	81.9	40.7	85	32.9	37.1	46.2	51.1	55.7	86	41.9	82.5	68.2	42.4	89	46.7	45.7	20.9	35.8	53.6	54.9	+12.9
	WeakSegDA [6]	94.0	62.7	86.3	36.5	32.8	38.4	44.9	51.0	86.1	43.4	**87.7**	66.4	36.5	87.9	44.1	58.8	23.2	35.6	55.9	56.4	+11.0
	Ours	**95.5**	**71.3**	**87.6**	**43.3**	**43.3**	**47.7**	**51.3**	**58.7**	**87.0**	**45.5**	86.4	**73.6**	**49**	**91.4**	**56.7**	**65.2**	**63.2**	**46.8**	**67.0**	**64.7**	**+2.7**
coarse	baseline	93.5	63.2	86.4	45.4	38.1	47.1	52	48.6	87.8	46.2	89.3	71.2	40.7	89.9	56.4	61.3	40.5	46.7	51.3	60.8	+6.6
	Coarse-to-fine [1]	**96.4**	**75.1**	**89.9**	**51.6**	47.3	49.6	53.7	62.2	89.5	45.2	91.0	71.4	46.4	**92.2**	**69.6**	72.9	51.5	51.1	61.7	66.7	+0.7
	Ours	95.5	71	89.2	49.3	**51.7**	**52.0**	**60.0**	**64.2**	**89.8**	**51.4**	**91.5**	**73.8**	**46.5**	91.5	69.4	**75.3**	**68.3**	**55.0**	**68.4**	**69.1**	**-1.7**
	Supervised	97.1	78.1	89.2	44.3	46.5	49.2	46.8	63.1	90.2	52.9	92.0	73.3	49.0	91.9	67.4	71.9	59.1	49.1	69.4	67.4	0.0

Table 1. Comparison results on GTA→Cityscapes. We report per-class IoU as well as overall mean IoU(mIoU). We show two comparisons, first, domain adaptation with no labels from target domain(UDA) vs domain adaptation with weak labels(image, point, coarse) from target domain. Second, comparison of our method(ours) with the baseline and existing methods(WeakSegDA [6], Coarse-to-fine) for each weak label. gap: performance gap for mIoU scores wrt to supervised setting. Lower value of gap is better. 'baseline': segmentation network trained on source labels and target weak labels.

Synthia → Cityscapes

	Method	road	sidewalk	building	wall*	fence*	pole*	tra. light	tra. sign	vege.	sky	person	rider	car	bus	motor.	bicycle	mIoU	mIoU*	gap
	Source	64.3	21.3	73.1	2.4	1.1	31.4	7.0	27.7	63.1	67.6	42.2	19.9	73.1	15.3	10.5	38.9	34.9	40.3	+33.6
UDA	CorDA [10]	93.3	61.6	85.3	19.6	5.1	37.8	36.6	42.8	84.9	90.4	69.7	41.8	85.6	38.4	32.6	53.9	55.0	62.8	+11.1
	ProDA [11]	87.8	45.7	84.6	37.1	0.6	44.0	54.6	37.0	88.1	84.4	74.2	24.3	88.2	51.1	40.5	45.6	55.5	62.0	+11.9
	DASS [4]	83.8	42.2	85.3	16.4	5.7	43.1	48.3	30.2	89.3	92.1	68.2	43.1	89.7	47.2	42.2	54.2	55.6	62.9	+11.0
image	baseline	83.6	34.1	74.7	22.1	9.5	37.8	41.4	41.4	62.2	32.3	2.0	32.8	69.4	32.6	0	1.8	36.1	39.1	+34.8
	WeakSegDA [6]	**92.3**	**51.9**	81.9	21.1	1.1	26.6	22.0	24.8	81.7	87.0	63.1	33.3	83.6	**50.7**	33.5	54.7	50.6	58.5	+15.4
	Ours	90.1	42.7	**85.6**	**29.6**	**24.5**	**46.3**	**47.2**	**51.2**	**87.6**	**88.8**	**70.6**	**34.9**	**84.3**	45.8	**38.2**	**64.3**	**61.3**	**63.9**	**+10.0**
point	baseline	88.8	55.8	75.4	29.2	14.3	40	35.6	43.5	55.3	60.5	45.9	39	74.6	34.9	26.6	56.8	48.5	53.3	+20.6
	WeakSegDA [6]	94.9	63.2	85.0	27.3	24.2	34.9	37.3	50.8	84.4	**88.2**	60.6	36.3	86.4	43.2	36.5	61.3	57.2	63.7	+10.2
	Ours	**95.4**	**68.7**	**85.4**	**37.5**	**29.3**	**44.0**	**48.9**	**56.4**	**86.8**	86.8	**70.6**	**47.1**	**89.7**	**50.8**	**41.1**	**65.8**	**62.8**	**68.7**	**5.2**
coarse	baseline	92.7	57.9	81.6	32.5	27.3	43.4	40.0	43.0	81.1	85.8	47.4	38.6	80.3	46.2	22.8	51.5	54.6	59.1	+14.8
	Coarse-to-fine [1]	**95.5**	**69.9**	87.3	38.4	29.7	44.9	40.1	53.7	87.0	**90.3**	70.9	39.9	87.8	53.6	35.4	61.6	61.6	67.2	+6.7
	Ours	93.4	68.6	**87.4**	**42.9**	**39.1**	**50.7**	**52.7**	**64.8**	**87.9**	77.3	**73.1**	**42.1**	**89.3**	**70.7**	**46.8**	**68.7**	**66.0**	**71.0**	**+2.9**
	Supervised	97.1	78.1	89.2	44.3	46.5	49.2	46.8	63.1	90.2	92.0	73.3	49.0	91.9	71.9	49.1	69.4	68.8	73.9	0.0

Table 2. Comparison results on Synthia→Cityscapes. We report per-class IoU as well as mean IoU for 16 classes(mIoU) and 13 classes(mIoU*) excluding * marked classes. We show two comparisons, namely 1) domain adaptation with no labels from target domain(UDA) vs domain adaptation with weak labels(image, point, coarse) from target domain. 2) Comparison of our method(ours) with baseline and existing methods(WeakSegDA [6]) for each weak label. gap: performance gap for mIoU scores wrt to supervised setting. Lower value of gap is better. 'baseline': segmentation network trained on source labels and target weak labels.

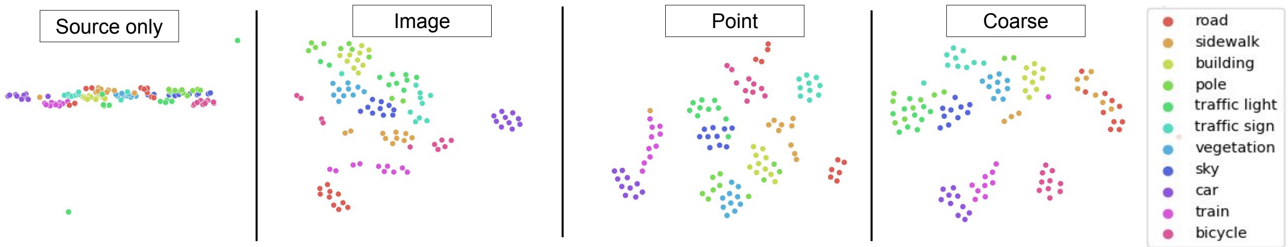

Figure 1. We show TSNE visualisation of pixel features for different weak labels(e.g. image, point and coarse labels) obtained from trained network via our framework. We also show features for 'source-only' which is features obtained by segmentation network trained only on source domain. We observe that features from our framework for WDASS forms better compact clusters than 'source-only'. Further our framework with point and coarse label training generate features that are more compact than with image label training. Setting: GTA5→Cityscapes; We randomly sample an image from Cityscapes validation set and generate its pixel features following different settings (e.g. source only, image, point etc)

labels where only one pixel is labeled per class in an image and with image label where no label is labeled. [1] shows that we can learn boundaries from source domain with labeled boundaries. We extend [1] for point and image labels and observe that it is also helpful for these weak labels(see Tab.2 main paper). We obtain the boundary loss($\mathcal{L}_{boundary}$) by matching the ground truth boundary and prediction boundary for source domain. Let y_s be the ground truth label mask from source domain and $\hat{y}_s$ be the corresponding predicted label. We obtain the boundary by computing the gradient of the labels. Specifically for ground truth boundary (Γ_{GT}) and prediction boundary (Γ_{pred}) we compute boundary as :

$$\Gamma_{GT} = ||\nabla y_s||_2; \Gamma_{pred} = ||\nabla \hat{y}_s||_2 \tag{1}$$

where ∇ is the gradient operator, which estimates gradient at each pixel by taking the central difference of label values for the pixel. Since $\hat{y}_s$ is not differentiable, we estimate $\hat{y}_s$ with Gumbel Softmax trick [3] making it differentiable. Further, from the boundary values from Γ_{GT} and Γ_{pred}, we select representative boundary pixels (p^+_{GT} for ground truth boundary pixels and p^+_{pred} for segmentation prediction boundary pixels) by thresholding them (threshold=$1e^{-8}$), i.e. p^+_{GT} are pixels where $\Gamma_{GT} > 1e^{-8}$ and p^+_{pred} are pixels with $\Gamma_{pred} > 1e^{-8}$. We define boundary loss $\mathcal{L}_{boundary}$ as,

$$\begin{aligned}\mathcal{L}_{boundary} = \lambda_1|\Gamma_{pred}(p^+_{GT}) - \Gamma_{GT}(p^+_{GT})| \\ +\lambda_2|\Gamma_{pred}(p^+_{pred}) - \Gamma_{GT}(p^+_{pred})|\end{aligned} \tag{2}$$

Following [7] we set λ_1 and λ_2 as 0.5. This boundary loss is only applied on source domain as it has fine boundary details.

Image loss, Sec. 3.4, main paper We apply image loss at the classification logits of the network as explained in Sec. 3.4 of the main paper (Eq. 12) following [6]. First we obtain image level probability for a class k, p^k_t using pixelwise class logit scores $m^{(i,k)}$:

$$p^k_t = \sigma(log\frac{1}{N}\sum_i^N \exp m^{(i,k)}) \tag{3}$$

As discussed in the main paper (Sec.3.2), this LogSumExp expression estimates smooth maximum over the logits, representing most activated pixel in the class map. With the image level class prediction probability p^k_t and image labels y^k_t, we obtain the image label loss ($\mathcal{L}^2_{image}$) using

$$\mathcal{L}^1_{image} = \sum_{k=1}^{K} -y^k_t log(p^k_t) - (1-y^k_t)log(1-p^k_t) \tag{4}$$

This image loss penalizes predictions of classes not present in the image and improves predictions for classes present in an image, thus overall improving performance.

Algorithm for our framework We show the summary of training procedure for our framework via **??**. Our framework has two key components, namely, Weak Label Guided Prototype Learning and Prototype based Contrastive Learning. For detailed information about the components, please see the main paper (Sec. 3.2 and 3.3).

Hyperparameters and implementation details for our framework Following previous works [6, 9, 11] we resize the image for training. For Cityscapes we resize the images to 512x1024 resolution, whereas for GTA5, we resize the image to 720x1280 resolution. We perform no resizing to Synthia images and use its original image size. For training we set the batch size to 4. We perform standard data augmentations for training. Specifically, we perform random resize with scale factor between [0.5,2] and random flip. Further, we set the crop size for training as 512x512 [6, 11]. Following [1, 2, 8], we also perform cross domain augmentations with every target image during training, where we cut paste pseudo labeled class mask [5] from target domain to real images. We set the pseudo label confidence threshold

Algorithm 1: Our framework for Weakly supervised Domain Adaptive Semantic Segmentation task

```
Input: Source data : $(x_s, y_s)_{j=1}^{n_s}$, Target data : $(x_t, y_t)_{j=1}^{n_t}$, where $y_t \in$ {image, point, coarse} labels; student network, $g_\theta$, pretrained on source
    dataset, teacher network, $h_\phi$; $n\_iterations$: 150000; $n_b$: batch-size; $\tau = 1$
Initialise: $\phi = \theta$ // teacher weight initialised as student weight
for $j \leftarrow 0$ to n_iterations do
    Get source data $(x_s, y_s)_{n=1}^{n_b}$, target data $(x_t, y_t)_{n=1}^{n_b}$

    /* Compute prototypes following Sec. 3.2                                   */
    Get pixel features for source and target, $f_s^i$, $f_t^i$
    Compute weight of features as similarity wrt anchor($w^{(i,k)}$),
    Compute class prototype $\eta_t^k$ as weighted average of features using weight $w^{(i,k)}$, Eq. 3, main paper
    Correct features using image loss $\mathcal{L}_{image}^1$, Eq. 6 main paper

    /* perform contrastive alignment, loss $\mathcal{L}_{intra}^t$, $\mathcal{L}_{intra}^s$, $\mathcal{L}_{inter}$        */
    Compute intra domain alignment loss $\mathcal{L}_{intra}^s$ for source, Eq. 7, main paper
    Compute intra domain alignment loss $\mathcal{L}_{intra}^t$ for target, Eq. 7, paper
    $\mathcal{L}_{intra} = \mathcal{L}_{intra}^t + \mathcal{L}_{intra}^s$
    Compute inter domain alignment loss $\mathcal{L}_{inter}$ for source to target alignment, Eq. 9, main paper
    $\mathcal{L}_{contrast} = 0.5\mathcal{L}_{intra} + 0.5\mathcal{L}_{inter}$

    Compute $\mathcal{L}_{base}$ (Eq. 11), $\mathcal{L}_{boundary}$ (Eq. 2 supplement), $\mathcal{L}_{image}$ Eq. 12 main paper

    Train $g_\theta$ with $\mathcal{L}_{base}$, $\mathcal{L}_{contrast}$, $\mathcal{L}_{image}$ and $\mathcal{L}_{boundary}$ as in Eq. 13 Sec. 3.4

    $\phi = \alpha\phi + (1 - \alpha)\theta$ // update teacher weight
end
```

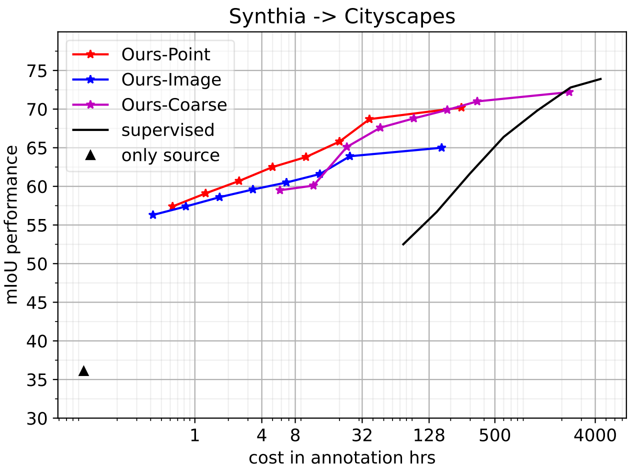

Figure 2. Annotation cost vs. performance. 1. Comparison of WDASS (point, image, coarse) vs. fully supervised training on the source (only source) and target (supervised). 2. Comparison within weakly supervised domain adaptation for various weak labels (Point, Image, Coarse labels). Results reported in mIoU for 13 classes for Synthia.

as 0.96 as in prior works. For training, we use SGD optimiser with poly learning rate scheduler having initial learning rate of 2.5x10e^{-4}. We set momentum of 0.9 and weight decay of 0.0001 and training for 150000 iterations. We perform multiscale evaluation, where we average the logits for different scales for an image. Specifically, we perform scaling with a factor of $\{0.5,1,2\}$. We make predictions using the averaged logits.

Visualisation of features We provide TSNE based feature visualisation for our pixel features in Fig. 1. We observe that features obtained for 'source only' setting, i.e. network trained only on source data are not properly separable for different classes. Our framework using additional weak labels (e.g. image, point and coarse labels) for WDASS task improves the feature representation by forming compact and separable feature representations for different classes. Further our framework trained with coarse and point weak labels generate more compact features than with image label training.

2. Qualitative Results

We perform qualitative comparison of our framework with prior works for GTA5→Cityscapes in Fig. 3 and for Synthia→Cityscapes in Fig. 4. For both settings our framework predicts better than the prior works. Particularly, for image based weak labels, our framework performs exceptionally well compared to baseline (weakSegDA [6]). For example, in Fig. 3, row 3, class 'bus' is not even predicted by baseline, whereas our framework predicts 'bus' better. We observe similar findings with the difficult Synthia→Cityscapes setting as well, see Fig. 4. The segmentation quality is best for coarse annotation, owing to availability of more labeled pixels from target domain. Further, even with just one labeled pixel per class for point label, our framework performs quite well in comparison to coarse annotation, see prediction of ‘bicycle’ in row 1 and ‘train’ in row 2 of Fig. 3. To summarise, our framework utilising additional cheap weak labels outperforms previous prior works and works best with coarse annotation.

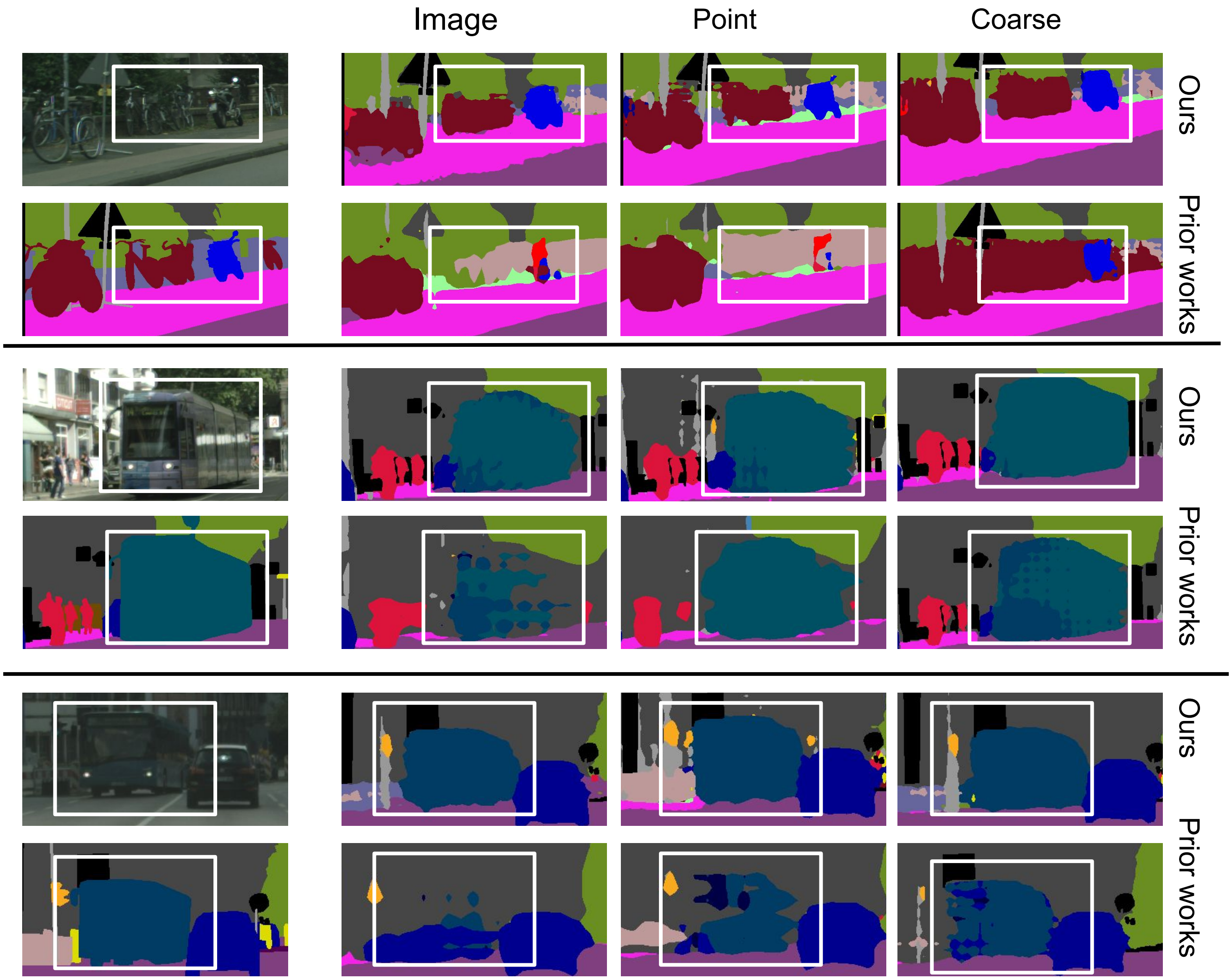

Figure 3. Qualitative results on GTA5→Cityscapes setting. On the left column we show the image and ground truth, whereas on right we compare our frameworks results with prior works. For image and point label we compare our framework performance with [6], whereas for coarse label, we compare with [1]. We present three different comparisons. In first row, we show our framework is makes better prediction for motorbike as well as bicycle class. For second row, we compare the the performance of class train, whereas for the last row, we show performance of class ‘bus’. Overall, our framework segments the classes better than the prior works for all weak labels. Please note that for better visualisation, we crop the region of interest from full image. Region of interest in white bounding boxes.

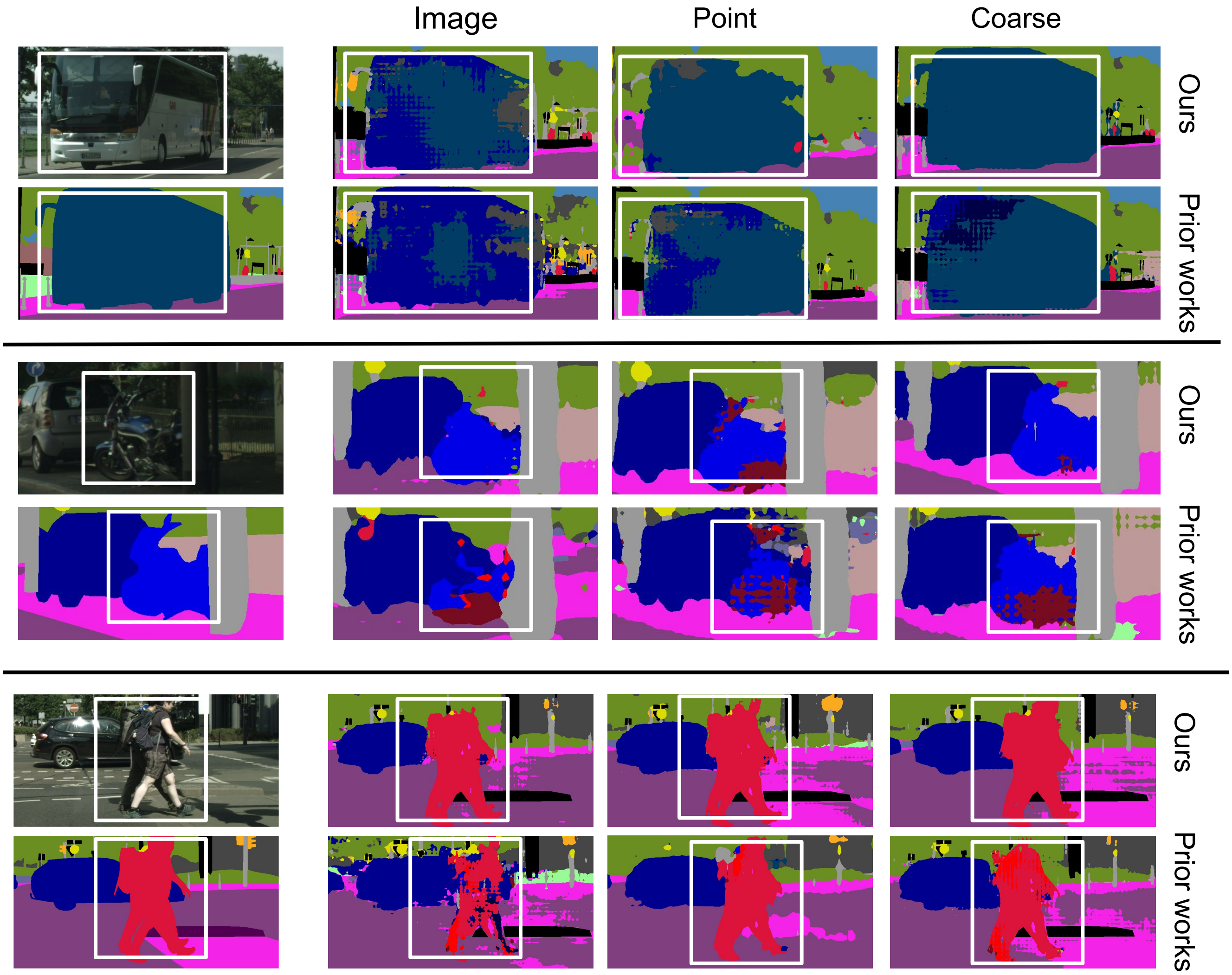

Figure 4. Qualitative results on Synthia→Cityscapes setting. On the left column we show the image and ground truth, whereas on right we compare our frameworks results with prior works. For image label we compare our framework performance with [6], whereas for coarse label, we compare with [1]. For point label we compare with baseline as in Tab. 1 main paper. We present three different comparisons. In first row, we show our framework is makes better prediction for bus class. For second row, we compare the the performance of class motorbike. Finally for the last row, we show the performance class 'person'. Overall, our framework segments the classes better than the prior works for all weak labels. Please note that for better visualisation, we crop the region of interest from full image. Region of interest in white bounding boxes.

References

[1] Anurag Das, Yongqin Xian, Yang He, Zeynep Akata, and Bernt Schiele. Urban scene semantic segmentation with low-cost coarse annotation. In *2023 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2023. 1, 2, 3, 6, 7

[2] Lukas Hoyer, Dengxin Dai, and Luc Van Gool. Daformer: Improving network architectures and training strategies for domain-adaptive semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9924–9935, 2022. 3

[3] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 3

[4] Geon Lee, Chanho Eom, Wonkyung Lee, Hyekang Park, and Bumsub Ham. Bi-directional contrastive learning for domain adaptive semantic segmentation. *arXiv preprint arXiv:2207.10892*, 2022. 1, 2

[5] Viktor Olsson, Wilhelm Tranheden, Juliano Pinto, and Lennart Svensson. Classmix: Segmentation-based data augmentation for semi-supervised learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1369–1378, 2021. 3

[6] Sujoy Paul, Yi-Hsuan Tsai, Samuel Schulter, Amit K Roy-Chowdhury, and Manmohan Chandraker. Domain adaptive semantic segmentation using weak labels. In *European conference on computer vision*, pages 571–587. Springer, 2020. 1, 2, 3, 4, 6, 7

[7] Towaki Takikawa, David Acuna, Varun Jampani, and Sanja Fidler. Gated-scnn: Gated shape cnns for semantic segmentation. In *ICCV*, pages 5229–5238, 2019. 3

[8] Wilhelm Tranheden, Viktor Olsson, Juliano Pinto, and Lennart Svensson. Dacs: Domain adaptation via cross-domain mixed sampling. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1379–1389, 2021. 3

[9] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7472–7481, 2018. 3

[10] Qin Wang, Dengxin Dai, Lukas Hoyer, Luc Van Gool, and Olga Fink. Domain adaptive semantic segmentation with self-supervised depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8515–8525, 2021. 1, 2

[11] Pan Zhang, Bo Zhang, Ting Zhang, Dong Chen, Yong Wang, and Fang Wen. Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12414–12424, 2021. 1, 2, 3