Geometric Modeling and Processing 2024 18TH

# PointeNet: A Lightweight Framework for Effective and Efficient Point Cloud Analysis

**Lipeng Gu**, Xuefeng Yan, Liangliang Nan, Dingkun Zhu,
Honghua Chen, Weiming Wang, Mingqiang Wei

Nanjing University of Aeronautics and Astronautics
Delft University of Technology
Hong Kong Metropolitan University

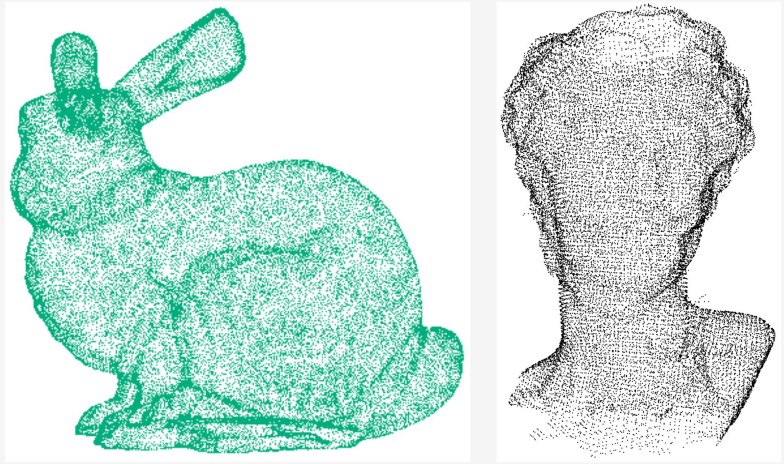

Co-organizers：

# Background

# Point Clouds

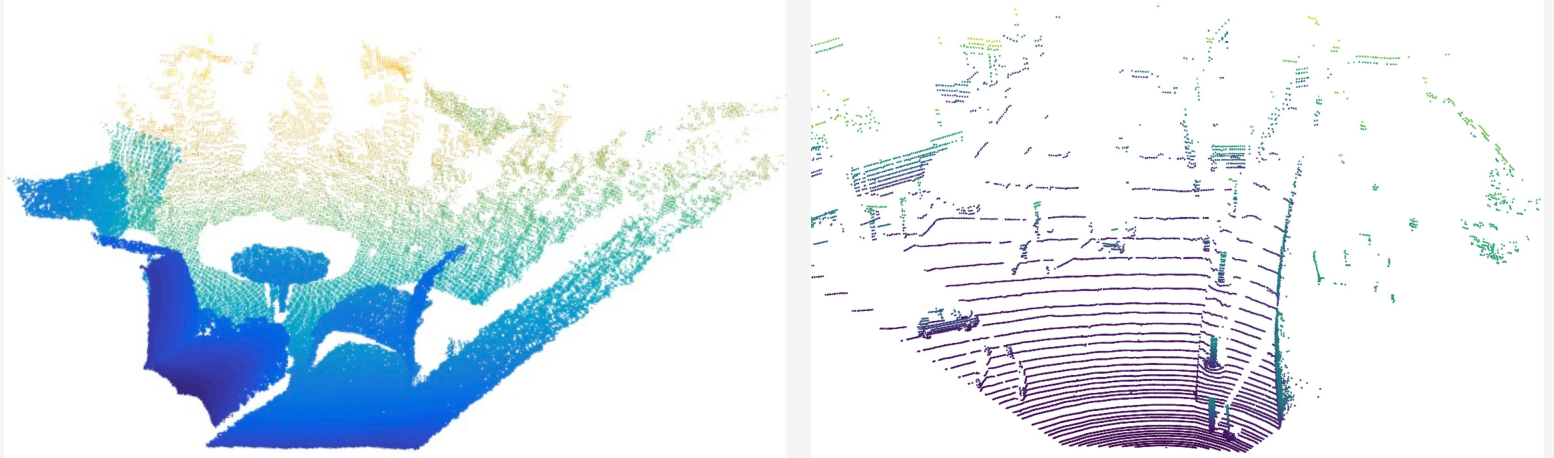
- **Irregular** and **unordered** points in 3D space
- LiDAR, RGB-D camera, and multi-view stereo

Object-level



Small size & evenly distributed

Scene-level



Large size & unevenly distributed at different distances

Figure 1: Some examples of point clouds

# Applications

- 3D Shape Classification
- Part Segmentation
- 3D Object Detection
- …

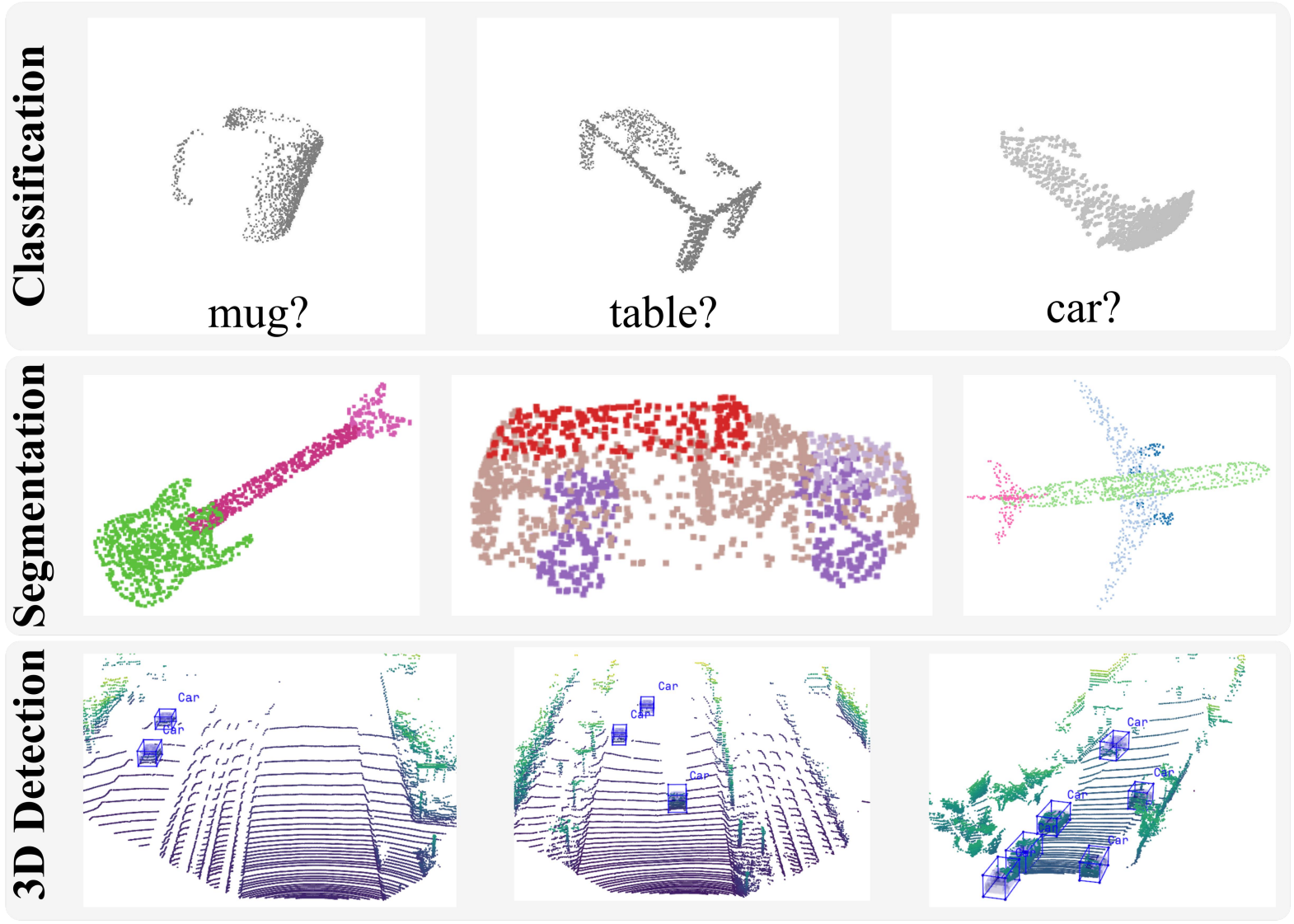



Figure 2: Applications of the point cloud analysis task.

# Related Works

- **PointNet** is the **pioneering** work

**Elaborate local feature extractor is crucial!**

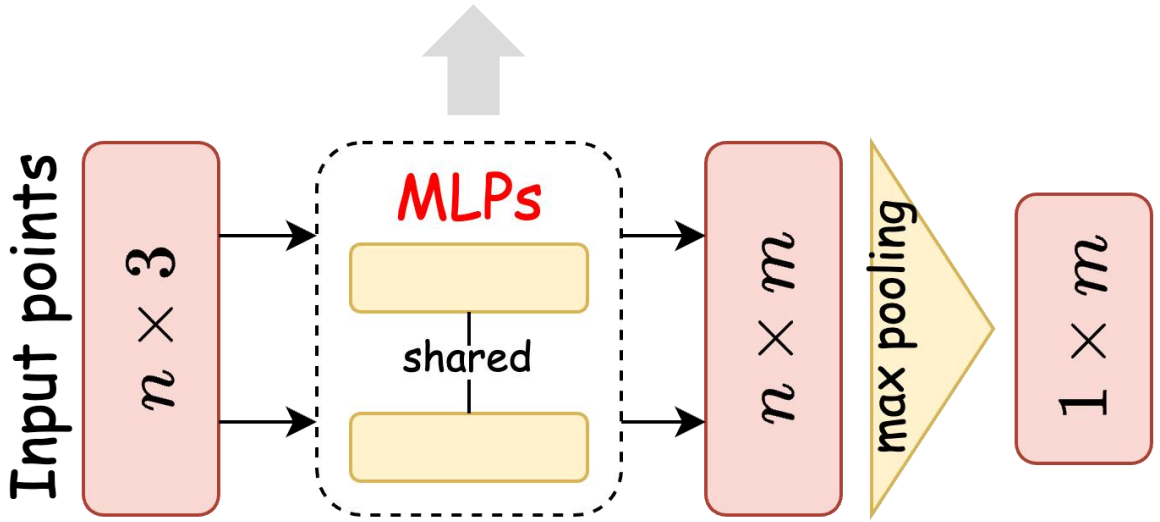



Figure 3: A simplified architecture of PointNet for point cloud classification.

# Related Works

## Elaborate local feature extractor

- **Convolution-based** [PAConv (Xu et al., 2021)、KPConv (Thomas et al., 2019)]
- **Graph- based** [3D-GCN (Lin et al., 2021)、DGCNN (Wang et al., 2019)]
- **Attention-based** [Point Transformer (Zhao et al., 2021)、PCT(Guo et al., 2021)]

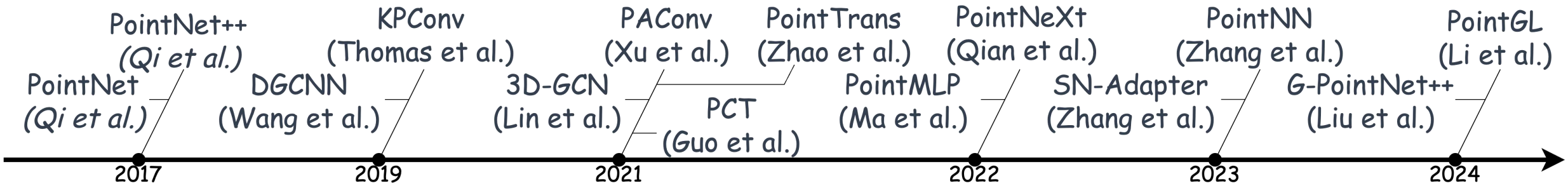



Figure 4: A chronological overview of deep learning-based point cloud methods.

# Related Works

**The burden of the network continues to increase**

- design **intricate** learnable operators
- **deepen** the network by stacking repeated blocks

## Self-attention Layers

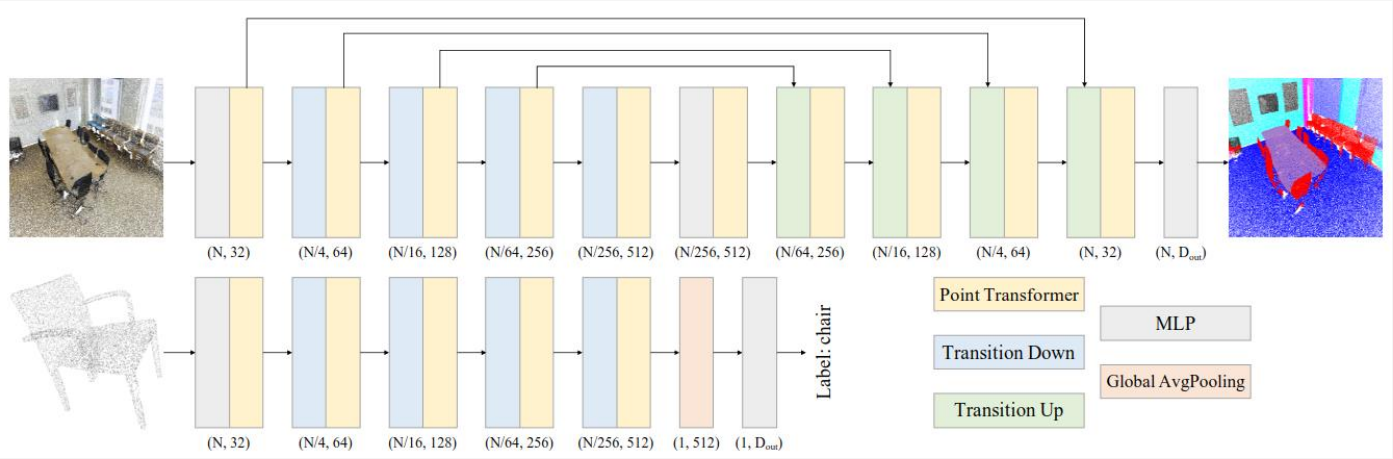



Figure 5: Point Transformer (Zhao et al., 2021)

## Deeper Network

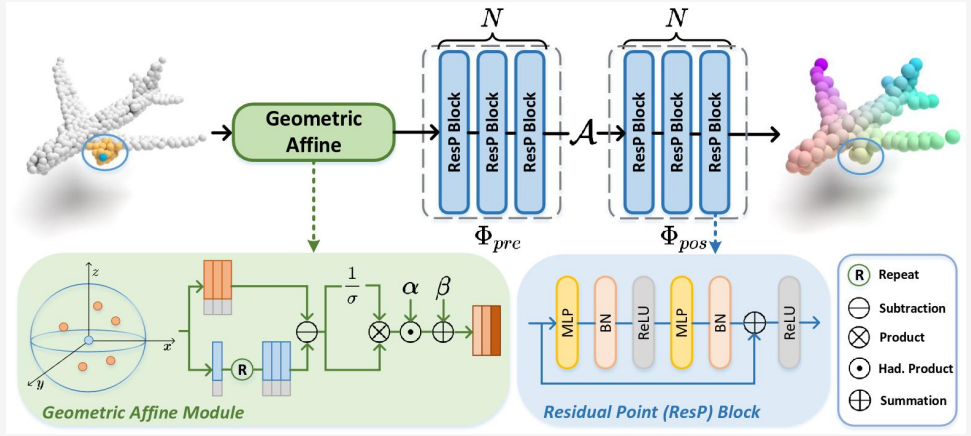



Figure 6: PointMLP (Ma et al., 2022)

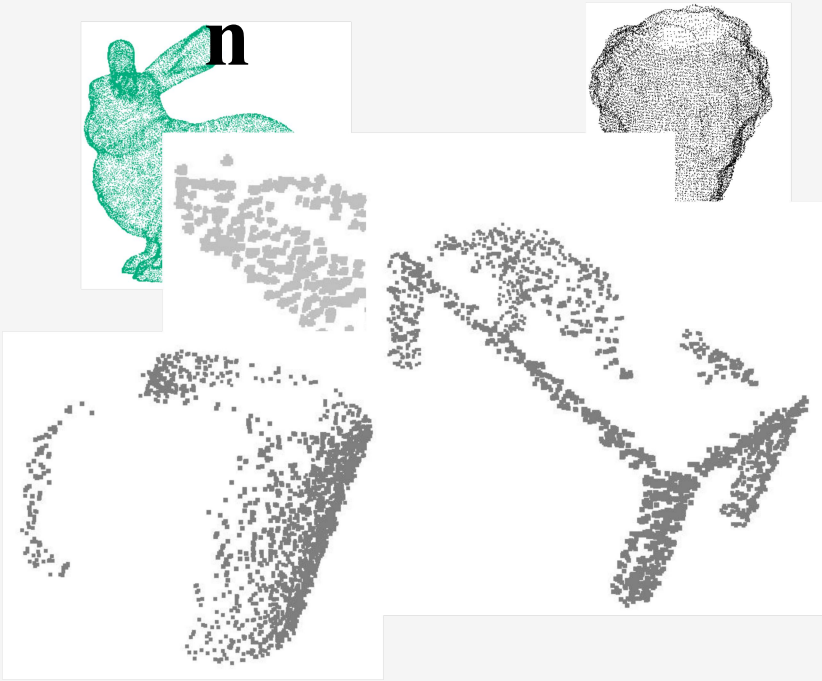
# Challenges



## Cost

- Learnable parameters
- Computational and memory burdens

## Applicatio n



- Tailored for single objects
- Not optimized for whole scenes

# Motivation



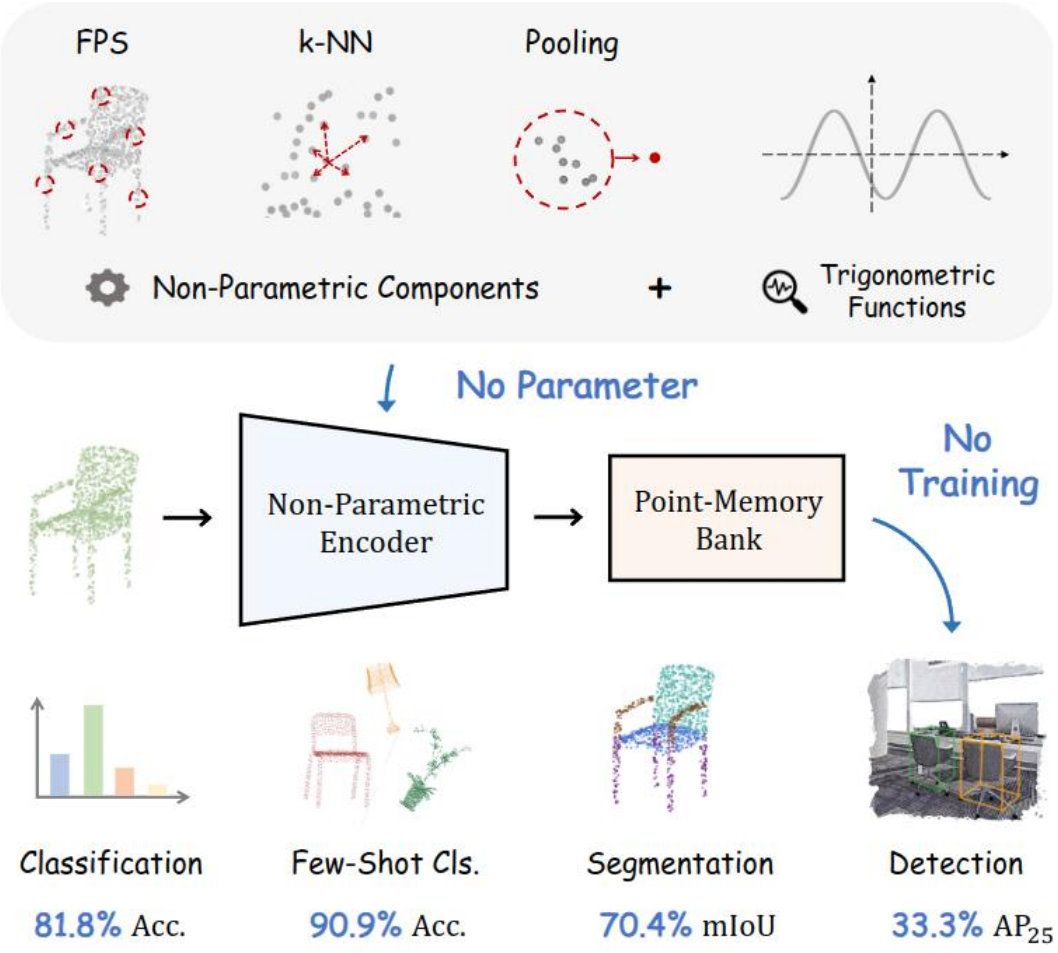



Figure 7: Point-NN (Zhang et al., 2023c)

- Non-Parametric Network
- Only neighborhood information

**Problem**

- **Simple** and **singular** features
- **Poor** performance
- Not optimized for scene-level tasks

**Solution**

- More point cloud features
- More attention to distant points
- Lightweight structure

# Methodology

- ✓ **Lightweight** [8.73% of PointMLP's parameters on ModelNet40]
- ✓ **Low training time** [38.7% of PointMLP's training time on ModelNet40]
- ✓ **Plug-and-play** [can be embedded into arbitrary 3D object detection network]

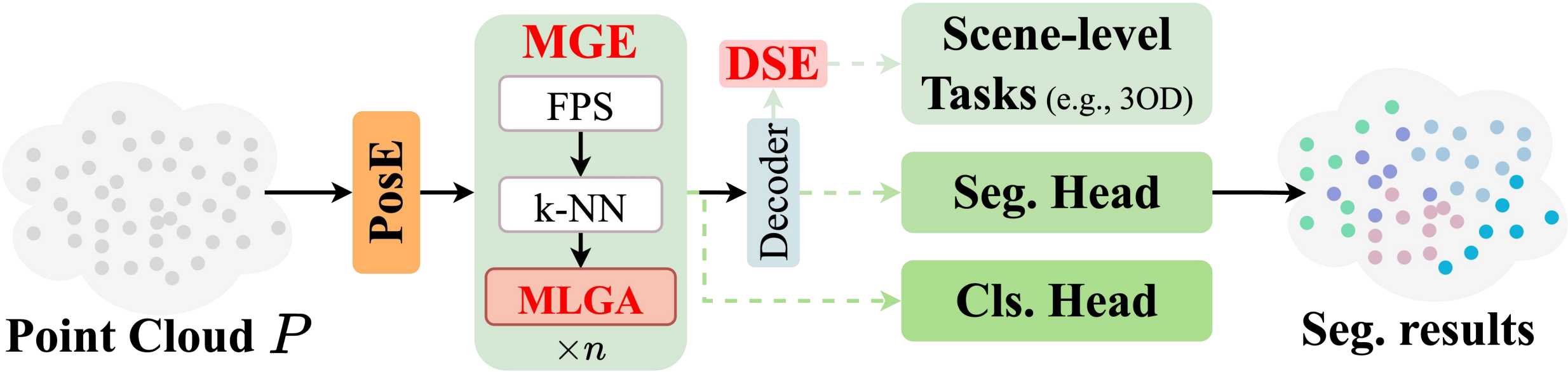



Figure 9: Overview of PointeNet

- Multivariate Geometric Encoding (**MGE**) module
- Distance-aware Semantic Enhancement (**DSE**) module
- Segmentation/classification head or arbitrary 3D object detection network

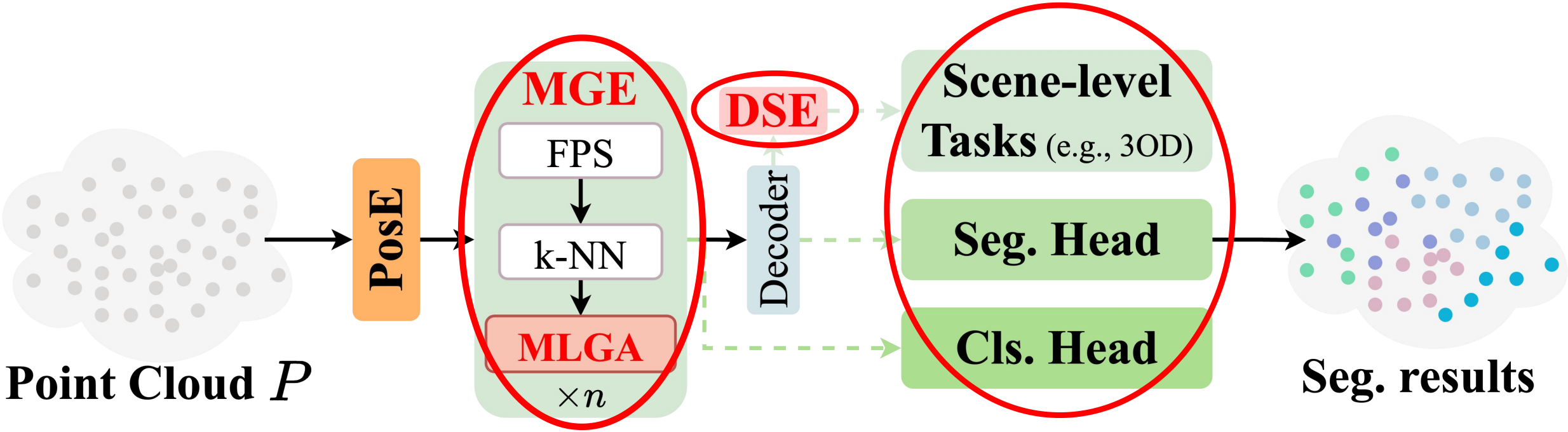



Figure 9: Overview of PointeNet

# Multivariate Geometric Encoding



- Multivariate geometric features
- FPS, k-NN, pooling, and Multivariate Local Geometric Aggregation (MLGA) with **minimal learnable parameters**

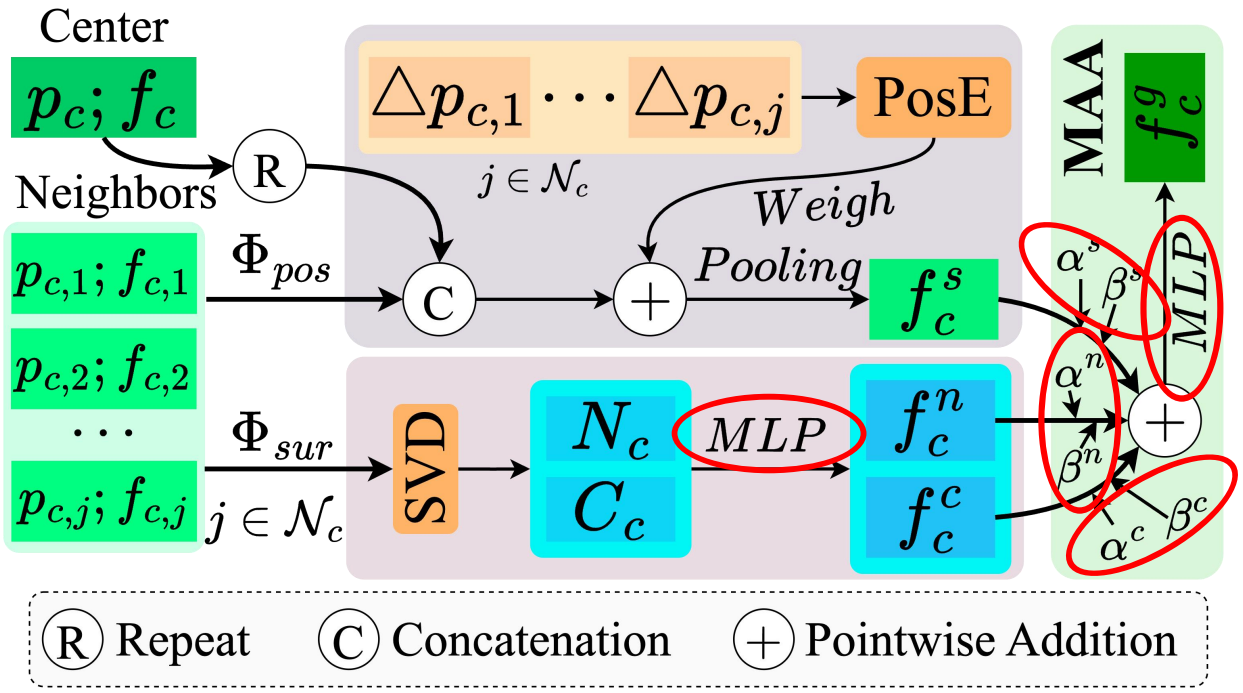



Figure 10: Overview of MLGA

**Spatial neighboring features** $f_c^s$

**Curvature features** $f_c^c$

**Normal features** $f_c^n$

**aggregate**

$$f_c^g = MLP\left(\sum_{i=\{s,c,n\}} (a^i \odot f_c^i + \beta^i)\right)$$

# Multivariate Geometric Encoding



The **spatial neighboring features** within a local region

$$f_c^s = (Concat(f_c, f_{c,j}) + PosE(\Delta p_{c,j})) \odot PosE(\Delta p_{c,j}), \quad j \in \mathcal{N}_c$$

where $PosE(\cdot)$ uses **trigonometric functions** for the position encoding.

The **curvature and normal features** within a local region

- calculate the matrix:

$$M_c = \frac{1}{\mathcal{N}_c} \sum_{j=1}^{\mathcal{N}_c} p_{c,j} p_{c,j}^T - \bar{p}_c \bar{p}_c^T$$

- perform SVD on the matrix to obtain the **eigenvalues** $c_1, c_2, c_3$ and corresponding **eigenvectors** $n_1, n_2, n_3$

# Multivariate Geometric Encoding



The **curvature and normal features** within a local region:

- use the eigenvector $n_3$ corresponding to the smallest eigenvalue $c_3$ as the *pseudo-normal vector*:

$$N_c = n_3$$

- normalize the three eigenvalues to obtain the *pseudo-curvature vector*:

$$C_c = \{c_1, c_2, c_3\}, \quad c_m = \frac{c_m}{\sum_{i=1}^{3} c_i}, \; m \in \{1,2,3\}$$

- use an MLP layer to obtain *curvature and normal features*:

$$f_c^n = MLP(N_c), \quad f_c^c = MLP(C_c)$$

# Distance-aware Semantic Enhancement



- **Distance-aware** semantic features
- **Adaptively differentiate** segmentation difficulty by region
- Only **four** FC layers and **two** MLP layers [a parameter cost of just 0.017 million]

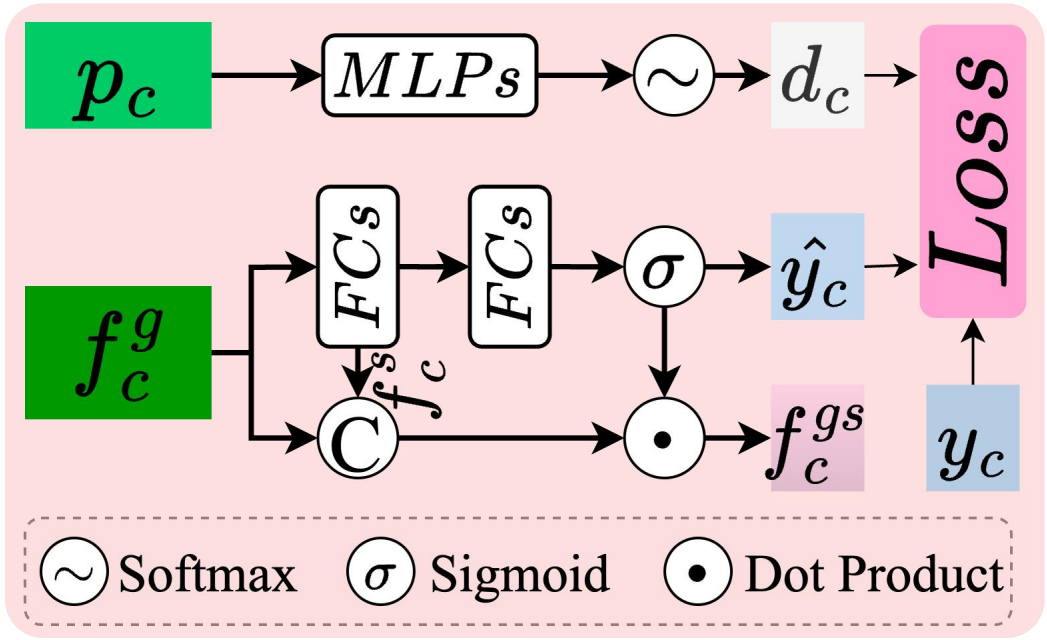



Figure 11: Overview of DSE

$$L_{seg} = -\alpha(1-p_t)^{\gamma}\log p_t$$

$$\boldsymbol{d_c}$$

$$L_{seg} = -d_c(1-p_t)^{\frac{1}{d_c}}\log p_t$$

Focal Loss (Lin et al., 2020)

# Distance-aware Semantic Enhancement



- The improved Focal Loss:

$$L_{seg} = -d_c(1-p_t)^{\frac{1}{d_c}}\log p_t, \quad p_t = (1-y_c)*(1-\hat{y}_c) + y_c * \hat{y}_c$$

- The final output features for decorating the input point cloud:

$$f^{gs} = Concat(f_c^g, f_c^s) * \hat{y}_c$$

# Experiments

# Shape classification on the synthetic ModelNet40



| Method | mAcc (%) | OA (%) | Param. | FLOPs | Train Time |
|---|---|---|---|---|---|
| PointNet | 86.0 | 89.2 | - | - | - |
| PointNet++ | - | 91.9 | 1.4 M | - | - |
| PointCNN | 88.1 | 92.5 | - | - | - |
| PointConv | - | 92.5 | 18.6 M | - | - |
| KPConv | - | 92.9 | 15.2 M | - | - |
| Point Transformer | 90.6 | 93.7 | - | - | - |
| EQ-PointNet++ | - | 93.2 | - | - | - |
| 3DCTN (Multi-scale) | **91.6** | 93.2 | 4.2 M | 3.7 G | - |
| PointNeXt-S | 90.8 | 93.2 | 1.4 M | 1.6 G | - |
| APES | - | <u>93.8</u> | - | - | - |
| FlatNet-P | - | 92.6 | - | - | - |
| PointMLP* | 90.8 | 93.3 | 12.6 M | 14.6 G | 6.2 h |
| Point-NN* | - | 81.3 | 0 M | 0 G | 0 h |
| PointeNet (C=36) (Ours) | <u>91.5</u> | **93.9** | 1.1 M | 2.0 G | 2.4 h |
| PointeNet (C=66) (Ours) | 91.4 | **93.9** | 3.2 M | 6.7 G | 4.4 h |

# Shape classification on the real-world ScanObjectNN



| Method | mAcc (%) | OA (%) | Param. | FLOPs | Train Time |
|---|---|---|---|---|---|
| PointNet | 63.4 | 68.2 | 3.5 M | - | - |
| PointNet++ | 75.4 | 77.9 | 1.7 M | - | - |
| PointCNN | 75.1 | 78.5 | 15.2 M | - | - |
| SpiderCNN | 69.8 | 73.7 | - | - | - |
| SimpleView | - | 80.5 | - | - | - |
| GBNet | 77.8 | 80.5 | 8.4 M | - | - |
| 3DCTN (Multi-scale) | 79.5 | 81.5 | 4.2 M | 3.7 G | - |
| PointNeXt-S | 85.8 | <u>87.7</u> | 1.4 M | 1.6 G | - |
| SN-Adapter-PointMLP | 84.6 | 86.3 | - | - | - |
| PointGL | 85.2 | 86.9 | - | - | - |
| PointWavelet-L | 85.8 | <u>87.7</u> | - | - | - |
| PointMLP* | 83.2 | 85.0 | 12.6 M | 14.6 G | 4.8 h |
| Point-NN* | - | 64.8 | 0 M | 0 G | 0 h |
| PointeNet (C=36) (Ours) | 84.5 | 86.1 | 0.9 M | 2.1 G | 3.1 h |
| PointeNet (C=66) (Ours) | <u>86.1</u> | 87.6 | 2.6 M | 7.1 G | 5.6 h |
| PointeNet (C=90) (Ours) | **86.4** | **87.9** | 4.5 M | 13.2 G | 7.8 h |

# Part segmentation on ShapeNetPart

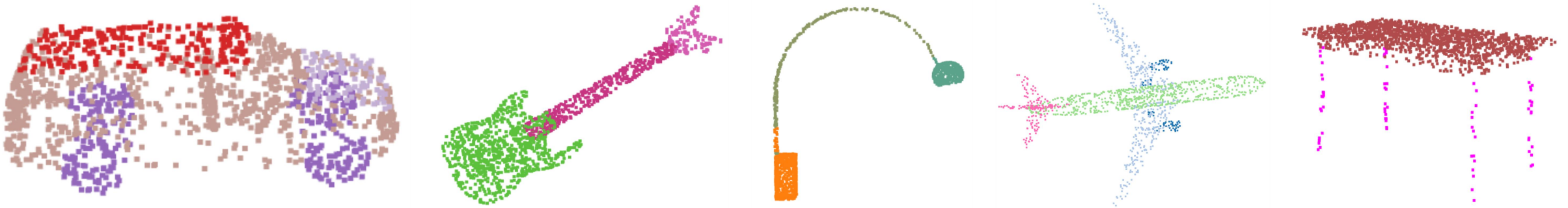


| Method | Cls. mIoU (%) | Inst. mIoU (%) | Param. | FLOPs | Train Time |
|---|---|---|---|---|---|
| PointNet | 80.4 | 83.7 | 8.3 M | - | - |
| PointNet++ | 81.9 | 85.1 | 1.8 M | - | - |
| Kd-Net | - | 82.3 | - | - | - |
| SpiderCNN | 82.4 | 85.3 | - | - | - |
| SPLATNet | 83.7 | 85.4 | - | - | - |
| CSANet | - | <u>85.7</u> | - | - | - |
| PointNeXt-S (C=160)* | 81.8 | 85.5 | 22.5 M | 110.2 G | 17 h |
| FlatNet-P | - | 84.9 | - | - | - |
| APES | 83.7 | **85.8** | - | - | - |
| G-PointNet++ | 82.4 | 85.5 | - | - | - |
| PointMLP* | <u>84.0</u> | <u>85.7</u> | 16.0 M | 5.8 G | 9.3 h |
| Point-NN* | - | 70.7 | 0 M | 0 G | 0 h |
| PointeNet (C=36) (Ours) | 83.7 | 85.2 | 5.3 M | 1.3 G | 3.6 h |
| PointeNet (C=66) (Ours) | **84.2** | 85.6 | 8.3 M | 2.8 G | 5.0 h |
| PointeNet (C=90) (Ours) | **84.2** | **85.8** | 12.3 M | 4.7 G | 6.2 h |

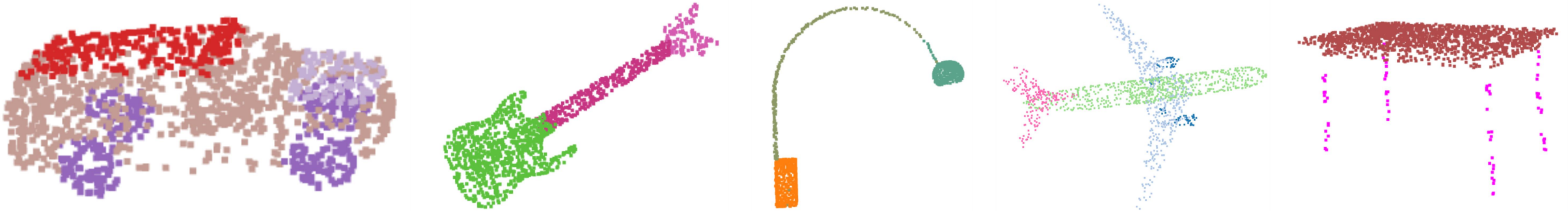
# Part segmentation on ShapeNetPart



Ground Truth



PointeNet



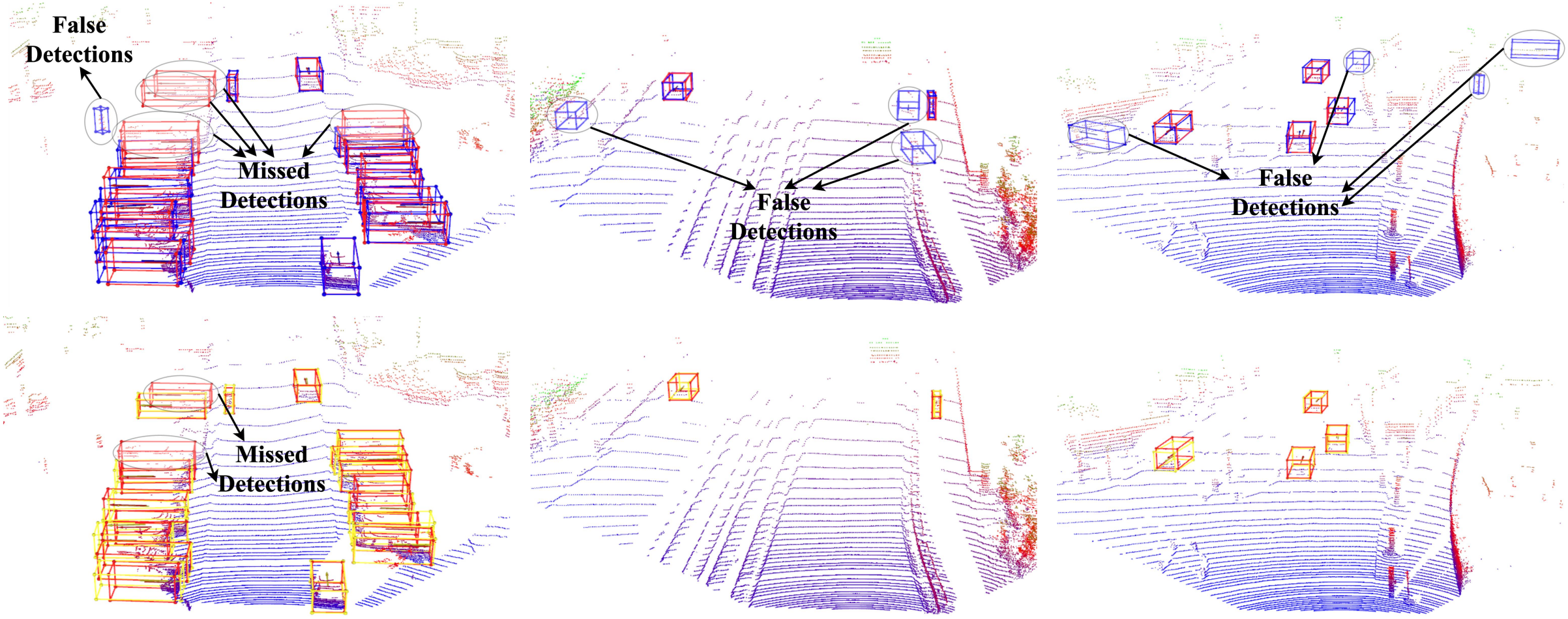
# Results of PointeNet on KITTI



| Method | Car (3D $AP_{R40}$) | | | Pedestrian (3D $AP_{R40}$) | | | Cyclist (3D $AP_{R40}$) | | |
|---|---|---|---|---|---|---|---|---|---|
| | Easy | Mod. | Hard | Easy | Mod. | Hard | Easy | Mod. | Hard |
| SECOND | 90.55 | 81.61 | 78.56 | 55.94 | 51.15 | 46.17 | 82.97 | 66.74 | 62.78 |
| PointPillars | 87.75 | 78.41 | 75.19 | 57.30 | 51.42 | 46.87 | 81.57 | 62.93 | 58.98 |
| PV-RCNN | 92.10 | 84.36 | 82.48 | 64.26 | 56.67 | 51.91 | 88.88 | 71.95 | 66.78 |
| PC-RGNN | 90.94 | 81.43 | 80.45 | - | - | - | - | - | - |
| Voxel-RCNN | 91.72 | 83.19 | 78.60 | - | - | - | - | - | - |
| VoxSeT | 91.02 | 82.01 | 79.04 | - | - | - | - | - | - |
| CAT-Det | 90.12 | 81.46 | 79.15 | - | - | - | - | - | - |
| VP-Net | 89.01 | 82.19 | 79.46 | - | - | - | - | - | - |
| FARP-Net | 89.91 | 83.99 | 79.21 | - | - | - | - | - | - |
| PointRCNN* | 91.28 | 80.78 | 78.37 | 65.12 | 56.43 | 49.24 | 90.02 | 72.42 | 67.44 |
| PointRCNN + PointeNet | **92.34** | 83.13 | 80.78 | **66.00** | **58.62** | **52.09** | **92.99** | **73.97** | **69.50** |
| *Improvement* | **+1.06** | **+2.35** | **+2.41** | **+0.88** | **+2.19** | **+2.85** | **+2.97** | **+1.55** | **+2.06** |
| 3DSSD* | 91.84 | 84.52 | 82.10 | 59.49 | 54.69 | 50.46 | 89.82 | 69.73 | 65.49 |
| 3DSSD + PointeNet | 92.23 | **85.18** | **82.66** | 61.76 | 56.00 | 51.59 | 88.82 | 72.29 | 67.75 |
| *Improvement* | **+0.39** | **+0.66** | **+0.56** | **+2.27** | **+1.31** | **+1.13** | -1.00 | **+2.56** | **+2.26** |

# Results of PointeNet on KITTI

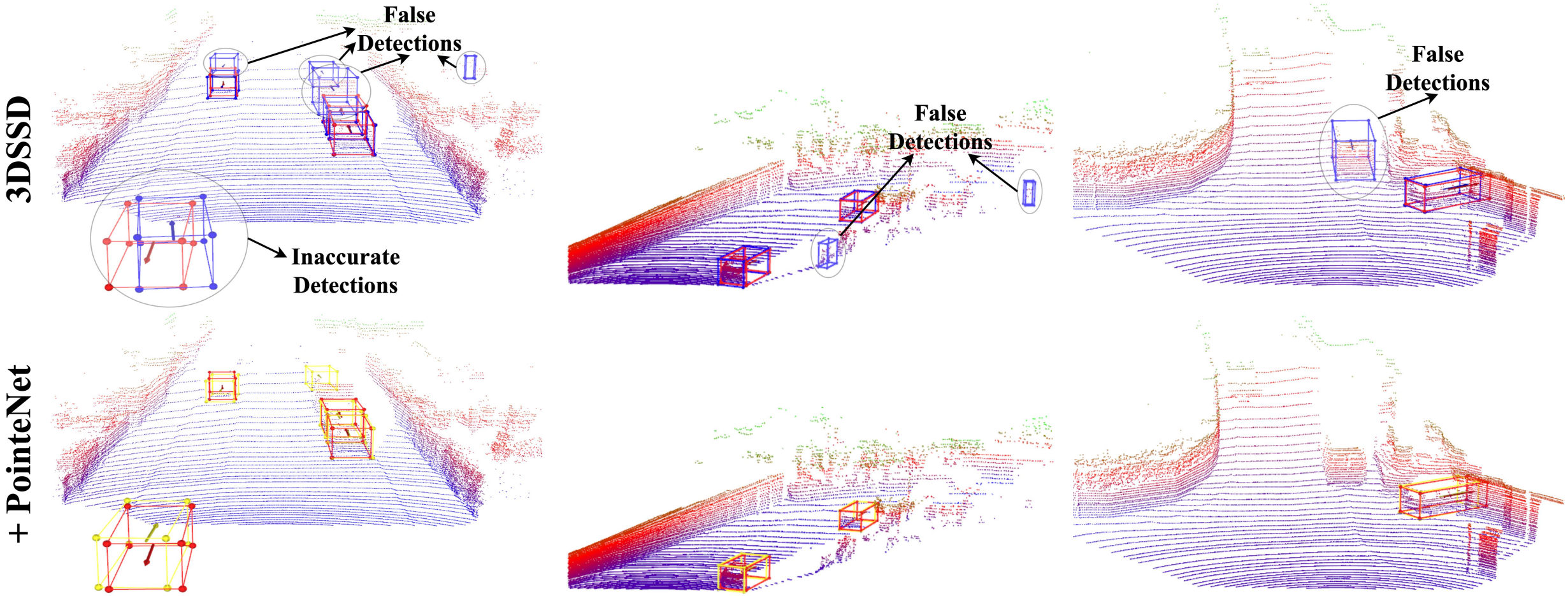





[The red 3D bounding boxes represent the ground truth, while the blue/yellow ones denote the prediction results]

# Results of PointeNet on KITTI






[The red 3D bounding boxes represent the ground truth, while the blue/yellow ones denote the prediction results]

# Ablation results



- Ablation results of the proposed MGE on ShapeNetPart

| Geometries | | | Aggregation | | Cls. | Inst. | Param. | FLOPs | Train |
|---|---|---|---|---|---|---|---|---|---|
| Neighboring | Normal | Curvature | MAA | Add. | mIoU (%) | mIoU (%) | | | Time |
| ✓ | | | | | 82.98 | 84.76 | 4.4 M | 1.3 G | 3.2 h |
| ✓ | ✓ | | ✓ | | 83.47 | 84.94 | 5.3 M | 1.3 G | 3.5 h |
| ✓ | | ✓ | ✓ | | 83.48 | 85.13 | 5.3 M | 1.3 G | 3.5 h |
| ✓ | ✓ | ✓ | | ✓ | 83.65 | 85.09 | 4.4 M | 1.3 G | 3.5 h |
| ✓ | ✓ | ✓ | ✓ | | **83.73** | **85.17** | 5.3 M | 1.3 G | 3.6 h |

- Ablation results of the proposed DSE on KITTI

| PointRCNN | MGE | DSE | | Car (3D $AP_{R40}$) | | |
|---|---|---|---|---|---|---|
| | | w/o $d$ | w/ $d$ | Easy | Mod. | Hard |
| ✓ | | | | 91.28 | 80.78 | 78.37 |
| ✓ | ✓ | | | 91.98 | 82.35 | 80.39 |
| ✓ | ✓ | ✓ | | 92.04 | 82.51 | 80.49 |
| ✓ | ✓ | | ✓ | **92.34** | **83.13** | **80.78** |

# The effect of noise



| Method | ShapeNetPart (C=66) | | ModelNet40 (C=36) | | ScanObjectNN (C=90) | |
|---|---|---|---|---|---|---|
| | Cls. mIoU (%) | Inst. mIoU (%) | mAcc (%) | OA (%) | mAcc (%) | OA (%) |
| $\sigma = 0$ | 84.2 | 85.6 | 91.5 | 93.9 | 86.4 | 87.9 |
| $\sigma = 0.1\% * r$ | 84.4 (+0.2) | 85.8 (+0.2) | 91.0 (−0.5) | 93.2 (−0.7) | 86.4 (−0.0) | 87.7 (−0.2) |
| $\sigma = 0.5\% * r$ | 84.3 (+0.1) | 85.6 (−0.0) | 90.4 (−1.1) | 93.1 (−0.8) | 85.6 (−0.8) | 86.9 (−1.0) |
| $\sigma = 1\% * r$ | 84.1 (−0.1) | 85.3 (−0.3) | 90.7 (−0.8) | 92.9 (−1.0) | 84.8 (−1.6) | 86.7 (−1.2) |
| $\sigma = 3\% * r$ | 82.0 (−2.2) | 84.0 (−1.6) | 89.1 (−2.4) | 91.9 (−2.0) | 82.4 (−4.0) | 84.2 (−3.7) |
| $\sigma = 5\% * r$ | 79.9 (−4.3) | 82.3 (−3.3) | 87.9 (−3.6) | 91.1 (−2.8) | 81.0 (−5.4) | 83.2 (−4.7) |

[$\sigma$denotes the standard deviation of the introduced Gaussian noise, specified relative to the radius $r$of the bounding sphere of a point cloud.]

# Conclusion

# Conclusion



- **Conclusions**
  - A lightweight network, dubbed PointeNet, for a more efficient point cloud analysis
  - A Multivariate Geometric Encoding module for lightweightly capturing and adaptively aggregating multivariate geometric features
  - A Distance-aware Semantic Enhancement (DSE) module tailored for autonomous driving scenarios to improve the performance of scene-level tasks
- **Future work**
  - Explore more potential point cloud features
  - Extend to more application scenarios
  - Explore the potential of integration with multi-modal large models
- **Code:** https://github.com/lipeng-gu/PointeNet.git

Geometric Modeling and Processing 2024

18TH

# THANKS!


Co-organizers:

山东大学 SHANDONG UNIVERSITY

青岛科技大学 Qingdao University of Science & Technology

山东财经大学 Shandong University of Finance and Economics