# DBQ-SSD: DYNAMIC BALL QUERY FOR EFFICIENT 3D OBJECT DETECTION

**Anonymous authors**
Paper under double-blind review

## 1 LIMITATION AND FUTURE WORK

With increasing the supervision on resource budget (increasing the value of $\lambda$.), the performance will decrease accordingly. We suspect that dropping too many point clouds may eliminate the part of the useful point cloud. Therefore, this paper targets to achieve a better trade-off between accuracy and inference speed, which maintaining or even achieving gain in accuracy, and significantly speeding up inference. In this paper, we can not specifying what point to drop but our detector equips strong ability to eliminate redundancy. Therefore, we look forward to inspire future works for focusing on dropping more redundant point cloud without performance degradation.

Table 1: Illustration of the architecture of DBQ-SSD. npoint denotes the number of sampled points, [radii] denotes the grouping radii, [nquery] denotes the number of grouping points, [dimension] denotes the feature dimensions. Aggregation indicates aggregation operation and MLP size.

| Module | npoint | [radii] | [nquery] | [dimension] | Aggregation |
|---|---|---|---|---|---|
| SA layer | 4096 | [0.2, 0.8] | [16, 32] | [[16, 16, 32], [32, 32, 64]] | MLP ($32 \rightarrow 64$) + MLP ($64 \rightarrow 64$) |
| SA layer | 1024 | [0.8, 1.6] | [16, 32] | [[64, 64, 128], [64, 96, 128]] | MLP ($128 \rightarrow 128$) + MLP ($128 \rightarrow 128$) |
| SA layer | 512 | [1.6, 4.8] | [16, 32] | [[128, 128, 256], [128, 256, 256]] | MLP ($256 \rightarrow 256$) + MLP ($256 \rightarrow 256$) |
| Vote layer | 256 | - | - | - | MLP ($256 \rightarrow 128 \rightarrow 3$) |
| SA layer | 256 | [4.8, 6.4] | [16, 32] | [[256,256,512], [256,512,1024]] | MLP ($512 \rightarrow 512$) + MLP ($1024 \rightarrow 512$) |

## 2 DETAILED DETECTOR ARCHITECTURE

We report the detailed architecture of our DBQ-SSD. DBQ-SSD is a single-stage point-based detector that consists of three Set Abstraction (SA) layers for extracting point features, and one SA layer for aggregating centroid-based instances. Each SA layer has two different groups for the spherical neighbor query. In addition, a vote layer is used to generate candidate points. A light-weight head is attached to the backbone to predict final results. The detailed architecture for KITTI is reported in Tab. 1.

The head consists of two parallel branches, *i.e.*, classification and regression branches. The corresponding architecture:

$$\textit{Classification branch: FC(}512) \rightarrow \text{FC}(256) \rightarrow \text{FC}(256) \rightarrow \text{FC}(3)$$

$$\textit{Regression branch: FC(}512) \rightarrow \text{FC}(256) \rightarrow \text{FC}(256) \rightarrow \text{FC}(30)$$

where the classification branch predicts 3 classes, while the regression branch predicts 12 classes of equally angle bins and their corresponding angle offsets, and the distance $(d_x, d_y, d_z)$ to its corresponding instance, as well as the size $(d_l, d_w, d_h)$. For Waymo scene, we use the same model setting and just adjust the scale of input point cloud to 16,384, 4,096, 2,048 and 1,024.

## 3 EXPERIMENTS ON TITTI *test* AND ONCE *val* SET

To verify the generalization, we evaluate our method on both KITTI *test* set and ONCE *val* set.

**KITTI *test* set.** As shown in Tab. 2, our DBQ-SSD achieves comparable performance compared with IA-SSD, while showing super inference speed nearly two times than IA-SSD.

**ONCE *val* set.** Because the official configuration file of IA-SSD is not released with respect to ONCE dataset, we reproduce the results according to the paper. As shown in Tab. 3, our method significantly improves the inference speed to **33 FPS (2.4x speedup)**, while maintaining comparable performance with IA-SSD. When adjusting the $\gamma$ to 0.1, our method achieves nearly **1 mAP** performance improvement while gaining **1.7x speedup**.

Table 2: Comparison with the state-of-the-art methods on the KITTI *test* set. Bold font is used to indicate the best performance. The speed is tested on a single GPU with with batch size of 16 and measured by FPS.

| Method | Type | 3D Car (IoU=0.7) | | | 3D Ped. (IoU=0.5) | | | 3D Cyc. (IoU=0.5) | | | Speed |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Easy | Mod. | Hard | Easy | Mod. | Hard | Easy | Mod. | Hard | |
| Voxel-based Methods | | | | | | | | | | | |
| VoxelNet | 1-stage | 77.47 | 65.11 | 57.73 | 39.48 | 33.69 | 31.5 | 61.22 | 48.36 | 44.37 | 4.5 |
| SECOND | 1-stage | 84.65 | 75.96 | 68.71 | 45.31 | 35.52 | 33.14 | 75.83 | 60.82 | 53.67 | 20 |
| PointPillars | 1-stage | 82.58 | 74.31 | 68.99 | 51.45 | 41.92 | 38.89 | 77.10 | 58.65 | 51.92 | 42.4 |
| 3D IoU Loss | 1-stage | 86.16 | 76.50 | 71.39 | - | - | - | - | - | - | 12.5 |
| Associate-3Ddet | 1-stage | 85.99 | 77.40 | 70.53 | - | - | - | - | - | - | 20 |
| SA-SSD | 1-stage | 88.75 | 79.79 | 74.16 | - | - | - | - | - | - | 25 |
| CIA-SSD | 1-stage | 89.59 | 80.28 | 72.87 | - | - | - | - | - | - | 32 |
| TANet | 2-stage | 84.39 | 75.94 | 68.82 | 53.72 | **44.34** | 40.49 | 75.70 | 59.44 | 52.53 | 28.5 |
| Part-$A^2$ | 2-stage | 87.81 | 78.49 | 73.51 | 53.10 | 43.35 | 40.06 | **79.17** | 63.52 | 56.93 | 12.5 |
| Point-Voxel Methods | | | | | | | | | | | |
| Fast Point R-CNN | 2-stage | 89.29 | 77.40 | 70.24 | - | - | - | - | - | - | 16.7 |
| STD | 2-stage | 87.95 | 79.71 | 75.09 | 53.29 | 42.47 | 38.35 | 78.69 | 61.59 | 55.30 | 12.5 |
| PV-RCNN | 1-stage | **90.25** | **81.43** | **76.82** | 52.17 | 43.29 | 40.29 | 78.60 | 63.71 | **57.65** | 12.5 |
| VIC-Net | 1-stage | 88.25 | 80.61 | 75.83 | 43.82 | 37.18 | 35.35 | 78.29 | 63.65 | 57.27 | 17 |
| HVPR | 1-stage | 86.38 | 77.92 | 73.04 | 52.47 | 43.96 | **40.64** | - | - | - | 36.1 |
| Point-based Methods | | | | | | | | | | | |
| PointRCNN | 2-stage | 86.96 | 75.64 | 70.70 | 47.98 | 39.37 | 36.01 | 74.96 | 58.82 | 52.53 | 10 |
| 3D IoU-Net | 2-stage | 87.96 | 79.03 | 72.78 | - | - | - | - | - | - | 10 |
| Point-GNN | 1-stage | 88.33 | 79.47 | 72.29 | 51.92 | 43.77 | 40.14 | 78.60 | 63.48 | 57.08 | 1.6 |
| 3DSSD | 1-stage | 88.36 | 79.57 | 74.55 | **54.64** | 44.27 | 40.23 | 82.48 | **64.10** | 56.90 | 25 |
| IA-SSD | 1-stage | 88.34 | 80.13 | 75.04 | 46.51 | 39.03 | 35.60 | 78.35 | 61.94 | 55.70 | 83 |
| IA-SSD (Reproduced) | 1-stage | 87.67 | 79.40 | 74.22 | 46.16 | 38.29 | 35.61 | 78.26 | 61.53 | 55.48 | 83 |
| DBQ-SSD | 1-stage | 87.93 | 79.39 | 74.40 | 47.59 | 38.08 | 35.61 | 78.18 | 62.80 | 55.70 | **162** |

Table 3: Comparison with the state-of-the-art methods on the ONCE *val* set. Bold font is used to indicate the best performance. The speed is tested on a single GPU with batch size of 16 and measured by FPS.

| Method | Vehicle | | | | Pedestrian | | | | Cyclist | | | | mAP | Speed |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Overall | 0-30m | 30-50m | ¿50m | Overall | 0-30m | 30-50m | ¿50m | Overall | 0-30m | 30-50m | ¿50m | | |
| PointPollars | 68.57 | 80.86 | 62.07 | 47.04 | 17.63 | 19.74 | 15.15 | 10.23 | 46.81 | 58.33 | 40.32 | 25.86 | 44.34 | - |
| SECOND | 71.19 | 84.04 | 63.02 | 47.25 | 26.44 | 29.33 | 24.05 | 18.05 | 58.04 | 69.96 | 52.43 | 34.61 | 51.89 | - |
| PV-RCNN | **77.77** | **89.39** | **72.55** | **58.64** | 23.50 | 25.61 | 22.84 | 17.27 | 59.37 | 71.66 | 52.58 | 36.17 | 53.55 | - |
| PointRCNN | 52.09 | 74.45 | 40.89 | 16.81 | 4.28 | 6.17 | 2.40 | 0.91 | 29.84 | 46.03 | 20.94 | 5.46 | 28.74 | - |
| IA-SSD | 70.30 | 83.01 | 62.84 | 47.01 | **39.82** | **47.45** | 32.75 | 18.99 | 62.17 | 73.78 | 56.31 | **39.53** | 57.43 | 14 |
| IA-SSD (Reproduced) | 70.48 | 84.16 | 63.77 | 49.27 | 38.22 | 44.14 | 33.10 | 20.41 | 61.90 | 73.94 | 55.44 | 38.37 | 56.87 | 14 |
| DBQ-SSD ($\lambda$=0.05) | 72.06 | 84.63 | 64.66 | 50.13 | 38.32 | 43.35 | 32.97 | 21.22 | 62.16 | 73.94 | 56.65 | 38.20 | 57.51 | 23 |
| DBQ-SSD ($\lambda$=0.10) | 72.14 | 84.81 | 64.27 | 50.22 | 37.83 | 43.88 | 32.18 | 20.29 | **62.99** | **75.13** | 56.65 | 38.91 | **57.65** | 24 |
| DBQ-SSD ($\lambda$=0.20) | 71.63 | 84.38 | 64.06 | 49.82 | 37.27 | 41.90 | **33.59** | **20.95** | 62.77 | 74.94 | **57.14** | 38.47 | 57.22 | 27 |
| DBQ-SSD ($\lambda$=0.30) | 70.66 | 83.28 | 63.66 | 48.88 | 37.46 | 42.35 | 32.94 | 22.21 | 62.51 | 74.46 | 56.65 | 38.01 | 56.88 | **33** |

## 4 VISUALIZATION

As shown in Fig. 1, Fig. 2, and Fig. 3, we provide the detailed visualization of predicted results for Waymo *val* set and KITTI *val* set. The conclusion is the same as KITTI scene. As the network depth increases, the foreground points are retained for classification and regression, while redundant background points are dropped. It reveals that our method can adaptively discard useless points for speeding up inference. It's worth noting that the discarding behavior of point clouds significantly differs between KITTI and Waymo scenes, which verifies that our method equips generalization.


Figure 1: Visualization results on Waymo *val* set. The red and green 3D boxes in figures are ground truth and prediction boxes. Green, cyan, and yellow represent *Car, Pedestrian, and Cyclist*. Red and white points represent activation and blocking points, respectively. "Small" and "Large" means the scale of group in MSG, and the digital in parentheses is the index of SA layer.

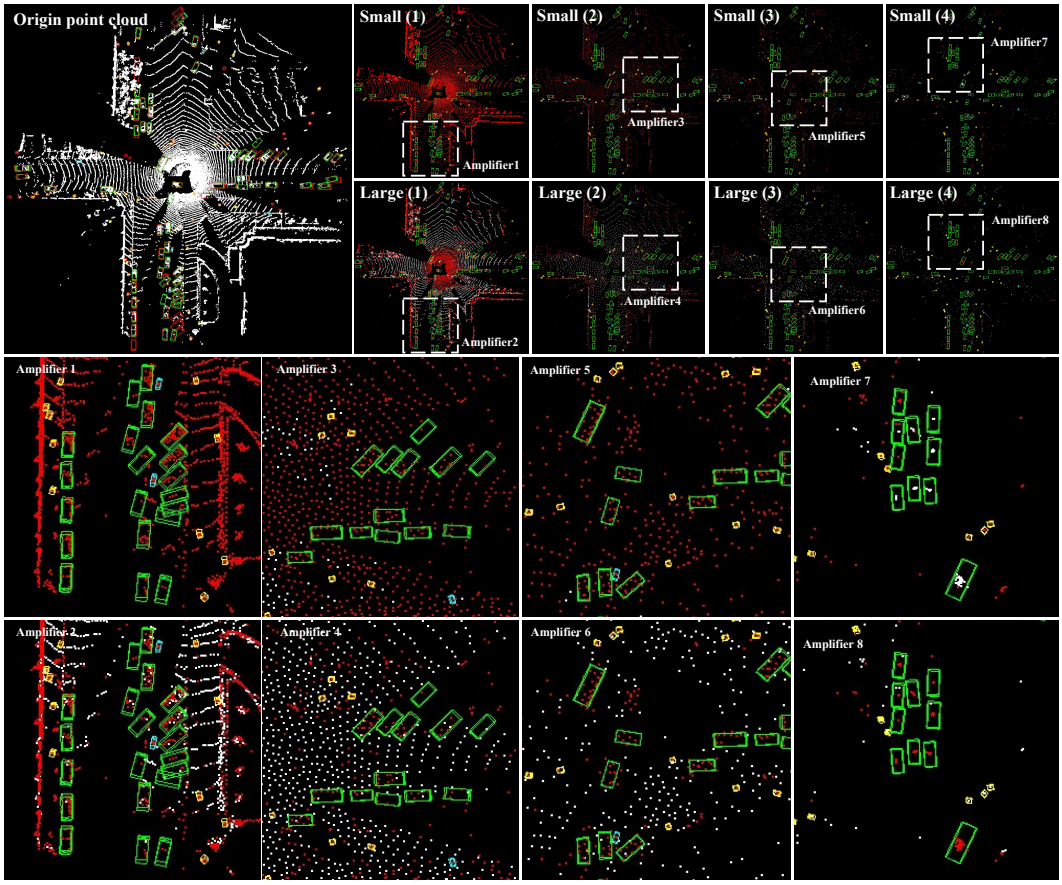


Figure 2: Another visualization results on Waymo *val* set.

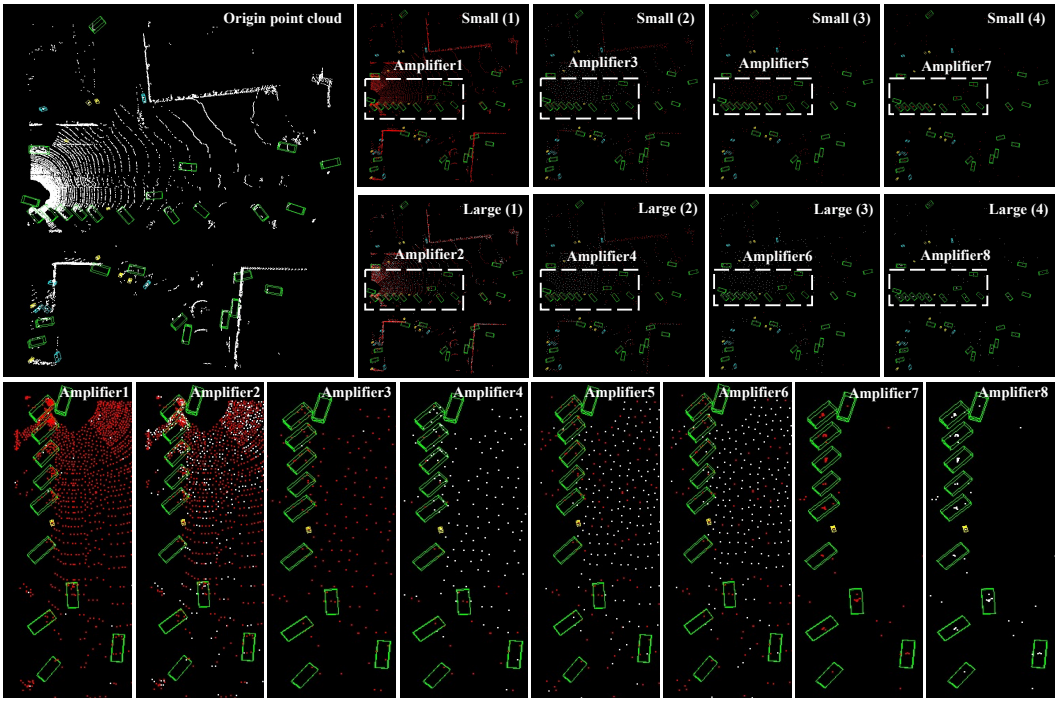


Figure 3: Detail visualization results on KITTI *val* set. The red and green 3D boxes in figures are ground truth and prediction boxes. Green, cyan, and yellow represent *Car, Pedestrian, and Cyclist*. Red and white points represent activation and blocking points, respectively. "Small" and "Large" means the scale of group in MSG, and the digital in parentheses is the index of SA layer.