JUNE 18-22, 2023
CVPR
VANCOUVER, CANADA

# Poly-PC: A Polyhedral Network for Multiple Point Cloud Tasks at Once

**TUE-AM-117**

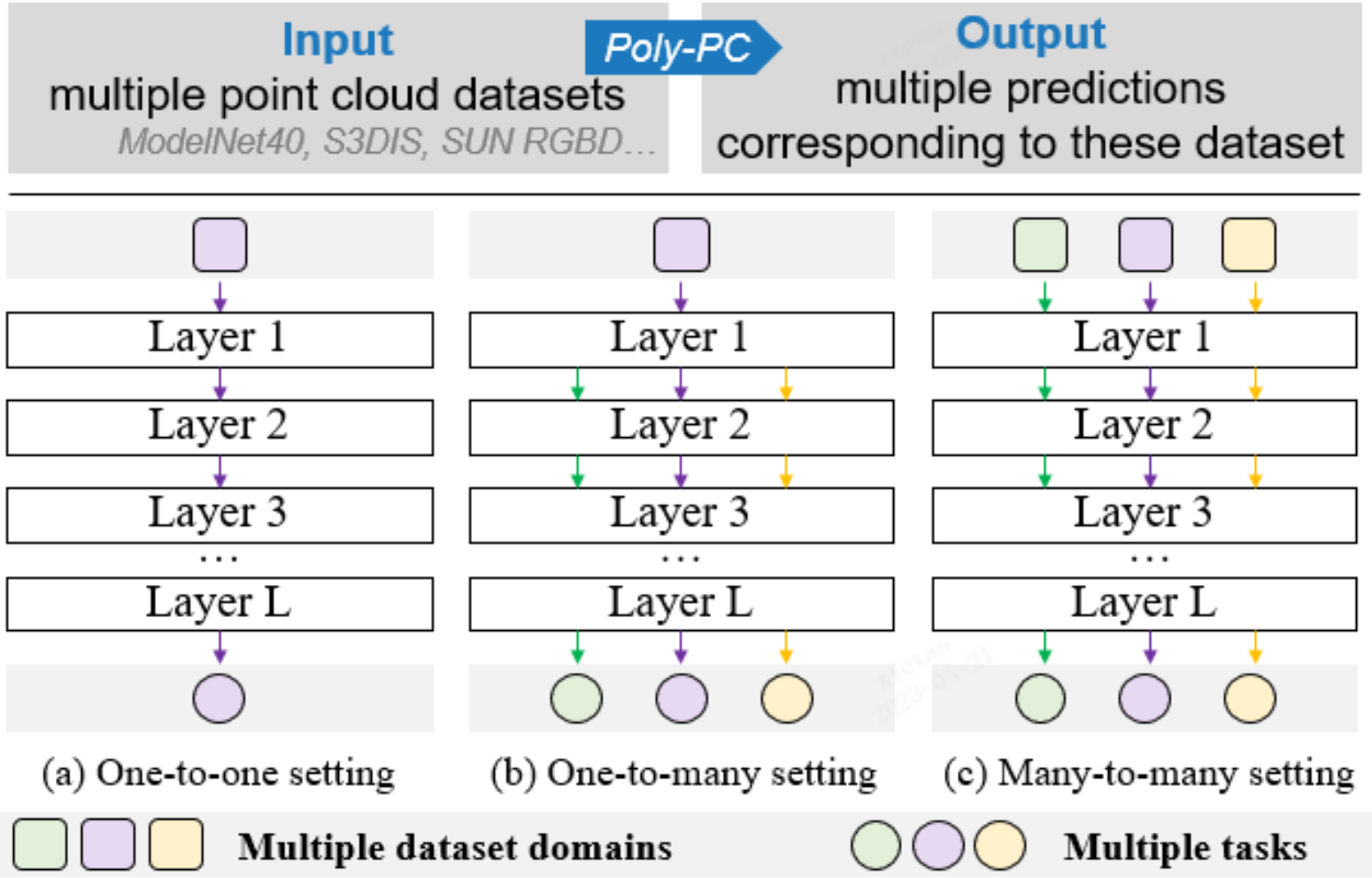
Tao Xie[1,3], Shiguang Wang[2,3], Ke Wang[1], Linqi Yang[1], Zhiqiang Jiang[1], Xingcheng Zhang[3], Kun Dai[1], Ruifeng Li[1], Jian Chen[2]

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

电子科技大学
University of Electronic Science and Technology of China

商汤
sensetime

# Problem formulation



JUNE 18-22, 2023
CVPR
VANCOUVER, CANADA

# Two challenges

**Ploy-PC**
*A unified framework*

- ***optimizes*** **multiple point cloud tasks** ***collectively under heterogeneous dataset*** *(i.e., Many-to-many setting)*

***Compared with***

- *classical point cloud networks (i.e., one-to-one setting)*
- *regular multi-task setting (i.e., one-to-many setting)*

***2 key challenges***

1. ***Not Feasible:*** *all point cloud tasks to directly share a single backbone*

2. input **Multiple dataset domains** *performing multiple tasks on point cloud*

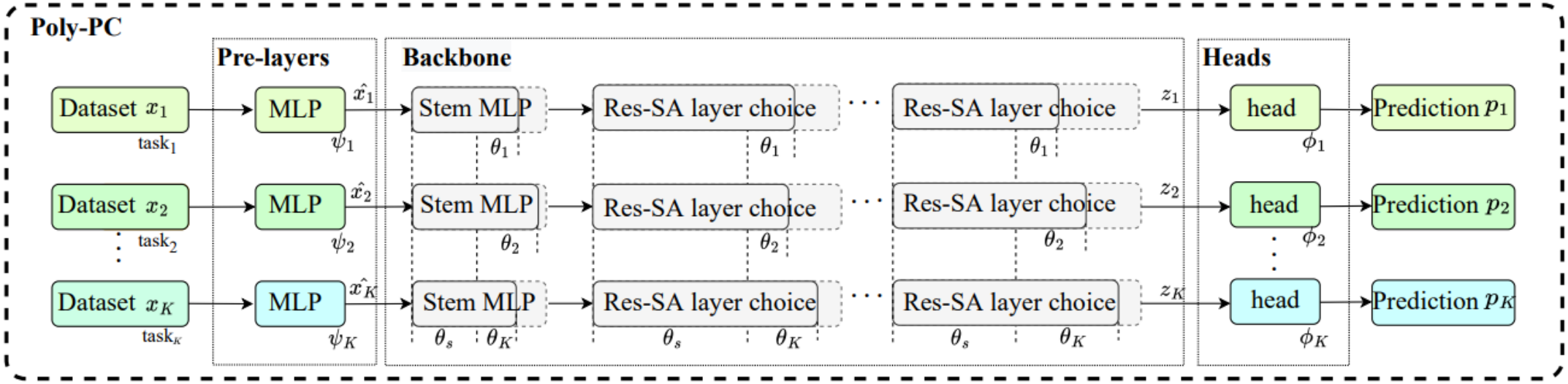
   *induced different task difficulty*

   *arises* ***considerable disparities*** *in:*
   directions and magnitude of different task gradients

   ***Consider task prioritization to prevent:***
   placing undue emphasis on easier tasks when optimizing the multi-task network

# Poly-PC

JUNE 18-22, 2023
CVPR
VANCOUVER, CANADA



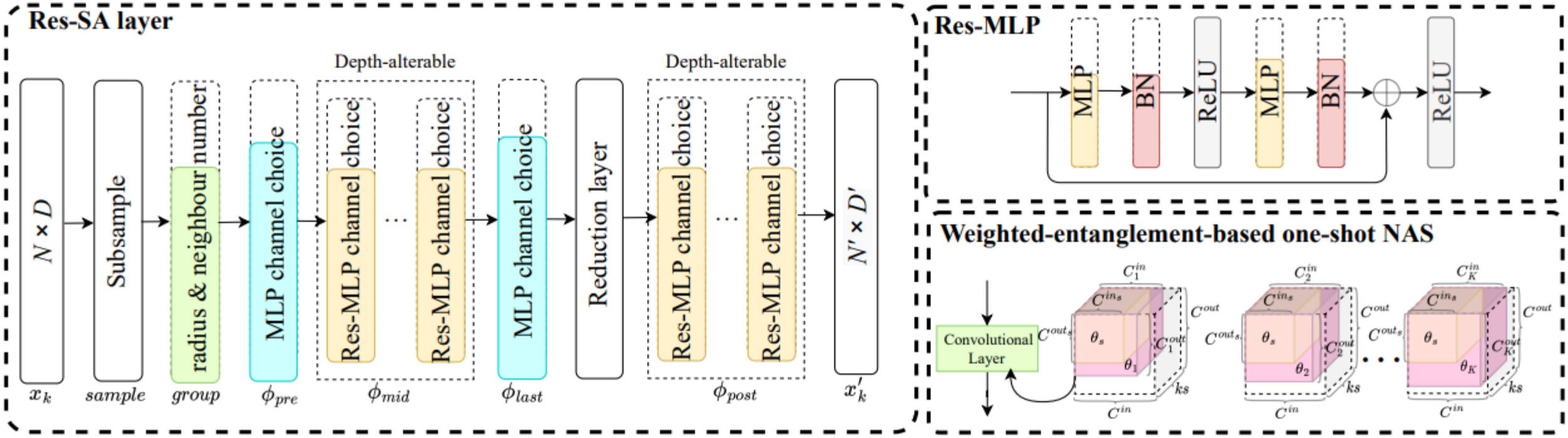

***Poly-PC** = supernet training for all tasks + evolutionary search*

**balance**
- *Weight-entanglement one-shot NAS technique*
- *Task-prioritization-based gradient*

1 ***Poly-PC*** extraordinarily well-trained

2 Identifying **optimal architectures** for each task under resource constraint

# Poly-PC



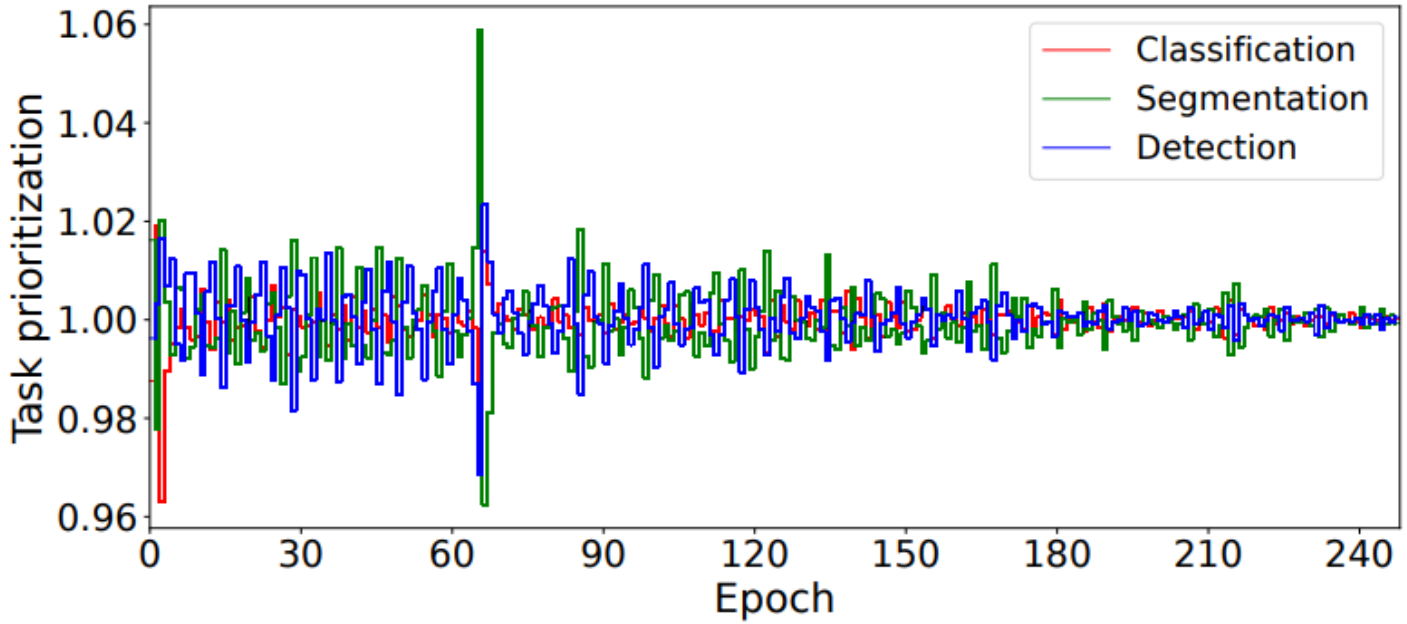

$$g_i = \Phi_{post}(R(\Phi_{last}(\Phi_{mid}(\Phi_{pre}([f_{i,j}; (p_{i,j} - p_i)/r])))), \tag{1}$$

# Task prioritization



$$r_k = \left(\frac{L_k^{t-1}/L_k^{t-2}}{L^{t-1}/L^{t-2}}\right)^{\alpha}, k = 1, 2..., K, \qquad (2)$$

$$TH_k^t = \frac{Kexp(r_k/T)}{\sum_{i=1}^{K} exp(r_i/T)}, k = 1, 2..., K, \qquad (3)$$

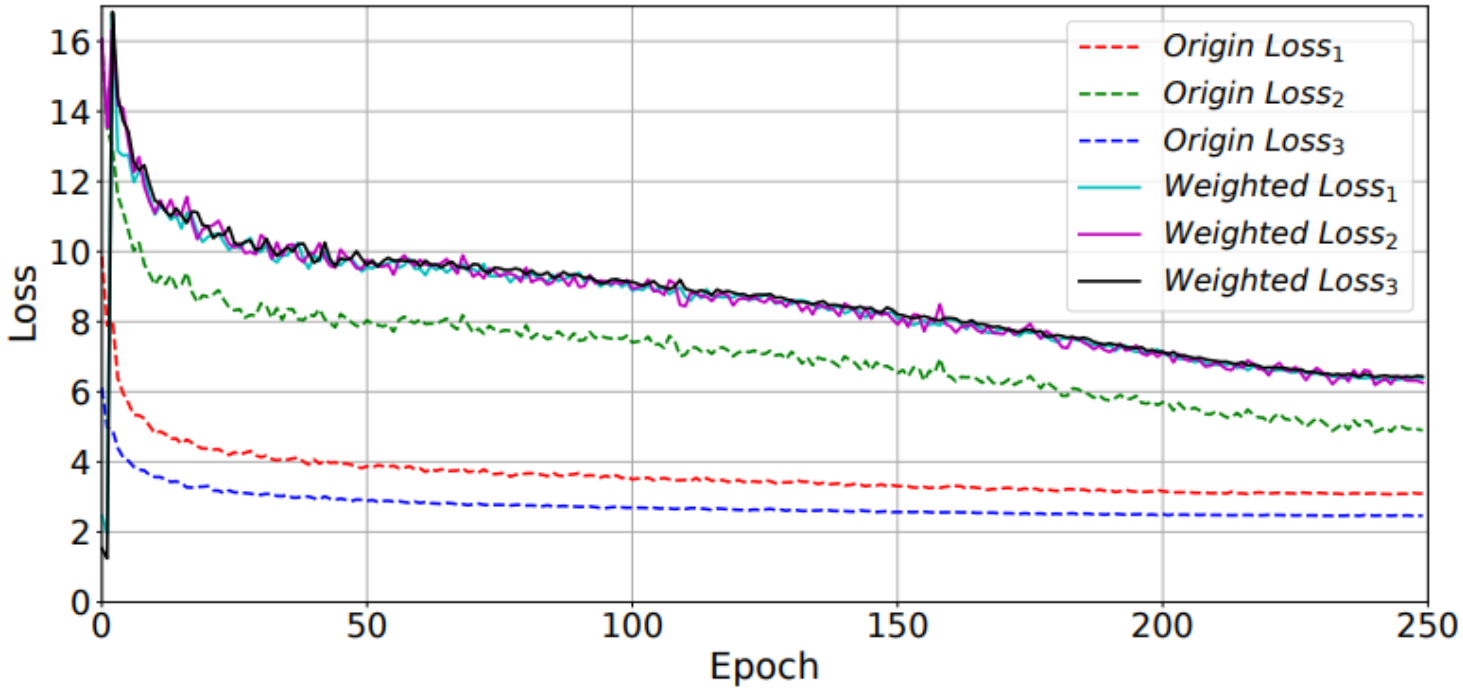


# Gradient-magnitude homogenization

$$w_k^t L_k^t = \frac{1}{K} L^t \Rightarrow w_k^t = \frac{L^t}{K L_k^t}. \tag{4}$$

$$\frac{L_k^t}{L^t} = \frac{L_k^{t-1}}{L^{t-1}} \Rightarrow w_k^t = \frac{L^{t-1}}{K L_k^{t-1}}. \tag{5}$$

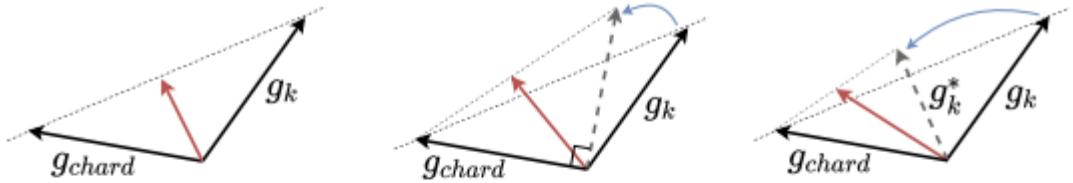
$$\hat{W}_k^t = T H_k^t w_k^t. \tag{6}$$



# Gradient-direction homogenization

$$\Psi(g_k, g_{chard})^{(i)} = \begin{cases} 1, sgn(g_k^{(i)}) = sgn(g_{chard}^{(i)}) \\ 0, otherwise \end{cases} \quad (8) \qquad (g_k^*)^{(i)} = \Psi(g_k, g_{chard})^{(i)} g_k^{(i)}, i = 1, 2..., n. \quad (9)$$

$$g_{\theta_s}^* = \frac{1}{K}(g_1^* + g_2^* + g_{chard} + ... + g_K^*), \quad (10)$$




(a) Gradient descent (b) PCGrad (c) Ours

Figure 2. The combined update vector of $g_k$ and $g_{chard}$ with naive gradient descent, PCGrad, and our algorithm. The naive gradient descent simply adds the two gradient vectors. PCGrad first projects $g_k$ onto the normal plane of $g_{chard}$ (vice versa), and then adds $g_{chard}$ and the projected gradient. Our algorithm constructs consensus vectors $g_k^*$ of $g_k$ to enable the direction of the final merged gradient vector closer to that of $g_{chard}$, compelling the network to concentrate on the difficult task in current epoch.

# Evolutionary Search

JUNE 18-22, 2023
CVPR
VANCOUVER, CANADA

| | | Stem Channel | Neigh Num | Group Radius | Res-MLP Num | Output Channel | Expansion rate | Reduction rate |
|---|---|---|---|---|---|---|---|---|
| stage1 | cls | (64, 72, 80) | (32, 40, 48) | (0.09, 0.1, 0.11) | (0, 1) | (128, 136, 144) | (4, 5) | (0.5, 0.75) |
| | det | (64, 72, 80) | (56, 64, 72) | (0.18, 0.2, 0.22) | (0, 1) | (128, 136, 144) | (4, 5) | (0.5, 0.75) |
| | seg | (32, 40, 48) | (40, 44, 48) | (0.09, 0.1, 0.11) | (0, 1) | (64, 72, 80) | (4, 5) | (0.5, 0.75) |
| stage2 | cls | - | (32, 40, 48) | (0.18, 0.2, 0.22) | (0, 1) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| | det | | (28, 32, 36) | (0.36, 0.4, 0.44) | (0, 1) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| | seg | | (40, 44, 48) | (0.18, 0.2, 0.22) | (0, 1) | (128, 136, 144) | (4, 5) | (0.5, 0.75) |
| stage3 | cls | - | (32, 40, 48) | (0.36 ,0.4, 0.44) | (0, 1) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| | det | | (12, 16, 20) | (0.72, 0.8, 0.88) | (0, 1) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| | seg | | (40, 44, 48) | (0.36, 0.4, 0.44) | (0, 1) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| stage4 | cls | - | (32, 40, 48) | (0.36, 0.4, 0.44) | (0, 1) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| | det | | (6, 8, 10) | (1.04, 1.2, 1.36) | (0, 1) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| | seg | | (40, 44, 48) | (0.72, 0.8, 0.88) | (0, 1) | (512, 536, 560) | (4, 5) | (0.5, 0.75) |

Table 1. The search space of Poly-PC (base). For simplicity, we put the search space of stem channel into stage 1.

| | | Stem Channel | Neigh Num | Group Radius | Res-MLP Num | Output Channel | Expansion rate | Reduction rate |
|---|---|---|---|---|---|---|---|---|
| stage1 | cls | (64, 72, 80) | (40, 48, 56) | (0.09, 0.1, 0.11) | (1, 2) | (128, 136, 144) | (4, 5) | (0.5, 0.75) |
| | det | (128, 136, 144) | (56, 64, 72) | (0.18, 0.2, 0.22) | (1, 2) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| | seg | (32, 40, 48) | (48, 56, 64) | (0.09, 0.1, 0.11) | (1, 2) | (64, 72, 80) | (4, 5) | (0.5, 0.75) |
| stage2 | cls | - | (40, 48, 56) | (0.18, 0.2, 0.22) | (1, 2) | (128, 136, 144) | (4, 5) | (0.5, 0.75) |
| | det | | (28, 32, 36) | (0.36, 0.4, 0.44) | (1, 2) | (512, 528, 544) | (4, 5) | (0.5, 0.75) |
| | seg | | (48, 56, 64) | (0.18, 0.2, 0.22) | (1, 2) | (128, 136, 144) | (4, 5) | (0.5, 0.75) |
| stage3 | cls | - | (40, 48, 56) | (0.36 ,0.4, 0.44) | (1, 2) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| | det | | (12, 16, 20) | (0.72, 0.8, 0.88) | (1, 2) | (512, 528, 544) | (4, 5) | (0.5, 0.75) |
| | seg | | (48, 56, 64) | (0.36, 0.4, 0.44) | (2, 3) | (256, 272, 288) | (4, 5) | (0.5, 0.75) |
| stage4 | cls | - | (40, 48, 56) | (0.36, 0.4, 0.44) | (1, 2) | (512, 536, 560) | (4, 5) | (0.5, 0.75) |
| | det | | (6, 8, 10) | (1.04, 1.2, 1.36) | (1, 2) | (512, 528, 544) | (4, 5) | (0.5, 0.75) |
| | seg | | (48, 56, 64) | (0.72, 0.8, 0.88) | (2, 3) | (512, 536, 560) | (4, 5) | (0.5, 0.75) |

Table 2. The search space of OFAT-PC (large). For simplicity, we put the search space of stem channel into stage 1.

# Experiments


| Method | Points | FLOPs (G) | Params (M) | OA |
|---|---|---|---|---|
| PointCNN [16] | 1k | 1.6 | 0.6 | 92.2 |
| PointConv [43] | 1k | - | 18.6 | 92.5 |
| Kpconv [39] | 7k | 1.7 | 15.2 | 92.9 |
| DGCNN [42] | 1k | 4.8 | 1.8 | 92.9 |
| DeepGCN [15] | 1k | 3.9 | 2.2 | 93.6 |
| ASSANet [31] | 1k | 2.4 | 3.6 | 92.4 |
| ASSANet-L [31] | 1k | 28.9 | 118.4 | 92.9 |
| PointMLP [22] | 1k | 31.4 | 12.6 | 93.7 |
| Point Trans. [54] | 1k | 9.4 | 13.9 | 93.7 |
| PointNet++ [30] | 1k | 3.2 | 1.5 | 90.7 |
| PointNet++ [30] | 5k | 3.2 | 1.5 | 91.9 |
| Poly-PC (base) | 1k | 2.9 | 1.9 | 92.6 (+1.9) |
| Poly-PC (large) | 1k | 5.9 | 5.5 | 93.7 (+3.0) |

**ModelNet40**

| Method | Flops (G) | Params (M) | OA | mIoU |
|---|---|---|---|---|
| PointCNN [16] | - | 0.6 | 85.9 | 57.3 |
| PCCN [2] | 7.3 | 5.4 | - | 58.3 |
| PAT [47] | - | 6.1 | - | 60.1 |
| Kpconv [39] | 2.1 | 14.9 | - | 67.1 |
| ASSANet [31] | 2.5 | 2.4 | - | 63.0 |
| ASSANet-L [31] | 36.2 | 115 | - | 66.8 |
| Point Trans. [54] | 7.8 | 5.6 | 90.8 | 70.4 |
| PointNeXT-S [32] | 3.6 | 0.8 | 87.9 | 63.4 |
| PointNeXT-B [32] | 8.8 | 3.8 | 89.4 | 67.5 |
| PointNet++* [30] | 1.0 | 1.0 | 85.8 | 56.9 |
| Poly-PC (base) | 1.0 | 1.0 | 88.2 (+2.4) | 63.0 (+6.1) |
| Poly-PC (large) | 5.6 | 5.6 | 89.5 (+3.7) | 66.0 (+9.1) |

**S3DIS**

| Method | Flops (G) | Params (M) | mAP@0.25 | mAP@0.5 |
|---|---|---|---|---|
| F-PointNet [28]† | - | 11.8 | 54.0 | - |
| ImVoteNet* [26]† | 241 | 42.5 | 64.5 | 38.6 |
| 3Detr [24] | 9.8 | 7.1 | 59.1 | 32.7 |
| MLCVNe [46] | 7.2 | 1.2 | 59.8 | - |
| H3DNet [53] | 14.5 | 6.3 | 60.1 | 39.0 |
| BRNet [7] | 8.0 | 3.2 | 61.1 | 43.7 |
| VENet [45] | 30.3 | 4.9 | 62.5 | 39.2 |
| Group-free* [21] | 11.2 | 19.8 | 63.0 | 45.2 |
| RBGNet [41] | 3.5 | 2.2 | 64.1 | 47.2 |
| VoteNet* [27] | 5.8 | 1 | 59.1 | 35.8 |
| Poly-PC (base) | 6.1 | 1 | 62.3 (+3.2) | 40.2 (+4.4) |
| Poly-PC (large) | 18.8 | 7.8 | 63.5 (+4.4) | 41.9 (+6.1) |

**SUN RGBD**

| Method | OA | mIoU | mAP@0.25 | S-Score | Params (t) |
|---|---|---|---|---|---|
| PointNet++ & VoteNet | 91.9 | 56.9 | 59.1 | 207.9 | 3.5M |
| Poly-PC (base) | 92.6 | 63.0 | 62.3 | 217.9 | 3.4M |
| Poly-PC (large) | 93.7 | 66.0 | 63.5 | 223.2 | 13.7M |
| PointCNN [16] | 92.2 | 57.3 | - | 149.5 | 1.2M |
| Kpconv [39] | 92.9 | 67.1 | - | 160.0 | 30.1M |
| ASSANet [31] | 92.4 | 63.0 | - | 155.4 | 6.0M |
| Poly-PC* (base) | 92.6 | 63.0 | - | 155.6 | 2.7M |
| Poly-PC* (large) | 93.7 | 66.0 | - | 159.7 | 8.5M |

**Overall performance**

# Experiments



- Incremental learning of Poly-PC
- Trained on ModelNet40, S3DIS, and SUN RGBD
- Shape classification results on the ScanObjectNN dataset

| Method | Points | Flops (G) | Params Count | OA |
|---|---|---|---|---|
| PointNet++ [10] | 1k | 3.2 | 100% | 77.9 |
| PointCNN [6] | 1k | - | 100% | 78.5 |
| DGCNN [16] | 1k | 2.4 | 100% | 78.1 |
| DRNet [12] | 1k | - | 100% | 80.3 |
| GBNet [13] | 1k | - | 100% | 80.5 |
| PRANet [2] | 1k | - | 100% | 82.1 |
| MVTN [4] | 1k | 1.8 | 100% | 82.8 |
| PointMLP [7] | 1k | - | 100% | 85.4 |
| **Poly-PC (base)** | **1k** | **3.3** | **88%** | **86.8** |
| **Poly-PC (large)** | **1k** | **5.8** | **50%** | **87.9** |