CIA-SSD: Confident IoU-Aware Single-Stage Object Detector From Point Cloud

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

Abstract

Existing single-stage detectors for locating objects in point clouds often treat object localization and category classification as separate tasks, so the localization accuracy and classification confidence may not well align. To address this issue, we present a new single-stage detector named the *Confident IoU-Aware Single-Stage object Detector* (CIA-SSD). First, we design the lightweight *Spatial-Semantic Feature Aggregation* module to adaptively fuse high-level abstract semantic features and low-level spatial features for accurate predictions of bounding boxes and classification confidence. Also, the predicted confidence is further rectified with our designed *IoU-aware confidence rectification module* to make the confidence more consistent with the localization accuracy. Based on the rectified confidence, we further formulate the *Distance-variant IoU-weighted NMS* to obtain smoother regressions and avoid redundant predictions. We experiment CIA-SSD on 3D car detection in the KITTI test set and show that it attains top performance in terms of the official ranking metric (moderate AP 80.28%) and above 32 FPS inference speed, outperforming all prior single-stage detectors. The code is available at https://github.com/Vegeta2020/CIA-SSD.

1 Introduction

To detect objects in autonomous driving, point clouds are often adopted to offer robust information. In general, there are two classes of methods to detect objects in point clouds: single-stage and two-stage. Though two-stage detectors usually attain higher average precisions benefited from an extra refinement stage, single-stage detectors are typically more efficient due to their simpler network structures. Also, the detection precisions of recent single-stage detectors (He et al. 2020; Yang et al. 2020; Shi and Rajkumar 2020) gradually approach that of the state-of-the-art two-stage detectors. The advantages of time efficiency and competitive precision motivate us to focus this work on single-stage detectors.

Existing 3D object detectors often treat object localization and category classification as separate tasks, so the localization accuracy and classification confidence may not align well (Jiang et al. 2018). Hence, two-stage detectors (Yang et al. 2019; Shi et al. 2020a) extract features from the region proposals generated by the first-stage backbone and predict the IoUs between the regressed bounding boxes and ground-truth boxes in the second stage to refine the confidence predictions. Compared with hard-category labels, the soft IoU labels are usually more consistent with the localization qualities, thus leading to more accurate confidence predictions.

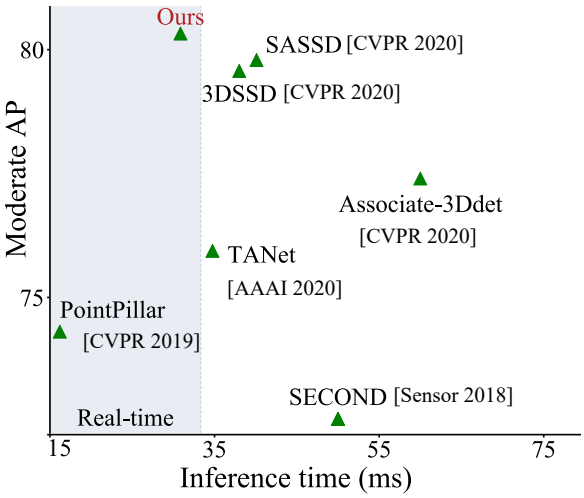


Figure 1: Our CIA-SSD attains top performance (official rank: moderate AP 80.28%) and real-time speed (30.76 ms) on 3D car detection in KITTI test set (Geiger et al. 2013), compared with the state-of-the-art single-stage detectors.

Compared with two-stage detectors, single-stage detectors cannot train features extracted from their predicted bounding boxes with a second-stage network. Also, their features are learned mostly based on the pre-defined anchors or classified positive points, so the resulting IoU predictions may not be as accurate as those in the two-stage networks. Hence, general single-stage detectors cannot effectively rectify confidence predictions like the two-stage ones.

To resolve this issue, SASSD (He et al. 2020), a very recent single-stage detector, exploits an interpolation approach to obtain the region proposal features for confidence rectification. Their approach is, however, very complex with the interpolation operation. In this work, we design a new *confidence rectification module* embedded in our *Confident IoU-*

Aware Single-Stage object Detector (CIA-SSD) to address the issue more elegantly. Our key idea is based on the finding that anchor-feature-based IoU predictions are discriminative especially between the precise and imprecise regressions of the bounding boxes. Thus, by utilizing a convex function to augment the discrimination, we polarize the effect of IoU predictions between the precise and imprecise regressions and effectively rectify the confidence in the post processing.

Besides, we design a lightweight *Spatial-Semantic Feature Aggregation* (SSFA) module to adaptively fuse high-level abstract semantic features and low-level spatial features for more accurate predictions of bounding boxes and classification confidence. Compared with the commonly-used 2D feature extraction module (Yan, Mao, and Li 2018; He et al. 2020), our SSFA module boosts the performance effectively with a moderate increase in the model complexity. Further, we notice that there are often more redundant false-positive predictions and strong oscillations for bounding boxes regressed at large distances from the viewpoint. Hence, we formulate a novel *Distance-variant IoU-weighted NMS* (DI-NMS) to obtain smoother regressions and reduce redundant predictions, by considering the depth factor not encountered in 2D NMS.

Figure 2 illustrates the quality of our results with an example, in which our method can find better-aligned bounding boxes and avoid redundant predictions, compared with the state-of-the-art single-stage detector SASSD. Below, we summarize the contributions of this work: (i) a lightweight spatial-semantic feature aggregation module that adaptively fuses high-level abstract semantic features and low-level spatial features for more accurate bounding box regression and classification; (ii) an IoU-aware confidence rectification module to alleviate the misalignment between the localization accuracy and classification confidence; and (iii) a distance-variant IoU-weighted NMS to smooth the regressions and avoid redundant predictions. CIA-SSD attains top performance (moderate AP 80.28%) and real-time speed (30.76 ms) on 3D car detection in KITTI test set (Geiger et al. 2013), as illustrated in Figure 1.

2 Related Work

There are two types of LiDAR-based 3D object detectors: (i) Two-stage detectors usually generate region proposals in the first stage, then feed these regions of interests into a second-stage network for refinement; and (ii) In contrast, single-stage detectors have simpler networks, since they directly regress class scores and bounding boxes in one stage.

Among the two-stage detectors, PointRCNN (Shi, Wang, and Li 2019) uses PointNet++ (Qi et al. 2017) as the backbone in both stages and devises an anchor-free strategy to generate 3D proposals. Based on PointRCNN, Part-A^2 (Shi et al. 2020b) replaces the PointNet++ backbone with a sparse convolutional network, and proposes RoI-aware point cloud feature pooling in the refinement. STD (Yang et al. 2019) exploits spherical anchors to generate proposals, together with a segmentation branch, to reduce the number of positive anchors. PV-RCNN (Shi et al. 2020a) uses set abstraction modules to extract point features from multi-scale voxel features in the first stage to refine the region proposals.

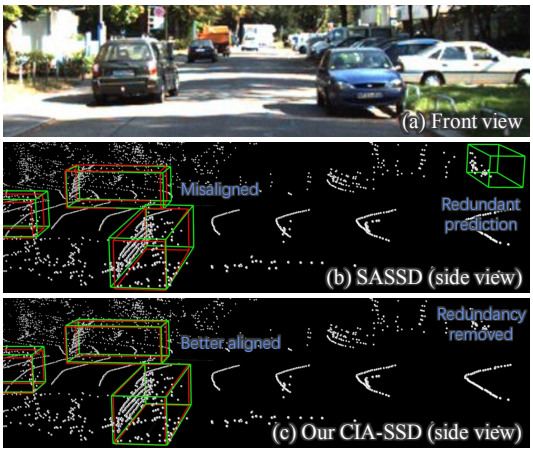


Figure 2: As shown in this example, our CIA-SSD (c) can predict bounding boxes (green) that better align with the ground truths (red) and avoid redundant predictions, as compared with the very recent single-stage detector SASSD (b).

Among the single-stage detectors, VoxelNet (Zhou and Tuzel 2018) voxelizes the point cloud and proposes a voxel feature encoding layer to extract point features in each voxel and produce fixed-length features for batch training. PointPillar (Lang et al. 2019) divides a point cloud into pillars instead of voxels for feature extraction, then utilizes a 2D convolutional detection architecture for object localization. SECOND (Yan, Mao, and Li 2018) exploits the sparse convolution (Liu et al. 2015) and submanifold sparse convolution (Graham, Engelcke, and Van Der Maaten 2018) to replace conventional 3D convolution. TANet (Liu et al. 2020) presents a triple attention module embedded in voxel feature extraction and combines it with a proposed cascaded refinement network. Very recently, Point-GNN (Shi and Rajkumar 2020) proposes a graph neural network to extract point features and achieves a decent performance. 3DSSD (Yang et al. 2020) proposes a fusion sampling strategy by combining both feature- and point-based farthest point sampling for better classification performance. SASSD (He et al. 2020) presents an auxiliary network in parallel with a sparse convolutional network to regress the box centers and semantic classes for each point with interpolated voxel features.

As we shall show in the results, recent single-stage detectors have achieved comparable performance (average precision) with the state-of-the-art two-stage detectors. Given their high efficiency, single-stage detectors have great potential for real-time applications. This motivates us to focus this research on developing a new single-stage detector CIA-SSD, which has attained top performance and real-time speed, compared with all previous single-stage detectors.

3 Confident IoU-Aware Single Stage Detector

Figure 3 shows our model's pipeline, which has three parts: (i) the sparse convolutional network (SPConvNet) for encoding the input point cloud; (ii) the SSFA module for extracting robust spatial-semantic features; and (iii) the multi-task head with a confidence function for rectifying the classification score and DI-NMS for post-processing.

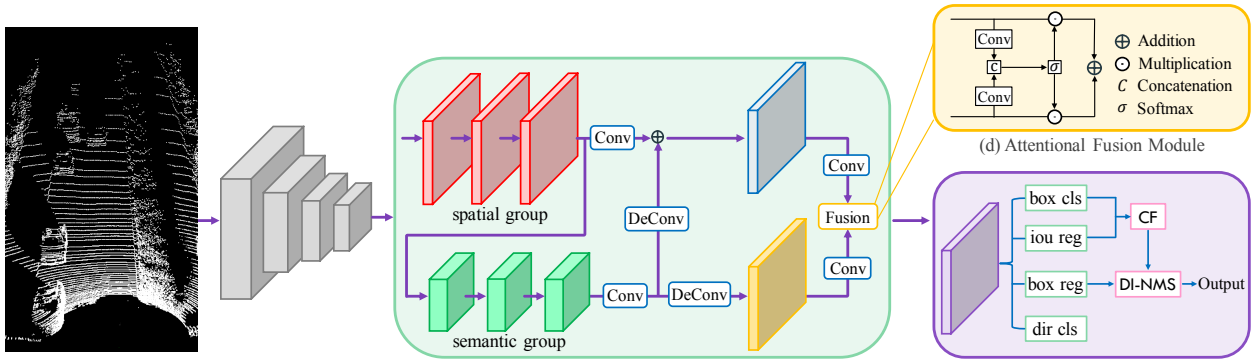


Figure 3: The pipeline of our proposed Confident IoU-Aware Single-Stage object Detector (CIA-SSD). First, we encode the input point cloud (a) with a sparse convolutional network denoted by SPConvNet (b), followed by our spatial-semantic feature aggregation (SSFA) module (c) for robust feature extraction, in which an attentional fusion module (d) is adopted to adaptively fuse the spatial and semantic features. Then, the multi-task head (e) realizes the object classification and localization, with our introduced confidence function (CF) for confidence rectification. In the end, we further formulate the distance-variant IoU-weighted NMS (DI-NMS) for post-processing. Note that "box cls," "iou reg," "box reg," and "dir cls" in (e) denote bounding box classification, IoU prediction regression, bounding box regression, and direction classification, respectively.

3.1 Point Cloud Encoder

To encode a point cloud, we voxelize it and calculate the mean coordinates and intensities of points in each voxel as the initial feature. We then utilize the sparse convolutional network SPConvNet (see Figure 3(b)), following the settings of SECOND (Yan, Mao, and Li 2018) to extract features from the sparse voxels. The SPConvNet consists of four blocks, each comprising several submanifold sparse convolution (SSC) (Graham, Engelcke, and Van Der Maaten 2018) layers and one sparse convolution (SC) (Liu et al. 2015) layer. Specifically, our four blocks have $\{2, 2, 3, 3\}$ SSC layers, respectively, and an SC layer appended to the end of each block for 2x downsampling on the 3D feature maps. Lastly, we transform the sparse voxel features to dense feature maps and concatenate the features in z to produce the BEV feature maps as inputs to the SSFA module.

3.2 Spatial-Semantic Feature Aggregation

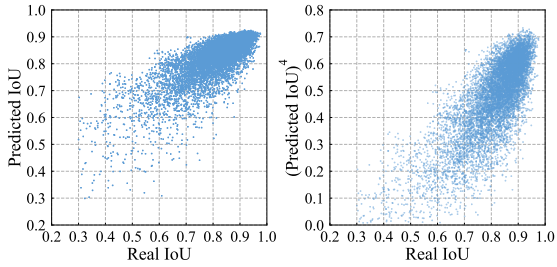
To detect cars in autonomous driving, we have to (i) regress precise car locations and also to (ii) classify each regressed bounding box as a positive/negative sample. In such processes, it is crucial to consider both *low-level spatial features* and *high-level abstract semantic features*. However, when we enrich the high-level abstract semantics in the feature maps, e.g., through stacked convolution layers, the quality of the low-level spatial information often declines as a result. Hence, the commonly-used BEV feature extraction module (Yan, Mao, and Li 2018; He et al. 2020), which applies stacked convolution layers, could not effectively obtain robust features with rich spatial information.

In this work, we design the *spatial-semantic feature aggregation* (SSFA) module. As shown in Figures 3 (c) & (d), our SSFA module contains two groups of convolution layers and an attentional fusion module at the end. The two convolution groups are named the *spatial* and *semantic* groups (with corresponding layer-wise features), indicated by the red and green blocks in Figure 3(c), respectively, and their outputs are named the spatial and semantic features, respectively. Specifically, we keep the dimensions (number of channels and feature map size) of the spatial feature to be the same as the input to avoid loss of spatial information. For the semantic group, we aim to gain more high-level abstract semantic information by taking the spatial feature as input, doubling the number of channels, and reducing the spatial size by half with an initial convolution layer of stride two. Also, we adopt a 2D DeConv layer to recover the dimensions of the semantic feature map to be the same as the spatial feature map before the element-wise addition to producing the enriched spatial feature (the blue block in Figure 3(c)). On the other hand, we use another 2D DeConv layer to produce the upsampled semantic feature (the yellow block in Figure 3(c)) before the attentional fusion.

To adaptively fuse the enriched spatial feature and the upsampled semantic feature, we adopt the attention module shown in Figure 3(d). First, we compress the channels of each feature to one and concatenate the results. We then use Softmax to normalize the two concatenated channels and split them into two BEV attention maps, in which Softmax builds the dependence between the two features for adaptive feature fusion. Lastly, we take the learned BEV attention maps as weights on the respective features and perform an element-wise addition ($\oplus$ in Figure 3(d)) to fuse the weighted features. The SSFA module helps extract more robust features with rich spatial and semantic information for more accurate predictions of bounding boxes and classification confidence (see the ablation study in Section 4.3).

3.3 IoU-Aware Confidence Rectification

To alleviate the misalignment between the localization accuracy and classification confidence without having an additional network stage, we design the *IoU-aware confidence rectification module* for post-processing the confidence. In 3D object detection, we usually define anchors by evenly distributing bounding boxes of fixed size on the BEV feature



(a) (b)

Figure 4: Scatterplots: (a) real IoUs vs. predicted IoUs, showing that high predicted IoU often associates with high real IoU; and (b) real IoUs vs. (predicted IoUs)$^{\beta}$ with $\beta = 4$, such that the rectified IoUs become more discriminative for us to locate good IoU predictions with high certainty.

maps; then, we can train the network by regressing the offsets between the anchors and ground truths. Suffering from the misaligned features of the anchors, the IoUs predicted by anchor-based single-stage detectors are often not as accurate as the IoUs predicted with region proposals in two-stage detectors. However, we observe from an experiment that the IoUs predicted with anchors are still rather discriminative.

Figure 4 shows the experimental result of IoU prediction ("iou reg" in Figure 3(e)) conducted on the KITTI validation set. Specifically, "real IoU" refers to the IoU computed between the anchor-based predicted bounding box and its nearest ground-truth box, whereas "predicted IoU" refers to the network-predicted IoU on the corresponding anchor-based predicted bounding box. From the scatterplot shown in Figure 4(a), we can see that although the predicted IoU cannot perfectly match the real one and the prediction deviation increases when the real IoU drops, high predicted IoUs often associate with high real IoUs, thus allowing us to differentiate between the precise and imprecise bounding box regressions. Typically, if an anchor feature leads to precise regression, the feature should also predict high IoU with high certainty, as the feature already contains sufficient location information. On the other hand, if an anchor feature produces imprecise regression, e.g., when the anchor is far away from the ground truths, the feature will likely lead to low-IoU prediction with high uncertainty.

To suppress the uncertainties of low-IoU predictions and further augment the discrimination between low-IoU and high-IoU predictions, we introduce the rectification item g:

$$g = i^{\beta} \ , \tag{1}$$

where i denotes the predicted IoU and β is a hyperparameter that controls the degrees of suppressing the low-IoU predictions and augmenting the high-IoU predictions. As shown in Figure 4(b), the predictions of high IoUs can become more discriminative, *e.g.*, when setting $\beta = 4$. Hence, we propose to rectify the classification score with g and formulate the following Confidence Function f in the multi-task head:

$$f = c \cdot g = c \cdot i^{\beta} \tag{2}$$

where c is the classification score of the predicted bounding box. By this means, we can polarize the effect of low-IoU and high-IoU predictions for better rectification of c.

Algorithm 1 Distance-variant IoU-weighted NMS

Require:
$\mathcal{C}$ is the $N\times7$ matrix of predicted bounding box parameters (x, y, z, w, l, h, r), where N is the number of bounding boxes, xyz denote box center, wlh denote box dimensions, and r denotes box orientation angle;
$\mathcal{S}$ is the set of N rectified confidence values of the corresponding predicted bounding boxes;
$\mathcal{I}$ is the set of N IoU prediction values of the corresponding predicted bounding boxes;
$\mathcal{A}$ is the $N\times7$ matrix of corresponding anchors for the predicted box parameters (x, y, z, w, l, h, r);
$\mathcal{C} = \{c_1, c_2, ..., c_N\}$; $\mathcal{I} = \{iou_1, iou_2, ..., iou_N\}$;
$\mathcal{S} = \{s_1, s_2, ..., s_N\}$; *and* $\mathcal{A} = \{a_1, a_2, ..., a_N\}$.

Ensure:
Selected bounding boxes: $\mathcal{B} = \emptyset, \mathcal{L} = \{1, 2, ..., N\}$

1: $dist = \{\|c_{i,[0,1]} - a_{i,[0,1]}\|_2 \mid c_i \in \mathcal{C}, a_i \in \mathcal{A}, i \in \mathcal{L}\}$
2: $\mathcal{S} = \{s_i \cdot (1 - softmax_i(dist)) \mid s_i \in \mathcal{S}, i \in \mathcal{L}\}$
3: $iou\_thres = 0.3, cnt\_thres = \mu$
4: **while** $\mathcal{L} \neq \emptyset$ **do**
5: $cnt = 0, idx = \arg\max_{i \in \mathcal{L}} \mathcal{S}, c' = c_{idx}$
6: $\mathcal{L}' = \{i \mid i \in \mathcal{L}, IoU(c_i, c') > iou\_thres, c_i \in \mathcal{C}\}$
7: $cnt = \sum_{i \in \mathcal{L}'} iou_i \cdot IoU(c_i, c'), c_i \in \mathcal{C}, iou_i \in \mathcal{I}$
8: $b = \frac{\sum_{i \in \mathcal{L}'} iou_i \cdot c_i \cdot e^{-(1-IoU(c_i,c'))^2/\sigma^2}}{\sum_{i \in \mathcal{L}'} iou_i \cdot e^{-(1-IoU(c_i,c'))^2/\sigma^2}}, c_i \in \mathcal{C}, iou_i \in \mathcal{I}$
$\sigma \propto \|c_{i,[0,1]}\|_2$
9: $\mathcal{L} \leftarrow \mathcal{L} - \mathcal{L}'$
10: if $cnt > cnt\_thres$, $\mathcal{B} \leftarrow \mathcal{B} \cup \{b\}$
11: **end while**
12: **return** $\mathcal{B}$

During the training, we train the IoU prediction branch simultaneously with the bounding box regression and classification branches and detach the predicted bounding boxes from the computation graph in the IoU predictions to avoid the gradients to back-propagate from the IoU prediction loss to the bounding box regressions. Only in the testing, we make use of the Confidence Function in Eq. (2) to rectify the classification score and produce the rectified classification confidence f for use in the following NMS process.

3.4 Distance-Variant IoU-Weighted NMS

Distant objects are often predicted with low classification confidence and high regression uncertainties, caused by the point sparsity at distant regions. Hence, we often observe (i) strong oscillations on the regressed bounding boxes and (ii) redundant false-positive predictions that do not overlap with any ground-truth bounding box (in which a few anchors are incorrectly activated to perform the regressions).

To avoid these issues, we formulate the *distance-variant IoU-weighted NMS* (DI-NMS) for post-processing the predictions. Algorithm 1 outlines the procedure: (i) In Steps 1-2, we calculate the L_2 distance between the BEV centers of each pair of anchor and predicted box, and take the distance offset to refine the confidence; (ii) In Steps 5-7, we

Figure 5: Snapshots of our 3D detection results on the KITTI validation set. The predicted and ground-truth bounding boxes are shown in green and red, respectively, and are projected back onto the color images (1^{st} & 3^{rd} rows) for visualization.

select box c' with the highest classification confidence and find indices of its overlapped boxes $\mathcal{L}'$ filtered with an IoU threshold, in which c' and $\mathcal{L}'$ are the candidate box and indices of auxiliary boxes, respectively. Next, we calculate a cumulative sum of products between the IoU predictions of the auxiliary boxes and their real IoUs with the candidate box, in which we design the cumulative sum to filter redundant (zero-IoU) false positives; and (iii) In Step 8, we use a Gaussian weighted average to obtain a smooth regression from the auxiliary boxes.

Note, auxiliary boxes that have large IoU predictions or high IoUs with the candidate box are assigned with large Gaussian weights. Also, considering that the localization uncertainty increases with distance, we utilize σ to adjust the smoothness degree of the Gaussian weights, in which σ is positively correlated with the BEV distance between the box center and viewpoint. Thus, we can use the smooth Gaussian weights to evenly consider the oscillated auxiliary boxes in distant predictions, while using differentiated weights to focus on the auxiliary boxes that are highly overlapped with the candidate box in short-range predictions.

3.5 Loss Function

For model optimization, we follow the conventional settings of box encoding and loss functions in (Yan, Mao, and Li 2018). Specifically, we use the Focal loss (Lin et al. 2017), Smooth-L_1 loss (Liu et al. 2016), and cross-entropy loss in the bounding box classification ($\mathcal{L}_{\text{cls}}$), box regression ($\mathcal{L}_{\text{box}}$), and direction classification ($\mathcal{L}_{\text{dir}}$). Besides, we encode the IoUs between the predicted boxes and the ground-truth boxes to keep them in the range of [-1, 1]:

$$iou_t = 2 \cdot (iou - 0.5) \tag{3}$$

where iou denotes the real IoU between the predicted bounding box and ground-truth box, and iou_t denotes the encoded target for the IoU prediction. Then, we apply the Smooth-L_1 loss on the IoU regressions to calculate the IoU prediction loss $\mathcal{L}_{\text{iou}}$. Similar to the bounding box regression loss $\mathcal{L}_{\text{box}}$, we calculate the IoU prediction loss $\mathcal{L}_{\text{iou}}$ only on the positive samples. The overall loss $\mathcal{L}$ is defined as

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \omega\mathcal{L}_{\text{box}} + \mu\mathcal{L}_{\text{dir}} + \lambda\mathcal{L}_{\text{iou}} \tag{4}$$

where we empirically set $\omega = 2.0$, $\mu = 0.2$, and $\lambda = 1.0$.

4 Experiments

We evaluate our CIA-SSD on the KITTI 3D object benchmark (Geiger et al. 2013) with 7,481 training samples and 7,518 test samples. The training samples are further divided into a training set (3,712 samples) and a validation set (3,769 samples). Following previous works, *e.g.*, SASSD (He et al. 2020) and SECOND (Yan, Mao, and Li 2018), we conducted experiments on the most commonly-used car category and evaluated the results by average precision (AP) with IoU threshold 0.7. Also, the dataset has three difficulty levels (easy, moderate, and hard) based on the object size, occlusion, and truncation levels. Figure 5 shows some of our predicted bounding boxes projected onto color images.

4.1 Implementation Details

We voxelize the input point cloud into a grid of resolutions [0.05, 0.05, 0.1] meters in ranges [0, 70.4], [-40, 40], and [-3, 1] meters along the x, y, and z axes, respectively. The anchors are pre-defined evenly on the BEV feature map with the same dimensions (width=1.6m, length=3.9m, height=1.56m) and two possible orientations (0° or 90°). Further, they are divided into three categories (positive, negative, and ignored) based on the matching strategy in (Zhou and Tuzel 2018) with IoU thresholds 0.45 and 0.6.

We consider four types of data augmentations to enhance our model's generalization ability. The first type is a global

Type	Method	Data	$AP_{3D}(\%)$		
			Easy	Mod	Hard
2-stage	MV3D (2017)	LiDAR+RGB	74.97	63.63	54.00
	F-PointNet (2018)	LiDAR+RGB	82.19	69.79	60.59
	AVOD (2018)	LiDAR+RGB	83.07	71.76	65.73
	PI-RCNN (2020)	LiDAR+RGB	84.37	74.82	70.03
	PointRCNN (2019)	LiDAR	86.96	75.64	70.70
	F-ConvNet (2019)	LiDAR+RGB	87.36	76.39	66.69
	3D IoU Loss (2019)	LiDAR	86.16	76.50	71.39
	Patches (2019)	LiDAR	88.67	77.20	71.82
	Fast PointRCNN (2019)	LiDAR	85.29	77.40	70.24
	UberATG-MMF (2019)	LiDAR+RGB	88.40	77.43	70.22
	Part-A^2 (2020b)	LiDAR	87.81	78.49	73.51
	3D IoU-Net (2020)	LiDAR	87.96	79.03	72.78
	STD (2019)	LiDAR	87.95	79.71	75.09
	3D-CVF (2020)	LiDAR+RGB	88.84	79.72	72.80
	PV-RCNN (2020a)	LiDAR	*90.25*	*81.43*	*76.82*
1-stage	VoxelNet (2018)	LiDAR	77.82	64.17	57.51
	ContFuse (2018)	LiDAR+RGB	83.68	68.78	61.67
	SECOND (2018)	LiDAR	83.34	72.55	65.82
	PointPillars (2019)	LiDAR	82.58	74.31	68.99
	TANet (2020)	LiDAR	84.39	75.94	68.82
	Associate-3Ddet (2020)	LiDAR	85.99	77.40	70.53
	Point-GNN (2020)	LiDAR	88.33	79.47	72.29
	3DSSD (2020)	LiDAR	88.36	79.57	**74.55**
	SASSD (2020)	LiDAR	88.75	79.79	74.16
	CIA-SSD (ours)	LiDAR	**89.59**	**80.28**	72.87

Table 1: Comparison with the state-of-the-art methods on the KITTI test set. The 3D average precisions of 40 sampling recall points for car detection are evaluated on the KITTI official server; from the official ranking metric "*Moderate AP*," we can see that our CIA-SSD attains the top performance compared with all 1-stage detectors and is comparable with the top 2-stage detector on this challenging problem.

augmentation on the entire point cloud, including random rotation, scaling, and flipping. The second type is a local augmentation on a portion of the point cloud around a ground-truth object, including random rotation and translation. The third type is a ground-truth augmentation following SECOND (Yan, Mao, and Li 2018). Last, we filter out objects with difficulty levels not attributed to easy, moderate, and hard to improve the quality of the positive samples, and take also objects of similar categories, such as van for car, as the targets to alleviate model confusion in the training.

In the SSFA module, the spatial and semantic groups have three stacked convolution layers of kernel 3x3 with a number of channels 128 and 256, respectively. After the spatial and semantic groups, there is one 1x1 convolution layer with 128 and 256 channels separately. The 2D DeConv layers have 3x3 kernels and 128 output channels with stride two. Before the attentional fusion, we use a 3x3 convolution layer with 128 output channels to transform each group feature. In the attentional fusion, the convolutional layers have 3x3 kernels and one output channel. Other network settings follow those in SECOND (Yan, Mao, and Li 2018).

We use the ADAM optimizer (Kingma and Ba 2014) with the cosine annealing learning rate (Loshchilov and Hutter 2016) to train our model with a batch size of four on a single GPU card for 60 epochs. Further, we empirically set $\beta = 4$ (in the confidence function), $\mu = 2.6$ (in DI-NMS), and $\sigma = \{0.0009, 0.009, 0.1, 1\}$ (in DI-NMS) for BEV distances in ranges [0, 20m), [20m, 40m), [40m, 60m), and [60m, 70.4m], respectively.

Method	$AP_{3D}(\%)$		
	Easy	Mod	Hard
VoxelNet (2018)	81.97	65.46	62.85
ContFuse (2018)	86.32	73.25	67.81
SECOND (2018)	87.43	76.48	69.10
TANet (2020)	88.21	77.85	75.62
PointPillars (2019)	-	77.98	-
Point-GNN (2020)	87.89	78.34	77.38
Associate-3Ddet (2020)	89.29	79.17	77.76
3DSSD (2020)	89.71	79.45	78.67
SASSD (2020)	**90.15**	**79.91**	78.78
CIA-SSD (ours)	90.04	79.81	**78.80**

Table 2: Comparison with the state-of-the-art single-stage detectors on the KITTI validation set, in which the 3D average precisions for car detection are based on 11 sampling recall points (vs. 40 points in the KITTI test set).

4.2 Comparison with State-of-the-Arts

We compare our CIA-SSD with the state-of-the-art methods listed in Table 1. As shown in the table, our model ranks the 1^{st} place in terms of moderate and easy AP, *i.e.*, 80.28% and 89.59% respectively, among all the single-stage detectors. The "moderate AP" is the official ranking metric for 3D detection on the KITTI official test server. Our CIA-SSD outperforms all the state-of-the-art single-stage detectors, including the very recent ones, including Point-GNN, 3DSSD, and SASSD by about 0.5 to 0.8 points under this metric. Besides, while two-stage detectors generally perform better than single-stage detectors due to the extra second-stage refinement, our proposed single-stage detector still outperforms most of the recent two-stage detectors, *e.g.*, 3D-CVF, STD, and Part-A^2 by about 0.6 to 1.8 points on moderate AP. Furthermore, we shall show the high efficiency of our model compared with two-stage detectors in Section 4.4.

Although our model sets the new state-of-the-art single-stage results on easy and moderate APs for the KITTI test set, the corresponding APs are slightly lower than some of the state-of-the-art results on the validation set, as shown in Table 2. Also, our hard AP is lower than the state-of-the-art methods on the test set, while being the top on the validation set. We argue that such inconsistency may be caused by the mismatched distributions between the KITTI val and test splits, as mentioned in Part-A^2 (Shi et al. 2020b).

4.3 Ablation Study

We adopt the 2D feature extraction module (Yan, Mao, and Li 2018) grouped by seven stacked convolutional layers, the normal NMS, and multi-task head without the IoU prediction branch as the baseline modules in the ablation study.

Effect of data processing Table 3 shows results that reveal the effect of each data processing technique on the baseline modules. Both global and local augmentations effectively improve easy and moderate APs. Also, the ground truth augmentation, training with the similar type of objects, and filtering objects with suitable difficulty levels for ground-truth augmentation boost all levels of AP effectively. All these techniques help to build up a strong baseline for better validation of our proposed modules.

glo.	*loc.*	*gt aug.*	*sim.*	*diff.*	Easy	Mod	Hard
					81.29	66.39	65.40
✓					86.53	75.73	74.77
✓	✓				87.18	76.33	68.64
✓	✓	✓			87.97	78.03	76.80
✓	✓	✓	✓		88.81	78.35	77.26
✓	✓	✓	✓	✓	**89.09**	**78.73**	**77.56**

Table 3: Ablation study on our implemented data processing techniques on the baseline modules, in which we report the 3D average precisions of 11 sampling recall points for car detection on the KITTI val split. Here, "glo.," "loc," "gt aug.," "sim.," and "diff." denote the global augmentation, local augmentation, ground truth augmentation, training with similar type of objects, and filtering objects with difficulty levels not attributed to easy, moderate, & hard, respectively.

SSFA	*CF*	*DI-NMS*	Easy	Mod	Hard
			89.09	78.73	77.56
✓			89.46	79.17	77.88
✓	✓		89.66	79.63	78.64
✓	✓	✓	**90.04**	**79.81**	**78.80**

Table 4: Ablation study on our modules: SSFA, CF, and DI-NMS. Here, we report the 3D average precisions of 11 sampling recall points for car detection on the KITTI val split.

Effect of SSFA module As shown in Table 4, our SSFA module contributes an improvement of 0.37, 0.44, and 0.32 on the moderate, easy, and hard APs, respectively. Besides, compared with the baseline 2D feature extraction module under a batch size of four, our SSFA module and the corresponding module in SASSD (He et al. 2020) increase our framework's GPU occupation by about 10% and 27%, respectively, validating that our SSFA module is lightweight.

Effect of confidence function In Table 4, our confidence function in the multi-task head improves the easy, moderate, and hard APs by 0.20, 0.46, and 0.76, respectively. Also, we use the metric Pearson Correlation Coefficient (PCC) (D., W., and L. 2008) to measure the correlation between the IoU and confidence of our predicted boxes and calculate the corresponding moderate AP for different β values in Table 5. We can see that setting $\beta = 4$ leads to the highest PCC and moderate AP. Besides, the AP still increases from 79.17 to 79.30 when setting $\beta = 0$ (*i.e.*, without rectifying the classification scores), because the IoU prediction optimizes the model to make the learned features aware of the relative locations between the predicted boxes and ground truths.

Effect of DI-NMS Our DI-NMS raises the easy AP (by 0.38), moderate AP (by 0.18), and hard AP (by 0.16); see Table 4. However, different from 2D detection with region proposals that often fully cover the objects, candidate boxes in 3D detection may not have good object coverage, due to the missing points around the boundary, so it is hard to produce well-aligned boxes with the weighted average in DI-NMS. Hence, the increase in AP with DI-NMS is lower than that with the SSFA module and confidence function.

β	0	1	2	3	4	5	6
AP_{Mod}	79.30	79.37	79.56	79.61	**79.63**	79.62	79.61
PCC	0.460	0.471	0.502	0.509	**0.511**	0.510	0.509

Table 5: Hyperparameter analysis on β in the confidence function. Here, we report the 3D moderate average precision (AP_{Mod}) and Pearson Correlation Coefficient (PCC) between the IoU and confidence of the predicted boxes for different β values in car detection on the KITTI val split.

1-stage	Point-GNN	Associate-3Ddet	SASSD	3DSSD	TANet	Ours (1-stage)
time (ms)	643	60	40.1	38	34.75	**30.76**

Table 6: Comparing the runtime (in millisecond) of our model with very recent state-of-the-art single-stage detectors, showing that our model runs the fastest among them.

2-stage	PointRCNN	Part-A^2	STD	Fast PointRCNN	Ours (1-stage)
time (ms)	100	80	80	65	**30.76**

Table 7: Comparing the runtime (in millisecond) of our model with state-of-the-art two-stage detectors, showing that our single-stage detector has much higher efficiency.

4.4 Runtime Analysis

Table 6 compares the runtime of our CIA-SSD with five very recent state-of-the-art single-stage detectors, and we can see that our CIA-SSD is the fastest one. Notice that our CIA-SSD is faster than SASSD as the number of channels in our SSFA module is only half of that of SA-SSD in most layers. Next, Table 7 compares the runtime of CIA-SSD with four recent two-stage detectors. From the table, we can see that CIA-SSD runs significantly faster than these two-stage detectors, confirming the high runtime efficiency of single-stage detectors. The average inference time of CIA-SSD is 30.76ms, including (i) 2.84ms for data processing before the network forwarding; (ii) 24.33ms for data processing in the network; and (iii) 3.59ms for post-processing to produce the final predictions. All the reported timing results were averaged from five runs of our program on an Intel Xeon Silver CPU and a single TITAN Xp GPU.

5 Conclusion

This paper presents a new object detector on point clouds, named Confident IoU-Aware Single-Stage object Detector (CIA-SSD). Our main contributions include the spatial-semantic feature aggregation module for extracting robust spatial-semantic features for object predictions, the formulation of a confidence function to rectify the classification score and alleviate the misalignment between the localization accuracy and classification confidence, and the distance-variant IoU-weighted NMS to obtain smoother results and avoid redundant (zero-IoU) false positives. The experimental results show that CIA-SSD achieves the state-of-the-art 3D detection performance on the official ranking metric (Moderate AP) for the KITTI benchmark, compared with all the existing single-stage detectors. Also, CIA-SSD attains real-time detection efficiency and runs the fastest, compared with the very recent state-of-the-art detectors.

Acknowledgement

We thank reviewers for the valuable comments. This work is funded by the Research Grants Council of the Hong Kong Special Administrative Region (Proj. no. CUHK 14201918).

References

Chen, X.; Ma, H.; Wan, J.; Li, B.; and Xia, T. 2017. Multi-View 3D Object Detection Network for Autonomous Driving. In *CVPR*, 1907–1915.

Chen, Y.; Liu, S.; Shen, X.; and Jia, J. 2019. Fast Point R-CNN. In *ICCV*, 9775–9784.

D., W. D.; W., M. I.; and L., S. R. 2008. Multivariate Probability Distributions. In *Mathematical Statistics with Applications*, 223–295. Belmont, California: Brooks/Cole, 7 edition.

Du, L.; Ye, X.; Tan, X.; Feng, J.; Xu, Z.; Ding, E.; and Wen, S. 2020. Associate-3Ddet: Perceptual-to-Conceptual Association for 3D Point Cloud Object Detection. In *CVPR*, 13329–13338.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision Meets Robotics: The KITTI Dataset. *The International Journal of Robotics Research* 32(11): 1231–1237.

Graham, B.; Engelcke, M.; and Van Der Maaten, L. 2018. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In *CVPR*, 9224–9232.

He, C.; Zeng, H.; Huang, J.; Hua, X.-S.; and Zhang, L. 2020. Structure Aware Single-stage 3D Object Detection from Point Cloud. In *CVPR*, 11873–11882.

Jiang, B.; Luo, R.; Mao, J.; Xiao, T.; and Jiang, Y. 2018. Acquisition of Localization Confidence for Accurate Object Detection. In *ECCV*, 784–799.

Kingma, D. P.; and Ba, J. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980* .

Ku, J.; Mozifian, M.; Lee, J.; Harakeh, A.; and Waslander, S. L. 2018. Joint 3D Proposal Generation and Object Detection from View Aggregation. In *IROS*, 1–8.

Lang, A. H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; and Beijbom, O. 2019. PointPillars: Fast Encoders for Object Detection from Point Clouds. In *CVPR*, 12697–12705.

Lehner, J.; Mitterecker, A.; Adler, T.; Hofmarcher, M.; Nessler, B.; and Hochreiter, S. 2019. Patch Refinement–Localized 3D Object Detection. *arXiv preprint arXiv:1910.04093* .

Li, J.; Luo, S.; Zhu, Z.; Dai, H.; Krylov, A. S.; Ding, Y.; and Shao, L. 2020. 3D IoU-Net: IoU Guided 3D Object Detector for Point Clouds. *arXiv preprint arXiv:2004.04962* .

Liang, M.; Yang, B.; Chen, Y.; Hu, R.; and Urtasun, R. 2019. Multi-task Multi-sensor Fusion for 3D Object Detection. In *CVPR*, 7345–7353.

Liang, M.; Yang, B.; Wang, S.; and Urtasun, R. 2018. Deep Continuous Fusion for Multi-sensor 3D Object Detection. In *ECCV*, 641–656.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal Loss for Dense Object Detection. In *ICCV*, 2980–2988.

Liu, B.; Wang, M.; Foroosh, H.; Tappen, M.; and Pensky, M. 2015. Sparse Convolutional Neural Networks. In *CVPR*, 806–814.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. SSD: Single Shot Multibox Detector. In *ECCV*, 21–37.

Liu, Z.; Zhao, X.; Huang, T.; Hu, R.; Zhou, Y.; and Bai, X. 2020. TANet: Robust 3D Object Detection from Point Clouds with Triple Attention. In *AAAI*, 11677–11684.

Loshchilov, I.; and Hutter, F. 2016. SGDR: Stochastic Gradient Descent with Warm Restarts. *arXiv preprint arXiv:1608.03983* .

Qi, C. R.; Liu, W.; Wu, C.; Su, H.; and Guibas, L. J. 2018. Frustum PointNets for 3D Object Detection from RGB-D Data. In *CVPR*, 918–927.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NeurIPS*, 5099–5108.

Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; and Li, H. 2020a. PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection. In *CVPR*, 10529–10538.

Shi, S.; Wang, X.; and Li, H. 2019. PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. In *CVPR*, 770–779.

Shi, S.; Wang, Z.; Shi, J.; Wang, X.; and Li, H. 2020b. 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* .

Shi, W.; and Rajkumar, R. 2020. Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud. In *CVPR*, 1711–1719.

Wang, Z.; and Jia, K. 2019. Frustum ConvNet: Sliding Frustums to Aggregate Local Point-Wise Features for Amodal 3D Object Detection. In *IROS*, 1742–1749.

Xie, L.; Xiang, C.; Yu, Z.; Xu, G.; Yang, Z.; Cai, D.; and He, X. 2020. PI-RCNN: An Efficient Multi-Sensor 3D Object Detector with Point-Based Attentive Cont-Conv Fusion Module. In *AAAI*, 12460–12467.

Yan, Y.; Mao, Y.; and Li, B. 2018. SECOND: Sparsely Embedded Convolutional Detection. *Sensors* 18(10): 3337.

Yang, Z.; Sun, Y.; Liu, S.; and Jia, J. 2020. 3DSSD: Point-based 3D Single Stage Object Detector. In *CVPR*, 11040–11048.

Yang, Z.; Sun, Y.; Liu, S.; Shen, X.; and Jia, J. 2019. STD: Sparse-to-Dense 3D Object Detector for Point Cloud. In *ICCV*, 1951–1960.

Yoo, J. H.; Kim, Y.; Kim, J. S.; and Choi, J. W. 2020. 3D-CVF: Generating Joint Camera and LiDAR Features Using Cross-View Spatial Feature Fusion for 3D Object Detection. In *ECCV*, 720–736.

Zhou, D.; Fang, J.; Song, X.; Guan, C.; Yin, J.; Dai, Y.; and Yang, R. 2019. IoU Loss for 2D/3D Object Detection. In *3DV*, 85–94.

Zhou, Y.; and Tuzel, O. 2018. Voxelnet: End-to-end Learning for Point Cloud Based 3D Object Detection. In *CVPR*, 4490–4499.