Department of Electronic Engineering, Tsinghua University

ICCV23 PARIS

Ada3D : Exploiting the Spatial Redundancy with Adaptive Inference for Efficient 3D Object Detection

Tianchen Zhao[12], Xuefei Ning[1*] , Ke Hong[1] , Zhongyuan Qiu[2], Pu Lu[1] , Yali Zhao[2] , Linfeng Zhang[1] , Lipu Zhou[3], Guohao Dai[4] , Huazhong Yang[1] , Yu Wang[1*]

1 Tsinghua University, 2Novauto, 3 Meituan, 4 Shanghai Jiao Tong University

(Corresponding Authors)*

Background: Advances in 3D Perception

- **3D Perception:** Key component of comprehending the 3D world

Autonomous Driving

Metaverse

Embodied AI

Background: Efficiency Challenge

Autonomous Driving: Efficiency Challenges

Low Latency Requirement:
The reaction speed of self-driving car should be faster than human driver (250ms)

\+

High Accuracy Requirement:
Perception should be accurate to avoid safety issues.

Budget Restrict:
Price of computing chips on self-driving cars are restricted.

\+

Power Restrict:
Energy-efficient embedded computing platforms(~30W) instead of consumer-level GPU (~300W) is adopted

High Efficiency & Accuracy Requirement

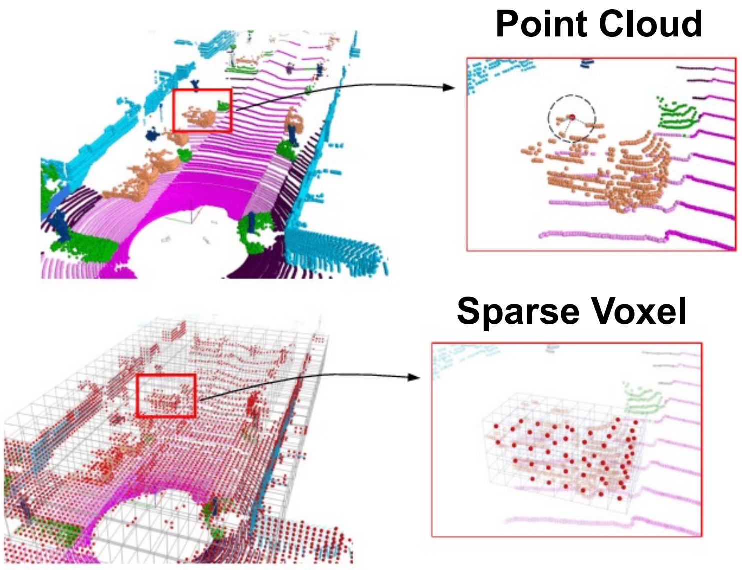
Conflict with

Restricted Hardware Resources

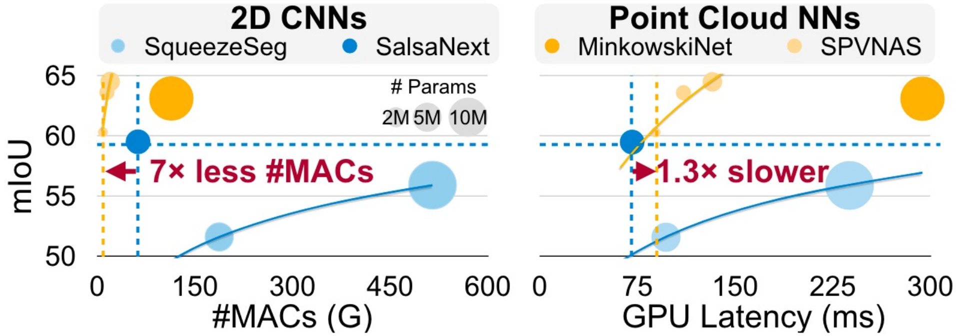
Calls for Efficiency Improvement !

Background: Voxel-based 3D Perception

➢ **Voxel-based 3D Perception:** Good Performance with Large Cost



(Figure from TPVNet[1])



PointACC[2]: Sparse Convolution with
7x less MACs runs 1.3x slower on GPU

[1] Xu et. al., RPVNet: A Deep and Efficient Range-Point-Voxel Fusion Network for LiDAR Point Cloud Segmentation, ICCV21
[2] Ling et al., PointAcc: Efficient Point Cloud Accelerator. MICRO21

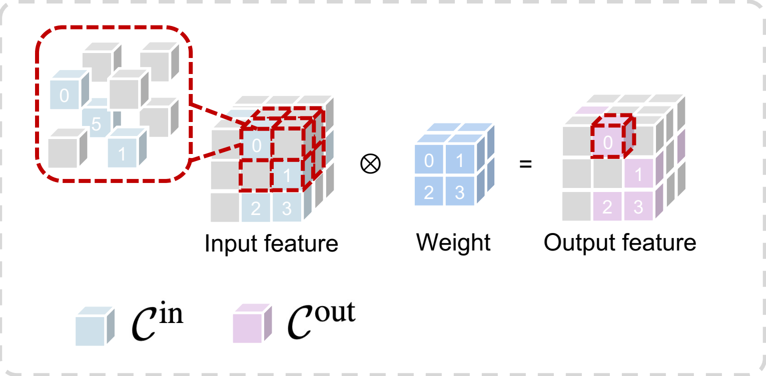
Preliminary: 3D Sparse Convolution

➢ **3D Sparse Convolution**: Convolution with sparse mask C_in & C_out

- Sparse Convolution: Enlarging the dense.
- Submanifold sparse convolution: $\mathcal{C}^{\text{out}} = \mathcal{C}^{\text{in}}$
  - To maintain the sparsity through convolution networks
  - Less computation with comparable performance

$$\mathbf{x}_{\mathbf{u}}^{\text{out}} = \sum_{\mathbf{i} \in \mathcal{N}^D(\mathbf{u}, \mathcal{C}^{\text{in}})} W_{\mathbf{i}} \mathbf{x}_{\mathbf{u}+\mathbf{i}}^{\text{in}} \text{ for } \mathbf{u} \in \mathcal{C}^{\text{out}}$$

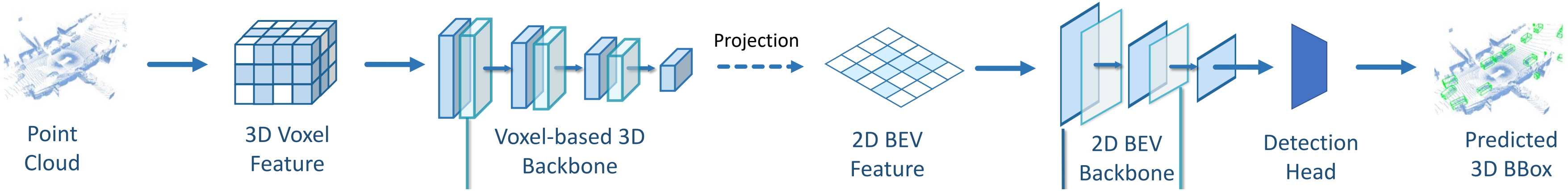
$$\mathcal{N}^D(\mathbf{u}, \mathcal{C}^{\text{in}}) = \{\mathbf{i} | \mathbf{u} + \mathbf{i} \in \mathcal{C}^{\text{in}}, \mathbf{i} \in \mathcal{N}^D\}$$



[1] Yin et. al., VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection, ICCV17

Preliminary: Voxel-based 3D Detection

➢**Voxel-based 3D Detection**: Framework overview



To BEV

SparseTensor
Coordinates: [N,3]
Feature: [N,C]

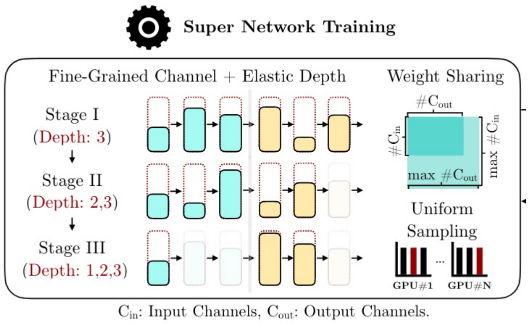
→ *Flatten in Z-axis* →

Dense Tensor
Feature: [B,C,W,H]

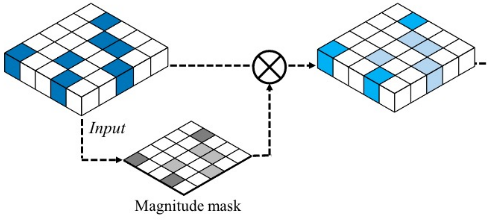
[1] Yin et. al., Center-based 3D Object Detection and Tracking, CVPR20

Prior Research: Efficient 3D Perception

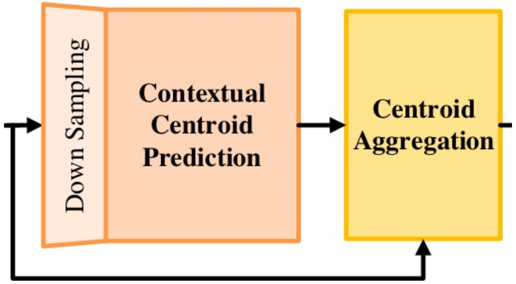
➢ **Prior research of efficient 3d perception**: Mainly focus on reducing the model-level redundancy



SPVNAS[1]
Use NAS (Neural Architecture Search) to search for model macro deisgn (depth/width)



SPSS-Conv[2]
Kernel-level pruning of the 3D sparse convolution.

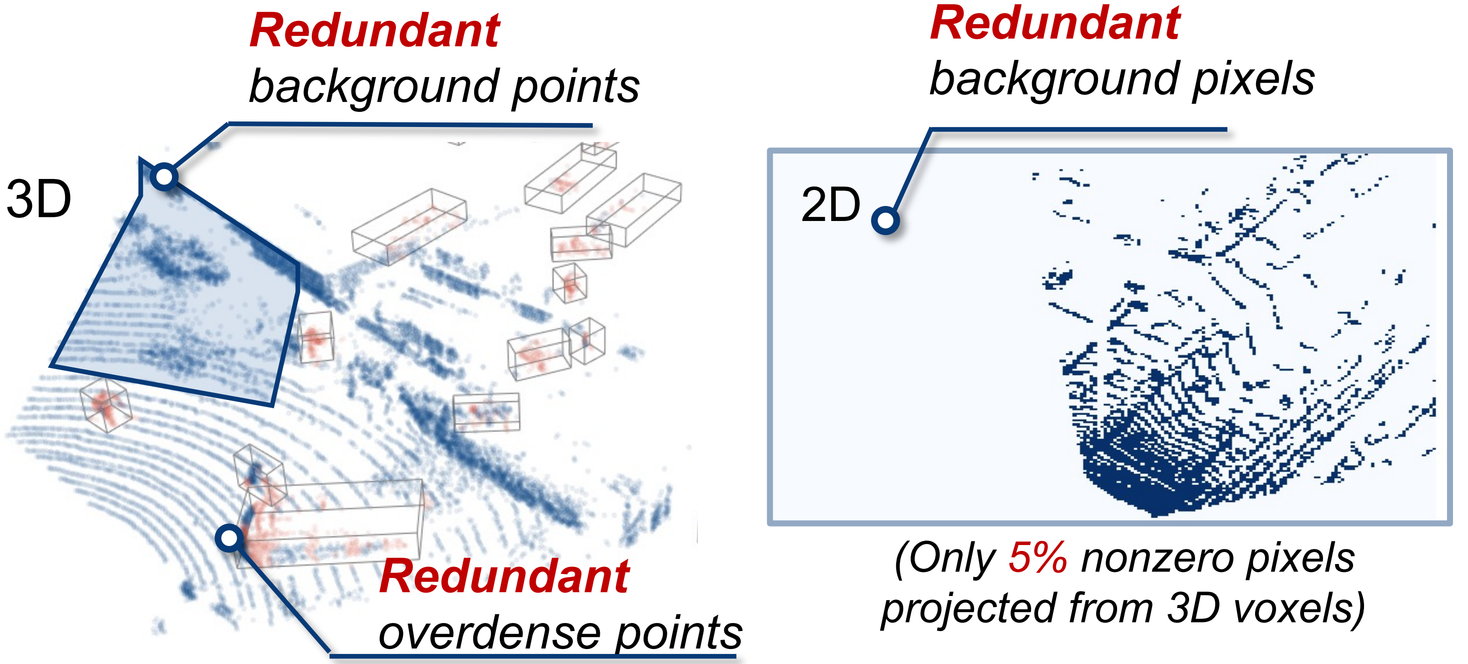


IA-SSD[3]
Design novel feature-based downsampling module to replace FPS (Furthest Point Sampling)

[1] Tang et. al., Searching Efficient 3D Architectures with Sparse Point-Voxel Convolution, ECCV20.
[2] Liu et al., Spatial Pruned Sparse Convolution for Efficient 3D Object Detection, NeurIPS22.
[3] Zhang et. al.,Not All Points Are Equal: Learning Highly Efficient Point-based Detectors for 3D LiDAR Point Clouds, CVPR22

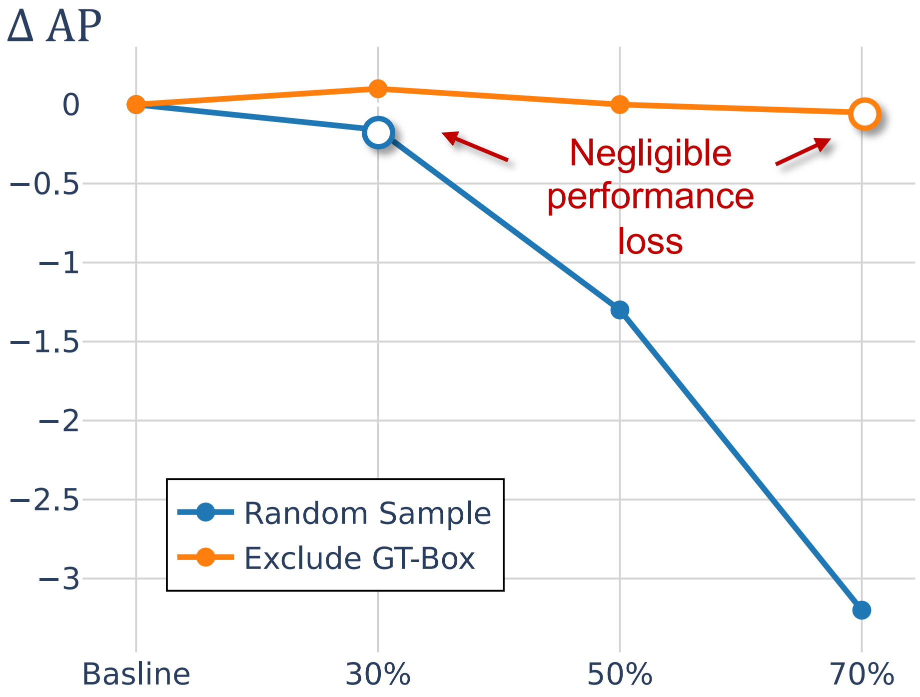
Key Motivation: Data Redundancy

➢ **Exploiting Data-level Redundancy**: Another approach of improving the efficiency of 3D Detector.



(Only 5% nonzero pixels projected from 3D voxels)

Exploiting Data-level Redundancy: Qualitative Results.



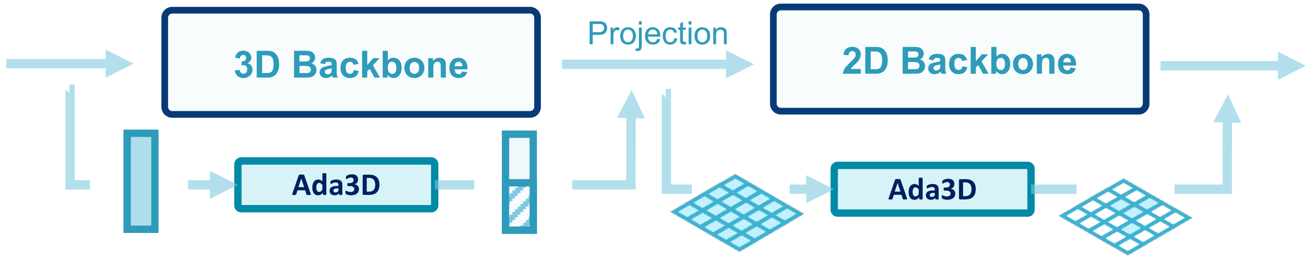
Oracle Experiment:

- When dropping inputs 70% points exclude gt-box, less than -0.1 AP.
- When random dropping 30% points, less than -0.5 AP

Large Redundancy Exists!

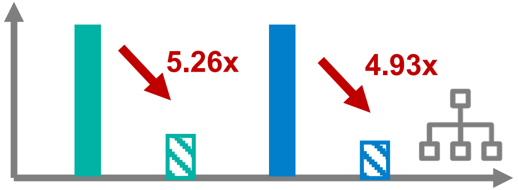
Key Contribution: Exploit Spatial Redundancy

- **<Ada3D : Exploiting the Spatial Redundancy with Adaptive Inference for Efficient 3D Object Detection>:** Adaptive Inference, discard redundant input during inference.

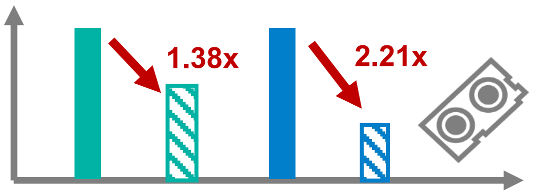


Reduce **40%** redundant 3D voxels

Reduce **80%** redundant 2D pixels



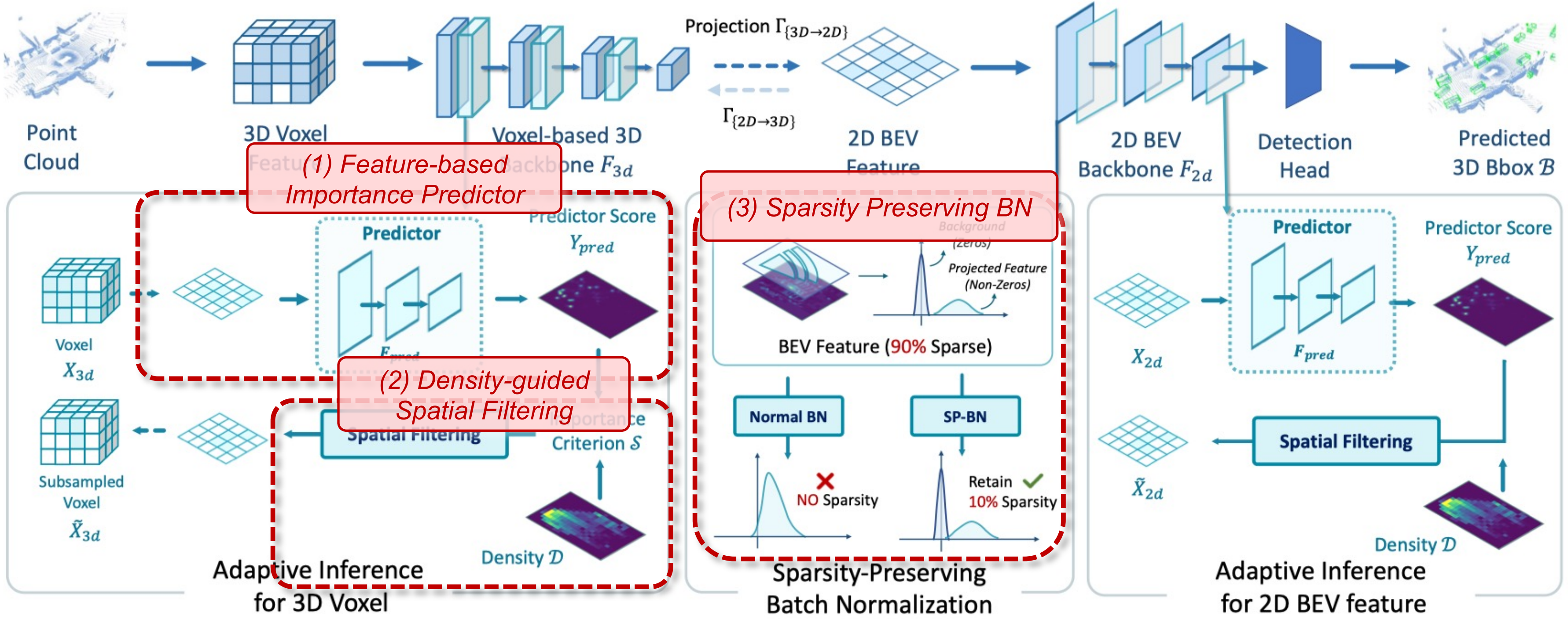
5x FLOPs and memory saving



1.38x GPU latency saving
2.21x GPU peak memory

Method: Overview

➢ Overview of Ada3D: 3 Key Components



Method-1: Importance Predictor

➢ **BEV-space Importance Predictor:** predict pixel-wise relative importance

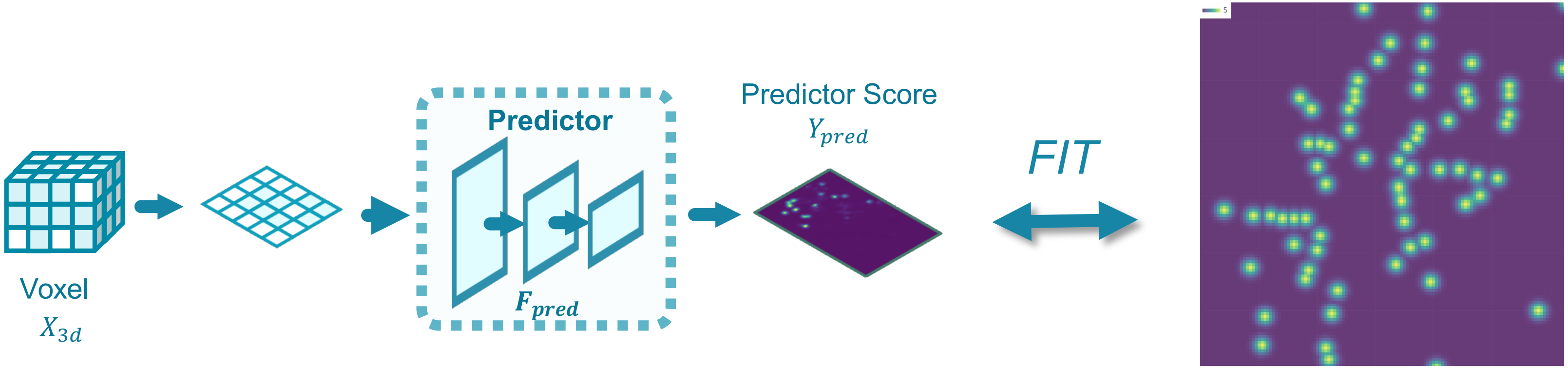
Lightweight Predictor:

- *Shared BEV-space Predictor*
- *5 Layer 2D Convolution Network*
- *Low Resolution*
- *Efficient Group Convolution*
- *<1% Computational Cost than Backbone*

Predictor Training Ground-truth:

Center-based Object Heatmap

(Gaussian Ball Rendered Object Center)

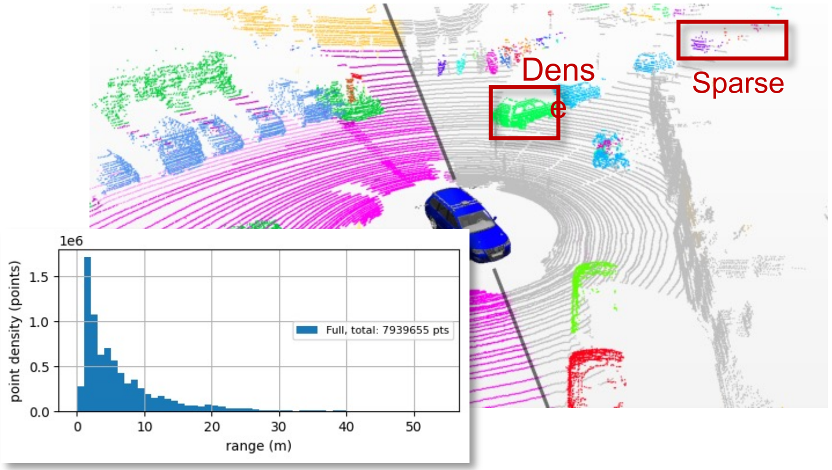


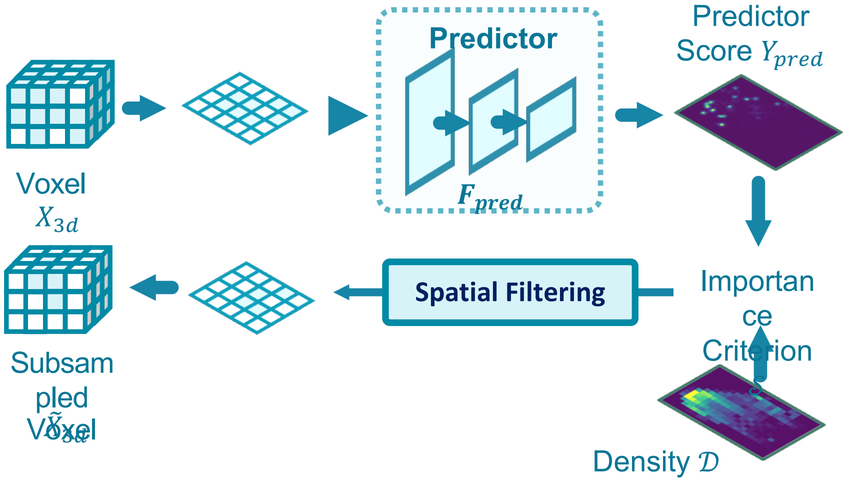
Method-2: Density-based Spatial Filtering

➢ **Density-guided Filtering:** Leverage the property of Lidar point cloud

Lidar Point Cloud: Local region are denser,
Remote region are sparse
Predictor tends to **output larger value for local dense part**

Compensation: Focus more on remote sparse region

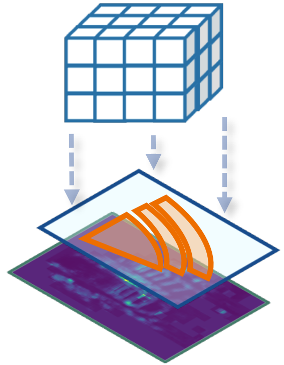




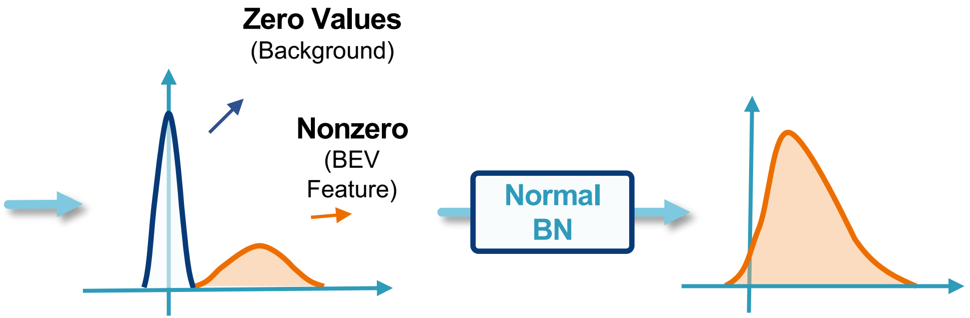
Method-3: Spatial-Preserving BN

- **“Normal BN”:** Sacrifices sparsity

2D BEV Backbone: Lose sparsity after 1st BN Layer

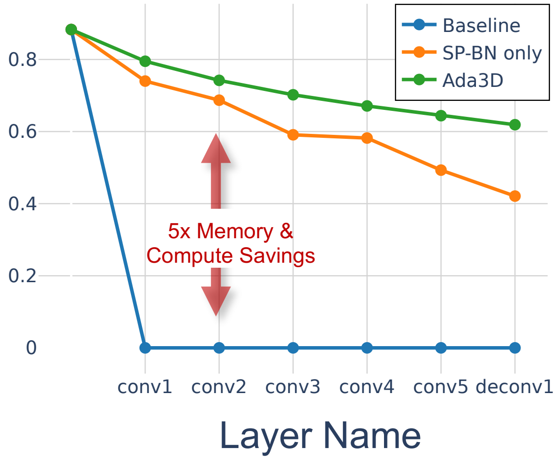


Sparse BEV feature map
Projected from
3D voxel feature



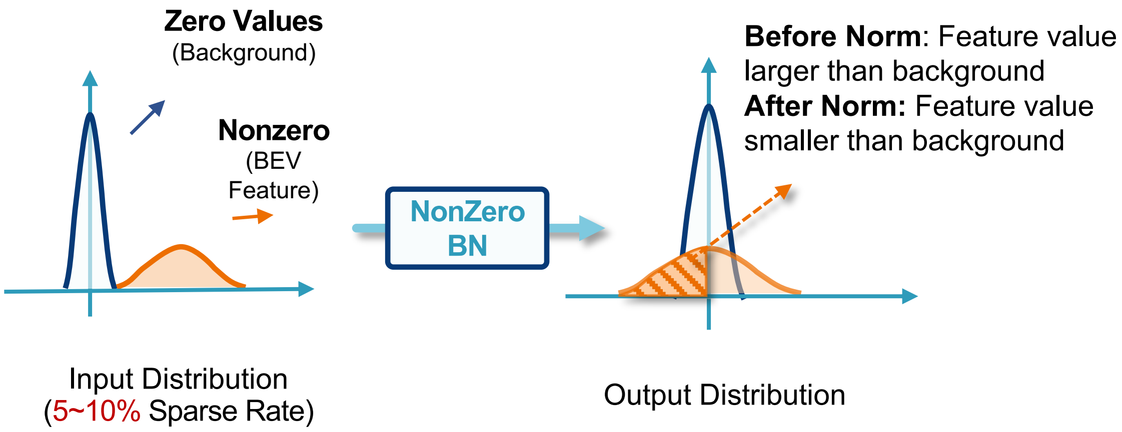
Input Distribution
(5~10% Sparse Rate)

Output Distribution
(**0%** Sparse Rate)



Method-3: Spatial-Preserving BN

- **Straight Forward Solution:** "Nonzero-BN", apply BN to nonzero elements only, when finetuning, **instable training** and **performance degradation**.



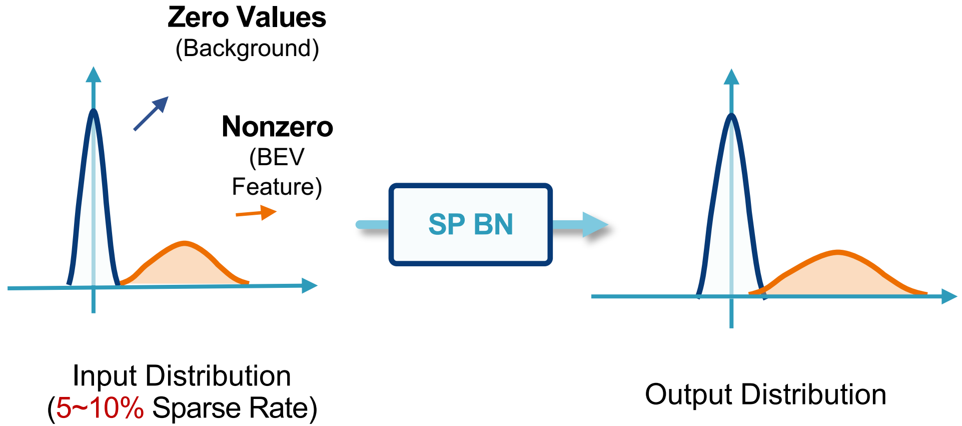
BN Type	Sparse	KITTI Mod. AP		
		Car.	Ped.	Cyc.
Normal BN	-	79.4	53.4	65.5
Without BN	✓	76.3	43.5	49.7
Nonzero BN	✓	74.5	39.4	47.3
SP-BN	✓	**79.1**	**53.3**	**65.6**

✗ Violating Relative Relations

Method-3: Spatial-Preserving BN

➢ **Simple Fix:** "SP-BN", skip zero mean in BN.

$$\hat{x}_i^{(k)} = \frac{x_i^{(k)}}{\sqrt{(\sigma_B^{(k)})^2 + \epsilon}},$$



BN Type	Sparse	KITTI Mod. AP		
		Car.	Ped.	Cyc.
Normal BN	-	79.4	53.4	65.5
Without BN	✓	76.3	43.5	49.7
Nonzero BN	✓	74.5	39.4	47.3
SP-BN	✓	**79.1**	**53.3**	**65.6**

✓ Retaining Sparsity

✓ Retaining Relative Relations

Experiments: KITTI Overview

Mehod	*FLOPs*	*Mem*	mAP	3D Car (IoU=0.7)			3D Ped. (IoU=0.5)			3D Cyc. (IoU=0.5)		
	Opt.	*Opt.*	*(Mod.)*	*Easy*	*Mod.*	*Hard*	*Easy*	*Mod.*	*Hard*	*Easy*	*Mod.*	*Hard*
VoxelNet [36]	-	-	49.05	77.47	65.11	57.73	39.48	33.69	31.50	61.22	48.36	44.37
SECOND [28]	-	-	57.43	84.65	75.96	68.71	45.31	35.52	33.14	75.83	60.82	53.67
PointPillars [12]	-	-	58.29	82.58	74.31	68.99	51.45	41.92	38.89	77.10	58.65	51.92
SA-SSD [8]	-	-	-	88.75	79.79	74.16	-	-	-	-	-	-
TANet [16]	-	-	59.90	84.39	75.94	68.82	53.72	44.34	40.49	75.70	59.44	52.53
Part-A^2 [21]	-	-	61.78	87.81	78.49	73.51	53.10	43.35	40.06	79.17	63.52	56.93
SPVCNN [25]	-	-	61.16	87.80	78.40	74.80	49.20	41.40	38.40	80.10	63.70	56.20
PointRCNN [20]	-	-	57.95	86.96	75.64	70.70	47.98	39.37	36.01	74.96	58.82	52.53
3DSSD [30]	-	-	55.11	87.73	78.58	72.01	35.03	27.76	26.08	66.69	59.00	55.62
IA-SSD [34]	-	-	60.30	88.34	80.13	75.10	46.51	39.03	35.60	78.35	61.94	55.70
CenterPoint [31]	-	-	59.96	88.21	79.80	76.51	46.83	38.97	36.78	76.32	61.11	53.62
CenterPoint-Pillar [31]	-	-	57.39	84.76	77.09	72.47	44.07	37.80	35.23	75.17	57.29	50.87
CenterPoint (Ada3D-B)	**5.26×**	**4.93×**	**59.85**	(-0.1)	79.41	75.63	46.91	39.11	36.43	76.09	61.04	53.73
CenterPoint (Ada3D-C)	**9.83×**	**8.49×**	**57.72**	(-2.2)	74.98	69.11	43.66	38.23	34.80	75.27	59.96	52.14

Without Sacrificing Loss, **5x** Computation/Memory Optimization

With Moderate Perf. Loss, **10x** Computation/Memory Optimization

Experiments: nuScenes & ONCE Overview

Method	*FLOPs Opt.*	*Mem. Opt.*	mAP	NDS
PointPillar [12]	-	-	44.63	58.23
SECOND [28]	-	-	50.59	62.29
CenterPoint-Pillar [31]	-	-	50.03	60.70
CenterPoint [31] *(voxel=0.1)*	-	-	55.43	64.63
CenterPoint-Ada3D *(voxel=0.1)*	2.32×	2.61×	54.80	63.53
CenterPoint [31] *(voxel=0.075)*	-	-	59.22	66.48
SPSS-Conv [15] *(voxel=0.075)*	1.14×	1.14×	57.80	65.69
CenterPoint-0.5W [31] *(voxel=0.075)*	2.78×	2.78×	57.19	64.08
CenterPoint-Ada3D *(voxel=0.075)*	3.34×	3.96×	58.62	65.68
VoxelNeXT [1]	-	-	60.50	66.60
VoxelNeXT-Ada3D [1]	1.19×	1.20×	59.75	65.84

Method	*FLOPs Opt.*	*Mem. Opt.*	mAP	Veh.	Ped.	Cyc
PointRCNN [20]	-	-	28.74	52.09	4.28	29.84
PointPillar [12]	-	-	44.34	68.57	17.63	46.81
SECOND [28]	-	-	51.89	71.16	26.44	58.04
PVRCNN [19]	-	-	53.55	77.77	23.50	59.37
CenterPoint [31]	-	-	63.99	75.69	49.80	66.48
CenterPoint *(Ada3D)*	2.32×	2.61×	62.68	73.43	49.09	65.53

2~3x

FLOPs/Memory Optimization

Experiments: KITTI Detailed

Method	Technique			FLOPs		Mem.		mAP	Car Mod.	Ped. Mod.	Cyc. Mod.
	IP	DG	SP-BN	3D	2D	3D	2D	(Mod.)	(IoU=0.7)	(IoU=0.5)	(IoU=0.5)
CenterPoint	-	-	-	1.00	1.00	1.00	1.00	66.1	79.4 (-)	53.4 (-)	65.5 (-)
CenterPoint (SP-BN)	-	-	✓	1.00	0.49	1.00	0.45	66.0	79.1 (-0.3)	53.3 (-0.1)	65.6 (+0.1)
CenterPoint (Ada3D-A)	✓	✓	✓	1.00	0.22	1.00	0.25	66.4	79.5 (+0.1)	53.6 (+0.2)	66.1 (+0.6)
CenterPoint (Ada3D-B)	✓	✓	✓	0.66	0.18	0.68	0.17	66.1	79.1 (-0.3)	54.0 (+0.6)	65.3 (-0.3)
CenterPoint (Ada3D-B w.o. DG)	✓	-	✓	0.64	0.18	0.66	0.16	65.1	78.8 (-0.6)	51.6 (-1.8)	64.9 (-0.6)
CenterPoint (Ada3D-C)	✓	✓	✓	0.39	0.08	0.43	0.07	65.4	77.6 (-1.8)	53.5 (+0.2)	65.1 (-0.4)

Ada3D-A: Filter 80% 2D pixels, **+0.3** mAP

Ada3D-B: Filter 40% 3D voxels, 80% 2D pixels, 5.26x FLOPs, 4.93x Memory Opt., **-0.0** mAP

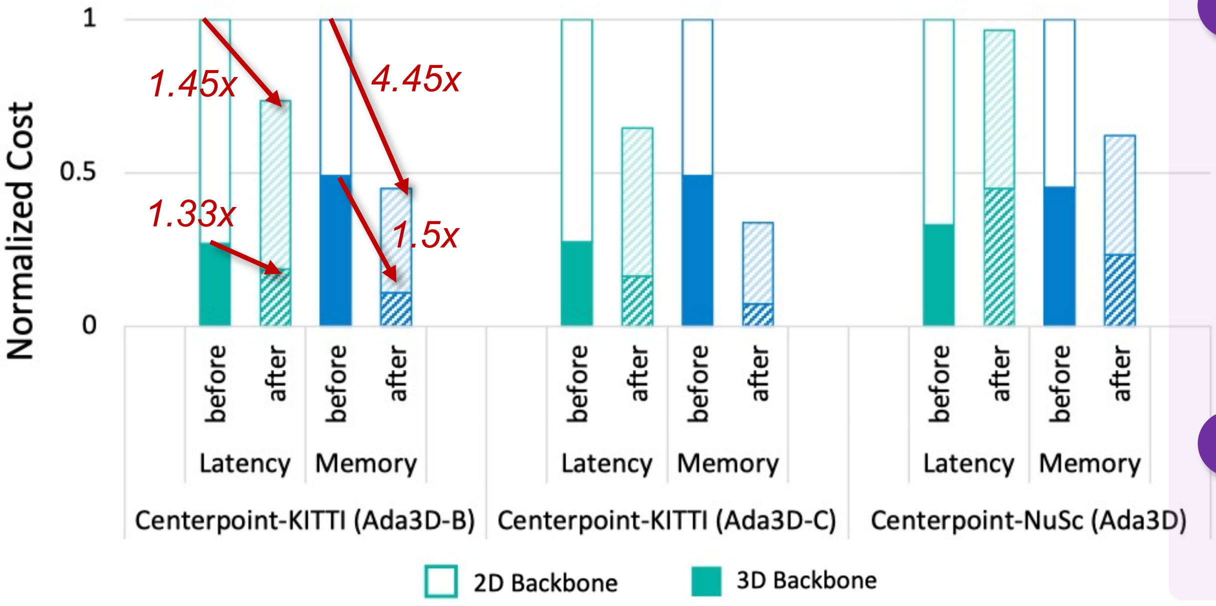
Ada3D-C: Filter 60% 3D voxels, 90% 2D pixels, 9.83x FLOPs, 8.49x Memory Opt., **-0.7** mAP

Experiments: Hardware Measurement

Ada3D-B: 5.26x FLOPs, 4.93x Memory Opt.

1.36x Latency, **2.22x** Peak Memory

Implementation: RTX3090, CUDA-11.1, Gather-Scatter GEMM SPConv v2.2.6



1 **Both 3D and 2D part requires optimization:**

Latency:	**FLOPs:**
3D: 9.7ms -> **1.33x**	3D: **1.51x**
2D: 26.2ms -> **1.45x**	2D: **4.54x**
Peak Mem.	**Memory.**
3D: 1003 MB -> **1.5x**	3D: **1.47x**
2D: 1039 MB -> **4.45x**	2D: **5.8x**

2 **3D part's latency & memory optimization grows linearly with FLOPs/Memroy while 2D part DONOT. (WHY?)**

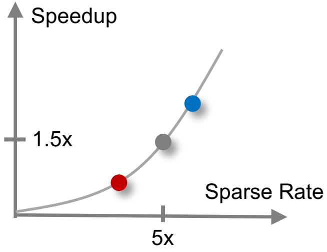
Experiments: Hardware Measurement

Ada3D-B: 5.26x FLOPs, 4.93x Memory Opt.

1.36x Latency, **2.22x** Peak Memory

Table 10: **The density and keep ratio for 3D layers of "Ada3D-B" on KITTI dataset.** The "Compress" is the reciprocal of keep rate.

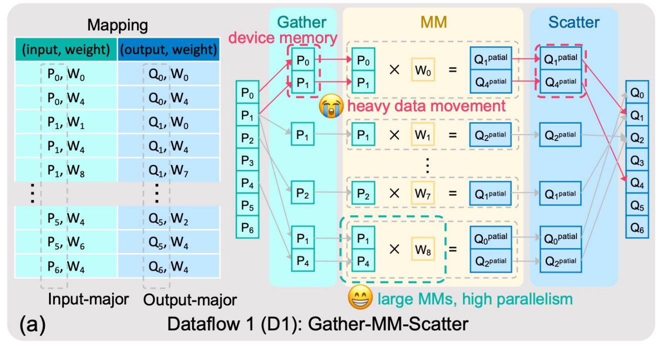
Layer	Density		Keep Rate	Compress
	Pre	Post		
3d_conv_1	0.0007	0.0005	71.43%	1.4000×
3d_conv_2	0.0098	0.0077	78.57%	1.2727×
3d_conv_3	0.0534	0.0305	57.12%	1.7508×
3d_conv_4	0.2198	0.1407	64.01%	1.5621×
3d_conv_5	0.2198	0.1407	64.01%	1.5621×
2d_conv_1	1.0000	0.0883	8.83%	11.3250×
2d_conv_2	1.0000	0.1336	13.36%	7.4850×
2d_conv_3	1.0000	0.1045	10.45%	9.5694×
2d_conv_4	1.0000	0.1416	14.16%	7.0621×
2d_conv_5	1.0000	0.1777	17.77%	5.6275×
2d_deconv_1	1.0000	0.2116	21.16%	4.7256×



Already at high sparsity

Increase sparsity from dense

Q Why theoretical metrics(FLOPs/Memory) have discrepancy with hardware measurement (Latency, Peak Memory)?



Computational flow of Sparse Convolution.The speedup does not linearly scale with data size
Less than **20%** density (5x sparsity) incur **1.5x** latency speedup

[1] Exploiting Hardware Utilization and Adaptive Dataflow for Sparse Convolution in 3D Point Clouds, MLSYS22

Analysis: Other Compression Method

Method	FLOPs Opt.	Mem. Opt.	mAP	KITTI Mod. Car.	KITTI Mod. Ped.	KITTI Mod. Cyc.
CenterPoint [28]	-	-	66.1	79.4	53.4	65.5
CenterPoint *(SPVNAS)*	1.07×	1.07×	65.5	79.2	52.1	65.3
CenterPoint *(SPVNAS+Ada3D)*	3.95×	4.35×	65.5	78.6	52.5	65.5

Could be combined with **Model-level Compression Method** to further reduce sparsity

CenterPoint [31] *(voxel=0.075)*	-	-	59.22	66.48
SPSS-Conv [15] *(voxel=0.075)*	1.14×	1.14×	57.80	65.69
CenterPoint-0.5W [31] *(voxel=0.075)*	2.78×	2.78×	57.19	64.08
CenterPoint-Ada3D *(voxel=0.075)*	3.34×	3.96×	58.62	65.68
VoxelNeXT [1]	-	-	60.50	66.60
VoxelNeXT-Ada3D [1]	1.19×	1.20×	59.75	65.84

Other Compression Method: *Only* optimize the *3D* backbone

Could be combined with recent **"Fully Sparse Detectors"** to further reduce sparsity

Analysis: Ablation Studies

Table 4: **Ablation studies and quantitve efficiency improvements of different Ada3D models on KITTI *val*.** "IP" stands for "importance predictor", "DG" for "density-guided spatial filtering", "SP-BN" for "sparsity preserving batch normalization". The "FLOPs" and "Mem." calculates the normalized resource consumption of the optimized model.

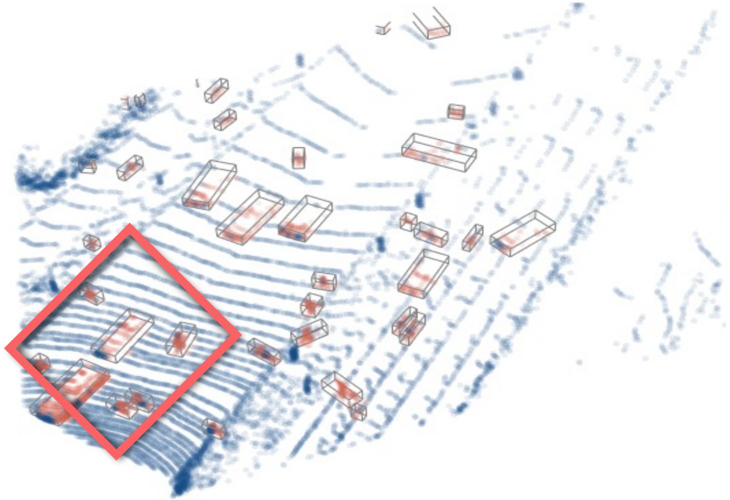
Method	*Technique*			*FLOPs*		*Mem.*		mAP	Car Mod.	Ped. Mod.	Cyc. Mod.
	IP	DG	SP-BN	*3D*	*2D*	*3D*	*2D*	(Mod.)	(IoU=0.7)	(IoU=0.5)	(IoU=0.5)
CenterPoint	-	-	-	1.00	1.00	1.00	1.00	66.1	79.4 (-)	53.4 (-)	65.5 (-)
CenterPoint (SP-BN)	-	-	✓	1.00	0.49	1.00	0.45	66.0	79.1 (-0.3)	53.3 (-0.1)	65.6 (+0.1)
CenterPoint (Ada3D-A)	✓	✓	✓	1.00	0.22	1.00	0.25	66.4	79.5 (+0.1)	53.6 (+0.2)	66.1 (+0.6)
CenterPoint (Ada3D-B)	✓	✓	✓	0.66	0.18	0.68	0.17	66.1	79.1 (-0.3)	54.0 (+0.6)	65.3 (-0.3)
CenterPoint (Ada3D-B w.o. DG)	✓	-	✓	0.64	0.18	0.66	0.16	65.1	78.8 (-0.6)	51.6 (-1.8)	64.9 (-0.6)
CenterPoint (Ada3D-C)	✓	✓	✓	0.39	0.08	0.43	0.07	65.4	77.6 (-1.8)	53.5 (+0.2)	65.1 (-0.4)

SP-BN reduces 50% 2D pixels
without sacrificing performance

1.0 less mAP drop
With Density Guidance

Analysis: Ablation Studies

➢ Effectiveness of BEV-space Importance Predictor



Point cloud painted with Predictor Heatmap

f_{score} IP	f_{score} DG	R_{drop}	R_{inbox} *3D*	R_{inbox} *2D*	KITTI Mod. AP Car.	KITTI Mod. AP Ped.	KITTI Mod. AP Cyc.
-	-	-	-	-	79.1	53.3	65.6
-	✓	25%	12.3%	9.4%	76.4	45.6	59.4
✓	-	25%	1.4%	1.1%	78.8	51.6	64.9
✓	✓	25%	**0.8%**	**0.0%**	**79.1**	**54.0**	**65.2**
-	✓	50%	17.6%	20.3%	72.1	39.4	55.6
✓	-	50%	6.8%	8.8%	76.9	50.2	63.7
✓	✓	50%	**5.2%**	**7.5%**	**77.6**	**53.5**	**65.1**

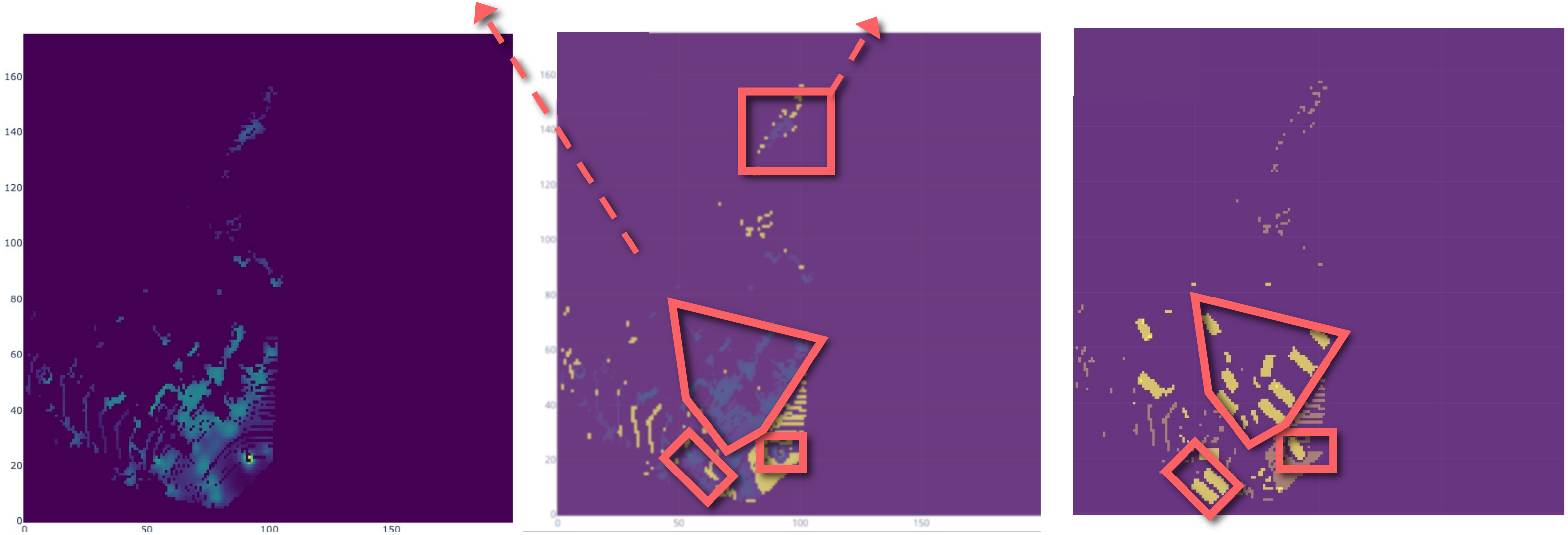
Low inbox rate: *Avoid dropping valuable points*

Analysis: Ablation Studies

➢ Effectiveness of Density-guided Filtering

Importance-Predictor:
(Filtered out boxes)

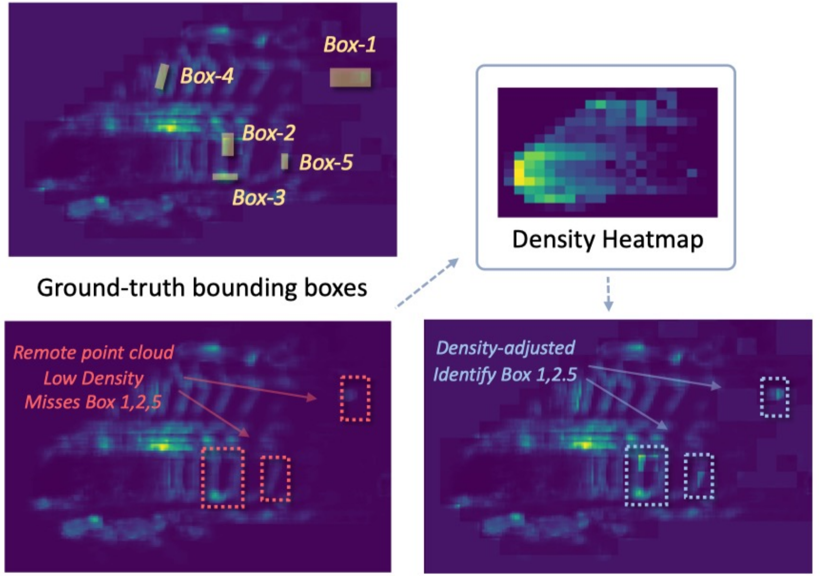
Density-Guidance:
(Keep remote sparse features)



Predicted Heatmap *Dropped Input* *Bounding Boxes*

Analysis: Ablation Studies

➢ **Density-guided Filtering:** Leverage the property of Lidar point cloud



Predictor importance score (without density-guidance)

Density-adjusted importance score (with density-guidance)

f_{score} IP	f_{score} DG	R_{drop}	R_{inbox} 3D	R_{inbox} 2D	KITTI Mod. AP Car.	KITTI Mod. AP Ped.	KITTI Mod. AP Cyc.
-	-	-	-	-	79.1	53.3	65.6
-	✓	25%	12.3%	9.4%	76.4	45.6	59.4
✓	-	25%	1.4%	1.1%	78.8	51.6	64.9
✓	✓	25%	**0.8%**	**0.0%**	**79.1**	**54.0**	**65.2**
-	✓	50%	17.6%	20.3%	72.1	39.4	55.6
✓	-	50%	6.8%	8.8%	76.9	50.2	63.7
✓	✓	50%	**5.2%**	**7.5%**	**77.6**	**53.5**	**65.1**

Low inbox rate: *Avoid dropping valuable points*

Thanks for Listening!

For ***more information about Ada3D***,
Please go to our **Project Page**:
https://a-suozhang.xyz/ada3d.github.io/
(🚧under construction🚧)
or contact me through
Email: suozhang1998@gmail.com
WeChat: ztc19980908

If you are interested in **Efficient Deep Learning Research**, Please go to our Group Website (NICS-EFC) for more information.
https://nicsefc.ee.tsinghua.edu.cn/
Visiting Student Welcomed!
We also conduct research about
Efficient AIGC tasks (e.g., LLM & Diffusion)

If you are interested in **Efficient and Intelligent Driving**, See NOVAUTO
A startup focuses on autonomous driving.
https://www.novauto.com.cn/

Project Page

WeChat