A The ONCE dataset

We publish the ONCE dataset, benchmark, develop kit, data format and annotation instructions at our website http://www.once-for-auto-driving.com. It is our priority to protect the privacy of third parties. We bear all responsibility in case of violation of rights, etc., and confirmation of the data license.

Dataset documentation. http://www.once-for-auto-driving.com/documentation.html shows the dataset documentation and intended uses.

Terms of use, privacy and License. The ONCE dataset is published under CC BY-NC-SA 4.0 license, which means everyone can use this dataset for non-commercial research purpose. The detailed Terms of use, privacy terms and license are in http://www.once-for-auto-driving.com/terms_of_use.html.

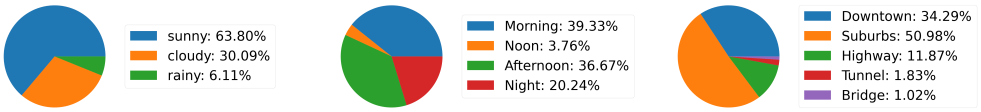
Data maintenance. http://www.once-for-auto-driving.com/download.html provides data download links for users. Data is stored in Google Drive for global users, and another copy of data is stored in BaiduYunPan for Chinese users. We will maintain the data for a long time and check the data accessibility in a regular basis.

Benchmark and code. http://www.once-for-auto-driving.com/benchmark.html provides benchmark results. The reproduction code will be released upon acceptance.

Data statistics. Figure 1 shows the proportions of different weather conditions, time periods and areas in the ONCE dataset.

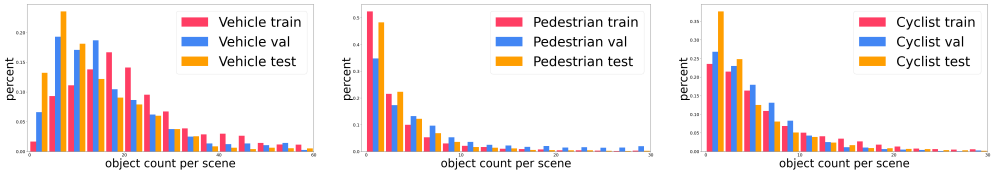
Annotation statistics. Figure 2 shows the distribution of the number of objects for annotated scenes. Our annotated set covers diverse object counts for different scenes, *e.g.* the vehicle count in a scene ranges from less than 1 to more than 50. Pedestrians and cyclists in a scene range from less than 1 to more than 30. The distributions of training, validation and testing splits are mostly similar but slightly different in some intervals, which guarantees stable evaluation results and encourages evaluated methods to have stronger generalizability across the three splits.

Limitations. The major limitation of our ONCE dataset is that currently we only annotate a small amount of scenes of the one million scenes, which may hamper the broader exploration on 3D object detection. To overcome the limitation, we plan to provide more annotations in the near future. We also plan to support more autonomous driving tasks in addition to 3D detection on the ONCE dataset.



(a) Weather conditions (b) Time periods (c) Areas

Figure 1: Proportions of different weather, time and areas in the ONCE dataset. Our dataset covers a wide range of domains with 6% scenes captured on rainy days and 20% scenes collected at night.



(a) Vehicle (b) Pedestrian (c) Cyclist

Figure 2: Distribution of annotation counts per scene. Our ONCE dataset is diverse in the number of objects in each scene. The vehicle count in each scene ranges from 0 to 60.

B Experiments

B.1 Models for 3D Object Detection

Method	Vehicle				Pedestrian				Cyclist				mAP
	overall	0-30m	30-50m	50m-inf	overall	0-30m	30-50m	50m-inf	overall	0-30m	30-50m	50m-inf	
Multi-Modality (point clouds + images)													
PointPainting [13]	66.17	80.31	59.80	42.26	44.84	52.63	36.63	22.47	62.34	73.55	57.20	40.39	57.78
Single-Modality (point clouds only)													
PointRCNN [8]	52.09	74.45	40.89	16.81	4.28	6.17	2.40	0.91	29.84	46.03	20.94	5.46	28.74
PointPillars [6]	68.57	80.86	62.07	47.04	17.63	19.74	15.15	10.23	46.81	58.33	40.32	25.86	44.34
SECOND [17]	71.19	84.04	63.02	47.25	26.44	29.33	24.05	18.05	58.04	69.96	52.43	34.61	51.89
PV-RCNN [9]	77.77	89.39	72.55	58.64	23.50	25.61	22.84	17.27	59.37	71.66	52.58	36.17	53.55
CenterPoints [19]	66.79	80.10	59.55	43.39	49.90	56.24	42.61	26.27	63.45	74.28	57.94	41.48	60.05

Table 1: Results of detection models on the validation split.

B.2 Self-Supervised Learning for 3D Object Detection

Method	Vehicle				Pedestrian				Cyclist				mAP
	overall	0-30m	30-50m	50m-inf	overall	0-30m	30-50m	50m-inf	overall	0-30m	30-50m	50m-inf	
baseline [17]	71.19	84.04	63.02	47.25	26.44	29.33	24.05	18.05	58.04	69.96	52.43	34.61	51.89
U_{small}													
BYOL [5]	68.02	81.01	60.21	44.17	19.50	22.16	16.68	12.06	50.61	62.46	44.29	28.18	46.04 (-5.85)
PointContrast [16]	71.07	83.31	64.90	49.34	22.52	23.73	21.81	16.06	56.36	68.11	50.35	34.06	49.98 (-1.91)
SwAV [2]	72.71	83.68	65.91	50.10	25.13	27.77	22.77	16.36	58.05	69.99	52.23	34.86	51.96 (+0.07)
DeepCluster [12]	73.19	84.25	66.86	50.47	24.00	26.36	21.73	16.79	58.99	70.80	53.66	36.17	52.06 (+0.17)
U_{medium}													
BYOL [5]	70.93	84.15	63.48	45.74	25.86	29.91	21.55	15.83	55.63	58.59	49.01	29.53	50.82 (-1.07)
PointContrast [16]	71.39	83.89	65.22	47.73	27.69	32.53	23.00	14.68	56.88	69.01	50.41	34.57	51.99 (+0.10)
SwAV [2]	72.51	83.39	65.46	51.08	27.08	29.94	25.19	17.13	57.85	69.87	52.38	33.78	52.48 (+0.59)
DeepCluster [12]	71.62	83.99	65.55	50.77	29.33	33.25	25.08	17.00	57.61	68.57	52.58	34.05	52.86 (+0.97)
U_{large}													
BYOL [5]	71.32	83.59	64.89	50.27	25.02	27.06	22.96	17.04	58.56	70.18	52.74	36.32	51.63 (-0.26)
PointContrast [16]	71.87	86.93	62.85	48.65	28.03	33.07	25.91	14.44	60.88	71.12	55.77	36.78	53.59 (+1.70)
SwAV [2]	72.46	83.09	66.66	51.50	29.84	34.15	26.22	17.61	57.84	68.79	52.21	35.39	53.38 (+1.49)
DeepCluster [12]	72.89	83.52	67.09	50.38	30.32	34.76	26.43	18.33	57.94	69.18	52.42	34.36	53.72 (+1.83)

Table 2: Results of self-supervised learning methods on the validation split.

B.3 Semi-Supervised Learning for 3D Object Detection

Method	Vehicle				Pedestrian				Cyclist				mAP
	overall	0-30m	30-50m	50m-inf	overall	0-30m	30-50m	50m-inf	overall	0-30m	30-50m	50m-inf	
baseline [17]	71.19	84.04	63.02	47.25	26.44	29.33	24.05	18.05	58.04	69.96	52.43	34.61	51.89
U_{small}													
Pseudo Label [7]	72.80	84.46	64.97	51.46	25.50	28.36	22.66	18.51	55.37	65.95	50.34	34.42	51.22 (-0.67)
Noisy Student [15]	73.69	84.69	67.72	53.41	28.81	33.23	23.42	16.93	54.67	65.58	50.43	32.65	52.39 (+0.50)
Mean Teacher [11]	74.46	86.65	68.44	53.59	30.54	34.24	26.31	20.12	61.02	72.51	55.24	39.11	55.34 (+3.45)
SESS [20]	73.33	84.52	66.22	52.83	27.31	31.11	23.94	19.01	59.52	71.03	53.93	36.68	53.39 (+1.50)
3DIoUMatch [14]	73.81	84.61	68.11	54.48	30.86	35.87	25.55	18.30	56.77	68.02	51.80	35.91	53.81 (+1.92)
U_{medium}													
Pseudo Label [7]	73.03	86.06	65.96	51.42	24.56	27.28	20.81	17.00	53.61	65.26	48.44	33.58	50.40 (-1.49)
Noisy Student [15]	75.53	86.52	69.78	55.05	31.56	35.80	26.24	21.21	58.93	69.61	53.73	36.94	55.34 (+3.45)
Mean Teacher [11]	76.01	86.47	70.34	55.92	35.58	40.86	30.44	19.82	63.21	74.89	56.77	40.29	58.27 (+6.38)
SESS [20]	72.11	84.06	66.44	53.61	33.44	38.58	28.10	18.67	61.82	73.20	56.60	38.73	55.79 (+3.90)
3DIoUMatch [14]	75.69	86.46	70.22	56.06	34.14	38.84	29.19	19.62	58.93	69.08	54.16	38.87	56.25 (+4.36)
U_{large}													
Pseudo Label [7]	72.41	84.06	64.54	50.05	23.62	26.80	20.13	16.66	53.25	64.69	48.52	33.47	49.76 (-2.13)
Noisy Student [15]	75.99	86.67	70.48	55.60	33.31	37.81	28.19	21.39	59.81	70.01	55.13	38.33	56.37 (+4.48)
Mean Teacher [11]	76.38	86.45	70.99	57.48	35.95	41.76	29.05	18.81	65.50	75.72	60.07	43.66	59.28 (+7.39)
SESS [20]	75.95	86.83	70.45	55.76	34.43	40.00	27.92	19.20	63.58	74.85	58.88	39.51	57.99 (+6.10)
3DIoUMatch [14]	75.81	86.11	71.82	57.84	35.70	40.68	30.34	21.15	59.69	70.69	54.92	39.08	57.07 (+5.18)

Table 3: Results of semi-supervised learning methods on the validation split.

C Implementation details

In this section, we provide implementation and training details for the 3D object detection benchmark.

C.1 Models for 3D Object Detection

Data split. We use the training split to train those 6 models. The performance is evaluated on the validation and testing split.

General configurations. Non Maximum Suppression (NMS) with the IoU threshold 0.01 is adopted for post-processing. Other configurations are kept the same with the official version of those models if not specially mentioned.

Learning scheme. All the 6 models are trained with an initial learning rate 0.003 under the cosine annealing learning scheme. We use the adam optimizer for all the models. The models are trained with the batch size 32 for 80 epochs.

Data augmentation. For all the models, we use random flip of the X and Y axis, random rotation from -45° to $+45^\circ$, random scaling from 0.95 to 1.05, and objects cut and paste on the input point cloud as augmentations. We didn't apply augmentations on segmentation maps for PointPainting.

PointRCNN. PointRCNN is a point-based 3D detector that generates proposals directly on point clouds. We sample 60000 points per frame and construct the segmentation backbone with 32000-4000-500-256 points. We use the mean size of each category for proposal generation.

PointPillars. PointPillars is a pioneering work that introduces pillar-based representation into 3D object detection. We set the pillar size as $0.2m \times 0.2m$ and also use the mean size as the anchor size.

SECOND. SECOND is a voxel-based detector that transforms point clouds into voxels for feature extraction. We set the voxel size as $0.1m \times 0.1m \times 0.2m$ and use the same anchors as PointPillars.

PV-RCNN. PV-RCNN is a point-voxel based detector that applies SECOND for proposal generation and then utilizes keypoints for RoI feature extraction. We sample 4096 keypoints per scene.

CenterPoints. CenterPoints introduces center-based target assignments to replace the anchor-based assignments. In addition to the center head, we use the same backbone as SECOND.

PointPainting. PointPainting uses CenterPoints as the 3D detector and HRNet trained on CityScapes to generate semantic segmentation results.

C.2 Self-Supervised Learning for 3D Object Detection

Data split. We conduct self-supervised pretraing on the unlabeled sets $U_\star$ and then use the training split to finetune models. The performance is evaluated on the validation and testing split.

General configurations. We use the voxel-based SECOND detector as the baseline model for all the methods. During the pretraining stage, we pretrain the backbone of SECOND detector on unlabeled subset. We pretrain those methods for 20 epochs on the 100k unlabeled subset U_{small}, 5 epochs on the 500k subset U_{medium} and 3 epochs on the 1 million subset U_{large}.

Learning scheme. For all the methods, the pretaining and finetuning learning rate is initialized as 0.003. We use the adam optimizer and the cosine annealing learning scheme for all the methods.

Multi-view augmentation setup. We generate multi-view of the original scenes by random flip, scaling with a scale factor sampled from [0.95, 1.05] and rotation around vertical yaw axis between [-10, 10] degrees. We also do downsampling by a factor sampled from [0.9, 1].

PointContrast. PointContrast defines a contrastive loss over the point-level features given a pair of overlapping partial scans. The objective is to minimize the distance between matched points (positive pairs) and maximize the distance between unmatched ones (negative pairs). In our setting, we sample a random geometric transformation to transform an original point cloud scene into 2 augmented views. After passing the scenes through SECOND backbone to obtain voxel-wise features, we randomly select 1024 voxels within each scene. The voxel-wise features will be passed through a two-layer MLP (with dimension 128, 64) to project into latent space, with batchnorm and ReLU. The latent space feature will be concatenated with initial feature and passed through a one-layer MLP with

dimension 64. The final features will be used for contrastive pretraining. We pretrain the model using Adam optimizer with the initial learning rate 0.001 and the batch size as 4.

DeepCluster. DeepCluster uses k-means clustering to give each instance a cluster id as the pseudo label and use the label to train the network. Since clustering method is designed to learn semantic representation, we randomly crop patches in 3D scenes as pseudo instances and pass the patches through the backbone to obtain patch-wise features. We project the features into latent space for clustering and pretraining. We choose the total cluster number as 100. Patch-wise feature will be passed through a two-layer MLP (with dimension 192, 128), with a batchnorm and ReLU layer to project the features to the latent space. This two-layer MLP will not be used in the finetune stage. We pretrain the backbone with Adam optimizer. The initial learning rate is 0.0048 with a cosine decay. The batch size is 256.

SwAV. SwAV improves DeepCluster by introducing prototypes, online clustering and swapped predictions. We use the same clustering and training settings as DeepCluster. For other configurations we follow the settings in the original paper.

BYOL. BYOL introduces two networks, referred to online network and target network, that can interact and learn from each other. Given a 3D scene, we train the online network to predict the target network's representation of an augmented view of the same scene. In particular, after passing the 3D scene through the backbone, we project the representation through a two-layer MLP (with dimension 4096, 512). After that, the predictor in the online network will project the embedding into a latent space as the final representation of the online network. The predictor is also a two-layer MLP (with dimension 4096, 512). We update the target network by a slow-moving averaging of the online network with parameter 0.999. To avoid the model collapsing to trivial solutions, we further introduce a contrastive regularization term. Specifically, we follow the design in PointContrast and ramdomly select some voxel-wise features. A contrastive loss is designed on different views of the same voxel between its online representation and target representation. We pretrain the model using Adam optimizer with the initial learning rate 0.001. The batch size is 4.

C.3 Semi-Supervised Learning for 3D Object Detection

Data split. We first utilize the training split to obtain pretrained teacher and student models, and then we apply semi-supervised learning on the unlabeled set $U_\star$. The performance is evaluated on the validation and testing split.

General configurations. We use the SECOND detector as the baseline model for all the methods, which guarantees a fair comparison among those methods. The pretraining process and configurations follow those in C.1.

Learning scheme. The initial learning rate is 0.003 for both the pretraining and semi-supervised learning process. We use the adam optimizer and the cosine annealing learning scheme for all the methods. For pretraining, the batch size is 32. For semi-supervised learning, the batch size of labeled data is 8 and the batch size of unlabeled data is 32. The pretraining process lasts 80 epochs. The semi-supervised learning process lasts 25 epochs for U_{small} and 5 epochs for U_{medium} and U_{large}.

Data augmentation. During the pretraining process, we use random flip of the X and Y axis, random rotation from -45° to $+45^\circ$, random scaling from 0.95 to 1.05, and objects cut and paste on the input point cloud as augmentations. During the semi-supervised learning process, we use the same random flip, random rotation and random scaling for the student model. We didn't apply augmentations on the teacher model.

Pseudo Label. We use the pretrained model to generate pseudo ground truth boxes for each unlabeled scene. The model is then trained with pseudo labels in the unlabeled scenes, as well as real labels in the training split. It is worth noting that we didn't apply any augmentation in the semi-supervised learning process, mainly to explore whether augmentations are necessary with a large amount of data.

Mean Teacher. Mean Teacher uses the teacher and student model for semi-supervised learning. We first load the pretrained weights for both two models, and then the teacher model produces pseudo ground truths to train the student model for the unlabeled subset. A consistency loss is introduced to regularize two models. Specifically, we first match the predicted boxes of student model with pseudo boxes of teacher model by the nearest-neighbor criterion. Then the Kullback–Leibler divergence of

class predictions of the matched pairs of boxes is applied as the consistency loss. The teacher model is updated by exponential moving average (EMA) of the student model.

Noisy Student. Noisy Student is a self-training approach in which the student is trained with noise, *i.e.*, strong augmentations, using pseudo labels provided by the teacher. After the first round of semi-supervised training, we make the student a new teacher for the second around.

SESS. Self-Ensembling Semi-Supervised (SESS) 3D object detection extends Mean Teacher by introducing another two consistency constraints: size consistency and center consistency, along with class consistency to the matched pairs of boxes. The teacher is also updated by EMA.

3DIoUMatch. 3DIoUMatch introduces an extra IoU prediction head on the detection model. The predicted IoUs are then used for filtering low-quality pseudo boxes. We reject IoU-guided lower-half suppression and the EMA update scheme, since those components are detrimental to the detection performance in our experiments.

C.4 Unsupervised Domain Adaptation for 3D Object Detection

Data split. We conduct unsupervised domain adaptation on the training split. The performance is evaluated on the validation split.

Learning scheme. We completely follow the same learning rate, optimizer and learning scheme used in [18].

Data augmentation. We completely follow the augmentations used in [18].

SN. Statistical Normalization (SN) is based on the observation that domain gap mainly comes from the differences of object size between different datasets, so this method normalizes the objects' size of the source dataset according to the object statistics on the target domain.

ST3D. ST3D contains two stages: the model is first trained on the source dataset with an augmentation method named random object scaling. Then the model is trained on the target dataset with the aid of pseudo labels and a memory bank.

D Visualization

Annotation example. We present an example of annotations in Figure 3.

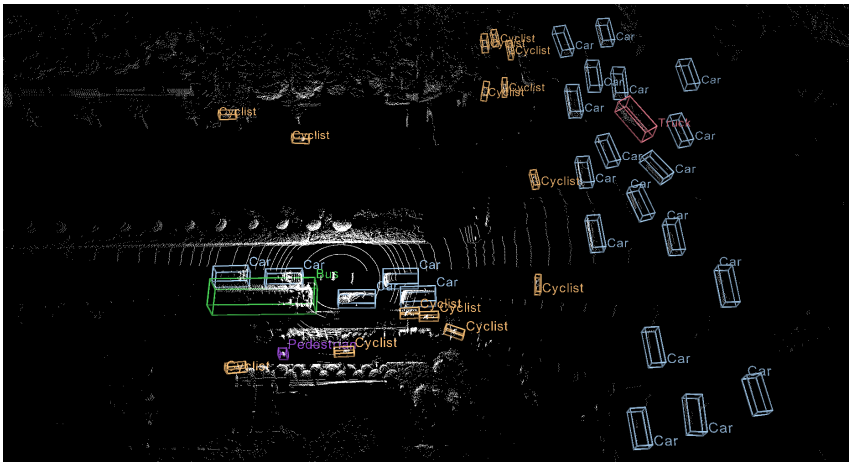


Figure 3: Example of 3D annotations.

3D annotations for RGB images. We present an example of 3D annotations on an RGB image in Figure 4.

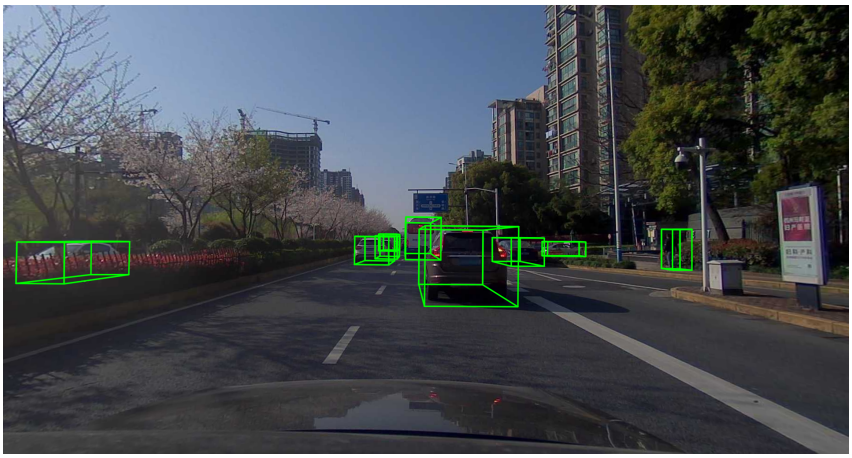


Figure 4: Example of 3D annotations on an RGB image.

Multi-modality alignments. We present an illustration of the alignments between point clouds and images in Figure 5.

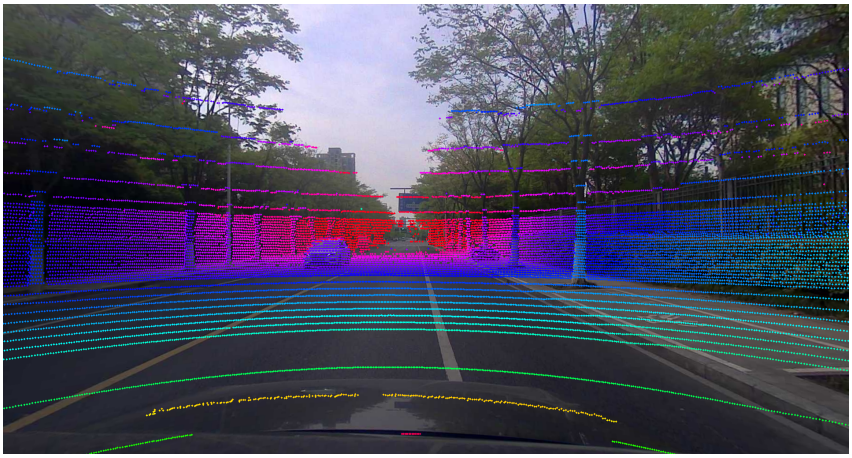


Figure 5: Alignment of point cloud and image.

Annotation system. We present an interface of the annotation system in Figure 6.

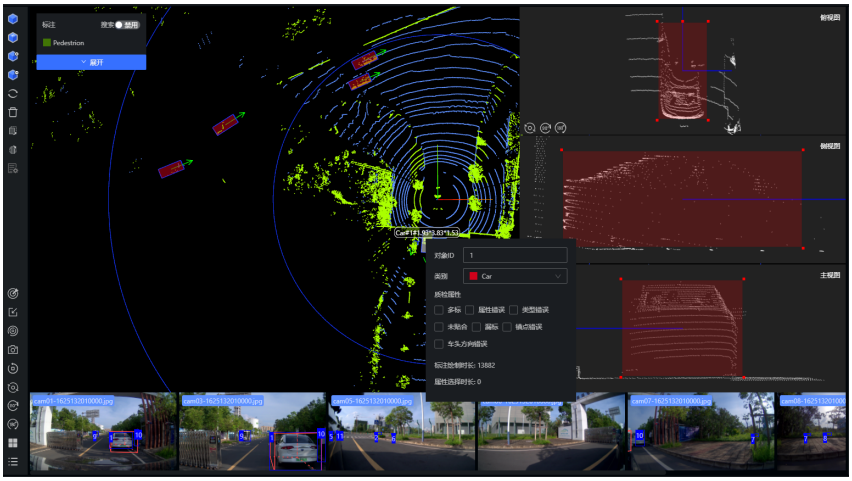


Figure 6: Annotation system.

E Evaluation Metric

Evaluation metric is critical for fair comparisons of different approaches on 3D object detection. Current 3D IoU-based evaluation metric AP_{3D} [4] faces the problem that objects with opposite orientations can both be matched to the ground truth with the IOU criterion. To resolve this problem, we extend [4] and take the object orientations into special consideration. In particular, we first re-rank the predictions according to their scores, and set those predicted boxes that have low 3D IoUs with all ground truths of the same category as false positives. The IoU thresholds are $0.7, 0.7, 0.7, 0.3, 0.5$ for car, bus, truck, pedestrian, cyclist respectively. Then we add additional filtering step in which we also set those predictions as false positives if their orientations θ cannot fall into the $\pm 90^{\circ}$ range of the matched ground truth orientations θ'. This step sets a more stringent criterion specially for orientations. The remaining matched predictions are treated as true positives. Finally, we determine 50 score thresholds with the recall rates r from 0.02 to 1.00 at the step 0.02 and we calculate the corresponding 50 precision rates to draw the precision-recall curve $p(r)$. The calculation of our orientation-aware AP_{3D}^{Ori} can be formulated as:

$$AP_{3D}^{Ori} = 100 \int_0^1 max\{p(r'|r' \geq r)\}dr. \tag{1}$$

We merge the car, bus and truck class into a super-class called vehicle following [10], so we officially report the AP_{3D}^{Ori} of vehicle, pedestrian and cyclist respectively in the following experiments. We still provide the evaluation interface of 5 classes for users. Mean AP (mAP) is thus obtained by averaging the scores of 3 categories. To further inspect the detection performance of different distances, we also provide AP_{3D}^{Ori} of 3 distance ranges: within 30m, 30-50m, and farther than 50m. This is obtained by only considering ground truths and predictions within that distance range.

Discussion on different evaluation metrics. Current evaluation metrics of 3D detection typically extend the Average Precision (AP) metric [3] of 2D detection to the 3D scenarios by changing the matching criterion between the ground truth boxes and predictions. The nuScenes dataset [1] uses the center distance between boxes on the ground plane as the matching criterion for AP calculation, in ignorance of the size and orientation of the objects. Although the nuScenes detection score (NDS) is proposed to take all factors into consideration, AP still accounts for 50% of the total NDS score, which shows strong preference to the accurate localization of object centers but less attention to the objects' size and orientation. The Waymo Open dataset [10] applies the Hungarian algorithm to match the ground truths and predictions, which may lead to a higher estimation of AP since objects with no overlaps can also be matched. The KITTI dataset [4] uses 3D Intersection over Union (IoU) above certain threshold as the matching criterion, but the predicted boxes with the opposite orientations of the ground truths can also be matched, which can be dangerous in practical. In this paper, we extend the 3D IoU-based evaluation metric AP of [4] and take the object orientations into special consideration. Our orientation-aware AP_{3D}^{Ori} metric is more stringent than [4]. Compared with the weighted-scoring method in [10], our method avoids repeated calculations of the orientation factor, since it has already participated in the computation of 3D rotated IoU. Compared with the distance-based matching scheme [1], our method puts equal weights on the object size, center and orientation.

References

[1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[2] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

[3] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[4] 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.

[5] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

[6] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019.

[7] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, 2013.

[8] 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.

[9] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020.

[10] 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.

[11] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *arXiv preprint arXiv:1703.01780*, 2017.

[12] Kai Tian, Shuigeng Zhou, and Jihong Guan. Deepcluster: A general clustering framework based on deep learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 809–825. Springer, 2017.

[13] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4604–4612, 2020.

[14] He Wang, Yezhen Cong, Or Litany, Yue Gao, and Leonidas J Guibas. 3dioumatch: Leveraging iou prediction for semi-supervised 3d object detection. *arXiv preprint arXiv:2012.04355*, 2020.

[15] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020.

[16] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020.

[17] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10): 3337, 2018.

[18] Jihan Yang, Shaoshuai Shi, Zhe Wang, Hongsheng Li, and Xiaojuan Qi. St3d: Self-training for unsupervised domain adaptation on 3d object detection. *arXiv preprint arXiv:2103.05346*, 2021.

[19] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. *arXiv preprint arXiv:2006.11275*, 2020.

[20] Na Zhao, Tat-Seng Chua, and Gim Hee Lee. Sess: Self-ensembling semi-supervised 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11079–11087, 2020.