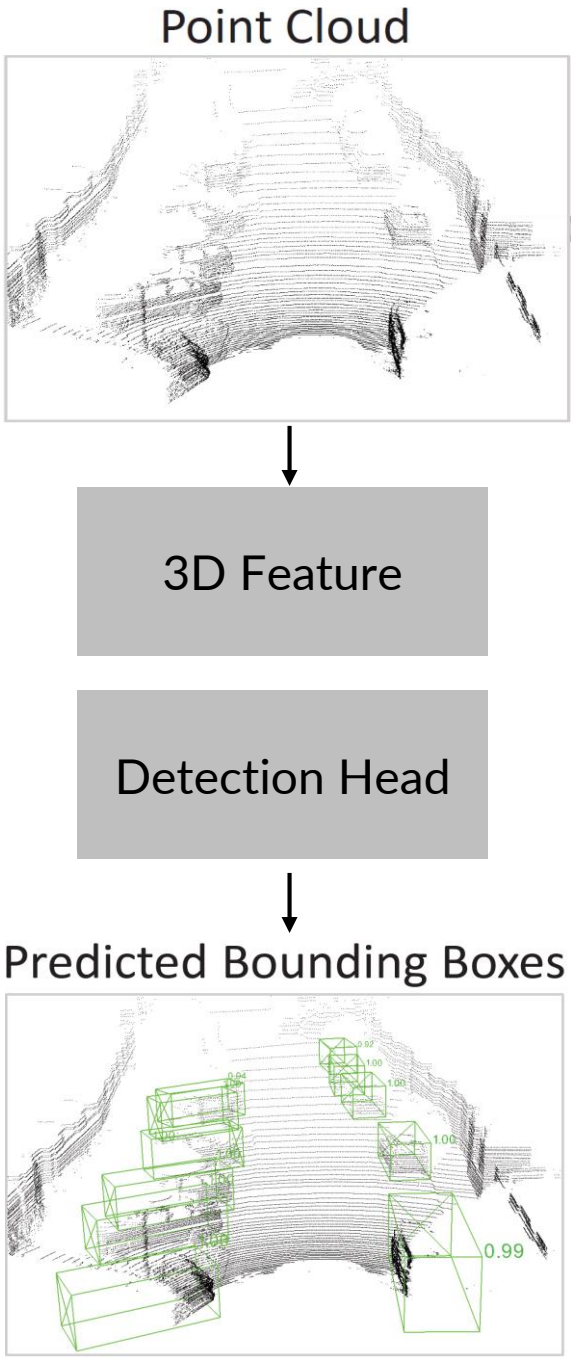
香港中文大學
The Chinese University of Hong Kong

MPPNet: Multi-Frame Feature Intertwining with Proxy Points for 3D Temporal Object Detection

Xuesong Chen*, Shaoshuai Shi*, Benjin Zhu, Ka Chun Cheung, Hang Xu, Hongsheng Li

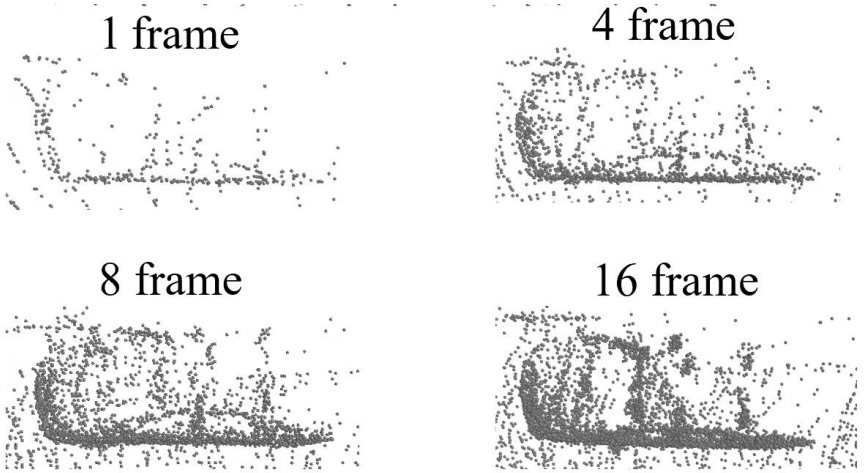
Background

- 3D object detection is the fundamental task for all autonomous driving systems
- 3D point clouds
  - Irregular sequences of points
  - Shape (N, 3): [x, y, z]
- 3D bounding box:
  - Localization: (cx, cy, cz)
  - Height, width, length
  - Heading direction in bird view
- Learning feature from point cloud
  - Point-based
  - Voxel-based



Motivation

- LiDAR sensors can only produce sparse point clouds
- Multi-frame point cloud sequences can help capture more complete object structure and position information
- However, naively concatenating multiple frames to train SOTA 3D object detectors doesn't necessarily improve the performance

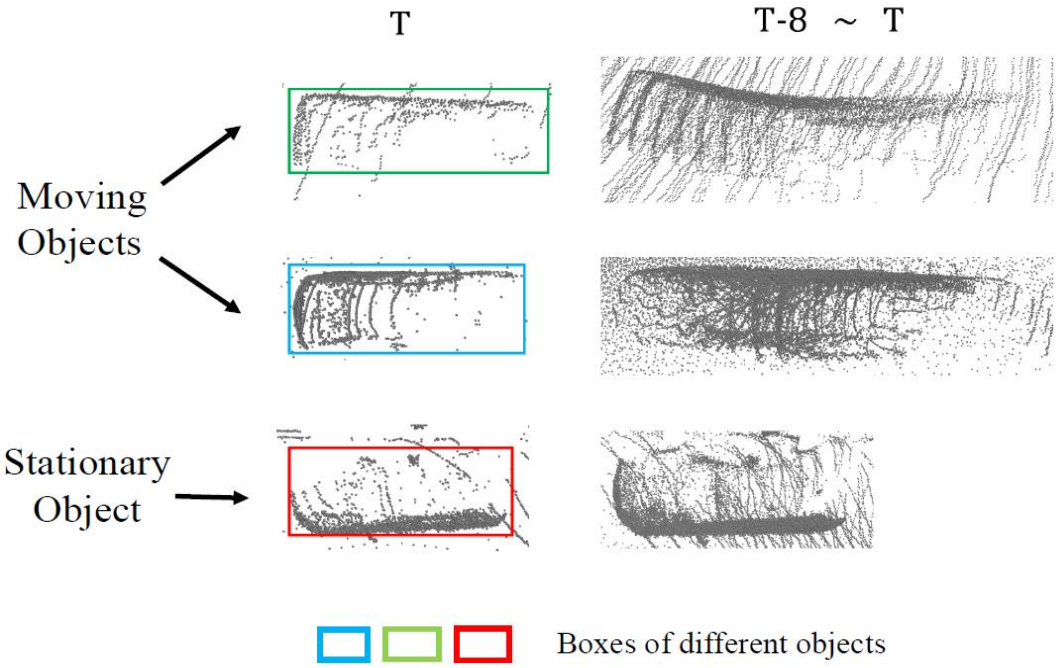


Pilot study with CenterPoint on Waymo validation set
Input: multi-frame concatenation

Frames	1-frame	4-frame	8-frame	12-frame	16-frame
mAPH@L2	64.50	**65.77**	65.69	65.33	64.69

Motivation

- Challenges of exploiting temporal information from point sequences
  - ◇ The concatenated point clouds of long sequences show various types of “tails”, posing challenges to the detectors
  - ◇ Large computation and memory cost for feature extraction from all LiDAR point in long sequences



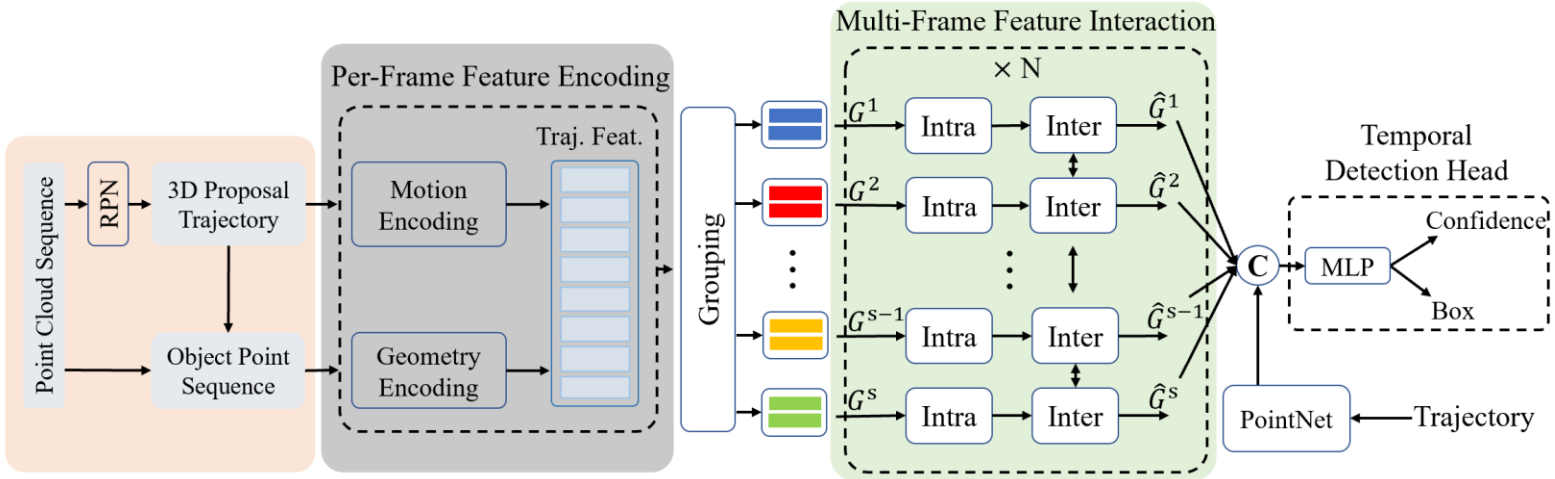
Pilot study with CenterPoint on Waymo validation set
Input: multi-frame concatenation

Frames	1-frame	4-frame	8-frame	12-frame	16-frame
mAPH@L2	64.50	**65.77**	65.69	65.33	64.69

Methods

- Two-stage framework
  - ◇ Stage-1: Trajectory proposal generation from past per-frame detections
  - ◇ Stage-2: Refine detection boxes at current time step
- Stage-2: a three-hierarchy architecture to reduce cost when processing long sequences

Stage-1: Trajectory Proposal Generation



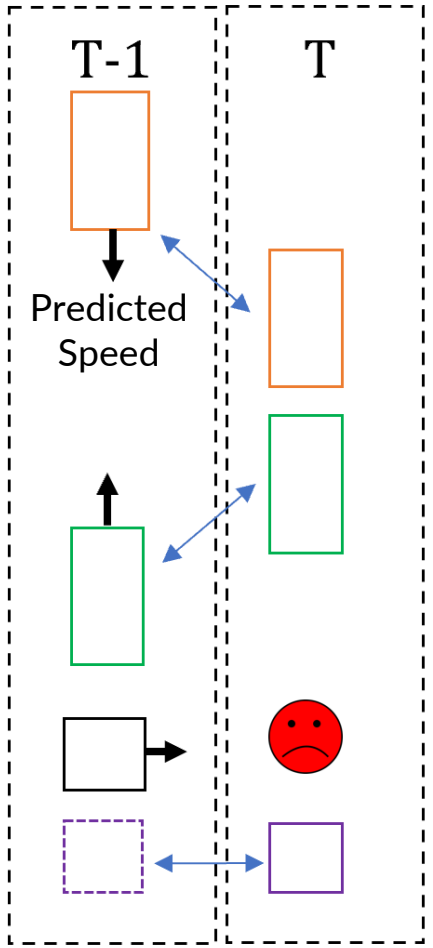
Stage-2: Box Refinement

Hierarchy I: Per-Frame Feature Encoding

Hierarchy II & III: Group-level and Trajectory-level Feature Propagation

Stage-1: Proposal Trajectory and Proposal Point Sequence

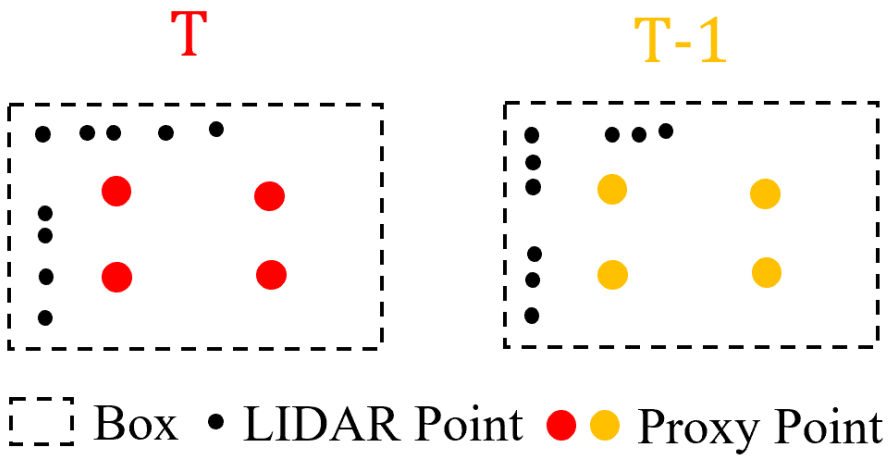
- **IoU based matching to generate trajectory proposal**
  - A 3D detector with 4-frame concatenation inputs to generate per-frame boxes and box speeds
  - Speed is employed to align proposals from different frames
  - The number of trajectory is equal to that of proposals at current time
  - Pad all trajectories to the same length (using the proposals of the current time step)
- **Using trajectory proposal to crop each proposal object's point sequence**
  - For each frame, random sample 128 LiDAR point for each proposal box



3 trajectories with length 2

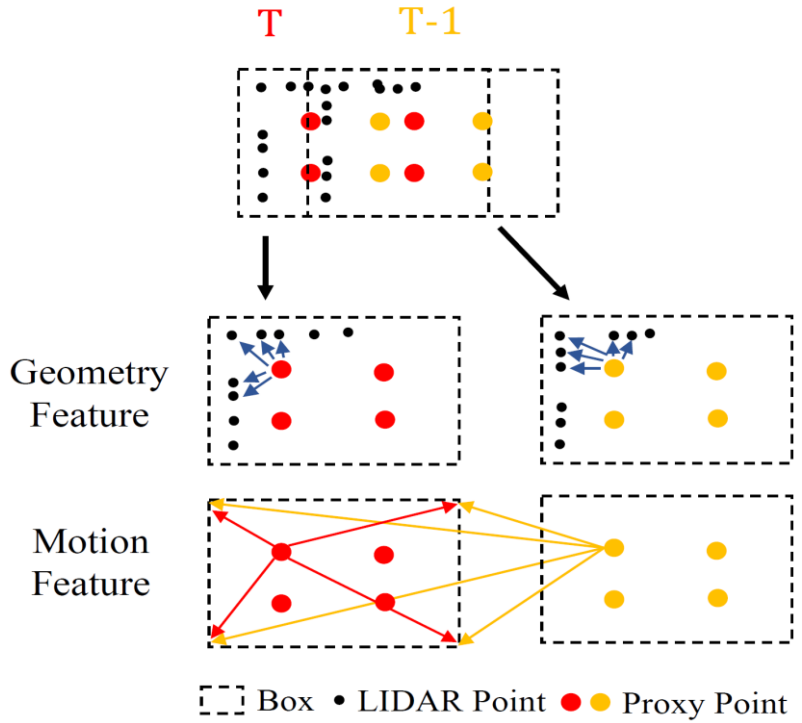
Stage-2: Proxy Points for Each Proposal Box

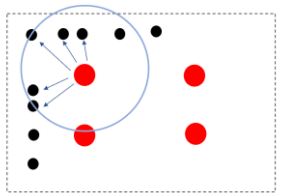
- **Why introducing proxy points?**
  - The number of point clouds of an object in different frames varies.
  - There is no definite cross-frame correspondence between points of the same object
- **What are proxy points?**
  - Uniformly distributed on the grid of each 3D proposal box and with time encoding
  - Fixed and consistent relative positions in each 3D proposal box
- **Advantages of proxy points**
  - Automatically aligned feature across different frames
  - Help to encode motion information



Hierarchy I: Per-frame Encoding with Proxy Point

- **Proposal-aware LiDAR point feature encoding**
  - ◇ Compute LiDAR points relative positions to the 8 corner point and 1 center point of the proposal (N, 3) → (N, 3x9)
  - ◇ A 3-layer MLP to encode points' proposal-aware positions
- **Decoupled object feature encoding with proxy points**
  - ◇ ***Geometric features:*** using set abstraction to pool LiDAR point features to the proxy points
  - ◇ ***Motion features:*** 1) Proxy points' relative displacements to ***current time T's*** box points; 2) Another 3-layer MLP for encoding
  - ◇ During inference, past frames' geometry feature can be stored to avoid redundant computation

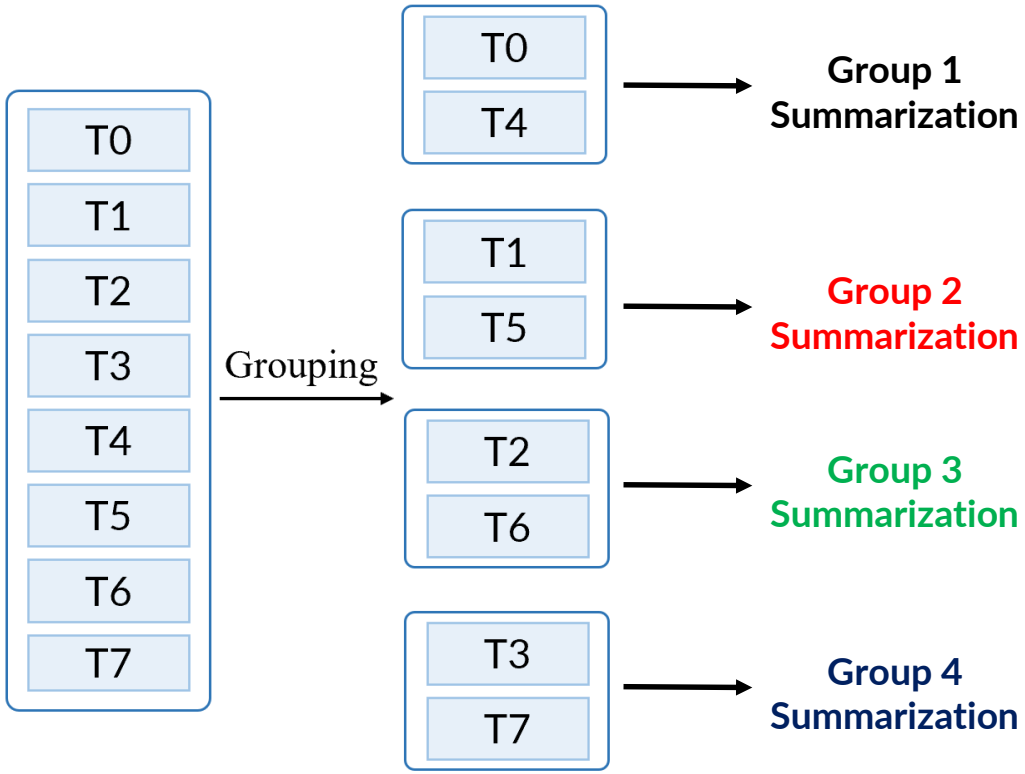




Set abstraction

Grouping

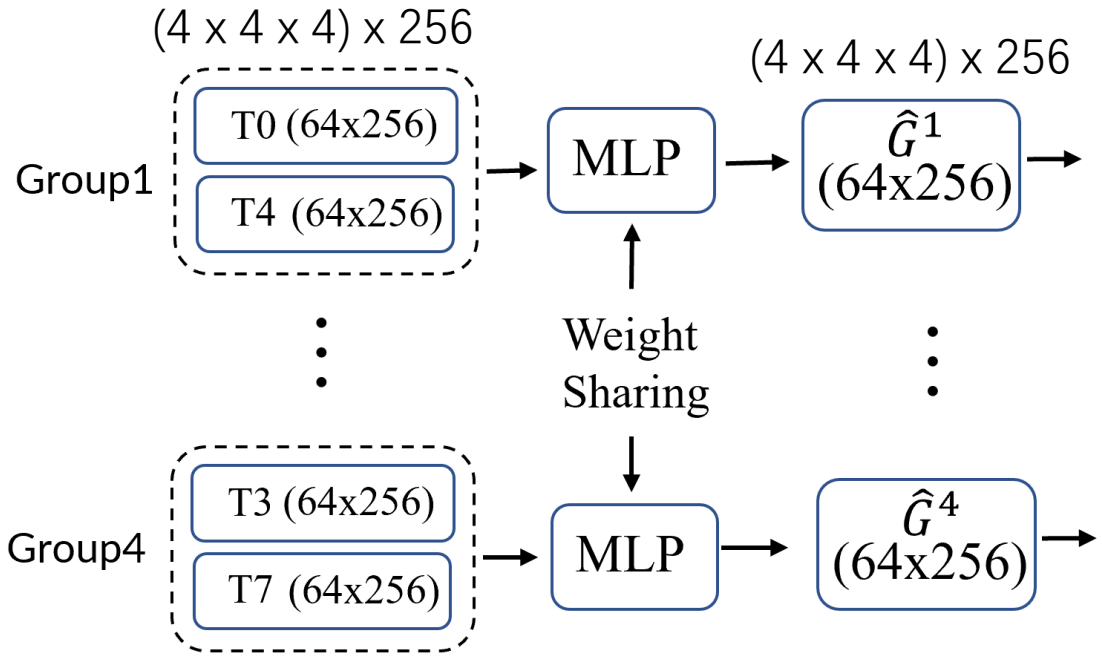
- Directly handling proxy points using attention of all frames are still time- and memory-consuming
- Divide a sequence into multiple groups and each group includes a sub-trajectory



Example: Grouping 8 frames into 4 groups

Initial Group Fusion

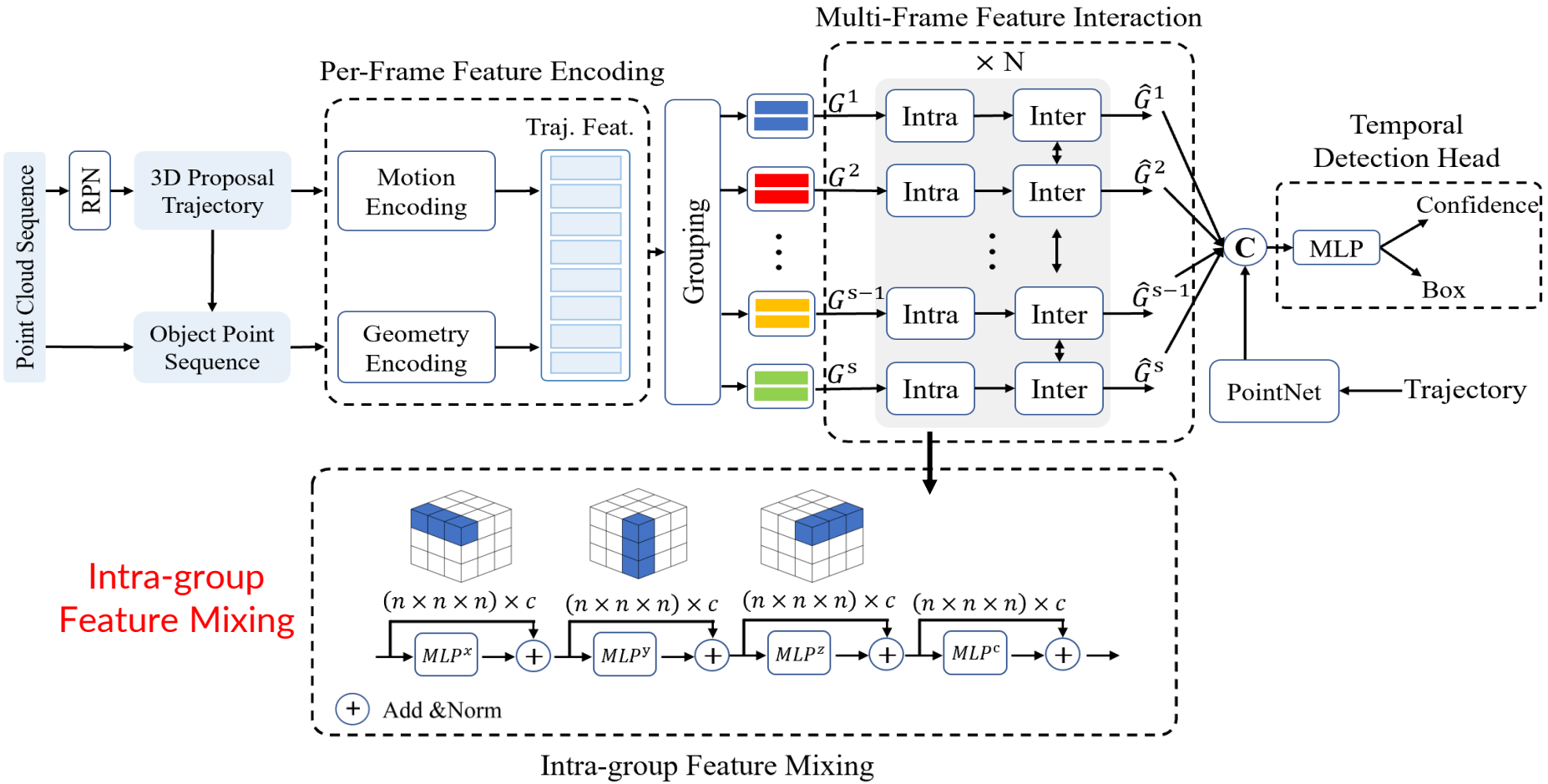
- Reduce each group's feature to a compact representation
- Intra- and inter-group feature fusion are conducted on grouped features, instead of each frame, to save computation and memory



Example: Grouping 8 frames into 4 groups

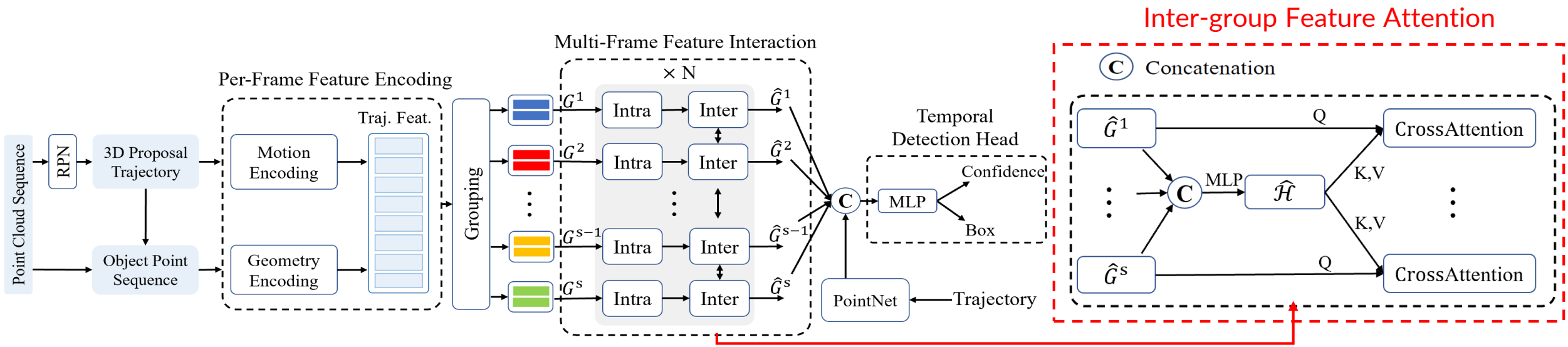
Hierarchy II: Intra-group Feature Mixing

- **Conduct feature fusion within each group via a 3D MLP-mixer**
  - ◇ Propagate information on *x*, *y*, *z*, and channel dimension with four 3-layer MLPs



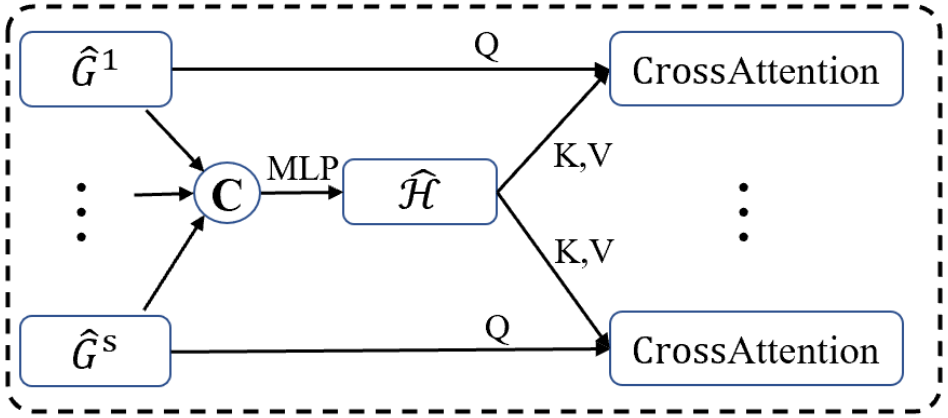
Hierarchy II: Inter-group Feature Attention

- **Take advantage of long sequences**
  - ◇ Linear projection of per-frame proxy-point features to query vectors
  - ◇ All proxy-point features projected to whole-sequence summarization key & value vectors
  - ◇ Employ cross-attention mechanism to update per-frame proxy-point features
  - ◇ Alternate and iterative intra- and inter-group fusion



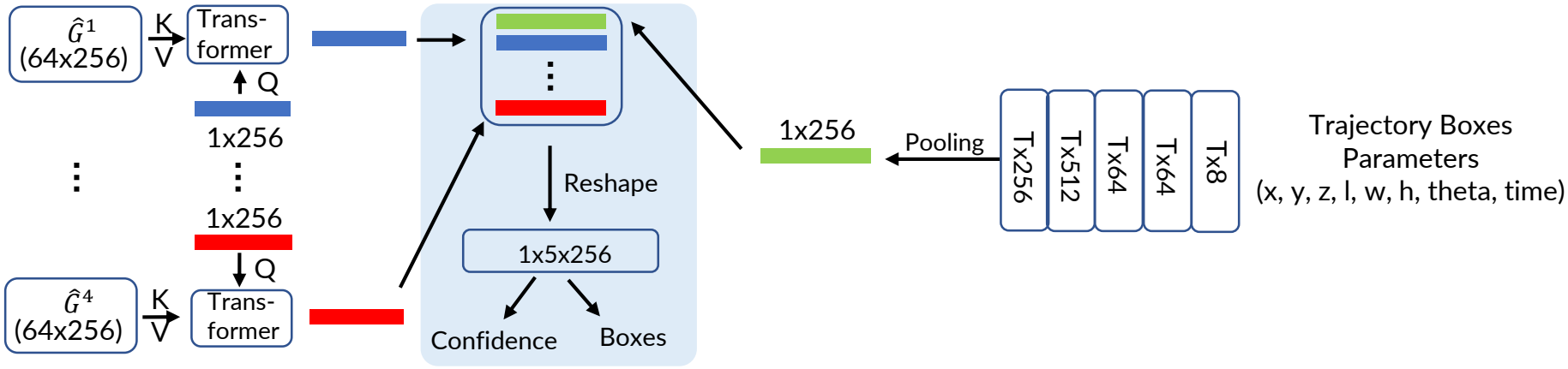
Hierarchy II: Inter-group Feature Attention

- **Design of cross attention**
  - ◇ **Index based Position Embedding:** 3-dim indexes from [0, 0, 0] to [N, N, N] to index all proxy point in each proposal and an MLP to encode the 3-dim index to position embedding
  - ◇ Use an MLP to mix all groups' proxy points to create summarizations of all the groups
  - ◇ Employ each group's proxy point as **Query** and the all-group summarization as shared **Key** and **Value**



Detection Head

- Each intra-group module is followed by a detection head, with trajectory boxes & trajectory proxy-point features as inputs
- ***Trajectory boxes*** are encoded by a PointNet
- ***Proxy points*** of each group is summarized by a transformer block
  - ◇ A zero-initialized learnable vector serves as the query (Q) to aggregate proxy points' feature (K, V) for each group with multi-head attention
- Queries are iteratively updated after each intra-group module and are supervised by losses
- Concat (trajectory boxes, proxy points) → FC layer → Confidence + Refined Box



Loss Function and Train Strategy

- The two stages of MPPNet are trained separately
  - ◇ Stage-1: follow the official training strategy and losses of CenterPoint with 4-frame inputs
- Loss functions of stage-2
  - ◇ Confidence: Cross Entropy
  - ◇ Boxes loss: regression loss of sizes (x, y, z, l, w, h, theta) and 8 corners' coordinates
  - ◇ Total loss:

$$\mathcal{L} = \mathcal{L}_{\text{conf}} + \alpha \mathcal{L}_{\text{reg}}$$

α is 2 in our experiments

Experiments

Waymo validation set results

Method	Frames	ALL (3D APH)		VEH (3D AP/APH)		PED(3D AP/APH)		CYC(3D AP/APH)	
		L1	L2	L1	L2	L1	L2	L1	L2
SECOND [36]	1	63.05	57.23	72.27/71.69	63.85/63.33	68.70/58.18	60.72/51.31	60.62/59.28	58.34/57.05
PointPillar [12]	1	63.33	57.53	71.60/71.00	63.10/62.50	70.60/56.70	62.90/50.20	64.40/62.30	61.90/59.90
LiDAR R-CNN [13]	1	66.20	60.10	73.50/73.00	64.70/64.20	71.20/58.70	63.10/51.70	68.60/66.90	66.10/64.40
RSN [28]	1	-	-	75.10/74.60	66.00/65.50	77.80/72.70	68.30/63.70	-	-
Pyramid [16]	1	-	-	76.30/75.68	67.23/66.68	-	-	-	-
PV-RCNN [23]	1	69.63	63.33	77.51/76.89	68.98/68.41	75.01/65.65	66.04/57.61	67.81/66.35	65.39/63.98
Part-A2 [26]	1	70.25	63.84	77.05/76.51	68.47/67.97	75.24/66.87	66.18/58.62	68.60/67.36	66.13/64.93
Centerpoint [40]	1	-	65.50	-	-/66.20	-	-/62.60	-	-/67.60
CT3D [22]	1	-	-	-	69.04/-	-	-	-	-
PV-RCNN++ [24]	1	75.21	68.61	79.10/78.63	70.34/69.91	80.62/74.62	71.86/66.30	73.49/72.38	70.70/69.62
3D-MAN [39]	16	-	-	74.53/74.03	67.61/67.14	-	-	-	-
† Centerpoint [40]	4	74.88	69.38	76.71/76.17	69.13/68.63	78.88/75.55	71.73/68.61	73.73/72.96	71.63/70.89
† CT3D-MF [22]	12	-	-	79.30/78.82	71.82/70.84	-	-	-	-
† CT3D-MF [22]	16	-	-	79.04/78.55	71.14/70.68	-	-	-	-
MPPNet (Ours)	4	79.83	74.22	81.54/81.06	74.07/73.61	84.56/81.94	77.20/74.67	77.15/76.50	75.01/74.38
MPPNet (Ours)	16	**80.40**	**74.85**	**82.74/82.28**	**75.41/74.96**	**84.69/82.25**	**77.43/75.06**	**77.28/76.66**	**75.13/74.52**

CT3D-MF is implemented by us as a baseline to MPPNet, which uses the same RPN model

Experiments

Waymo testing set results

Method	Frames	ALL (3D APH) L1	ALL (3D APH) L2	VEH (3D AP/APH) L1	VEH (3D AP/APH) L2	PED(3D AP/APH) L1	PED(3D AP/APH) L2	CYC(3D AP/APH) L1	CYC(3D AP/APH) L2
StarNet [17]	1	-	-	61.50/61.0	54.90/54.50	67.80/59.90	61.10/54.00	-	-
PointPillar [12]	1	-	-	68.60/68.1	60.50/60.10	68.00/55.50	61.40/50.10	-	-
CenterPoint [40]	1	-	-	80.20/79.70	72.20/71.80	78.30/72.10	72.20/66.40	-	-
PV-RCNN++ [24]	1	75.65	70.21	81.62/73.86	81.20/73.47	80.41/74.12	74.99/69.00	71.93/69.28	70.76/68.15
RSN [28]	3	-	-	80.70/80.30	71.90/71.60	78.90/75.60	70.70/67.80	-	-
Centerpoint [40]	2	77/18	71.93	81.05/80.59	73.42/72.99	80.47/77.28	74.56/71.52	74.60/73.68	72.17/71.28
PV-RCNN Ens [23]	2	76.90	71.52	81.06/80.57	73.69/73.23	80.31/76.28	73.98/70.16	75.10/73.84	72.38/71.16
Pyramid [16]	2	-	-	81.77/81.32	74.87/74.43	-	-	-	-
3D-MAN [39]	16	-	-	78.71/78.28	70.37/69.98	69.97/65.98	63.98/60.26	-	-
MPPNet (Ours)	16	**80.59**	**75.67**	**84.27/83.88**	**77.29/76.91**	**84.12/81.52**	**78.44/75.93**	**77.11/76.36**	**74.91/74.18**

Note: TTA (test time augmentation) and model ensemble are not used

Experiments

Waymo testing set results (with TTA and model ensemble)

Method Name	Object Type (ALL_NS)	Sensors (All)	Frames [-p, +f]	Latency (s) (Show all)	AP / L1	APH / L1	AP / L2	APH / L2	Date (Pacific Daylight Time)
1 MPPNetEns-MMLab	ALL_NS	L	[-15, +0]		0.8548	0.8414	0.8091	0.7960	2022-09-02 13:57
2 BEVFusion-TTA	ALL_NS	CL	[-2, +0]		0.8568	0.8438	0.8080	0.7953	2022-08-16 23:04
3 3DAM_Ens-Shanghai AI Lab	ALL_NS	L	[-4, +0]		0.8528	0.8378	0.8065	0.7919	2022-07-19 01:40
4 LIVOX_Detection	ALL_NS	L	[-6, +0]		0.8482	0.8354	0.8022	0.7896	2022-05-10 21:18
5 MT3D	ALL_NS	L	[-3, +0]		0.8503	0.8367	0.8006	0.7873	2022-06-15 05:31
6 MT-Net	ALL_NS	L	[-2, +0]		0.8470	0.8322	0.7989	0.7845	2022-07-10 22:27
7 DeepFusion-Ens	ALL_NS	CL	[-4, +0]		0.8437	0.8322	0.7954	0.7841	2022-03-15 07:59
8 3dal-ens	ALL_NS	L	[--4, +0]		0.8463	0.8309	0.7968	0.7820	2022-07-02 02:08
9 InceptioLidar	ALL_NS	L	[-9, +0]		0.8380	0.8246	0.7915	0.7784	2022-02-28 23:09
10 AFDetV2-Ens	ALL_NS	L	[-1, +0]		0.8407	0.8263	0.7904	0.7764	2021-12-06 21:18

Experiments

Performance comparison with 3D-MAN

Method	Frames	mAPH@L2
PointPillar	1	54.69
PointPillar-ES	1	55.13
PointPillar-FT	1	64.37
3D-MAN w/ PointPillar	16	67.14 (+12.45)
Ours w/ PointPillar-ES	16	71.39 (+16.26)
Ours w/ PointPillar-FT	16	72.90 (+8.53)

Effects of numbers of proxy points

# (Proxy Point)	mAPH@L2
$3 \times 3 \times 3$	72.54
$4 \times 4 \times 4$	73.08
$5 \times 5 \times 5$	72.98

Effects of input trajectory length

Method	CT3D-MF	MPPNet
4-frame	70.19	72.63
8-frame	70.71 (+0.52)	73.22 (+0.59)
12-frame	70.84 (+0.65)	73.55 (+0.92)
16-frame	70.68 (+0.49)	73.81 (+1.18)

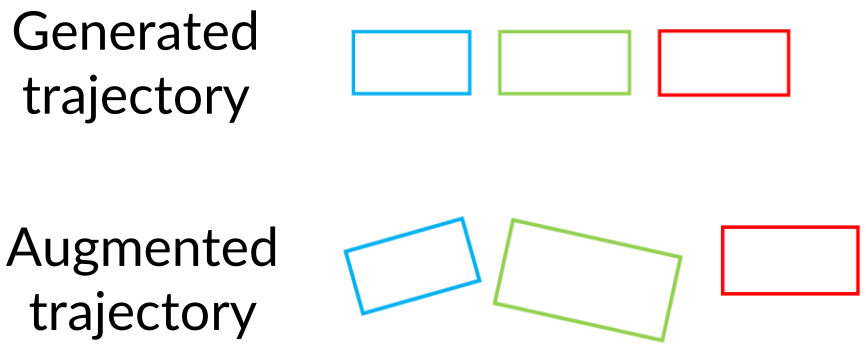
Effects of the numbers of recurrency of intra-inter block

Iteration of intra-inter block	mAPH@L2
1	72.78
2	73.08
3	73.02

Experiments

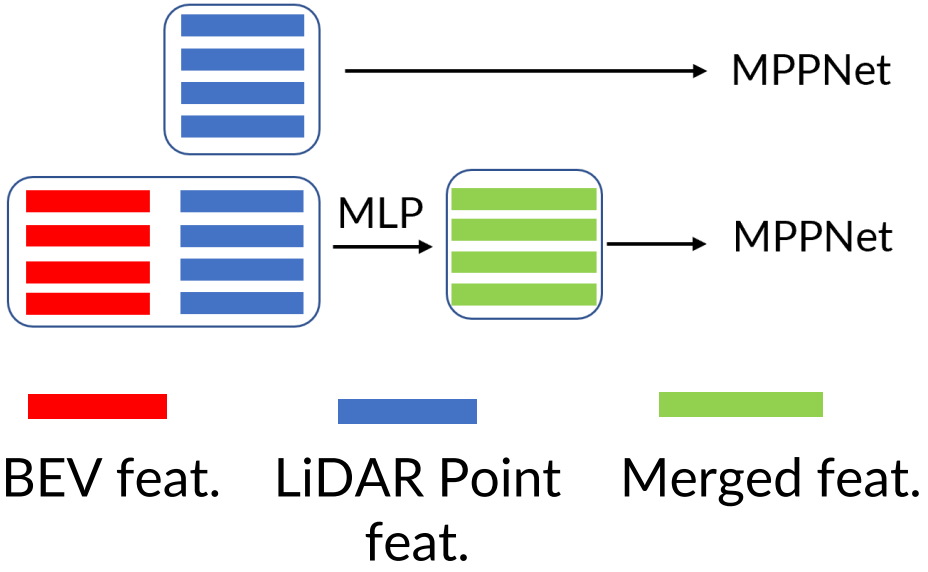
Effects of trajectory augmentation, intermediate supervision and grouping strategy

Training Strategy	MPPNet	w/o Traj. Aug	w/o Int. Loss
	73.08	72.62 (-0.46)	72.47 (-0.61)
Grouping Strategy	Stride 4	Stride 1	-
	74.21	74.06	-



Effects of using RPN's BEV feature

Feature Source	mAPH@L2
Point feat.+ BEV feat.	72.99
Point feat.	73.08



Experiments

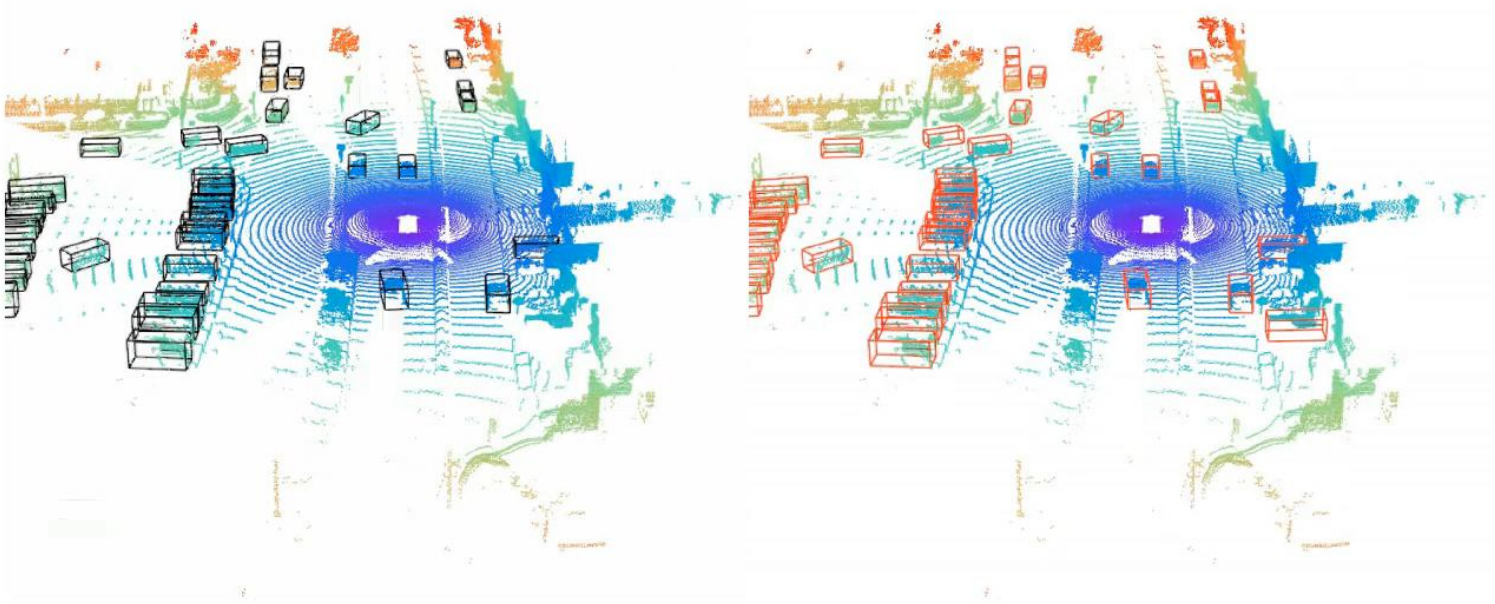
Effects of different components in MPPNet

Proxy Point	Boxes's Embedding	Per-frame Feature Encoding	Multi-frame Feature Fusion Intra	Multi-frame Feature Fusion Inter	mAPH@L2
✓	✓	Geometry+Motion	3D MLP Mixer	Cross-Attn	73.08
×	✓	Geometry+Motion	Self-Attn	Cross-Attn	71.13 (-1.95)
✓	✓	Geometry+Motion	Self-Attn	Cross-Attn	72.71 (-0.37)
✓	✓	Geometry	3D MLP Mixer	Cross-Attn	72.78 (-0.30)
✓	✓	Integrated	3D MLP Mixer	Cross-Attn	72.89 (-0.19)
✓	✓	Geometry+Motion	3D MLP Mixer	×	72.36 (-0.72)
✓	✓	Geometry+Motion	3D MLP Mixer	Cross-Attn w/o PE	72.97 (-0.11)
✓	✓	Geometry+Motion	3D MLP Mixer	Cross-Attn w/o Sum.	72.95 (-0.13)
✓	×	Geometry+Motion	3D MLP Mixer	Cross-Attn	72.98 (-0.10)

All components proposed in MPPNet contribute to the final performance

Conclusion

- We propose a two stage temporal 3D point clouds object detection framework
- Proxy points help to implicitly align points to facilitate inter-frame info fusion
- Intra- and inter-group feature fusion for achieving efficient multi-frame fusion



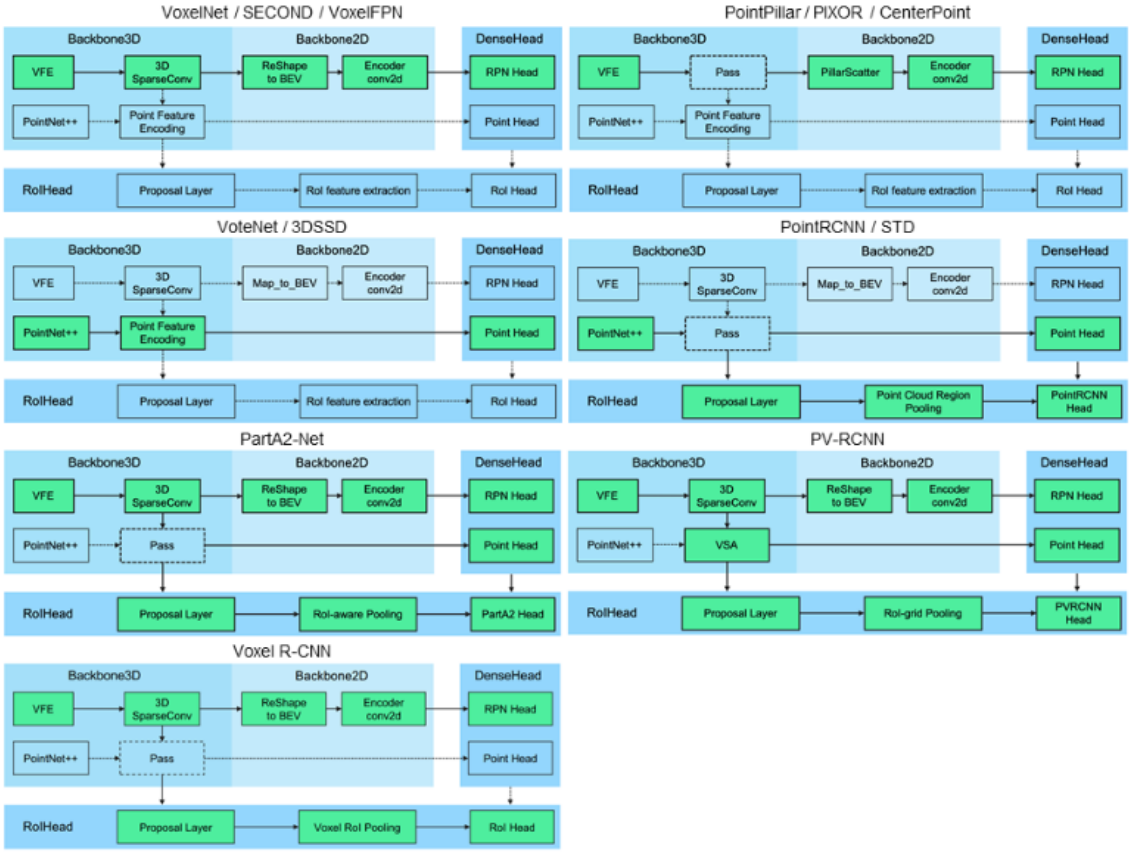
Non-empty Ground-truth Boxes

MPPNet Prediction Boxes

OpenPCDet

- Code is available at OpenPCDet: https://github.com/open-mmlab/OpenPCDet

Supports a large family of different 3D detectors



A large pool of model zoos

Performance@(train with 100% Data)	Vec_L1	Vec_L2	Ped_L1	Ped_L2	Cyc_L1	Cyc_L2
SECOND	72.27/71.69	63.85/63.33	68.70/58.18	60.72/51.31	60.62/59.28	58.34/57.05
CenterPoint-Pillar	73.37/72.86	65.09/64.62	75.35/65.11	67.61/58.25	67.76/66.22	65.25/63.77
Part-A2-Anchor	77.05/76.51	68.47/67.97	75.24/66.87	66.18/58.62	68.60/67.36	66.13/64.93
PV-RCNN (CenterHead)	78.00/77.50	69.43/68.98	79.21/73.03	70.42/64.72	71.46/70.27	68.95/67.79
PV-RCNN++	79.10/78.63	70.34/69.91	80.62/74.62	71.86/66.30	73.49/72.38	70.70/69.62
PV-RCNN++ (ResNet)	79.25/78.78	70.61/70.18	81.83/76.28	73.17/68.00	73.72/72.66	71.21/70.19
PV-RCNN++ (ResNet, 2 frames)	80.17/79.70	72.14/71.70	83.48/80.42	75.54/72.61	74.63/73.75	72.35/71.50
MPPNet (4 frames)	81.54/81.06	74.07/73.61	84.56/81.94	77.20/74.67	77.15/76.50	75.01/74.38
MPPNet (16 frames)	82.74/82.28	75.41/74.96	84.69/82.25	77.43/75.06	77.28/76.66	75.13/74.52

Thank you!