SE-SSD: Self-Ensembling Single-Stage Object Detector From Point Cloud

Wu Zheng Weiliang Tang Li Jiang Chi-Wing Fu
The Chinese University of Hong Kong
{wuzheng, lijiang, cwfu}@cse.cuhk.edu.hk tangwl123@foxmail.com

Abstract

We present Self-Ensembling Single-Stage object Detector (SE-SSD) for accurate and efficient 3D object detection in outdoor point clouds. Our key focus is on exploiting both soft and hard targets with our formulated constraints to jointly optimize the model, without introducing extra computation in the inference. Specifically, SE-SSD contains a pair of teacher and student SSDs, in which we design an effective IoU-based matching strategy to filter soft targets from the teacher and formulate a consistency loss to align student predictions with them. Also, to maximize the distilled knowledge for ensembling the teacher, we design a new augmentation scheme to produce shape-aware augmented samples to train the student, aiming to encourage it to infer complete object shapes. Lastly, to better exploit hard targets, we design an ODIoU loss to supervise the student with constraints on the predicted box centers and orientations. Our SE-SSD attains top performance compared with all prior published works. Also, it attains top precisions for car detection in the KITTI benchmark (ranked 1st and 2nd on the BEV and 3D leaderboards[1]*, respectively) with an ultra-high inference speed. The code is available at* https://github.com/Vegeta2020/SE-SSD.

1. Introduction

To support autonomous driving, 3D point clouds from LiDAR sensors are often adopted to detect objects near the vehicle. This is a robust approach, since point clouds are readily available regardless of the weather (fog vs. sunny) and time of the day (day vs. night). Hence, various point-cloud-based 3D detectors have been proposed recently.

To boost the detection precision, an important factor is the quality of the extracted features. This applies to both single-stage and two-stage detectors. For example, the series of works [24, 4, 25, 23] focus on improving the region-proposal-aligned features for a better refinement with a second-stage network. Also, many methods [3, 10, 29, 12, 33, 19] try to extract more discriminative multi-modality features by fusing RGB images and 3D point clouds. For single-stage detectors, Point-GNN [26] adapts a graph neural network to obtain a more compact representation of point cloud, while TANet [17] designs a delicate triple attention module to consider the feature-wise relation. Though these approaches give significant insights, the delicate designs are often complex and could slow down the inference, especially for the two-stage detectors.

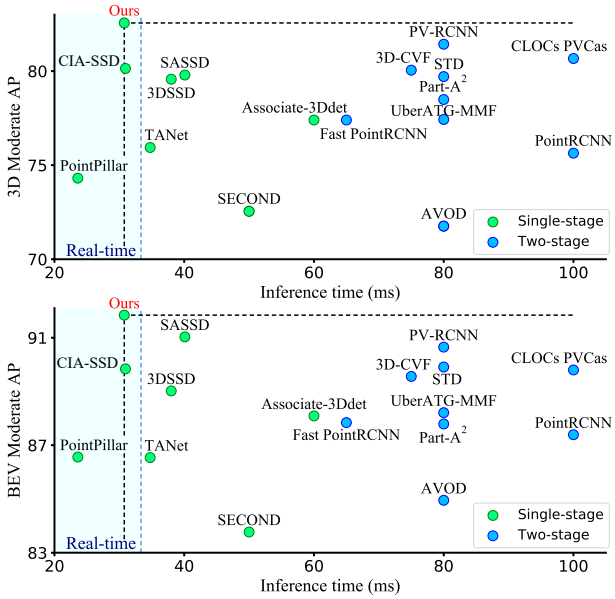

Figure 1. Our SE-SSD attains top precisions on both 3D and BEV car detection in KITTI benchmark [6] with real-time speed (30.56 ms), clearly outperforming all state-of-the-art detectors. Please refer to Table 1 for a detailed comparison with more methods.

To meet the practical need, especially in autonomous driving, 3D object detection demands high efficiency on top of high precision. Hence, another stream of works, *e.g.*, SA-SSD [8] and Associate-3Ddet [5], aim to exploit auxiliary tasks or further constraints to improve the feature representation, without introducing additional computational overhead during the inference. Following this stream of works, we formulate the Self-Ensembling Single-Stage object Detector (SE-SSD) to address the challenging 3D detection task based only on LiDAR point clouds.

[1] On the date of CVPR deadline, *i.e.*, Nov 16, 2020

To boost the detection precision, while striving for high efficiency, we design the SE-SSD framework with a pair of teacher SSD and student SSD, as inspired by [27]. The teacher SSD is ensembled from the student. It produces relatively more precise bounding boxes and confidence, which serve as soft targets to supervise the student. Compared with manually-annotated hard targets (labels), soft targets from the teacher often have higher entropy, thus offering more information [9] for the student to learn from. Hence, we exploit both soft and hard targets with our formulated constraints to jointly optimize the model, while incurring no extra inference time. To encourage the bounding boxes and confidence predicted by the student to better align with the soft targets, we design an effective IoU-based matching strategy to filter soft targets and pair them with student predictions, and further formulate a consistency loss to reduce the misalignment between them.

On the other hand, to enable the student SSD to effectively explore a larger data space, we design a new augmentation scheme on top of conventional augmentation strategies to produce augmented object samples in a shape-aware manner. By this scheme, we can encourage the model to infer the complete object shape from incomplete information. It is also a plug-and-play and general module for 3D detectors. Furthermore, hard targets are still essential in the supervised training, as they are the final targets for the model convergence. To better exploit them, we formulate a new orientation-aware distance-IoU (ODIoU) loss to supervise the student with constraints on both the center and orientation of the predicted bounding boxes. Overall, our SE-SSD is trained in a fully-supervised manner to best boost the detection performance, in which all the designed modules are needed only in the training, so there are no extra computation during the inference.

In summary, our contributions include (i) the Self-Ensembling Single-Stage object Detector (SE-SSD) framework, optimized by our formulated consistency constraint to better align predictions with the soft targets; (ii) a new augmentation scheme to produce shape-aware augmented ground-truth objects; and (iii) an Orientation-aware Distance-IoU (ODIoU) loss to supervise the detector using hard targets. Our SE-SSD attains state-of-the-art performance on both 3D and BEV car detection in the KITTI benchmark [6] and demonstrates ultra-high inference speed (32 FPS) on commodity CPU-GPU, clearly outperforming all prior published works, as presented in Figure 1.

2. Related Work

In general, 3D detectors are categorized into two types: (i) *single-stage detectors* regress bounding box and confidence directly from features learned from the input, and (ii) *two-stage detectors* use a second stage to refine the first-stage predictions with region-proposal-aligned features. So, two-stage detectors often attain higher precisions benefited from the extra stage, whereas single-stage detectors usually run faster due to simpler network structures. Recent trend (see Figure 1 and Table 1) reveals that the precisions of single-stage detectors [8, 31] gradually approach those of the two-stage detectors [23, 25, 32]. This motivates us to focus this work on developing a single-stage detector and *aim for both high precision and high speed*.

Two-stage Object Detectors Among these two-stage detectors, PointRCNN [24] uses PointNet [21] to fuse semantic features and raw points from region proposals for a second-stage refinement. Part-A^2 [25] exploits a voxel-based network to extract region proposal features to avoid ambiguity and further improve the feature representation. Similarly, STD [32] converts region-proposal semantic features to a compact representation with voxelization and reduce the number of anchors to improve the performance. PV-RCNN [23] utilizes both point-based and voxel-based networks to extract features from the voxels and raw points inside the region proposal. 3D-CVF [33] obtains semantics from multi-view images and fuses them with point features in both stages, whereas CLOCs PVCas [19] fuses semantic features from images and points to refine the predicted confidence.

Single-stage Object Detectors VoxelNet [38] proposes the voxel feature encoding layer to extract features from point clouds. PointPillar [11] divides a point cloud into pillars for efficient feature learning. SECOND [30] modifies the sparse convolution [7, 15] to efficiently extract features from sparse voxels. TANet [17] proposes the triple attention module to consider feature-wise relation in the feature extraction. Point-GNN [26] exploits a graph neural network to learn point features. 3DSSD [31] combines feature- and point-based sampling to improve the classification. Associate-3Ddet [5] extracts feature from complete point clouds to supervise the features learned from incomplete point clouds, encouraging the model to infer from incomplete points. SA-SSD [8] adopts an auxiliary network in parallel with the backbone to regress box centers and semantic classes to enrich the features. CIA-SSD [8] adopts a lightweight BEV network for extraction of robust spatial-semantic features, combined with an IoU-aware confidence rectification and DI-NMS for better post processing. Inspired by [27], SESS [34] addresses the detection in indoor scenes with a semi-supervised strategy to reduce the dependence on manual annotations.

Different from prior works, we aim to exploit both soft and hard targets to refine features in a fully-supervised manner through our novel constraints and augmentation scheme. Also, compared with *all prior* single- and two-stage detectors, our SE-SSD attains the *highest average precisions for both 3D and BEV car detection* in the KITTI benchmark [6] and exhibits *very high efficiency*.

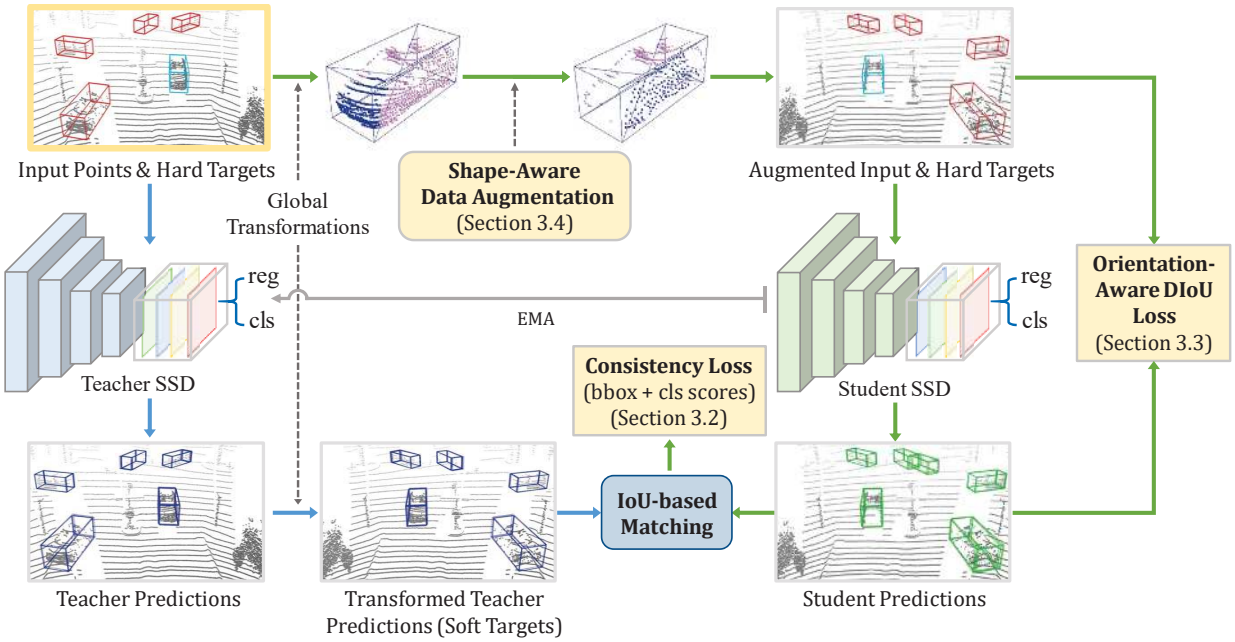

Figure 2. The framework of our Self-Ensembling Single-Stage object Detector (SE-SSD) with a teacher SSD and a student SSD. To start, we feed the input point cloud to the teacher to produce relatively precise bounding boxes and confidence, and take these predictions (after global transformations) as soft targets to supervise the student with our *consistency loss* (Section 3.2). On the top branch, we apply the same global transformations to the input, then perform our *shape-aware data augmentation* (Section 3.4) to generate augmented samples as inputs to the student. Further, we formulate the *Orientation-aware Distance-IoU loss* (Section 3.3) to supervise the student with hard targets, and update the teacher parameters based on the student parameters with the exponential moving average (EMA) strategy. In this way, the framework can boost the precisions of the detector significantly without incurring extra computation during the inference.

3. Self-Ensembling Single Stage Detector

3.1. Overall Framework

Figure 2 shows the framework of our self-ensembling single-stage object detector (SE-SSD), which has a teacher SSD (left) and a student SSD (right). Different from prior works on outdoor 3D object detection, we simultaneously employ and train two SSDs (of same architecture), such that the student can explore a larger data space via the augmented samples and be better optimized with the associated soft targets predicted by the teacher. To train the whole SE-SSD, we first initialize both the teacher and student with a pre-trained SSD model. Then, started from an input point cloud, our framework has two processing paths:

- In the first path (blue arrows in Figure 2), the teacher produces relatively precise predictions from the raw input point cloud. Then, we apply a set of global transformations on the prediction results and take them as soft targets to supervise the student SSD.
- In the second path (green arrows in Figure 2), we perturb the same input by the same global transformations as in the first path plus our shape-aware data augmentation (Section 3.4). Then, we feed the augmented input to the student, and train it with (i) our consistency loss (Section 3.2) to align the student predictions with the soft targets; and (ii) when we augment the input, we bring along its hard targets (Figure 2 (top right)) to supervise the student with our orientation-aware distance-IoU loss (Section 3.3).

In the training, we iteratively update the two SSD models: optimize the student with the above two losses and update the teacher using only the student parameters by a standard exponential moving average (EMA). Thus, the teacher can obtain distilled knowledge from student and produce soft targets to supervise student. So, we call the final trained student a *self-ensembling single-stage object detector*.

Architecture of Teacher & Student SSD The model has the same structure as [35] to efficiently encode point clouds, but we remove the Confidence Function and DI-NMS. It has a sparse convolution network (SPConvNet), a BEV convolution network (BEVConvNet), and a multi-task head (MTHead). BEV means bird's eye view. After point cloud voxelization, we find mean 3D coordinates and point density per voxel as the initial feature, then extract features using SPConvNet, which has four blocks ($\{2, 2, 3, 3\}$ submanifold sparse convolution [7] layers) with a sparse convolution [15] layer at the end. Next, we concatenate the sparse 3D feature along z into 2D dense ones for feature extraction with the BEVConvNet. Lastly, we use MTHead to regress bounding boxes and perform classification.

3.2. Consistency Loss

In 3D object detection, the patterns of point clouds in pre-defined anchors may vary significantly due to distances and different forms of object occlusion. Hence, samples of the *same* hard target may have very different point patterns and features. In contrast, soft targets can be more informative per training sample, and helps to reveal the difference between data samples of the same class [9]. This motivates us to treat the relatively more precise teacher predictions as soft targets and employ them to jointly optimize the student with hard targets. Accordingly, we formulate a consistency loss to optimize the student network with soft targets.

We first design an effective IoU-based matching strategy before calculating the consistency loss, aiming to pair the non-axis-aligned teacher and student boxes predicted from the very sparse outdoor point clouds. To obtain high-quality soft targets from the teacher, we first filter out those predicted bounding boxes (for both teacher and student) with confidence less than threshold τ_c, which helps reduce the calculation of the consistency loss. Next, we calculate the IoU between every pair of remaining student and teacher bounding boxes, and filter out the pairs with IoUs less than threshold τ_I, thus avoiding to mislead the student with unrelated soft targets; We denote N and N' as the initial and final number of box pairs, respectively. Thus, we keep only the highly-overlapping student-teacher pairs. Lastly, for each student box, we pair it with the teacher bounding box that has the largest IoU with it, aiming to increase the confidence of the soft targets. Compared with hard targets, the filtered soft targets are often closer to the student predictions, as they are predicted based on similar features. So, soft targets can better guide the student to fine-tune the predictions and reduce the gradient variance for better training.

Different from the IoU loss, Smooth-L_1 loss [16] can evenly treat all dimensions in the predictions, without biasing toward any specific one, so the features corresponding to different dimensions can also be evenly optimized. Hence, we adopt it to formulate our consistency loss for bounding boxes ($\mathcal{L}^c_{box}$) to minimize the misalignment errors between each pair of teacher and student bounding boxes:

$$\mathcal{L}^c_{box} = \frac{1}{N'}\sum_{i=1}^{N}\mathbb{1}(IoU_i > \tau_I)\sum_{e}\frac{1}{7}\mathcal{L}^c_{\delta_e}$$
$$\text{and}\ \ \delta_e = \begin{cases} |e_s - e_t| & \text{if } e \in \{x, y, z, w, l, h\} \\ |sin(e_s - e_t)| & \text{if } e \in \{r\} \end{cases} \tag{1}$$

where $\{x, y, z\}$, $\{w, l, h\}$, and r denote the center position, sizes, and orientation of a bounding box, respectively, predicted by teacher (subscript t) or student (subscript s), $\mathcal{L}^c_{\delta_e}$ denotes the Smooth-L_1 loss of δ_e, and IoU_i denotes the largest IoU of the i-th student bounding box with all teacher bounding boxes. Next, we formulate the consistency loss for classification score ($\mathcal{L}^c_{cls}$) to minimize the difference in predicted confidence of student and teacher:

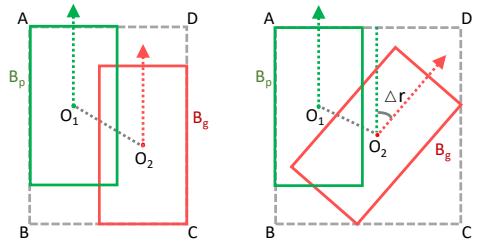

Figure 3. Illustrating axis-aligned bounding boxes (left) and non-axis-aligned bounding boxes (right) in Bird's Eye View (BEV), where the red and green boxes represent the ground truth and predicted box, respectively. To handle non-axis-aligned bounding boxes (right), our ODIoU loss impose constraints on both the box center distance $|O_1O_2|$ and the orientation difference $\triangle r$ in BEV.

$$\mathcal{L}^c_{cls} = \frac{1}{N'}\sum_{i=1}^{N}\mathbb{1}(IoU_i > \tau_I)\mathcal{L}^c_{\delta_c}$$
$$\text{and}\ \ \delta_c = |\sigma(c_s) - \sigma(c_t)| \tag{2}$$

where $\mathcal{L}^c_{\delta_c}$ denotes the Smooth-L_1 loss of δ_c, and $\sigma(c_s)$ and $\sigma(c_t)$ denote the sigmoid classification scores of student and teacher, respectively. Here, we adopt the sigmoid function to normalize the two predicted confidences, such that the deviation between the normalized values can be kept inside a small range. Combining Eqs (1) and (2), we can obtain the overall consistency loss as

$$\mathcal{L}_{cons} = \mathcal{L}^c_{cls} + \mathcal{L}^c_{box} \tag{3}$$

where we empirically set the same weight for both terms.

3.3. Orientation-Aware Distance-IoU Loss

In supervised training with hard targets, Smooth-L_1 loss [16] is often adopted to constrain the bounding box regression. However, due to long distances and occlusion in outdoor scenes, it is hard to acquire sufficient information from the sparse points to precisely predict all dimensions of the bounding boxes. To better exploit hard targets for regressing bounding boxes, we design the Orientation-aware Distance-IoU loss (ODIoU) to focus more attention on the alignment of box centers and orientations between the predicted and ground-truth bounding boxes; see Figure 3.

Inspired by [36], we impose a constraint on the distance between the 3D centers of the predicted and ground-truth bounding boxes to minimize the center misalignment. More importantly, we design a novel orientation constraint on the predicted BEV angle, aiming to further minimize the orientation difference between the predicted and ground-truth boxes. In 3D object detection, such a constraint is significant for the precise alignment between the non-axis-aligned boxes in the bird's eye view (BEV). Also, we empirically

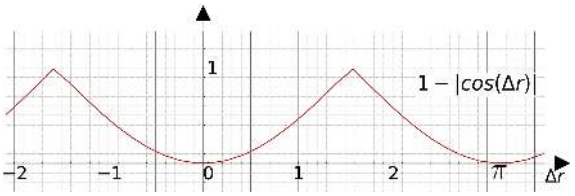

Figure 4. We formulate the orientation constraint $(1 - |cos(\triangle r)|)$ in the ODIoU loss to precisely minimize the orientation difference between bounding boxes; note the gradient of the term.

find that this constraint is an important means to further boost the detection precision. Compared with Smooth-L_1 loss, our ODIoU loss enhances the alignment of box centers and orientations, which are easy to infer from the points distributed on the object surface, thus leading to a better performance. Overall, our ODIoU loss is formulated as

$$\mathcal{L}^s_{box} = 1 - IoU(B_p, B_g) + \frac{c^2}{d^2} + \gamma(1 - |cos(\triangle r)|) \quad (4)$$

where B_p and B_g denote the predicted and ground-truth bounding boxes, respectively, c denotes the distance between the 3D centers of the two bounding boxes (see $|O_1O_2|$ in Figure 3), d denotes the diagonal length $|AC|$ of the minimum cuboid that encloses both bounding boxes; $\triangle r$ denotes the BEV orientation difference between B_p and B_g; and γ is a hyper-parameter weight.

In our ODIoU loss formula, $(1 - |cos(\triangle r)|)$ is an important term we designed specifically to encourage the predicted bounding box to rotate to the nearest direction that is parallel to the ground-truth orientation. When $\triangle r$ equals $\frac{\pi}{2}$ or $-\frac{\pi}{2}$, *i.e.*, the orientations of the two boxes are perpendicular to each other, so the term attains its maxima. When $\triangle r$ equals 0, π, or $-\pi$, the term attains its minima, which is zero. As shown in Figure 4, we can further look at the gradient of $(1 - |cos(\triangle r)|)$. When the training process minimizes the term, its gradient will help to bring $\triangle r$ to 0, π, or $-\pi$, which is the nearest location to minimize the loss. It is because the gradient magnitude is positively correlated to the angle difference, thus promoting fast convergence and smooth fine-tuning in different training stages.

Besides, we use the Focal loss [14] and cross-entropy loss for the bounding box classification ($\mathcal{L}^s_{cls}$) and direction classification ($\mathcal{L}^s_{dir}$), respectively. Hence, the overall loss to train the student SSD is

$$\mathcal{L}_{\text{student}} = \mathcal{L}^s_{\text{cls}} + \omega_1\mathcal{L}^s_{\text{box}} + \omega_2\mathcal{L}^s_{\text{dir}} + \mu_t(\mathcal{L}^c_{\text{cls}} + \mathcal{L}^c_{\text{box}}) \quad (5)$$

where $\mathcal{L}^s_{\text{box}}$ is the ODIoU loss for regressing the boxes, and the loss weights ω_1, ω_2, and μ_t are hyperparameters. Futher, our SSD can be pre-trained with the same settings as SE-SSD but without the consistency loss and teacher SSD.

3.4. Shape-Aware Data Augmentation

Data augmentation is important to improve a model's generalizability. To enable the student SSD to explore a larger data space, we design a new shape-aware data augmentation scheme to boost the performance of our detector. Our insight comes from the observation that the point cloud patterns of ground-truth objects could vary significantly due to occlusions, changes in distance, and diversity of object shapes in practice. So, we design the shape-aware data augmentation scheme to mimic how point clouds are affected by these factors when augmenting the data samples.

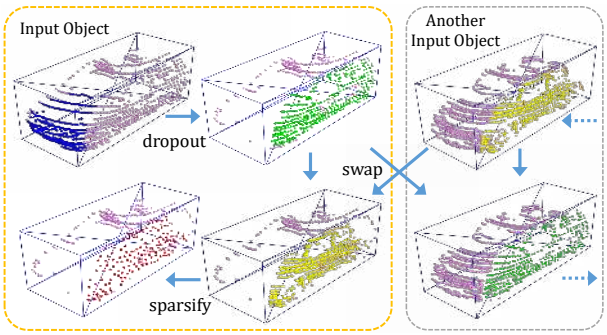

Figure 5. Illustrating how the shape-aware data augmentation scheme works on a ground-truth object. We divide the object's point cloud into six pyramidal subsets (one for each face of the object's bounding box), and then independently augment each subset by random dropout, swap, and/or sparsify operations.

By design, our shape-aware data augmentation scheme is a plug-and-play module. To start, for each object in a point cloud, we find its ground-truth bounding box centroid and connect the centroid with the box faces to form pyramidal volumes that divide the object points into six subsets. Observing that LiDAR points are distributed mainly on object surfaces, the division is like an object disassembly, and our augmentation scheme efficiently augments each object's point cloud by manipulating these divided point subsets like disassembled parts.

In details, our scheme performs the following three operations with randomized probabilities p_1, p_2, and p_3, respectively: (i) *random dropout* removes all points (blue) in a randomly-chosen pyramid (Figure 5 (top-left)), mimicking a partial object occlusion to help the network to infer a complete shape from the remained points. (ii) *random swap* randomly selects another input object in the current scene and swap a point subset (green) with the point subset (yellow) in the same pyramid of the other input object (Figure 5 (middle)), thus increasing the diversity of object samples by exploiting the surface similarity across objects. (iii) *random sparsifying* subsamples points in a randomly-chosen pyramid using farthest point sampling [22], mimicking the sparsity variation of points due to changes in distance from LiDAR camera; see the sparsified points (red) in Figure 5.

Furthermore, before the shape-aware augmentation, we perform a set of global transformations on the input point cloud, including a random translation, flipping, and scaling; see "global transformations" in Figure 2.

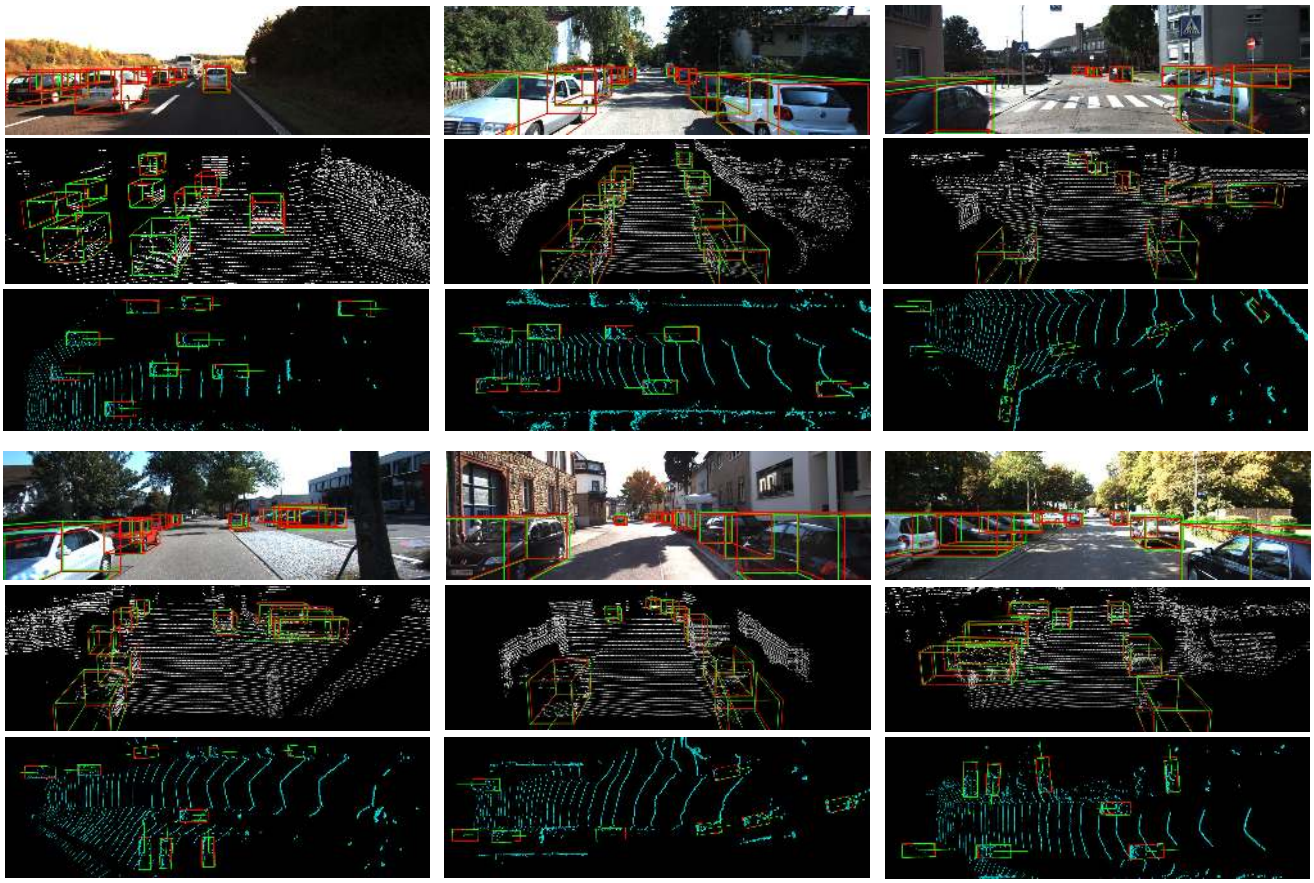

Figure 6. Snapshots of our 3D and BEV detection results on the KITTI validation set. We render the predicted and ground-truth bounding boxes in green and red, respectively, and project them back onto the color images for visualization.

4. Experiments

We evaluate our SE-SSD on the KITTI 3D and BEV object detection benchmark [6]. This widely-used dataset contains 7,481 training samples and 7,518 test samples. Following the common protocol, we further divide the training samples into a training set (3,712 samples) and a validation set (3,769 samples). Our experiments are mainly conducted on the most commonly-used car category and evaluated by the average precision with an IoU threshold 0.7. Also, the benchmark has three difficulty levels in the evaluation: easy, moderate, and hard, based on the object size, occlusion, and truncation levels, in which the moderate average precision is the official ranking metric for both 3D and BEV detection on the KITTI website. We will *release our code on GitHub* upon the publication of this work.

Figure 6 shows 3D bounding boxes (2nd & 5th rows) and BEV bounding boxes (3rd & 6th rows) predicted by our SE-SSD model for six different inputs, demonstrating its high-quality prediction results. Also, for a better visualization of the results, we project and overlay the 3D predictions onto the corresponding images (1st & 4th rows). Please refer to the supplemental material for more experimental results.

4.1. Implementation Details

Data preprocessing We only use LiDAR point clouds as input and voxelize all points in ranges [0, 70.4], [-40, 40], and [-3, 1] meters into a grid of resolution [0.05, 0.05, 0.1] along x, y, and z, respectively. We empirically set hyperparameters p_1=0.25, p_2=0.05, and p_3=0.1 (see Section 3.4). Besides shape-aware data augmentation, we adopt three types of common data augmentation: (i) mix-up [30], which randomly samples ground-truth objects from other scenes and add them into the current scene; (ii) local augmentation on points of individual ground-truth object, *e.g.*, random rotation and translation; and (iii) global augmentation on the whole scene, including random rotation, translation, and flipping. The former two are for preprocessing the inputs to both teacher and student SSDs.

Training details We adopt the ADAM optimizer and cosine annealing learning rate [18] with a batch size of four for 60 epochs. We follow [27] to ramp up μ_t (Eq. (5)) from 0 to 1 in the first 15 epoches using a sigmoid-shaped function $e^{-5(1-x)^2}$. We set τ_c and τ_I (Section 3.2) as 0.3 and 0.7, respectively, ω_1 and ω_2 (Eq. (5)) as 2.0 and 0.2, respectively, the EMA decay weight as 0.999, and γ (Eq. (4)) as 1.25.

	Method	Reference	Modality	$3D$				BEV				Time (ms)
				Easy	Mod	Hard	mAP	Easy	Mod	Hard	mAP	
Two-stage	MV3D [3]	CVPR 2017	LiDAR+RGB	74.97	63.63	54.00	64.2	86.62	78.93	69.80	78.45	360
	F-PointNet [20]	CVPR 2018	LiDAR+RGB	82.19	69.79	60.59	70.86	91.17	84.67	74.77	83.54	170
	AVOD [10]	IROS 2018	LiDAR+RGB	83.07	71.76	65.73	73.52	89.75	84.95	78.32	84.34	100
	PointRCNN [24]	CVPR 2019	LiDAR	86.96	75.64	70.70	77.77	92.13	87.39	82.72	87.41	100
	F-ConvNet [28]	IROS 2019	LiDAR+RGB	87.36	76.39	66.69	76.81	91.51	85.84	76.11	84.49	470*
	3D IoU Loss [37]	3DV 2019	LiDAR	86.16	76.50	71.39	78.02	91.36	86.22	81.20	86.26	80*
	Fast PointRCNN [4]	ICCV 2019	LiDAR	85.29	77.40	70.24	77.64	90.87	87.84	80.52	86.41	65
	UberATG-MMF [12]	CVPR 2019	LiDAR+RGB	88.40	77.43	70.22	78.68	93.67	88.21	81.99	87.96	80
	Part-A^2 [25]	TPAMI 2020	LiDAR	87.81	78.49	73.51	79.94	91.70	87.79	84.61	88.03	80
	STD [32]	ICCV 2019	LiDAR	87.95	79.71	75.09	80.92	94.74	89.19	86.42	90.12	80
	3D-CVF [33]	ECCV 2020	LiDAR+RGB	89.20	80.05	73.11	80.79	93.52	89.56	82.45	88.51	75
	CLOCs PVCas [19]	IROS 2020	LiDAR+RGB	88.94	80.67	77.15	82.25	93.05	89.80	86.57	89.81	100*
	PV-RCNN [23]	CVPR 2020	LiDAR	90.25	81.43	76.82	82.83	94.98	90.65	86.14	90.59	80*
	De-PV-RCNN [1]	ECCVW 2020	LiDAR	88.25	81.46	76.96	82.22	92.42	90.13	85.93	89.49	80*
One-stage	VoxelNet [38]	CVPR 2018	LiDAR	77.82	64.17	57.51	66.5	87.95	78.39	71.29	79.21	220
	ContFuse [13]	ECCV 2018	LiDAR+RGB	83.68	68.78	61.67	71.38	94.07	85.35	75.88	85.1	60
	SECOND [30]	Sensors 2018	LiDAR	83.34	72.55	65.82	73.9	89.39	83.77	78.59	83.92	50
	PointPillars [11]	CVPR 2019	LiDAR	82.58	74.31	68.99	75.29	90.07	86.56	82.81	86.48	**23.6**
	TANet [17]	AAAI 2020	LiDAR	84.39	75.94	68.82	76.38	91.58	86.54	81.19	86.44	34.75
	Associate-3Ddet [5]	CVPR 2020	LiDAR	85.99	77.40	70.53	77.97	91.40	88.09	82.96	87.48	60
	HotSpotNet [2]	ECCV 2020	LiDAR	87.60	78.31	73.34	79.75	94.06	88.09	83.24	88.46	40*
	Point-GNN [26]	CVPR 2020	LiDAR	88.33	79.47	72.29	80.03	93.11	89.17	83.90	88.73	643
	3DSSD [31]	CVPR 2020	LiDAR	88.36	79.57	74.55	80.83	92.66	89.02	85.86	89.18	38
	SA-SSD [8]	CVPR 2020	LiDAR	88.75	79.79	74.16	80.90	95.03	91.03	85.96	90.67	40.1
	CIA-SSD [35]	AAAI 2021	LiDAR	89.59	80.28	72.87	80.91	93.74	89.84	82.39	88.66	30.76
	SE-SSD (ours)	-	LiDAR	**91.49**	**82.54**	**77.15**	**83.73**	**95.68**	**91.84**	**86.72**	**91.41**	30.56

Table 1. Comparison with the state-of-the-art methods on the KITTI *test* set for car detection, with 3D and BEV average precisions of 40 sampling recall points evaluated on the KITTI server. Our SE-SSD attains the highest precisions for all difficulty levels with a very fast inference speed, outperforming all prior detectors. "*" means the runtime is cited from the submission on the KITTI website.

Method	$3D_{R40}$			BEV_{R40}			$3D_{R11}$
	Easy	Moderate	Hard	Easy	Moderate	Hard	Moderate
3DSSD [31]	-	-	-	-	-	-	79.45
SA-SSD [8]	92.23	84.30	81.36	-	-	-	79.91
De-PV-RCNN [1]	-	84.71	-	-	-	-	83.30
PV-RCNN [23]	92.57	84.83	82.69	95.76	91.11	88.93	83.90
SE-SSE (ours)	**93.19**	**86.12**	**83.31**	**96.59**	**92.28**	**89.72**	**85.71**

Table 2. Comparison with latest best two single- and two-stage detectors on KITTI *val* split for car detection, in which "R40" and "R11" mean 40 and 11 sampling recall points for AP, respectively.

4.2. Comparison with State-of-the-Arts

By submitting our prediction results to the KITTI server for evaluation, we obtain the 3D and BEV average precisions of our model on the KITTI test set and compare them with the state-of-the-art methods listed in Table 1.

As shown in the table, our model ranks the 1^{st} place among all state-of-the-art methods for both 3D and BEV detections in all three difficulty levels. Also, the inference speed of our model ranks the 2^{nd} place among all methods, about 2.6 times faster than the latest best two-stage detector Deformable PV-RCNN [1]. In 3D detection, our one-stage model attains a significant improvement of 1.1 points on moderate AP compared with PV-RCNN [1] and Deformable PV-RCNN [23]. For single-stage detectors, our model also outperforms all prior works by a large margin, outperforming the previous one-stage detector SA-SSD [8] by an average of 2.8 points for all three difficulty levels and with shorter inference time (reduced by $\sim$25%). Our large improvement in APs comes mainly from a better model optimization by exploiting both soft and hard targets, and the high efficiency of our model is mainly due to the nature of our proposed methods, *i.e.*, we refine features in SSD without incurring extra computation in the inference.

In BEV detection, our model also leads the best single- and two-stage detectors by around 0.8 points on average. Besides, we calculate the mean average precision (mAP) of three difficulty levels for comparison. Our higher mAPs indicate that SE-SSD attains a more balanced performance compared with others, so our method is more promising to address various cases more consistently in practice. Further, we compare our SE-SSD with latest best two single- and two-stage methods on KITTI *val* split. As shown in Table 2, our 3D and BEV moderate APs with 11 or 40 recall points both outperform these prior methods.

4.3. Ablation Study

Next, we present ablation studies to analyze the effectiveness of our proposed modules in SE-SSD on KITTI *val* split. Table 3 summarizes the ablation results on our consistency loss ("Cons loss"), ODIoU loss ("ODIoU"), and shape-aware data augmentation ("SA-DA"). For ODIoU loss, we replace it with the Smooth-L_1 loss in this ablation

Cons loss	ODIoU	SA-DA	Easy	Moderate	Hard
-	-	-	92.58	83.22	80.15
-	-	✓	93.02	83.70	80.68
-	✓	-	93.07	83.85	80.78
✓	-	-	93.13	84.15	81.17
✓	✓	-	93.17	85.81	83.01
✓	✓	✓	**93.19**	**86.12**	**83.31**

Table 3. Ablation study on our designed modules. We report the 3D average precisions of 40 sampling recall points on KITTI *val.* split for car detection. Cons Loss and SA-DA denote our consistency loss and shape-aware data augmentation, respectively.

study, since we cannot simply remove it like Cons loss and SA-DA. All reported APs are with 40 recall points.

Effect of consistency loss As first and fourth rows in Table 3 show, our consistency loss boosts the moderate AP by about 0.9 point. This large improvement shows that using the more informative soft targets can contribute to a better model optimization. For the slight increase in easy AP, we think that the predictions of the baseline on the easy subset are already very precise and thus are very close to the hard targets already. Importantly, by combining hard labels with the ODIoU loss in the optimization, our SE-SSD further boosts the moderate and hard APs by about 2.6 points, as shown in the fifth row in Table 3. This demonstrates the effectiveness of jointly optimizing the model by leveraging both hard and soft targets with our designed constraints.

Further, we analyze the effect of the consistency loss for bounding boxes ("reg") and confidence ("cls") separately to show the effectiveness of the loss on both terms. As Table 4 shows, the gain in AP from the confidence term is larger, we argue that the confidence optimization may be more effective to alleviate the misalignment between the localization accuracy and classification confidence. Also, we evaluate the box and confidence constraints [34] designed on the box-centers distance matching strategy and obtain a much lower AP ("dist"), we think that the underlying reason is related to their design, which is to address axis-aligned boxes and so is not suitable for our task.

Effect of ODIoU loss As first and third rows in Table 3 show, our ODIoU loss improves the moderate AP by about 0.6 points compared with the Smooth-L_1 loss. This gain in AP is larger than the one with the SA-DA module, thus showing the effectiveness of the constraints on both the box-centers distance and orientation difference in the ODIoU loss. Also, the gain in AP on the hard subset is larger than others, which is consistent with our expectation that even sparse points on the object surface could provide sufficient information to regress the box centers and orientations.

Effect of shape-aware data augmentation In Table 3, the first two rows indicate that our shape-aware data augmentation (SA-DA) scheme brings an average improvement of about 0.5 points on the baseline model. Based on the pre-trained SSD, SA-DA further improves the moderate and hard APs of SE-SSD by about 0.3 points, as indicated in the last two rows in Table 3. These gains in AP show the effectiveness of our SA-DA on boosting the performance by enhancing the object diversity and model generalizability.

Type	baseline	dist	reg only	cls only	cls + reg
Moderate AP	83.22	80.38	83.65	83.83	**84.15**

Table 4. Ablation study on our consistency loss, in which "cls" and "reg" mean our consistency loss on confidence and bounding boxes, respectively, and "dist" means the box and confidence constraints based on a box-centers distance matching strategy.

Type	baseline	nms filter	gt filter	stu filter
Moderate AP	83.22	83.49	80.73	**84.15**

Table 5. Ablation study on our IoU-based matching strategy, in which "nms", "gt", and "stu" mean that we filter soft targets with NMS, ground truths, and student predictions, respectively.

IoU-Based Matching Strategy Also, we compare different ways of filtering soft targets, *i.e.*, removing soft targets that (i) overlap with each other using NMS ("nms filter"), (ii) do not overlap with any ground truth ("gt filter"), and (iii) do not overlap with student boxes for less than an IoU threshold ("stu filter"). We can see from Table 5 that our proposed "stu filter" attains the largest gain in AP, as it keeps the most related and informative soft targets for the student predictions, compared with other strategies.

4.4. Runtime Analysis

The overall inference time of SE-SSD is only 30.56ms, including 2.84ms for data preprocessing, 24.33ms for network forwarding, and 3.39ms for post-processing and producing the final predictions. All evaluations were done on an Intel Xeon Silver CPU and a single TITAN Xp GPU. Our method attains a faster inference speed compared with SA-SSD [8], as our BEVConvNet structure is simpler and we further optimized our voxelization code.

5. Conclusion

This paper presents a novel self-ensembling single-stage object detector for outdoor 3D point clouds. The main contributions include the SE-SSD framework optimized by our formulated consistency constraint with soft targets, the ODIoU loss to supervise the network with hard targets, and the shape-aware data augmentation scheme to enlarge the diversity of training samples. The series of experiments demonstrate the effectiveness of SE-SSD and each proposed module, and show the advantage of high efficiency. Overall, our SE-SSD outperforms all state-of-the-art methods for both 3D and BEV car detection in the KITTI benchmark and attains an ultra-high inference speed.

Acknowledgments. This work is supported by the Hong Kong Centre for Logistics Robotics.

References

[1] Prarthana Bhattacharyya and Krzysztof Czarnecki. Deformable PV-RCNN: Improving 3D object detection with learned deformations. *arXiv preprint arXiv:2008.08766*, 2020. 7

[2] Qi Chen, Lin Sun, Zhixin Wang, Kui Jia, and Alan Yuille. Object as hotspots: An anchor-free 3D object detection approach via firing of hotspots. 2019. 7

[3] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3D object detection network for autonomous driving. In *CVPR*, 2017. 1, 7

[4] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast point R-CNN. In *ICCV*, 2019. 1, 7

[5] Liang Du, Xiaoqing Ye, Xiao Tan, Jianfeng Feng, Zhenbo Xu, Errui Ding, and Shilei Wen. Associate-3Ddet: Perceptual-to-conceptual association for 3D point cloud object detection. In *CVPR*, pages 13329–13338, 2020. 1, 2, 7

[6] 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. 1, 2, 6

[7] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3D semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, pages 9224–9232, 2018. 2, 3

[8] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3D object detection from point cloud. In *CVPR*, pages 11873–11882, 2020. 1, 2, 7, 8

[9] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2, 4

[10] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L. Waslander. Joint 3D proposal generation and object detection from view aggregation. *CoRR*, 2017. 1, 7

[11] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast encoders for object detection from point clouds. In *CVPR*, pages 12697–12705, 2019. 2, 7

[12] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3D object detection. In *CVPR*, pages 7345–7353, 2019. 1, 7

[13] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3D object detection. In *ECCV*, 2018. 7

[14] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 5

[15] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. Sparse convolutional neural networks. In *CVPR*, pages 806–814, 2015. 2, 3

[16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *ECCV*, pages 21–37, 2016. 4

[17] Zhe Liu, Xin Zhao, Tengteng Huang, Ruolan Hu, Yu Zhou, and Xiang Bai. TANet: Robust 3D object detection from point clouds with triple attention. In *AAAI*, pages 11677–11684, 2020. 1, 2, 7

[18] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 6

[19] Su Pang, Daniel Morris, and Hayder Radha. CLOCs: Camera-LiDAR object candidates fusion for 3D object detection. *arXiv preprint arXiv:2009.00784*, 2020. 1, 2, 7

[20] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3D object detection from RGB-D data. *CoRR*, 2017. 7

[21] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*, pages 652–660, 2017. 2

[22] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, pages 5099–5108, 2017. 5

[23] 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 *CVPR*, pages 10529–10538, 2020. 1, 2, 7

[24] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. PointRCNN: 3D object proposal generation and detection from point cloud. In *CVPR*, pages 770–779, 2019. 1, 2, 7

[25] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1, 2, 7

[26] Weijing Shi and Raj Rajkumar. Point-GNN: Graph neural network for 3D object detection in a point cloud. In *CVPR*, pages 1711–1719, 2020. 1, 2, 7

[27] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, pages 1195–1204, 2017. 2, 6

[28] Zhixin Wang and Kui Jia. Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection. In *IROS*, 2019. 7

[29] Liang Xie, Chao Xiang, Zhengxu Yu, Guodong Xu, Zheng Yang, Deng Cai, and Xiaofei He. PI-RCNN: An efficient multi-sensor 3D object detector with point-based attentive cont-conv fusion module. In *AAAI*, pages 12460–12467, 2020. 1

[30] Yan Yan, Yuxing Mao, and Bo Li. SECOND: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 2, 6, 7

[31] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3DSSD: Point-based 3D single stage object detector. In *CVPR*, pages 11040–11048, 2020. 2, 7

[32] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: Sparse-to-dense 3D object detector for point cloud. In *ICCV*, pages 1951–1960, 2019. 2, 7

[33] Jin Hyeok Yoo, Yeocheol Kim, Ji Song Kim, and Jun Won Choi. 3D-CVF: Generating joint camera and LiDAR fea-

tures using cross-view spatial feature fusion for 3D object detection. In *ECCV*, 2020. 1, 2, 7

[34] Na Zhao, Tat-Seng Chua, and Gim Hee Lee. SESS: Self-ensembling semi-supervised 3D object detection. In *CVPR*, pages 11079–11087, 2020. 2, 8

[35] Wu Zheng, Weiliang Tang, Sijin Chen, Li Jiang, and Chi-Wing Fu. CIA-SSD: Confident IoU-aware single-stage object detector from point cloud. In *AAAI*, 2021. 3, 7

[36] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-IoU loss: Faster and better learning for bounding box regression. In *AAAI*, pages 12993–13000, 2020. 4

[37] Dingfu Zhou, Jin Fang, Xibin Song, Chenye Guan, Junbo Yin, Yuchao Dai, and Ruigang Yang. IoU loss for 2D/3D object detection. In *3DV*, pages 85–94, 2019. 7

[38] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-end learning for point cloud based 3D object detection. In *CVPR*, pages 4490–4499, 2018. 2, 7