Supplementary Materials
Unsupervised Domain Adaptive 3D Detection with Multi-Level Consistency

1. Overview

We provide additional implementation details, experiment results and visualizations that are not included in the main paper due to space constraint.

- Section 2. We describe the implementation details and the experimental results of MLC-Net with 3DSSD [10] used as the base model.
- Section 3. We provide the implementation details of our MLC-Net (PointRCNN [6] as the base model) and other baselines that are discussed in the main paper.
- Section 4. Additional experimental results.
- Section 5. More qualitative visualization results.

2. Implementation based on 3DSSD

We demonstrate that our proposed method is detector-agnostic by adapting MLC-Net to one-stage detector 3DSSD.

2.1. MLC-Net on 3DSSD

Being a one-stage detector, 3DSSD differs from two-stage detectors that it does not generate region proposals. Instead, as shown in Figure 1, 3DSSD first employs a modified PointNet++ [4] model to extract point cloud features and downsample the points. A candidate generation layer is then applied to predict candidate shifts R which are the offsets of object locations relative to the downsampled points. The corrected points are treated as candidate points and candidate grouping is performed to generate instance-level features, and the final predictions S are predicted by the prediction head. Please refer to [10] for more details.

Despite that 3DSSD does not have the region proposal stage as PointRCNN, we highlight that MLC-Net is highly compatible as long as there are point-based operations and final instance predictions. Catering to the model architecture, we compute the consistency loss based on the difference between the student and teacher predictions of R and S. For candidate shifts R, we establish the point correspondences by passing the sampling index of the teacher model to the student model. As a result, both models sample the same points and the one-to-one matching of points is established. Similar to our implementation based on PointRCNN where region proposals of the teacher model is used for feature pooling at the student model, we copy the candidate points from the teacher to the student model for candidate grouping to obtain instance-level features. This operation guarantees the correspondences of final predictions S. The rest of the operations are similar to that of our method introduced in the main paper.

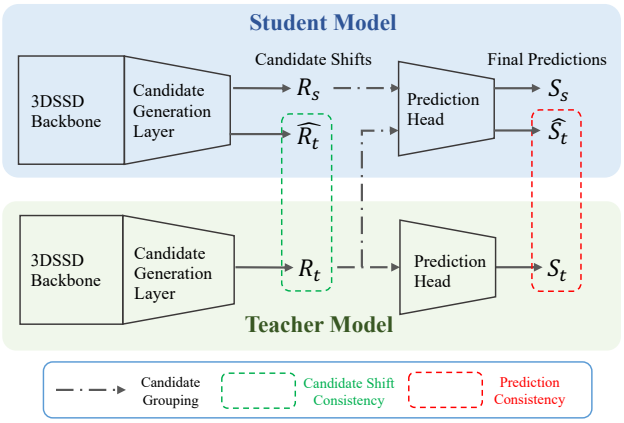

Figure 1: Architecture of MLC-Net with 3DSSD as the base model. The consistency loss is computed based on the candidates shifts R and the final predictions S. Candidate points generated by the teacher model is passed to the student model to establish correspondence between student prediction $\hat{S}_t$ and teacher prediction S_t.

2.2. Experiments

We evaluate MLC-Net implemented with 3DSSD on KITTI [2] and Waymo [7] datasets. For KITTI to Waymo transfer, we pretrain the model on KITTI dataset for 80 epochs and finetune with our proposed method for 20 epochs. For Waymo to KITTI transfer, we pretrain the model on Waymo dataset for 40 epochs and finetune with MLC-Net for 5 epochs. Note that all the source domain examples are used once in each training epoch with the target domain data randomly sampled or resampled to match the number. The number of epochs for the Waymo to KITTI

Table 1: Performance comparison on Waymo validation dataset for transfer from KITTI to Waymo with 3DSSD as the base model.

KITTI ! Waymo				
Methods	AP/L1	APH/L1	AP/L2	APH/L2
Direct Transfer	3.29	3.26	2.78	2.75
Wide-Range Aug	16.67	16.48	14.73	14.56
OT [9]	24.56	24.23	22.70	22.39
SN [9]	25.95	25.61	24.00	23.67
Ours	**29.87**	**29.27**	**26.80**	**26.27**

Table 2: Performance comparison on KITTI validation dataset for transfer from Waymo to KITTI with 3DSSD as the base model.

Waymo ! KITTI			
Methods	Easy	Moderate	Hard
Direct Transfer	6.31	6.41	6.25
Wide-Range Aug	37.83	35.34	34.05
OT [9]	45.42	40.50	41.04
SN [9]	47.81	45.92	46.46
Ours	**56.86**	**48.74**	**48.32**

transfer task is lower because the Waymo dataset has a substantial larger number of samples as compared to KITTI. During finetuning, we use the ADAM [3] optimizer with a learning rate of 0.001. The batch size is set to 32.

Table 1 and Table 2 report the performance comparison for KITTI to Waymo and Waymo to KITTI transfer tasks, respectively. It can be seen that MLC-Net consistently outperforms the Statistical Normalization (SN) method for both tasks. It is interesting to observe that the cross-domain detection performance of 3DSSD is lower than PointRCNN on these tasks, while 3DSSD has stronger in-domain performance as reported in [10]. This could be attributed to the one-stage design of 3DSSD that no bounding box refinement is performed, which makes it less robust to scale variations.

3. Implementation Details

In this section, we provide implementation details of MLC-Net with PointRCNN as the base model as well as other baselines used in comparison.

3.1. MLC-Net on PointRCNN

For our PointRCNN-based implementation of MLC-Net discussed in the main paper, we build our method on the official implementation [8]. For all the experiments, we first pretrain the base model with source data and load the student and teacher models with the same pretrained weights as initialization. For the pretraining, the default configurations provided in [8] are used. We then train the models with our proposed method with ADAM optimizer and a learning rate of 0.0001. A batch size of 32 is used. When KITTI is the source domain, we train the MLC-Net for 20 epochs and set m as 0.99 and α as 0.05. When transferring to KITTI, we conduct training for 5 epochs set m and α as 0.999 and 0.001, respectively. For all the experiments, the source domain random scaling augmentation range is set to [0.7, 1.3], while the target domain input augmentation h and RoI augmentation ξ both use a range of [0.9, 1.1]. The probability threshold ε is set to 0.99 and the loss weight coefficient λ is set to 0.1. We follow [9] and conduct all the evaluations on the car category. We implement our method using Pytorch and run the experiments with 8 NVIDIA V100 GPUs.

Table 3: Ablation study of random data augmentation. Both input and RoI augmentations force the model to be more adaptive to scale variations, and are found to be useful. They are also complementary to each other: applying both achieves the best result.

Input Aug	RoI Aug	AP/L1	APH/L1	AP/L2	APH/L2
		34.34	33.89	30.86	30.45
X		36.23	35.81	32.57	32.19
	X	35.86	35.44	32.22	31.85
X	X	**38.21**	**37.74**	**34.46**	**34.04**

3.2. Other Baselines

For the comparing methods in Section 4.2 of the main paper, Direct Transfer uses a default random scaling input augmentation range of [0.95, 1.05], while Wide-Range Aug refers to a wide random scaling input augmentation range of [0.7, 1.3], which is the same as the setting for MLC-Net. To adapt DA-Faster [1] to PointRCNN, we apply two domain discriminators to align the feature representations. One of the discriminators is applied to the global features obtained from the PointNet++ [4] backbone, while the other discriminator is applied to the instance-level features obtained from point cloud region pooling.

4. Additional Experiment Results

Effectiveness of Augmentations. Moreover, we evaluate the significance of input augmentation h and RoI augmentation ξ in Table 3. The use of augmentations consistently improve the performance of the model as random perturbations force the network to adapt to a wide range of distributions. We highlight that data augmentation is able to further boost the performance of MLC-Net, which already outperforms all baselines and state-of-the-art methods without any augmentation.

Additional Comparison with SF-UDA^{3D} [5]. As introduced in the related works, SF-UDA^{3D} proposes to address the 3D domain adaptive detection problem by leveraging the temporal coherence of target predictions. Specifically, the model trained on the source domain is used to generate predictions given target domain inputs of different scales. The best scale is selected by comparing the prediction consistency over a number of consecutive frames. Subsequently, target predictions of the best scale are used to finetune the pretrained model. SF-UDA^{3D} requires consecutive point cloud frames as the input, which is not di-

Table 4: Comparision with SF-UDA^{3D} on nuScenes $\rightarrow$ KITTI transfer on the base model of PointRCNN. * indicates the results are reprinted from the original paper. Our MLC-Net outperforms SF-UDA^{3D} without any requirement for temporal information.

Methods	Require Sequence	Easy	Moderate	Hard
Direct Transfer		49.1	39.6	35.5
SF-UDA^{3D} [5]*	X	68.8	49.8	45.0
Ours		**71.3**	**55.4**	**49.0**

rectly comparable to our proposed method which only requires single-frame input. Nevertheless, we compare the performance on the nuScenes *!* KITTI transfer task where the same evaluation metrics are used for both methods with the same PointRCNN base model. Table 4 shows that our proposed MLC-Net outputs SF-UDA^{3D} without leveraging any temporal information.

5. Qualitative Results

As shown in Figure 2 and Figure 3 ,we provide additional visualization of cross-domain detection results of different methods on multiple transfer tasks. It can be observed that directly applying a model trained on the source domain (Direct Transfer) suffers from significantly degraded performance due to geometric mismatches. While all the domain adaptation approaches demonstrate effectiveness in correcting the scale, Output Transform often under-corrects or over-corrects the predictions and cause inaccurate localization. This can be attributed to the global offset applied to all the bounding boxes, whereas individual predictions require different corrections. MLC-Net is able to mitigate the geometric mismatches effectively on various transfer tasks, which demonstrates the domain adaption capability of our proposed method.

References

[1] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3339–3348, 2018.

[2] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[4] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[5] Cristiano Saltori, Stéphane Lathuiliére, Nicu Sebe, Elisa Ricci, and Fabio Galasso. Sf-uda 3d: Source-free unsupervised domain adaptation for lidar-based 3d object detection. *arXiv preprint arXiv:2010.08243*, 2020.

[6] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019.

[7] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020.

[8] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. https://github.com/open-mmlab/OpenPCDet, 2020.

[9] Yan Wang, Xiangyu Chen, Yurong You, Li Erran Li, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. Train in germany, test in the usa: Making 3d object detectors generalize. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11713–11723, 2020.

[10] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048, 2020.

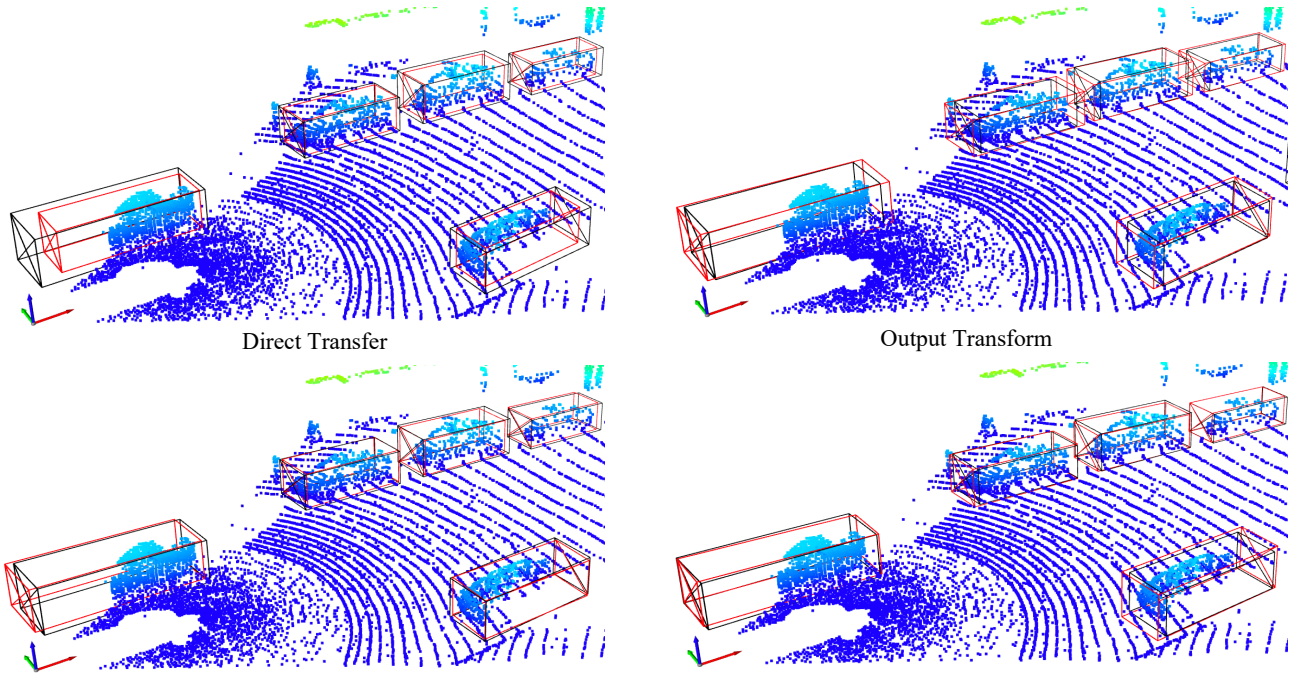

Figure 2: Qualitative results on Waymo validation dataset for KITTI to Waymo transfer.

! "#$%& '#()*+$# , -&.-& '#()*+/#0

1&(&"*&"%(2 3/#0(2"4(&"/) 5 6 7 8 3$& 9 , -#*:

Figure 3: Qualitative results on KITTI validation dataset for Waymo to KITTI transfer.