# A More Implementation Details

Our experiment sets is based on OpenPCDet [9].

## A.1 Range and Voxelization Settings

**KITTI.** The input point cloud is cut by [0, 70.4$m$] in X axis, [-40$m$, 40$m$] in Y axis, [-3$m$, 1$m$] in Z axis in the KITTI [4] dataset. The voxelization size is set to (0.05$m$, 0.05$m$, 0.1$m$) for SECOND [10], Voxel RCNN [3] and PV RCNN [8].

**nuScenes.** On the nuScenes [1] dataset, the input point cloud is clipped to [-51.2$m$, 51.2$m$] in X axis, [-51.2$m$, 51.2$m$] in Y axis, [-5$m$, 3$m$] in Z axis. The voxelization size is set to (0.1$m$, 0.1$m$, 0.2$m$) for both SECOND [10] and CenterPoint [11].

## A.2 Data Augmentations Settings

Following the above approachs [3, 8] sets, we use random flipping, global scaling, global rotation, and ground-truth (GT) sampling [10] in KITTI and nuScenes, and additional translation in nuScenes. Random flipping is used along X axis for KITTI and both X and Y axis for nuScenes with a random factor from [0.95, 1.05]. Random global rotation is added between $[-\pi/2, \pi/2]$ for KITTI and $[-\pi/4, \pi/4]$ for nuScenes. GT sampling copies and pastes annotated objects from one frame to another frame.

## A.3 Training Settings

The models are optimized by AdamW [7] optimizer with max learning rate 0.003, weight decay 0.01, and momentum 0.85 to 0.95 in KITTI and nuScenes. We train the models with 80 epochs in KITTI, 20 epochs for SECOND and CenterPoint in nuScenes. The batch size is 2 for Voxel RCNN and PV RCNN, 4 for SECOND and CenterPoint. The random seed is fixed as 666 to exclude its affect in all experiments.

## A.4 3D Backbone Network Settings

For all the methods, the channels of convolution in each block are [16, 32, 64, 64]. And every convolutions include a convulotion layer (sparse convolution, submanifold sparse convolution or drop sparse convolution), a batch normalization layer [5] and a ReLU activation layer.

## A.5 Focal Loss Settings

We use the $\alpha - balanced$ variant of the focal loss [6] in section 3.2 of main paper, which is definied as

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}log(p_t), \tag{A-1}$$

$$p_t = \begin{cases} p & y = 1 \\ 1-p & otherwise \end{cases}, \tag{A-2}$$

where, $p \in [0, 1]$ is the modal's estimated probability for the class with label $y = 1$, $y \in \{-1, +1\}$ is the ground-truth class, $\alpha_t\ in[0, 1]$ is a balanced weighting factor, $\gamma$ is focusing parameter. We set voxels in ground bounding box as foreground, and others as background. The parameters $\alpha$ and $\gamma$ are set as 0.25 and 2.0, respectively.

# B More Ablation Study

## B.1 The logic of Threshold $\theta_p$.

The logic of the threshold $\theta_p$ appears in section 3.2. Two types of logic are proposed. The first method is to use The original value of $\theta_p$ as the threshold and is called the value-based method. The second method is to use The proportional value of $\theta_p$ as the threshold and is referred to as proportion-based method. The results are shown in the Table. A-1. The proportion-based method is better than the value-based method because the features deleted by the proportional threshold are more stable.

Table A-1: Ablations on $\theta_p$ of SECOND in $AP_{3D}(R40)$ on KITTI validation set.

| Method | $\theta_p$ | $Car$ | | | $Ped.$ | $Cyc.$ |
|---|---|---|---|---|---|---|
| | | $Easy$ | $Mod.$ | $Hard$ | $Mod.$ | $Mod.$ |
| SECOND | - | 90.46 | 81.66 | 78.67 | 50.86 | **68.41** |
| $+Drop\ Conv$ | Value | 89.77 | 81.29 | 78.55 | 50.87 | 68.16 |
| | Proportion | 90.64 | **81.94** | 79.08 | **51.13** | 63.02 |

Table A-2: Ablations on $\theta_p$ of SECOND, Voxel RCNN and PV RCNN in $AP_{3D}(R40)$ on KITTI validation set.

| Method | $\theta_p$ | $Car$ | | | $Ped.$ | $Cyc.$ |
|---|---|---|---|---|---|---|
| | | $Easy$ | $Mod.$ | $Hard$ | $Mod.$ | $Mod.$ |
| SECOND | - | 90.46 | 81.66 | 78.67 | 50.86 | **68.41** |
| $+Drop\ Conv$ | 0.9 | 90.27 | 81.61 | 78.78 | **51.13** | 66.87 |
| | 0.95 | 90.64 | **81.94** | 79.08 | **51.13** | 63.02 |
| | 0.99 | 88.66 | 81.26 | 78.30 | 50.12 | 66.69 |
| Voxel RCNN | - | 92.45 | 85.20 | 82.81 | - | - |
| $+Drop\ Conv$ | 0.7 | 92.09 | 85.10 | 82.64 | - | - |
| | 0.8 | 92.52 | **85.37** | 82.92 | - | - |
| | 0.9 | 92.52 | 85.00 | 82.68 | - | - |
| PV RCNN | - | 92.31 | 84.76 | 82.53 | 56.46 | 71.73 |
| $+Drop\ Conv$ | 0.6 | 91.41 | 84.37 | 82.23 | 48.60 | **72.53** |
| | 0.7 | 92.05 | **84.88** | 82.56 | **60.47** | 70.98 |
| | 0.8 | 91.81 | 84.26 | 82.38 | 58.32 | 71.03 |

### B.2 The Value of threshold $\theta_p$.

$\theta_p$ appears in the section 3.2 valid positions map. Through $\theta_p$ can extract valid positions and delete invalid positions. We set $\theta_p$ with 0.9, 0.95 and 0.99, then train the network respectively. The results are shown in the Table.A-2. The combination effect of SECOND and $Drop\ Conv$ is the best, while the $\theta_p$ is 0.95. In the case of 0.9 and 0.99, the effect is poor, even lower than the baseline result. So this parameter is very important for $Drop\ Conv$. Compared with SECOND, it has an increase of 0.28% in moderate car detection with $Drop\ Conv$. It is notable that we find the effects of different networks are sensitive to the parameter. We take Voxel RCNN as the baseline and carry out the same experiment. The results are shown in the Table. A-2. The optimal $\theta_p$ is 0.8 for Voxel RCNN. It has an increase of 0.17% in moderate car detection with $Drop\ Conv$. A reasonable explanation is that SECOND is simpler than Voxel RCNN. Too many deleted voxels will lead to features loss. The results show that PV RCNN has bigger $\theta_p$ than SECOND and Voxel RCNN. This also proves that more complex networks have bigger optimal $\theta_p$. The complex networks surport dilation features delection.

### B.3 Add or Replace

When designing, we considered whether to add or replace the convolution layers of blocks. A comparative experiment is used to select. For the approach of add, one layer $Drop\ Conv$ is added behind the first, second, and third blocks. For the approach of replace, the $Drop\ Conv$ is used to replace the first submanifold sparse convolution of the first, second, and third blocks. The results are shown in the Table.A-3. We find that replacing the first submanifold sparse convolution can achieve better results than adding one layer in the end. Adding a layer actually increases the depth of the network. This may cause the network over fit. Replacing the middle layer of convolution blocks still maintains the network structure, while the $Drop\ Conv$ can also inhibit feature dilation, so as to achieve better results.

Table A-3: Ablations on add or replace in $AP_{3D}(R40)$ on KITTI validation set.

| Method | Forms | $Car$ | | | $Ped.$ | $Cyc.$ |
|---|---|---|---|---|---|---|
| | | $Easy$ | $Mod.$ | $Hard$ | $Mod.$ | $Mod.$ |
| SECOND | - | 90.46 | 81.66 | 78.67 | 50.86 | **68.41** |
| $+Drop\ Conv$ | Add | 90.64 | 81.52 | 78.46 | 50.75 | 62.02 |
| | Replace | 90.64 | **81.94** | 79.08 | **51.13** | 63.02 |

## B.4 Blocks with Drop Sparse Convolution

In order to verify which blocks need to be modified by $Drop\ Conv$, we designed an experiment to find the optimal modification. The blocks structure is shown as Fig.2 in the main paper. We set the first, the first two, the first three, and all blocks to replace $Drop\ Conv$. The results are shown in the Table.A-4. We find that when only the first block is replaced, the detection results significantly deteriorated. However, when replacing the first two blocks, the detection effect will return to the same as baseline. After replacing the first three blocks, the detection effect is optimal. When all blocks are replaced, the detection performance of the network decreases instead. This may be because replacing only the first blocks results in less useful information. By the time of the last block, the submanifold dilation had led to inaccurate positioning and limited improvements could be made. So, it is important to choose a reasonable use blocks for $Drop\ Conv$.

Table A-4: Ablations on replace blocks in $AP_{3D}(R40)$ on KITTI validation set.

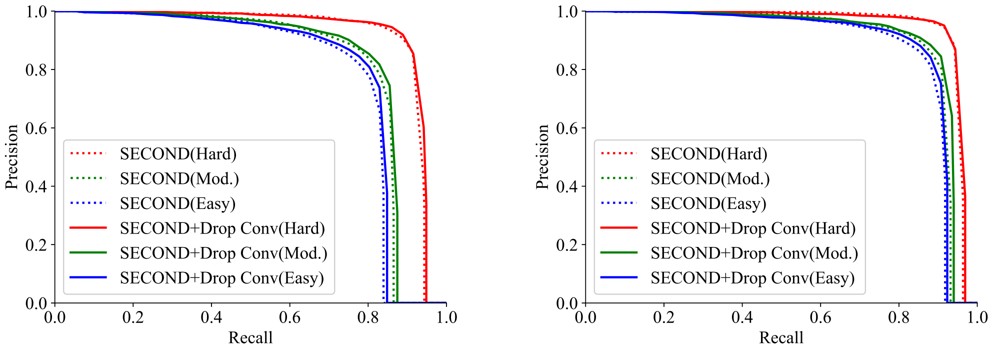
| Method | blocks | $Car$ | | | $Ped.$ | $Cyc.$ |
|---|---|---|---|---|---|---|
| | | $Easy$ | $Mod.$ | $Hard$ | $Mod.$ | $Mod.$ |
| SECOND | - | 90.46 | 81.66 | 78.67 | 50.86 | **68.41** |
| $+Drop\ Conv$ | [1] | 90.64 | 81.31 | 78.50 | 49.24 | 64.18 |
| | [1,2] | 90.60 | 81.65 | 78.64 | **51.43** | 68.06 |
| | [1,2,3] | 90.64 | **81.94** | 79.08 | 51.13 | 63.02 |
| | [1,2,3,4] | 90.86 | 81.50 | 78.55 | 49.53 | 62.13 |

## B.5 Size of Encoded Features

The encoded feature is proposed in the section 3.1 feature encoding. The output features can better represent the position of valid features by feature encoding. We design a ablation experiment with 64, 128 and 256 encoding feature output sizes, and other parameter settings are the same as the original. The results are shown in the Table.A-5. We find that when the encoded feature has output size with 64, $Drop\ Conv$ has a significant improvement, indicating the improvement effect of $Drop\ Conv$ on the network. When the encoded feature output size is increased to 128, the detection result reaches the optimal. However, continuing to increase the encoded feature output size to 256 will instead result in a lower detection effect. This maybe too many coding features make network difficult to learn parameters. Although encoded features output size seem to have little impact on $Drop\ Conv$, an appropriate encoded features output size can still significantly improve the effect of $Drop\ Conv$.

Table A-5: Ablations on encoding size in $AP_{3D}(R40)$ on KITTI validation set.

| Method | Size | $Car$ | | | $Ped.$ | $Cyc.$ |
|---|---|---|---|---|---|---|
| | | $Easy$ | $Mod.$ | $Hard$ | $Mod.$ | $Mod.$ |
| SECOND | - | 90.46 | 81.66 | 78.67 | 50.86 | **68.41** |
| $+Drop\ Conv$ | 64 | 90.73 | 81.86 | 78.73 | 50.86 | 65.76 |
| | 128 | 90.64 | **81.94** | 79.08 | 51.13 | 63.02 |
| | 256 | 88.79 | 81.40 | 78.43 | **51.22** | 65.32 |



(a) 3D

(b) Bird's eye view

Figure A-1: The PR curve of SECOND on KITTI validation set. The dashed line represents the results of SECOND on hard, moderate and easy difficulties. The solid line represents the results of SECOND+$Drop\ Conv$ on the three difficulties.

### B.6 Runtime

The runtime of baselines and the proposed method is shown as Table. A-6. The inference is test on on a single NVIDIA GeForce 3090 GPU with CUDA 11.1 and SpConv [2] 2.1.25. The inference time only increase a little time, cause the focal loss do not work in inference stage which save many times. The inference time of PV RCNN only adds 0.2 $ms$, because its $\theta_p$ is 0.7, which make convolution features simpler.

Table A-6: Ablations on inference time with KITTI validation set.

| Runtime $(ms)$ | Baseline | +$Drop\ Conv$ |
|---|---|---|
| SECOND | 29.0 | 30.6 |
| Voxel RCNN | 39.4 | 41.2 |
| PV RCNN | 56.4 | 56.6 |

## C More Visualization

### C.1 PR Curves

We visualized the precision recall (PR) curve on the KITTI validation set using SECOND and SECOND+$Drop\ Conv$. As shown in the Fig. A-1. It can be seen that as the recall increases, the precision of the baseline decreases faster. This indicates that negative samples have a greater impact on the baseline and have a negative impact on performance. After using $Drop\ Conv$, the dilation during sparse convolution was suppressed and the detection precision was improved.

### C.2 Qualitative Results

The visualization result is provided to verify the advantages of the proposed method, we use Voxel RCNN as a baseline to visualize the improvements brought by $Drop\ Conv$ in different scenes. The results are shown in the Fig.A-2. As we mentioned earlier, we hope to reduce submanifold dilation through $Drop\ Conv$ to improve positioning accuracy. This can be clearly seen in the red dotted line in scene (a). In the Voxel RCNN results, the positioning of the vehicle is not accurate, and there is a significant deviation in the vehicle orientation. However, after adding $Drop\ Conv$, the dilated features are suppressed and the features interfering with localization are reduced, resulting in more accurate localization of the detection network. However, the results of Voxel RCNN are also counted as meeting the standard, so although the proposed method corrects vehicle positioning, the metrics improvement is not significant. The improvement brought by the proposed method can be clearly

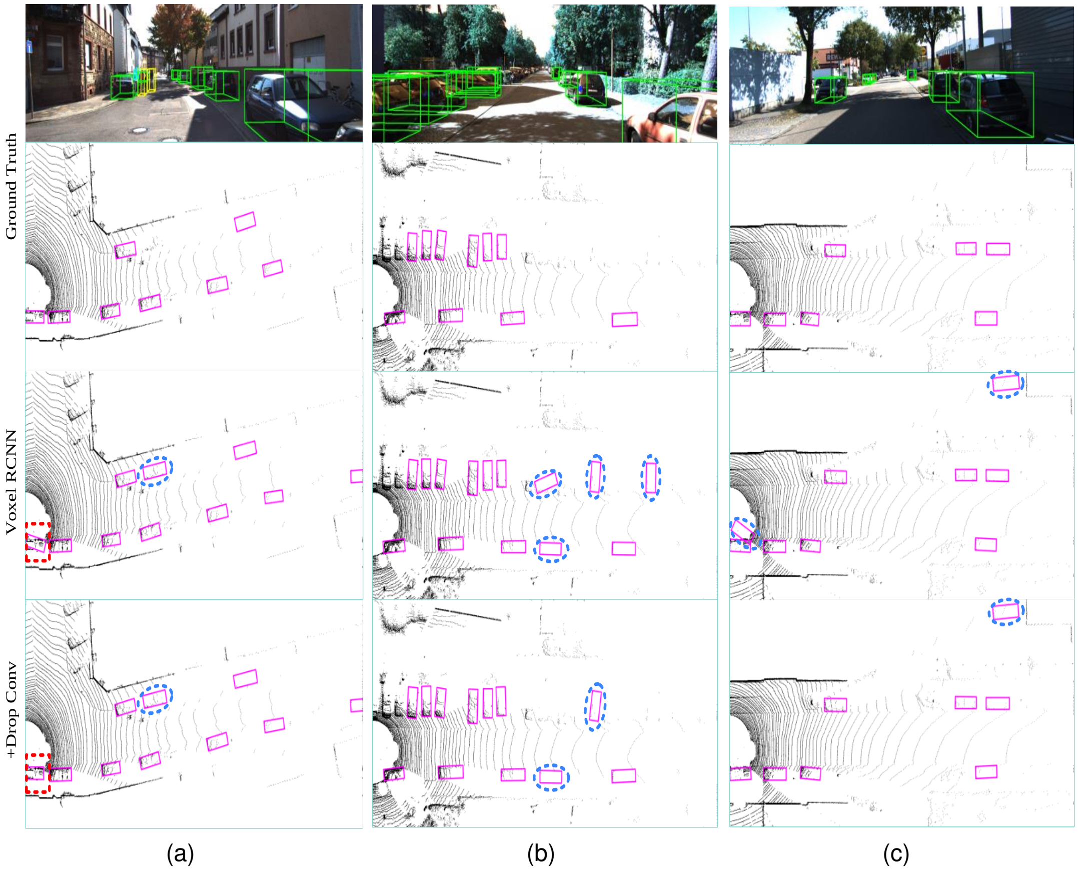


Figure A-2: The result of Voxel RCNN in different scenes. The top row is the pictures with ground truth. The second row is the point cloud with ground truth in BEV. The third row is is the point cloud with prediction of Voxel RCNN in BEV. The last row is is the point cloud with prediction of Voxel RCNN and $Drop\ Conv$ in BEV. The green lines are bounding box. The areas circled by red dotted lines show position optimized with $Drop\ Conv$. The areas circled by blue dotted lines show the wrong results.

seen in the remaining scenes. In different scenes, the proposed method can suppress some false detection objects. This is because after the submanifold dilation is suppressed, some similar objects cannot expand into detectable features, significantly improving the effectiveness of the network.

In order to show more vislization details, we add the qualitative results of SECOND and PV RCNN in KITTI validation set. The results is shown as Fig. A-3 and Fig. A-4. The proposed method has significant improvements on car class to position objects. But the improvement effect on small object detection is not satisfactory.

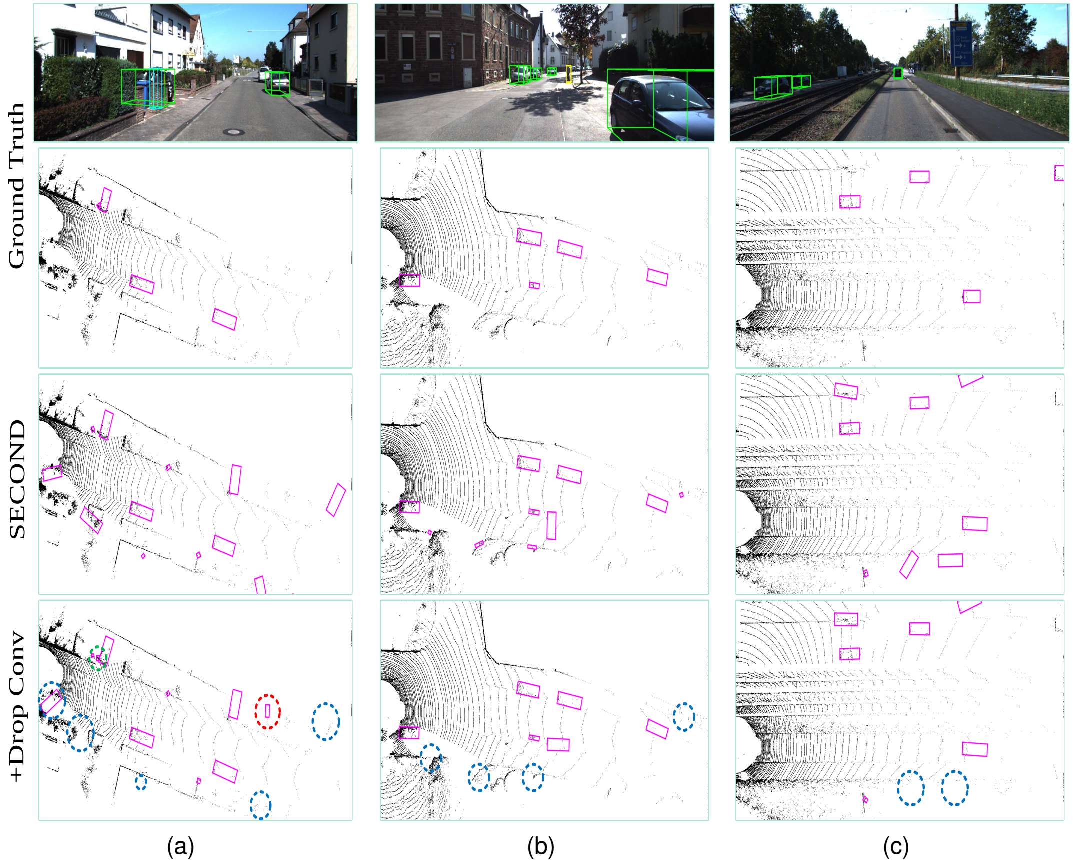


Figure A-3: The result of SECOND in different scenes. The top row is the pictures with ground truth. The second row is the point cloud with ground truth in BEV. The third row is is the point cloud with prediction of Voxel RCNN in BEV. The last row is is the point cloud with prediction of Voxel RCNN and *Drop Conv* in BEV. The green lines are bounding box. The areas circled by red dotted lines show position optimized with *Drop Conv*. The areas circled by blue dotted lines show baseline wrong detection but the proposed method correct. The areas circled by green dotted lines show baseline not detection but the proposed method correct. The areas circled by red dotted lines show baseline right detection but the proposed method wrong.

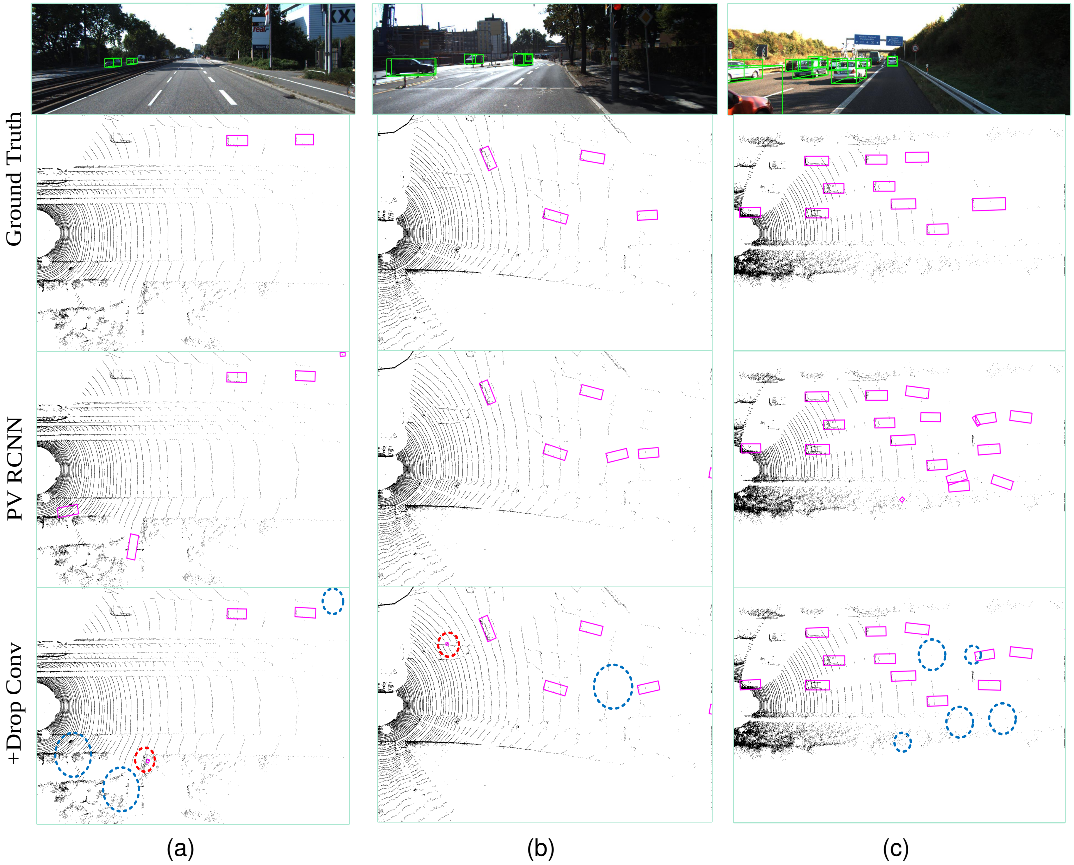


Figure A-4: The result of PV RCNN in different scenes. The top row is the pictures with ground truth. The second row is the point cloud with ground truth in BEV. The third row is is the point cloud with prediction of Voxel RCNN in BEV. The last row is is the point cloud with prediction of Voxel RCNN and $Drop\ Conv$ in BEV. The green lines are bounding box. The areas circled by red dotted lines show position optimized with $Drop\ Conv$. The areas circled by blue dotted lines show baseline wrong detection but the proposed method correct. The areas circled by green dotted lines show baseline not detection but the proposed method correct. The areas circled by red dotted lines show baseline right detection but the proposed method wrong.

# 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] Spconv Contributors. Spconv: Spatially sparse convolution library. `https://github.com/traveller59/spconv`, 2022.

[3] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 1201–1209, 2021.

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In Conference on Computer Vision and Pattern Recognition (CVPR), 2012.

[5] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning, pages 448–456. pmlr, 2015.

[6] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision, pages 2980–2988, 2017.

[7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101, 2017.

[8] 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.

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

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

[11] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 11784–11793, 2021.