DBQ-SSD: Dynamic Ball Query for Efficient 3D Object Detection

Jinrong Yang · *Lin Song* · *Songtao Liu* · *Weixin Mao*
Zeming Li · *Xiaoping Li* · *Hongbin Sun* · *Jian Sun* · *Nanning Zheng*

International Conference on Learning Representation (ICLR)
May 1st – 5th, 2023

Introduction

Background

- *Due to sparse, unordered, and semantically deficient of point cloud in autonomous driving, processing raw data is cumbersome and costly.*

Figure from WOD

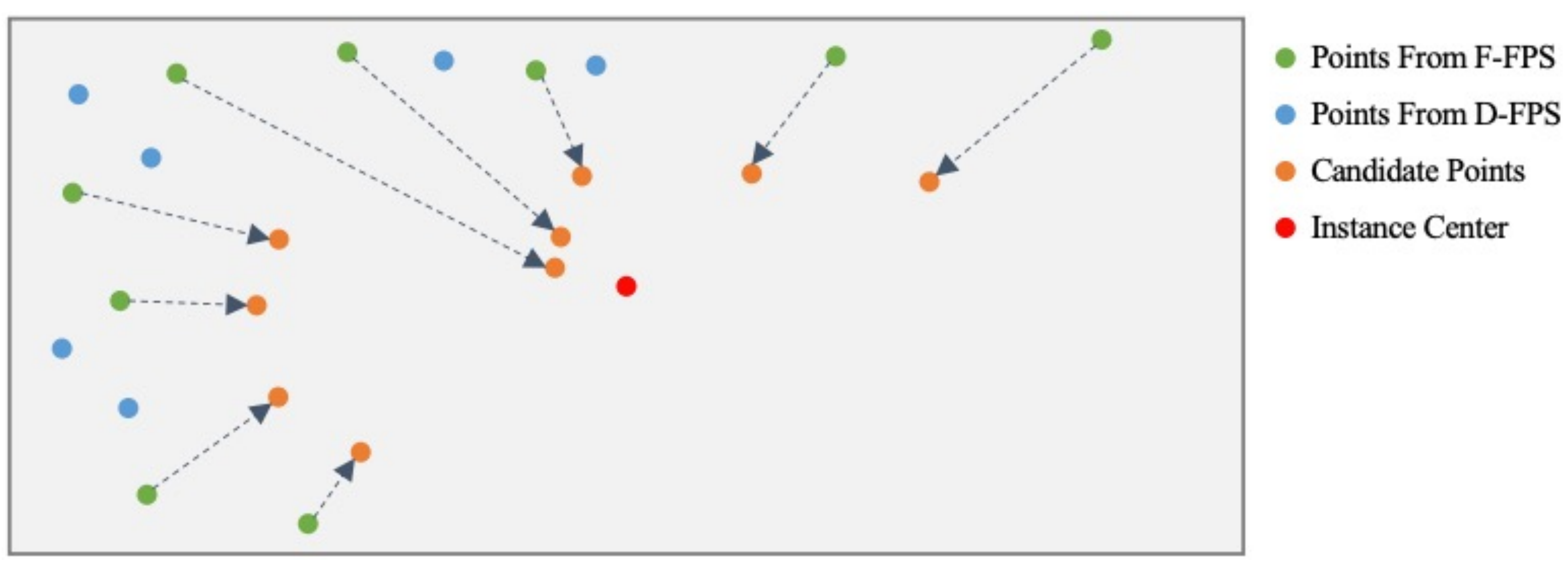
Introduction

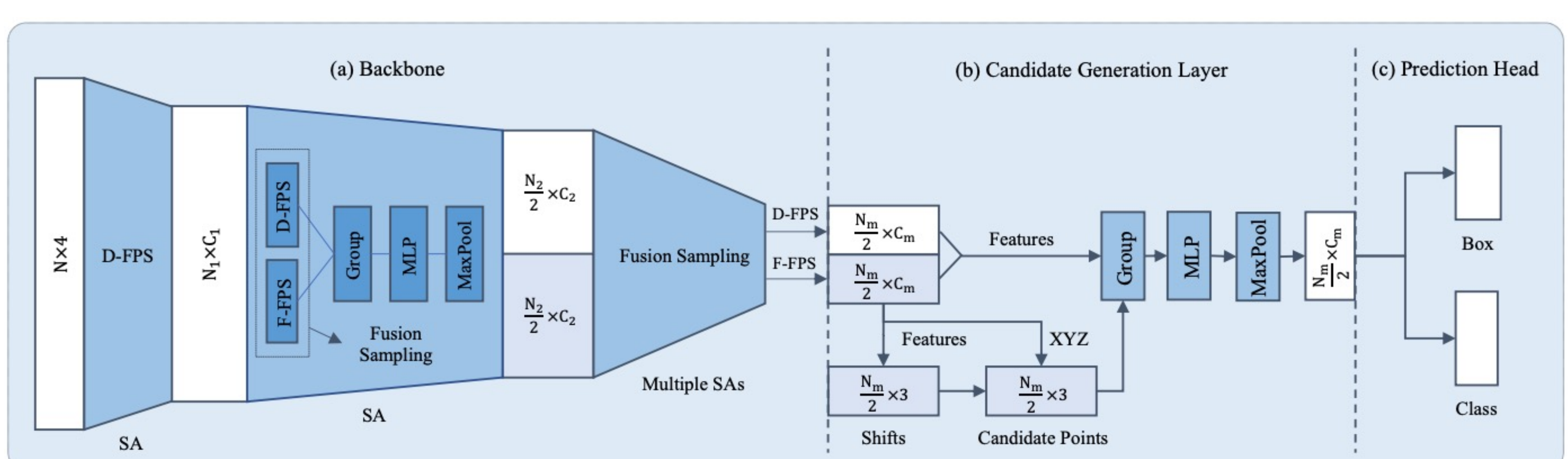
Existing methods

- *Employing the PointNet++ framework, many single-stage point-based 3D detectors are proposed to efficient process point cloud and extract informative point feature.*

3DSSD

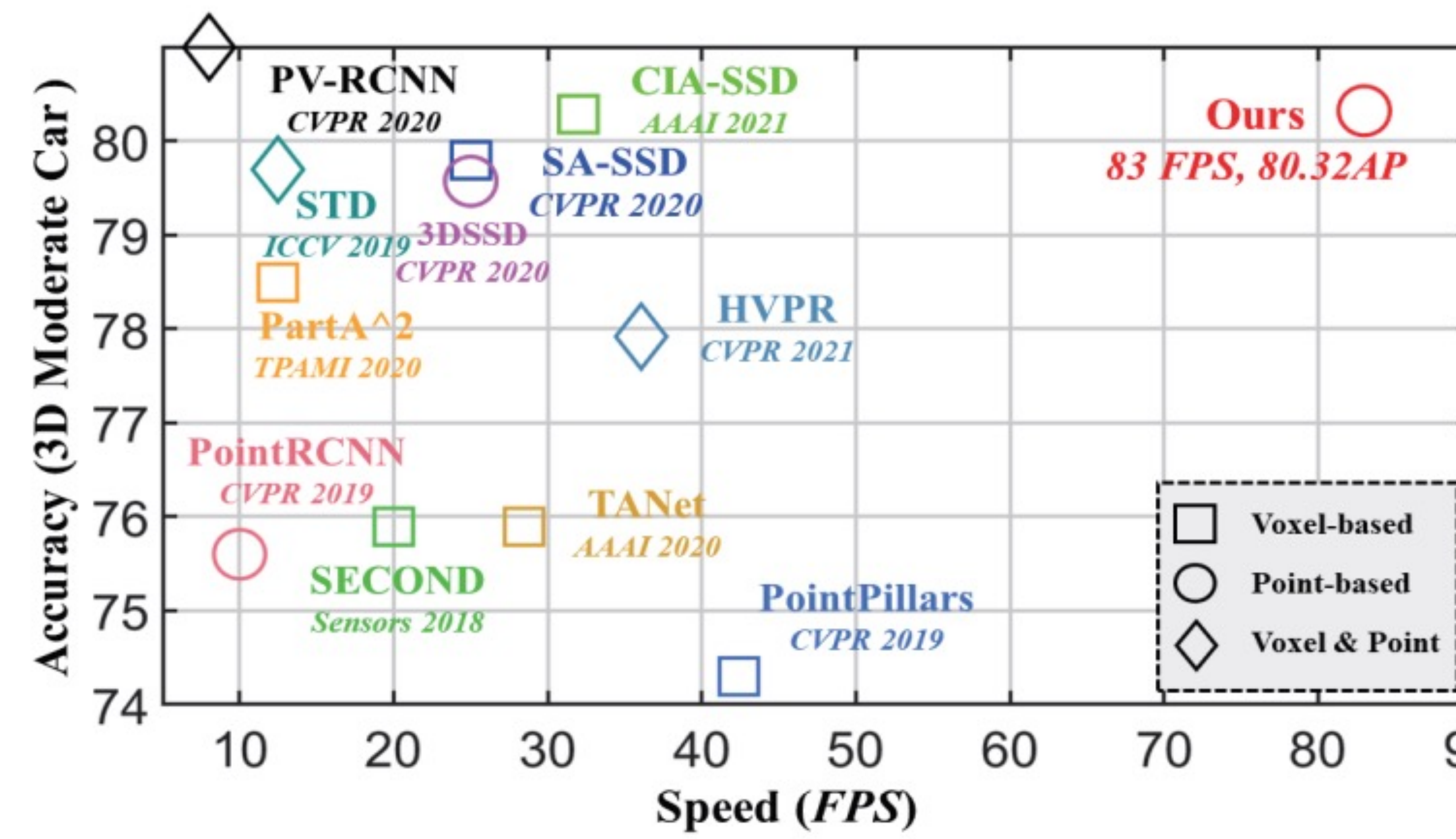
- *propose feature similarity-based FPS to replace Distance-based FPS for recalling more foreground point features*
- *encoder-only architecture without needing the feature propagation (FP) layers*

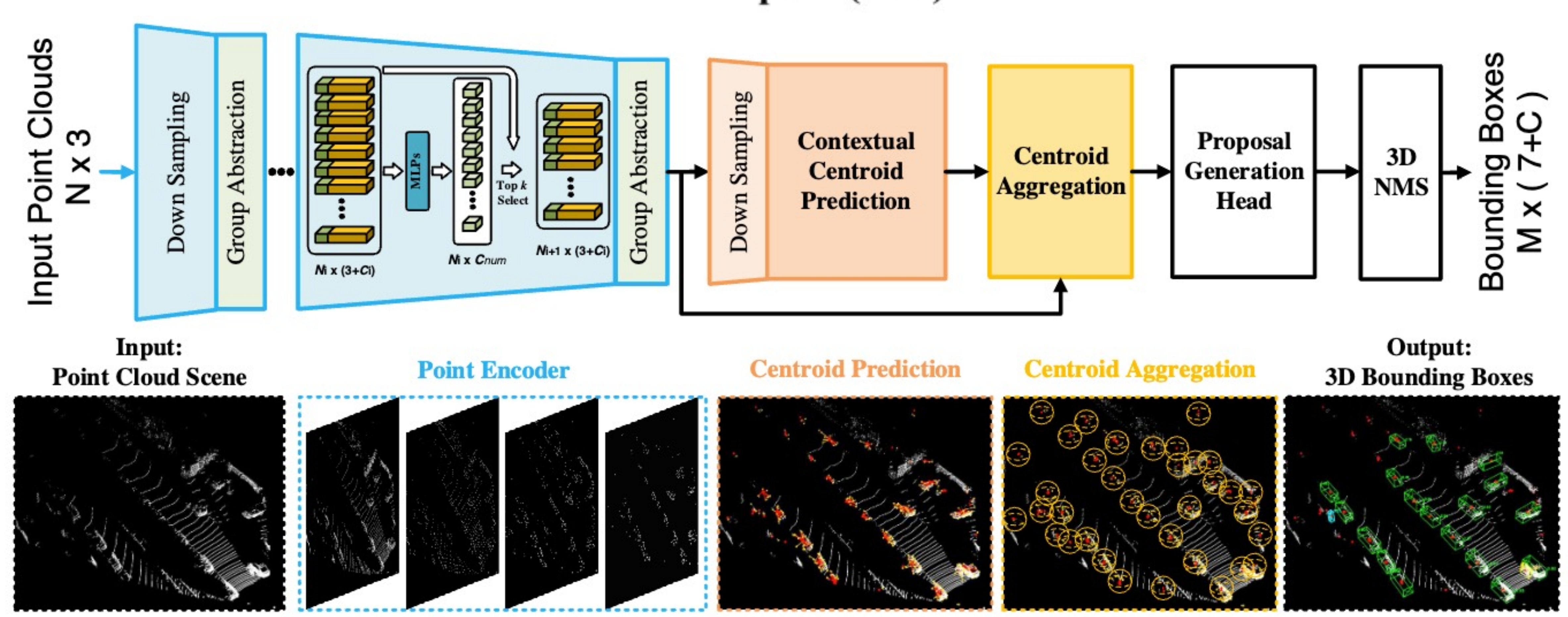




IA-SSD

- *Predict classification score for each point feature, and use top-k selection to carry out more efficient FPS operation*





Motivation, analysis, insight

Our motivations

- *Existing methods focus on processing foreground point features (small proportion) more efficient.*
- *Can we filter out redundancy of background point features?*
- *Can we also further drop redundance model?*

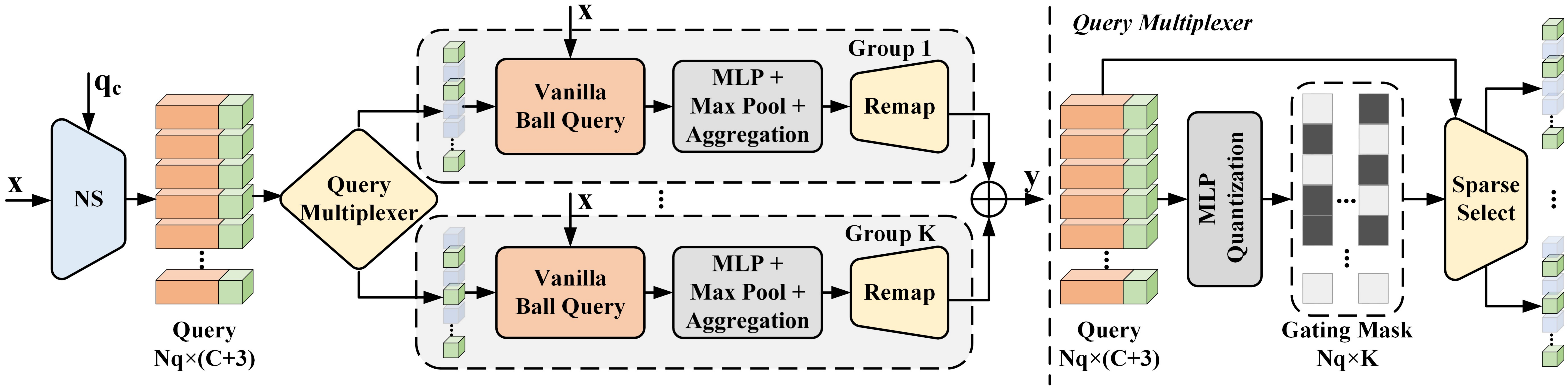
Analysis and insights

(a) Latency (b) Background ratio (c) Scale distribution

- *The MLP network occupies over half latency of overall model*
- *Tremendous spatial redundancy exists in background point features that appear in each stage of the detector*
- *The size of each object is varying, making it unusable to align each receptive field of the conventional multi-scale grouping (MSG) and suffering from branch redundancy in MSG*

Method

This work introduce Dynamic Ball Querying (DBQ) to replace the vanilla ball querying operation

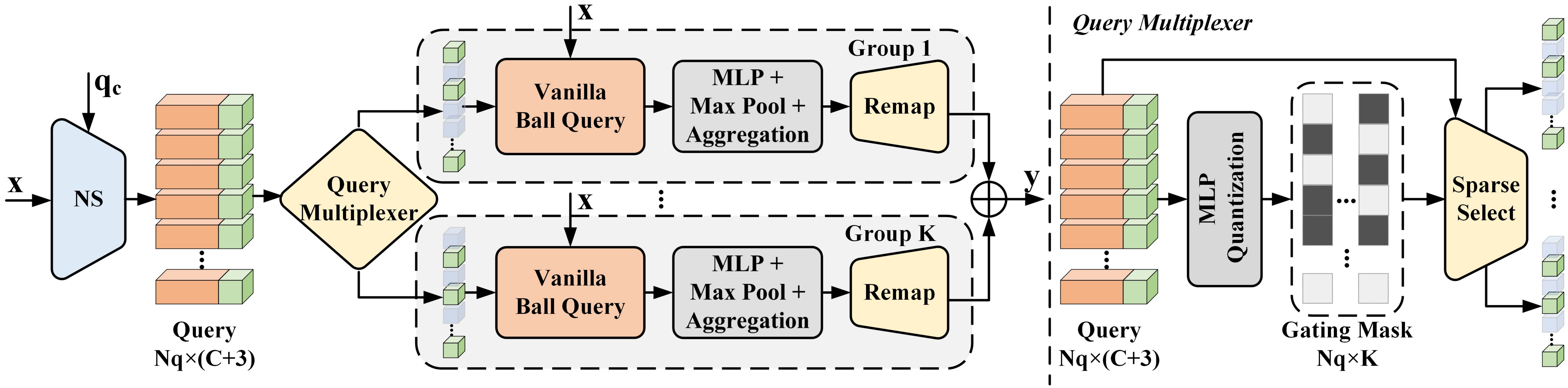


Inference

- *Query Multiplexer uses a light MLP network to predict gating masks for each group (scale), building on MLP network part [motivation 1]*
- *Each group can generate sparse activations to drop background point features [motivation 2]*
- *Gating masks with positive values enable corresponding point feature to through corresponding groups. (3 cases: all activation, all blocking, or partial activation) [motivation 3]*
- *Finally, remapping activation point features to dense form (like unpooling), in which blocking point feature fills zero*

Method

This work introduce Dynamic Ball Querying (DBQ) to replace the vanilla ball querying operation



Training

- *Non-differentiable of sparse selection: Gumbel-Sigmoid technique*
- *The degree of sparse is data-driven*
- *Introduce a latency constraint as a training target (budget loss) to achieve a balance between effectiveness and efficiency. We establish latency map Ψ for each group in each SA layer.*
- *Interesting, the latency budget γ is set to 0 for avoiding parameter adjustment, so we only need to scale λ (control the trade-off between effectiveness and efficiency)*

$$\Psi = \frac{\sum_l \sum_k \Psi_{l,k}(\sum_i \mathbf{m}^l(i,k)))}{\sum_l \sum_k \Psi_{l,k}(N_q^l)}$$

$$\mathcal{L} = \mathcal{L}_{\text{tasks}} + \lambda \mathcal{L}_{\text{budget}}, \text{ where } \mathcal{L}_{\text{budget}} = |\Psi - \gamma|$$

Experiment Results

Evaluation on KITTI datasets

Method	Type	3D Car (IoU=0.7) Easy	3D Car (IoU=0.7) Mod.	3D Car (IoU=0.7) Hard	3D Ped. (IoU=0.5) Easy	3D Ped. (IoU=0.5) Mod.	3D Ped. (IoU=0.5) Hard	3D Cyc. (IoU=0.5) Easy	3D Cyc. (IoU=0.5) Mod.	3D Cyc. (IoU=0.5) Hard	Speed
Voxel-based Methods											
VoxelNet (Zhou & Tuzel, 2018)	1-stage	77.47	65.11	57.73	39.48	33.69	31.5	61.22	48.36	44.37	4.5
SECOND (Yan et al., 2018)	1-stage	84.65	75.96	68.71	45.31	35.52	33.14	75.83	60.82	53.67	20
PointPillars (Lang et al., 2019)	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 (Zhou et al., 2019)	1-stage	86.16	76.50	71.39	-	-	-	-	-	-	12.5
Associate-3Ddet (Du et al., 2020)	1-stage	85.99	77.40	70.53	-	-	-	-	-	-	20
SA-SSD He et al. (2020)	1-stage	88.75	79.79	74.16	-	-	-	-	-	-	25
CIA-SSD (Zheng et al., 2020)	1-stage	89.59	80.28	72.87	-	-	-	-	-	-	32
TANet Liu et al. (2020)	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 Chen et al. (2019)	2-stage	89.29	77.40	70.24	-	-	-	-	-	-	16.7
STD (Yang et al., 2019)	2-stage	87.95	79.71	75.09	53.29	42.47	38.35	78.69	61.59	55.30	12.5
PV-RCNN (Shi et al., 2020a)	1-stage	**90.25**	**81.43**	**76.82**	52.17	43.29	40.29	78.60	63.71	**57.65**	12.5
VIC-Net (Jiang et al., 2021)	1-stage	88.25	80.61	75.83	43.82	37.18	35.35	78.29	63.65	57.27	17
HVPR (Noh et al., 2021)	1-stage	86.38	77.92	73.04	52.47	43.96	**40.64**	-	-	-	36.1
Point-based Methods											
PointRCNN (Shi et al., 2019)	2-stage	86.96	75.64	70.70	47.98	39.37	36.01	74.96	58.82	52.53	10
3D IoU-Net (Li et al., 2020a)	2-stage	87.96	79.03	72.78	-	-	-	-	-	-	10
Point-GNN (Shi & Rajkumar, 2020)	1-stage	88.33	79.47	72.29	51.92	43.77	40.14	78.60	63.48	57.08	1.6
3DSSD (Yang et al., 2020b)	1-stage	88.36	79.57	74.55	**54.64**	44.27	40.23	82.48	**64.10**	56.90	25
IA-SSD (Zhang et al., 2022)	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**

Experiment Results

Evaluation on WOD, ONCE datasets

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

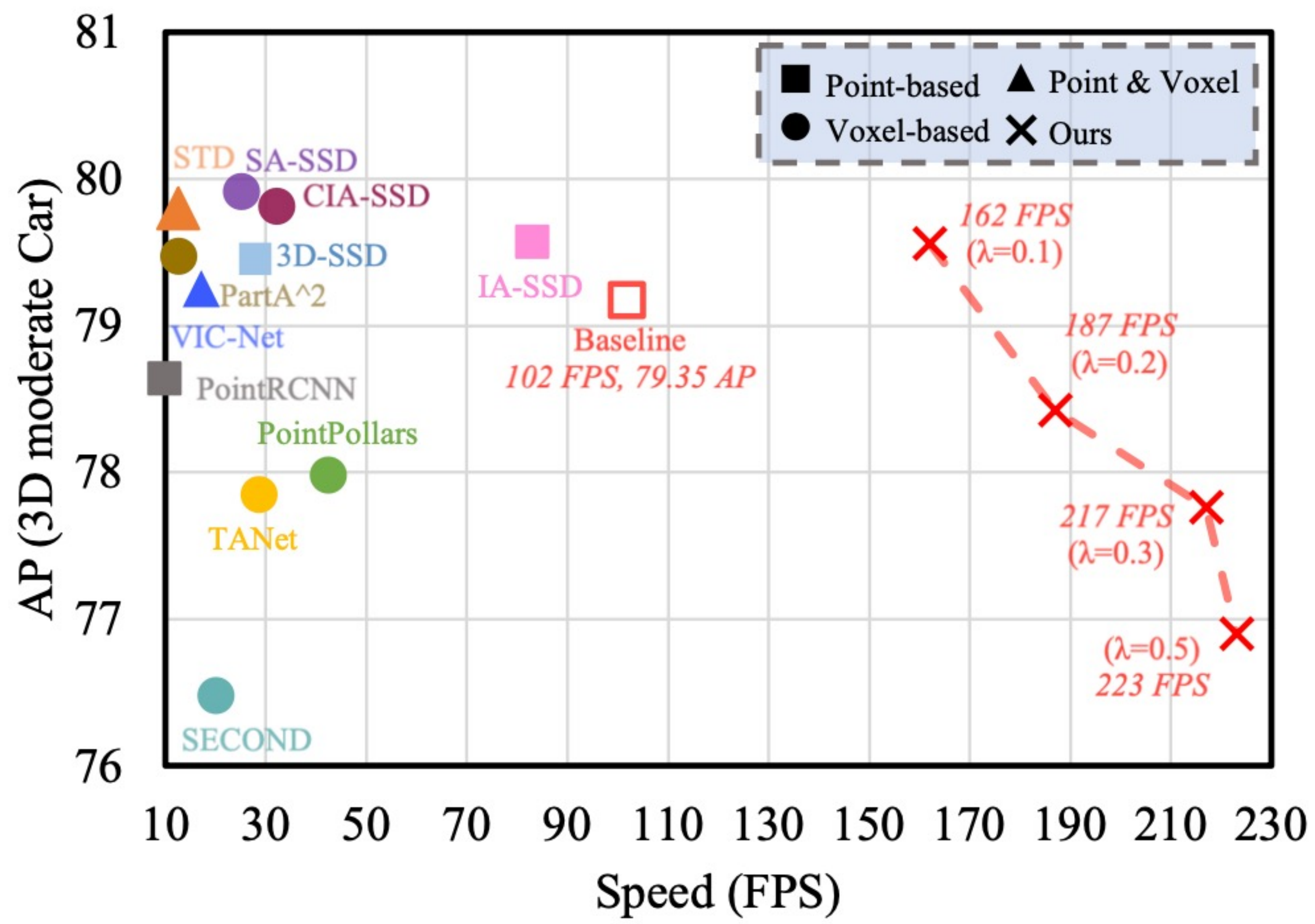
Method	Vehicle (LEVEL 1)		Vehicle (LEVEL 2)		Ped. (LEVEL 1)		Ped. (LEVEL 2)		Cyc. (LEVEL 1)		Cyc. (LEVEL 2)		speed
	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH	
PointPollars (Lang et al., 2019)	60.67	59.79	52.78	52.01	43.49	23.51	37.32	20.17	35.94	28.34	34.60	27.29	-
SECOND (Yan et al., 2018)	68.03	67.44	59.57	59.04	61.14	50.33	53.00	43.56	54.66	53.31	52.67	51.37	-
Part-A^2 (Shi et al., 2020b)	71.82	71.29	64.33	63.82	63.15	54.96	54.24	47.11	65.23	63.92	62.61	61.35	-
PV-RCNN (Shi et al., 2020a)	**74.06**	**73.38**	**64.99**	**64.38**	62.66	52.68	53.80	45.14	63.32	61.71	60.72	59.18	-
IA-SSD (Zhang et al., 2022)	70.53	69.67	61.55	60.80	**69.38**	**58.47**	**60.30**	50.73	67.67	65.30	64.98	62.71	14
Efficient Baseline	71.15	70.30	62.49	61.73	68.38	58.21	59.75	50.80	**68.64**	**66.23**	**66.09**	63.78	20
DBQ-SSD (λ=0.1)	70.56	69.82	61.81	61.15	68.89	58.07	60.15	50.60	66.58	63.98	64.22	61.66	**30**
DBQ-SSD (λ=0.05)	71.58	71.03	64.13	63.61	69.18	**58.47**	60.22	**50.81**	68.29	66.01	**66.09**	**63.86**	27

Table 6: 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.

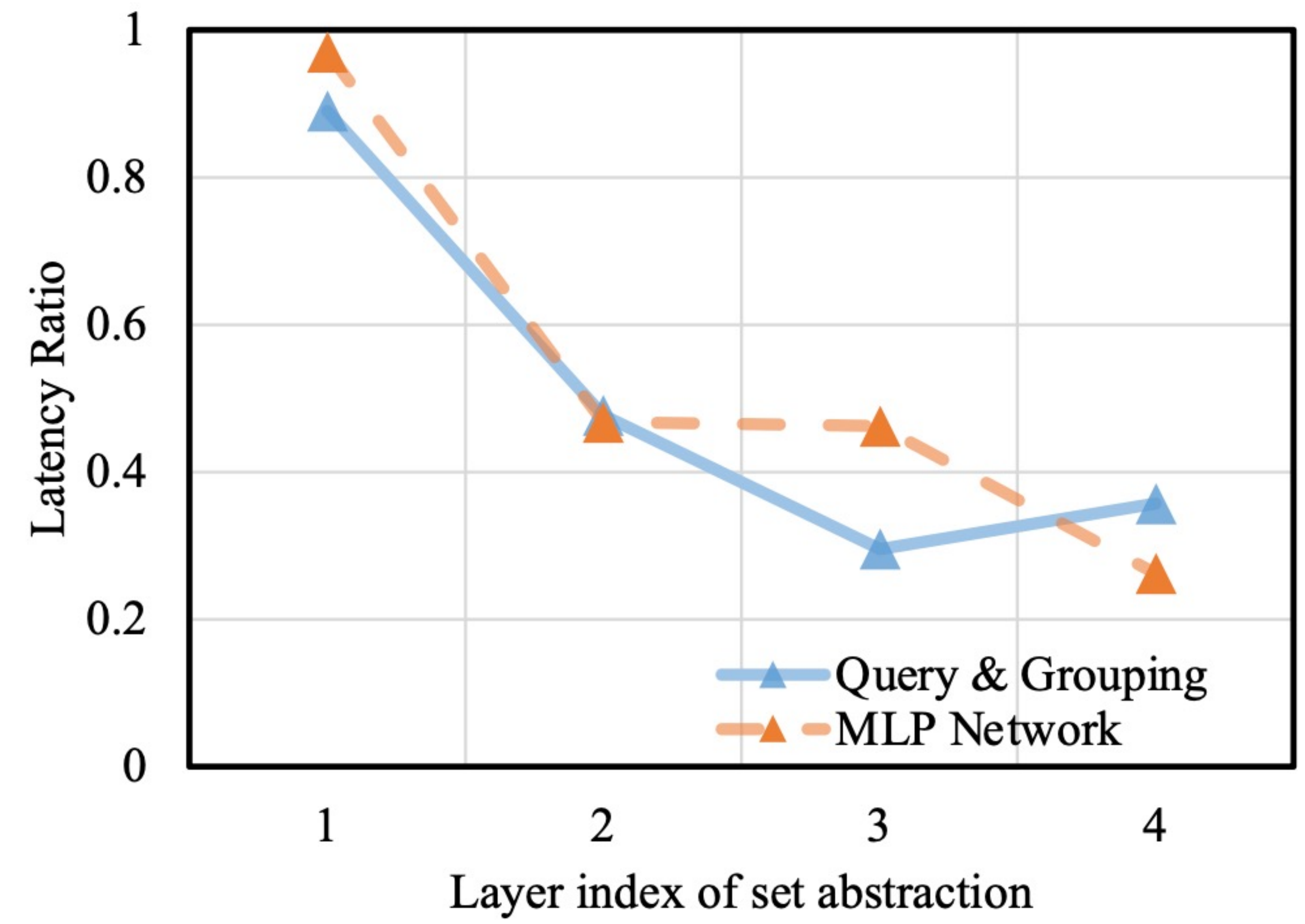
Method	Vehicle				Pedestrian				Cyclist				mAP	Speed
	Overall	0-30m	30-50m	>50m	Overall	0-30m	30-50m	>50m	Overall	0-30m	30-50m	>50m		
PointPollars	68.57	80.86	62.07	47.04	17.63	19.74	15.15	10.23	46.81	58.33	40.32	25.86	44.34	-
SECOND	71.19	84.04	63.02	47.25	26.44	29.33	24.05	18.05	58.04	69.96	52.43	34.61	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 (λ=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 (λ=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 (λ=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 (λ=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**

Experiment Results

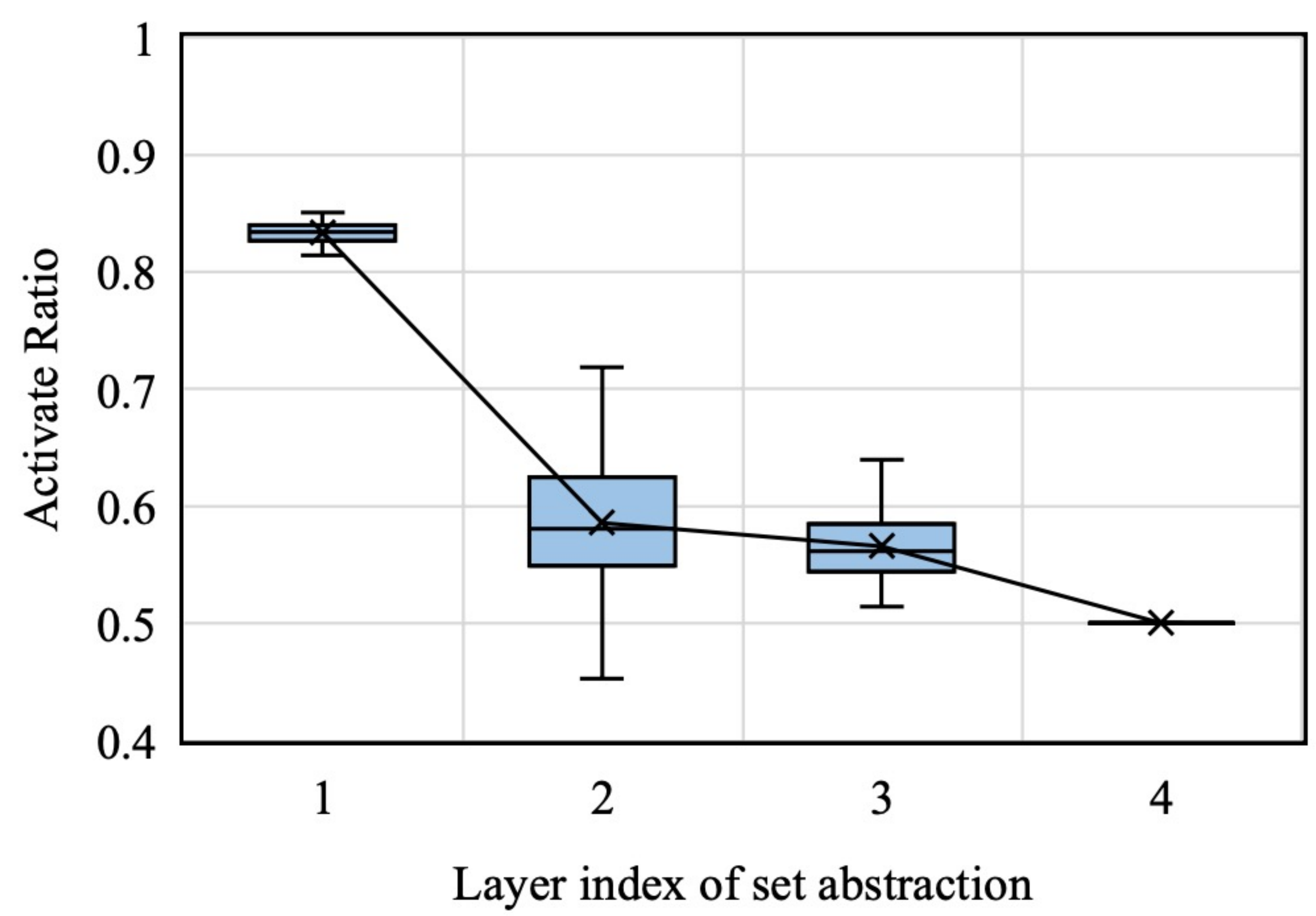
Ablation study on KITTI datasets

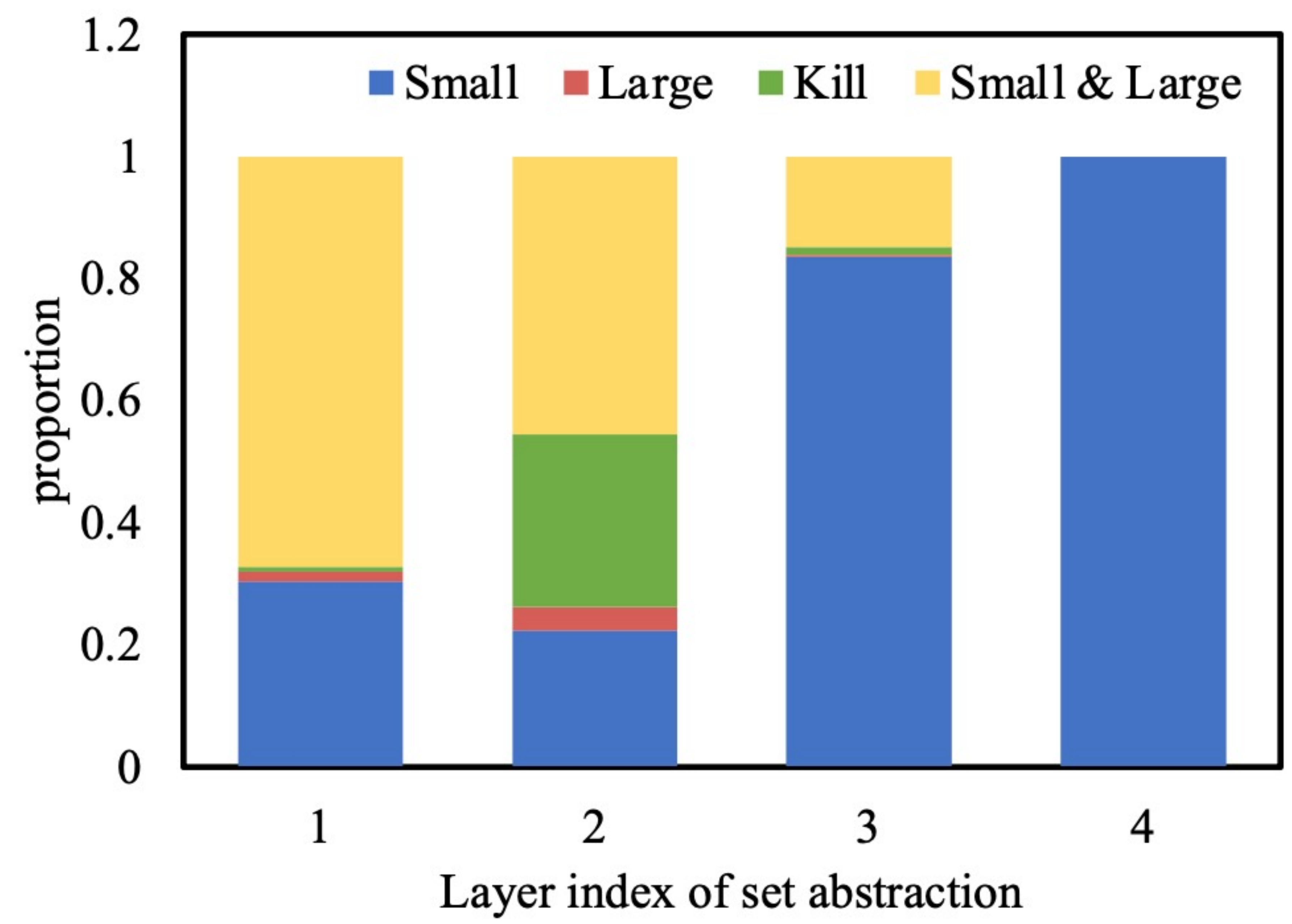


(a) Performance & Latency



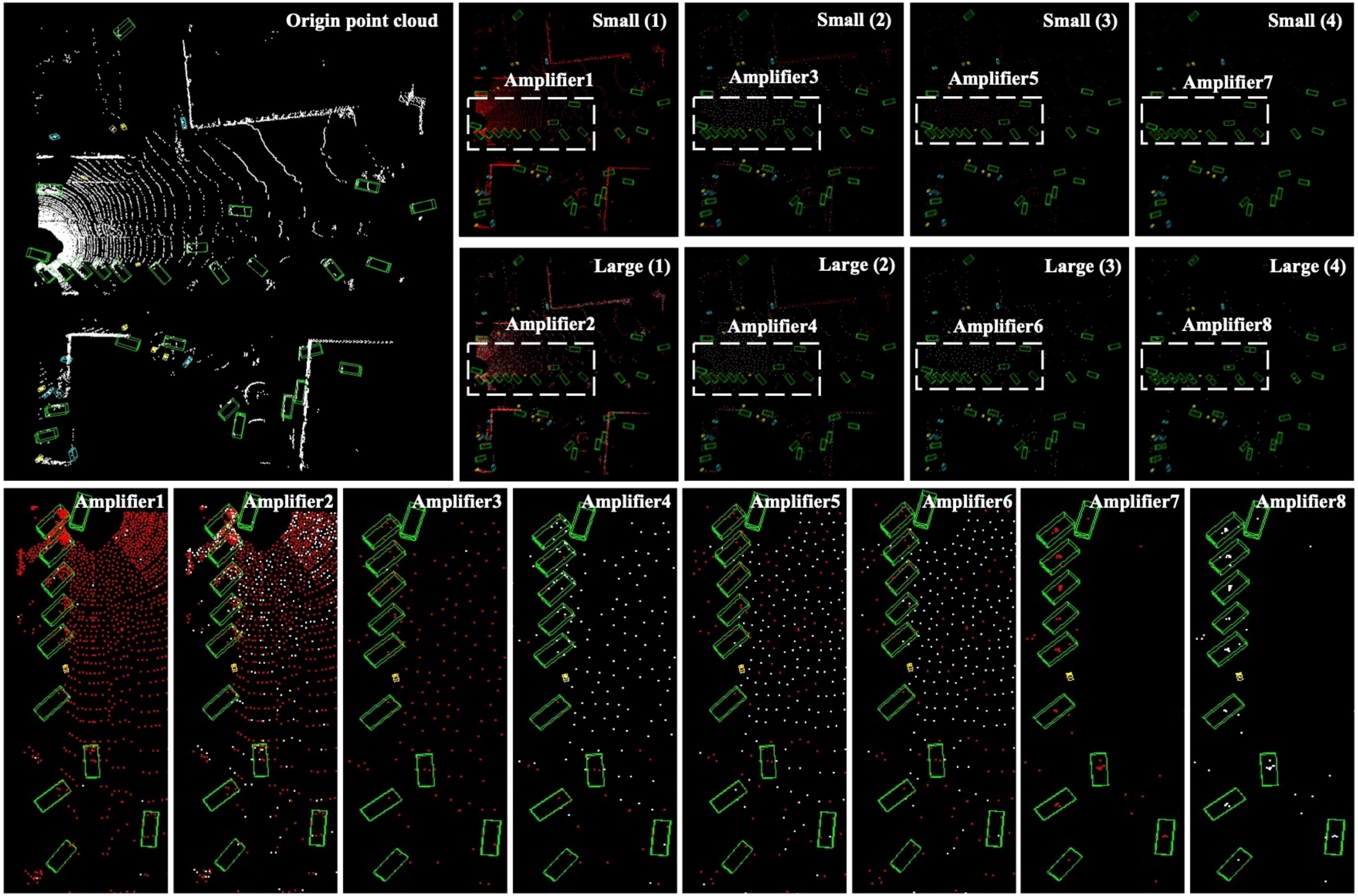
(b) Layer ($\lambda = 0.1$)





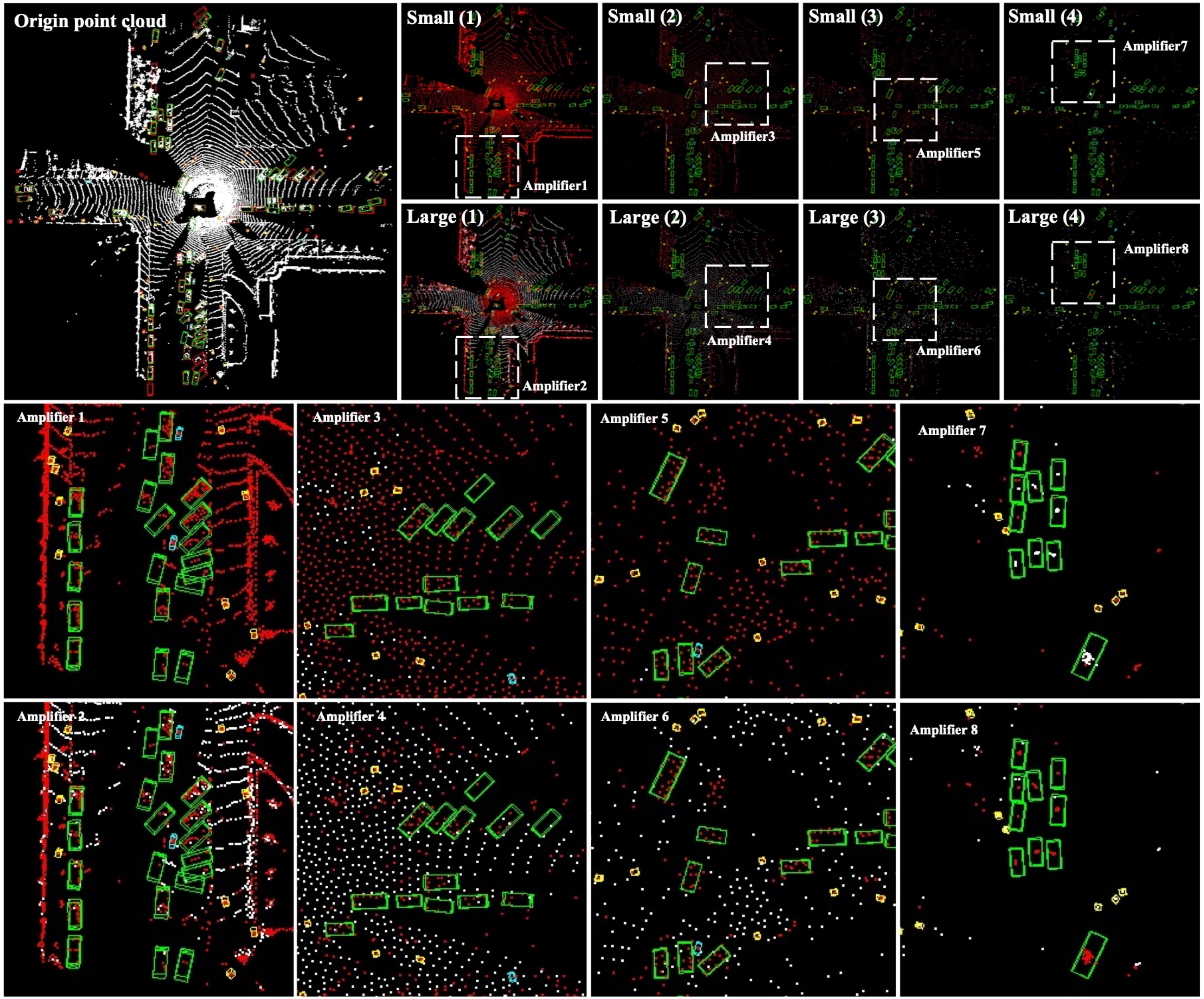
Experiment Results

Visualization on KITTI datasets



Experiment Results

Visualization on WOD datasets



Thanks

https://github.com/yancie-yjr/DBQ-SSD