CO-Net: Learning Multiple Point Cloud Tasks at Once with A Cohesive Network —Supplementary Material—

A. Implementation Details

A.1. Search space of CO-Net

Inspired by Poly-PC [16], we introduce a comprehensive search space to identify optimal backbone for various point cloud tasks, with the components of the backbone in each stage including: (1) neighbour points number, (2) group radius, (3) Res-MLP block numbers in Φ_{red} and Φ_{exp}, reduction rate in Φ_{red}, (4) expansion rate in Φ_{exp}, and (5) output channels of current stage. As illustrated in the main paper, each task has a unique search space. In addition, we also search for the output channel of the stem MLP. The detailed description of the searching space for CO-Net (base) is in Table 1 and CO-Net (large) in Table 2.

A.2. Supernet training

CO-Net is designed to optimize K different point cloud tasks simultaneously, which involves discovering K mappings from K different datasets $\{\mathcal{X}_k\}$ to a task-specific set of labels $\{\mathcal{Y}_k\}$, $k = 1, 2, ..., K$. To handle the fact that each task has its own dataset domain, we distribute the K tasks across different GPUs, with each task corresponding to a separate GPU. At each iteration, one subnet will be sampled uniformly for each task from the pre-defined search space, whose corresponding weights will be updated while the rest will be frozen. In addition, we optimize the task-shared parameters in the global group, while the task-specific parameters in the task-specific group, which is implemented by defining multiple communication groups in DistributedDataParallel of PyTorch. During this process, no gradient or weight update is applied to unsampled parameters in the supernet. We train CO-Net for a total of 100K iterations using 12 Tesla A100 GPUs, with classification, segmentation, and detection tasks taking up 4 GPUs respectively.

A.3. Evolution search.

To identify optimal subnets for each task, an evolution search is performed on the well-trained CO-Net. We evaluate and pick the subnets in accordance to the manager of the evolution algorithm, based on their performance of the corresponding task. The proxy score involves overall accuracy for 3D point classification, mean classwise intersection-over-union (mIoU) for 3D semantic segmentation, and mean average precision (mAP) at 0.25 threshold (mAP$_{0.25}$) for 3D object detection. Evolution search is started by selecting 50 random architectures as seeds and picking the top 10 architectures as parents to generate the next generation via crossover and mutation. Specifically, during the crossover, we select two candidates at random and combine them to produce a new architecture. For mutation, each candidate holds a probability P_d of mutating its depth first then a probability P_m of mutating each block to produce a new one. We apply $P_d = 0.2$ and $P_m = 0.4$ for all our experiments.

B. Incremental learning analysis

Generalization to ModelNet40. In this part, we demonstrate the ability of CO-Net to enable incremental learning by adding the Human-made Object Classification dataset ModelNet40 [14]. ModelNet40 is a point cloud dataset for 3D shape classification, containing 40 categories, with 9843 examples in the training set and 2468 examples in the validation set. The ModelNet40 dataset poses challenges for point cloud analysis methods due to its occlusions and noise. Having been trained on three datasets, CO-Net is fine-tuned on the ModelNet40 dataset with task-specific parameters while freezing all task-shared parameters. Particularly, the searched architecture for ScanObjectNN is applied to conduct the incremental experiment on the ModelNet40 dataset. Moreover, CO-Net is optimized by the initial learning rate 0.008 with cosine annealing and AdamW optimizer with weight decay 0.05. The batch size is set to 16, and the total epochs are set to 100. The results are reported in Table 3. CO-Net (base) and CO-Net (large) outperform Pointnet++ by significant margins, achieving 92.1/92.9 overall accuracy (OA), which is also comparable to the state-of-art methods. We can also observe that CO-Net (base) and CO-Net (large) require only 62% and 58% of the total parameters, respectively, demonstrating that a substantial number of parameters are task-shared, thus, our network can expand gracefully as the number of tasks increases.

Catastrophic forgetting analysis for CO-Net. We leverage the trained model of CO-Net on ScanObjectNN dataset to finetune on ModelNet40 dataset without freezing the task-shared parameters to get the new model, whose overall accuracy on ScanObjectNN is 0.1%. Instead, our CO-Net can generalize to a new task while retaining its performances on the learned tasks, further illustrating that CO-Net can evade catastrophic forgetting.

C. Per-class evaluation

We evaluate the performance of our approach on SUN RGBD dataset, taking into account different IoU thresholds for each category. Specifically, we report the results on 10 classes of SUN RGBD dataset in Table 4 and Table 5 with 0.25 and 0.5 box IoU thresholds, respectively. Our proposed approach demonstrates significant improvement

		Stem Channel	Neigh Num	Group Radius	Res-MLP Num	Output Channel	Expansion ratio	Reduction rate
stage1	cls	(56, 64, 72)	(32, 40, 48)	(0.09, 0.1, 0.11)	(1, 2)	(112, 128, 144)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	det	(64, 72, 80)	(56, 64, 72)	(0.18, 0.2, 0.22)	(1, 2)	(128, 136, 144)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	seg	(32, 40, 48)	(40, 44, 48)	(0.09, 0.1, 0.11)	(1, 2)	(64, 72, 80)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
stage2	cls	-	(32, 40, 48)	(0.18, 0.2, 0.22)	(1, 2)	(224, 256, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	det		(28, 32, 36)	(0.36, 0.4, 0.44)	(1, 2)	(256, 272, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	seg		(40, 44, 48)	(0.18, 0.2, 0.22)	(1, 2)	(128, 136, 144)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
stage3	cls	-	(32, 40, 48)	(0.36 ,0.4, 0.44)	(1, 2)	(224, 256, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	det		(12, 16, 20)	(0.72, 0.8, 0.88)	(1, 2)	(256, 272, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	seg		(40, 44, 48)	(0.36, 0.4, 0.44)	(1, 2)	(256, 272, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
stage4	cls	-	(32, 40, 48)	(0.36, 0.4, 0.44)	(1, 2)	(224, 256, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	det		(12, 16, 20)	(1.04, 1.2, 1.36)	(1, 2)	(256, 272, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	seg		(40, 44, 48)	(0.72, 0.8, 0.88)	(1, 2)	(512, 536, 560)	(3, 3.5, 4)	(0.25, 0.5, 0.75)

Table 1: The search space of CO-Net (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 ratio	Reduction rate
stage1	cls	(64, 72, 80)	(32, 40, 48)	(0.09, 0.1, 0.11)	(2, 3, 4)	(128, 136, 144)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	det	(128, 136, 144)	(56, 64, 72)	(0.18, 0.2, 0.22)	(2, 3, 4)	(256, 272, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	seg	(64, 72, 80)	(40, 48, 56)	(0.09, 0.1, 0.11)	(2, 3, 4)	(128, 136, 144)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
stage2	cls	-	(32, 40, 48)	(0.18, 0.2, 0.22)	(2, 3, 4)	(128, 136, 144)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	det		(28, 32, 36)	(0.36, 0.4, 0.44)	(2, 3, 4)	(512, 528, 544)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	seg		(40, 48, 56)	(0.18, 0.2, 0.22)	(2, 3, 4)	(128, 136, 144)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
stage3	cls	-	(32, 40, 48)	(0.36 ,0.4, 0.44)	(2, 3, 4)	(256, 272, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	det		(12, 16, 20)	(0.72, 0.8, 0.88)	(2, 3, 4)	(512, 528, 544)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	seg		(40, 48, 56)	(0.36, 0.4, 0.44)	(2, 3, 4)	(256, 272, 288)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
stage4	cls	-	(32, 40, 48)	(0.36, 0.4, 0.44)	(2, 3, 4)	(512, 536, 560)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	det		(12, 16, 20)	(1.04, 1.2, 1.36)	(2, 3, 4)	(512, 528, 544)	(3, 3.5, 4)	(0.25, 0.5, 0.75)
	seg		(40, 48, 56)	(0.72, 0.8, 0.88)	(2, 3, 4)	(512, 536, 560)	(3, 3.5, 4)	(0.25, 0.5, 0.75)

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

over the baseline VoteNet [7] and attains comparable performance than previous state-of-the-art methods Group-free [4] in almost all categories, emphasizing the significance of a robust backbone for enhancing the detection performance.

D. Visualization

Fig. 1 illustrates the detection results obtained by the baseline VoteNet [7] and CO-Net (large). It is evident that CO-Net (large) outperforms the baseline method VoteNet with fewer false positives and more accurate bounding boxes on the SUN RGBD dataset, which further highlights the potentiality of a strong backbone for boosting the detection performance.

E. Methodology

CO-Net aims to optimize K different tasks on point cloud concurrently, that is, discovering K mappings from K different datasets $\{\mathcal{X}_k\}$ to a task-specific set of labels $\{\mathcal{Y}_k\}$, $k = 1, 2, ..., K$. In this work, we focus on the design and optimization of CO-Net, which encourages all tasks to share as many parameters as feasible for efficient storage while ensuring superior performance for all tasks.

Method	Points	FLOPs (G)	#Params-e	OA
PointNet++ [8]	1k	3.2	100%	90.7
PointNet++ [8]	5k	3.2	100%	91.9
PointWeb [19]	1k	-	100%	92.3
PointConv [13]	1k	-	100%	92.5
Kpconv [10]	7k	-	100%	92.9
DGCNN [12]	1k	4.8	100%	92.9
DeepGCN [3]	1k	3.9	100%	93.6
ASSANet [9]	1k	2.4	100%	92.4
ASSANet-L [9]	1k	-	100%	92.9
PointMLP [5]	1k	-	100%	93.7
Point Trans. [20]	1k	-	100%	93.7
CO-Net (base)	1k	2.9	62.15%	92.1
CO-Net (large)	1k	5.9	58.11%	92.9

Table 3: Incremental learning. #Params-e means that the percentage of additional parameters that each method needs when generalizing to a new task.

F. Details on CO-Net Architecture

F.1. Network input

ScanObjectNN. CO-Net (base/large) leverages a randomly subsampled point cloud with a size of $N_{cls} \times C_{cls}^{in}$ as input for 3D point classification. N_{cls} represents the number of sampled points, which is set to 1024, and $C_{cls}^{in} = 6$ indicates the number of input coordinates and the normal-

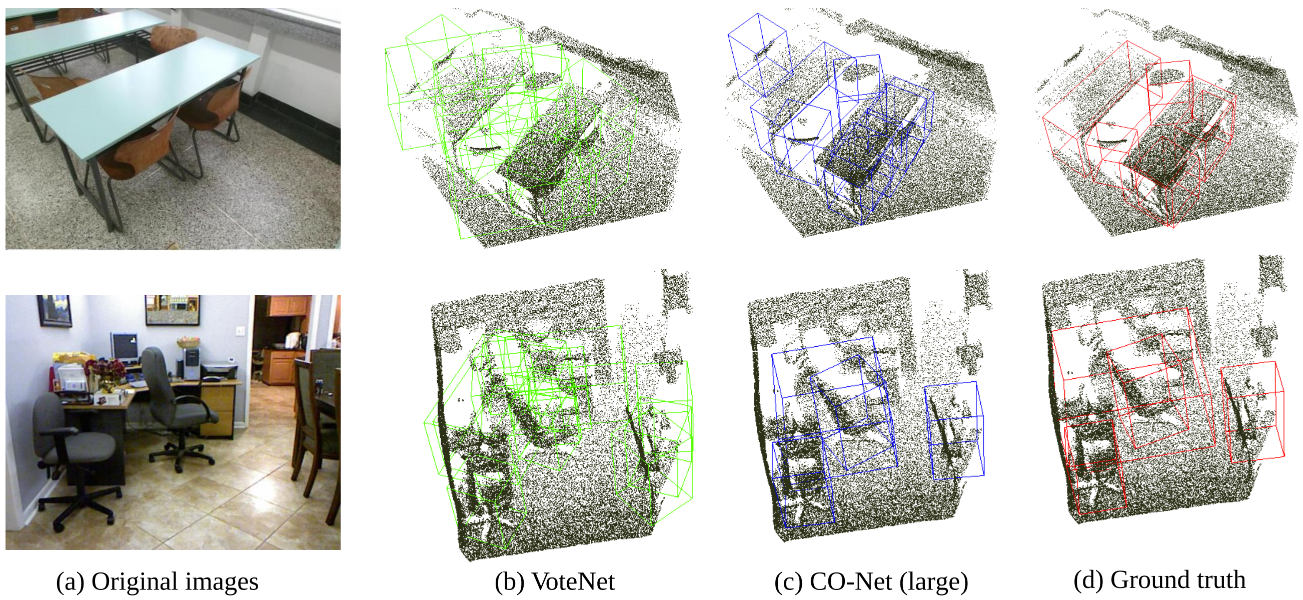


Figure 1: **Qualitative comparison results of 3D object detection on ScanNet V2.** Note that color is only used for better visualization and not utilized in Point-NAS. From left to right: 3D object detection by VoteNet, CO-Net (large), and the ground truth.

Method	Input	Bathtub	Bed	Bookshelf	Chair	Desk	Dresser	Nightstand	Sofa	Table	Toilet	#Params (M)	mAP@0.25
VoteNet [7]	Geo-only	74.4	83.0	28.8	75.3	22.0	29.8	62.2	64.0	47.3	90.1	**1**	59.1
MLCVNet [15]	Geo-only	79.2	85.8	31.9	75.8	26.5	31.3	61.5	66.3	50.4	89.1	1.2	59.8
3Detr [6]	Geo-only	77.6	81.8	27.5	68.0	28.7	28.6	56.6	58.3	50.0	90.3	7.1	59.1
Group-free(L6,O256) [4]	Geo-only	80.0	87.8	32.5	79.4	32.6	36.0	66.7	70.0	53.8	91.1	19.8	63.0
DisARM [2]	Geo-only	76.7	86.2	**35.4**	78.4	31.0	34.6	66.3	68.1	51.2	86.9	1.3	61.5
RBGNet(R66, O256) [11]	Geo-only	**80.7**	**88.4**	34.6	**82.8**	32.1	**38.8**	66.8	71.1	54.7	**91.4**	3.7	**64.1**
CO-Net (base)	Geo-only	76.3	85.9	33.1	79.1	33.4	33.6	68.6	71.1	53.9	91.2	1.1	62.6
CO-Net (large)	Geo-only	77.1	85.9	34.5	80.2	**35.1**	38.3	**68.7**	**72.4**	**55.0**	89.4	6.6	63.7

Table 4: 3D object detection results per category on the SUN RGBD dataset, evaluated with mAP@0.25 IoU. #Params refers to the number of parameters. Group-free (L6, O256) denotes a model with a 6-layer decoder (i.e., 6 attention modules) and 256 object candidates. RBGNet(R66, O256) denotes a model with 66 rays and 256 object candidates.

Method	Input	Bathtub	Bed	Bookshelf	Chair	Desk	Dresser	Nightstand	Sofa	Table	Toilet	#Params (M)	mAP@0.50
VoteNet [7]	Geo-only	45.4	53.4	6.8	56.5	5.9	12.0	38.6	49.1	21.3	68.5	**1**	35.8
H3DNet* [18]	Geo-only	47.6	52.9	8.6	60.1	8.4	20.6	45.6	50.4	27.1	69.1	-	39.0
BRNet [1]	Geo-only	55.5	63.8	9.3	61.6	10.0	**27.3**	**53.2**	56.7	28.6	70.9	-	43.7
Group-free (L6,O256) [4]	Geo-only	64.0	**67.1**	**12.4**	62.6	**14.5**	21.9	49.8	**58.2**	**29.2**	**72.2**	19.8	**45.2**
CO-Net (base)	Geo-only	54.6	57.7	7.0	59.8	10.4	24.9	50.1	55.3	26.4	65.1	1.1	41.1
CO-Net (large)	Geo-only	**64.2**	65.8	11.4	**63.3**	12.0	24.2	51.1	56.6	28.6	68.8	6.6	44.6

Table 5: 3D object detection results per category on the SUN RGBD dataset, evaluated with mAP@0.50 IoU. #Params refers to the number of parameters. Group-free (L6, O256) denotes a model with a 6-layer decoder (i.e., 6 attention modules) and 256 object candidates.

ized coordinate of each point.

S3DIS. In the semantic segmentation task, the input for CO-Net (base/large) is a randomly subsampled point cloud with a size of $N_{seg} \times C_{seg}^{in}$, where $Nseg$ is the number of sampled points, specified to 4096 for CO-Net (base) and 4096×4 for CO-Net (large), and $C_{seg}^{in} = 9$ represents the input coordinate, color and normalized input coordinate of each point.

SUN RGBD. For object detection, CO-Net(base/large) employs a randomly sampled point cloud of size $N_{det} \times C_{det}^{in}$, where N_{det} is adjusted to 20k, representing the number of sampled points and $C_{det}^{in} = 4$ refers to the input coordinate and its corresponding height (distance to floor). Notably, we estimate floor height as the 1% percentile of heights of all points.

F.2. Subsample points number in each stage

CO-Net consists of four stages (as illustrated in Appendix A.1), each of which downsamples the number of input points with the furthest points sampling based on 3D Euclidean distance (D-FPS) [8] in the first stage and the furthest points sampling based on feature distance (F-FPS) [17] in the rest stages. For point classification task, CO-Net (base/large) downsamples the number of points to 512, 256, 64 and 16 in four stages respectively. For point segmentation task, CO-Net (base) downsamples the number of points to 1024, 256, 64 and 16 in four stages respectively, while the four stages in CO-Net (large) downsample the points number to 1024×4, 256×4, 64×4 and 16×4. For point object detection task, the four stages in CO-Net (base/large) downsample the points number to 2048, 1024, 512 and 256 respectively.

F.3. Training recipe for VoteNet

In our implementation, the mAP@0.25 and mAP@0.5 of VoteNet [7] reach 61.4 and 37.9 respectively. The training recipe for VoteNet is given as: (1) using the AdamW optimizer with a weight decay of 0.05; (2) setting the batchsize to 16; (3) employing 8 Tesla A100 GPUs to train the VoteNet with a total of 180 epochs; (4) setting the initial learning rate to 0.008, decayed by 10x at the 120-th epoch and the 160-th epoch; (5) applying gradnorm clip to stabilize the training dynamics; (6) xyz coordinate and the height feature of each point are considered as the initial point features; (7) using the furthest points sampling based on 3D Euclidean distance (D-FPS) [8] in the first stage and the furthest points sampling based on feature distance (F-FPS) [17] in the rest stages to downsample the incoming points; (8) using the same data augmentation in VoteNet to augment the training data.

F.4. Structures for each task

Fig. 2 and Fig. 3 visualize the architectures for all tasks searched by CO-Net.

G. Limitation

In this work, we introduce CO-Net, a cohesive network that jointly learns multiple point cloud tasks under heterogeneous dataset domains. CO-Net is able to achieve superior performance while maintaining the storage efficiency of model deployment and generalizing to new tasks. In light of the experimental results, we note that the parameters of the head network for certain tasks are relatively large, indicating that there may be room for further parameter sharing beyond the backbone. As a result, in future work, we intend to investigate the feasibility of an encoder-decoder structure that enables parameter sharing across the entire network for all tasks.

References

[1] Bowen Cheng, Lu Sheng, Shaoshuai Shi, Ming Yang, and Dong Xu. Back-tracing representative points for voting-based 3d object detection in point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8963–8972, 2021. 3

[2] Yao Duan, Chenyang Zhu, Yuqing Lan, Renjiao Yi, Xinwang Liu, and Kai Xu. Disarm: displacement aware relation module for 3d detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16980–16989, 2022. 3

[3] Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9267–9276, 2019. 2

[4] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2949–2958, 2021. 2, 3

[5] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. In *International Conference on Learning Representations*, 2021. 2

[6] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2906–2917, 2021. 3

[7] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9277–9286, 2019. 2, 3, 4

[8] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2, 4

[9] Guocheng Qian, Hasan Hammoud, Guohao Li, Ali Thabet, and Bernard Ghanem. Assanet: An anisotropic separable set abstraction for efficient point cloud representation learning. *Advances in Neural Information Processing Systems*, 34:28119–28130, 2021. 2

[10] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for

point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6411–6420, 2019. 2

[11] Haiyang Wang, Shaoshuai Shi, Ze Yang, Rongyao Fang, Qi Qian, Hongsheng Li, Bernt Schiele, and Liwei Wang. Rbgnet: Ray-based grouping for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1110–1119, 2022. 3

[12] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 2

[13] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019. 2

[14] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 1

[15] Qian Xie, Yu-Kun Lai, Jing Wu, Zhoutao Wang, Yiming Zhang, Kai Xu, and Jun Wang. Mlcvnet: Multi-level context votenet for 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10447–10456, 2020. 3

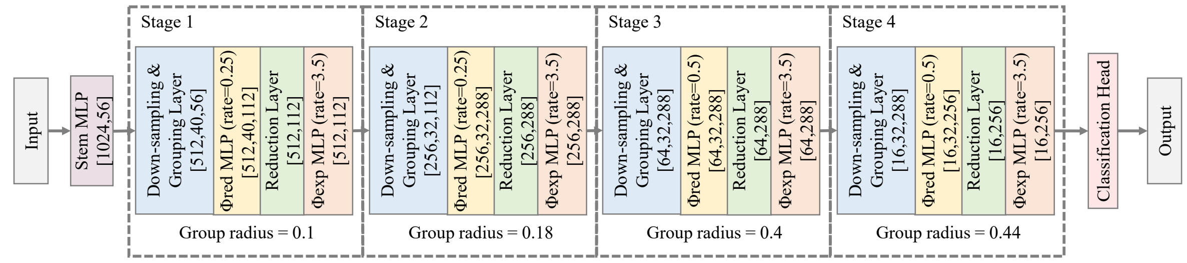
[16] Tao Xie, Shiguang Wang, Ke Wang, Linqi Yang, Zhiqiang Jiang, Xingcheng Zhang, Kun Dai, Ruifeng Li, and Jian Cheng. Poly-pc: A polyhedral network for multiple point cloud tasks at once. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1233–1243, 2023. 1

[17] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048, 2020. 4

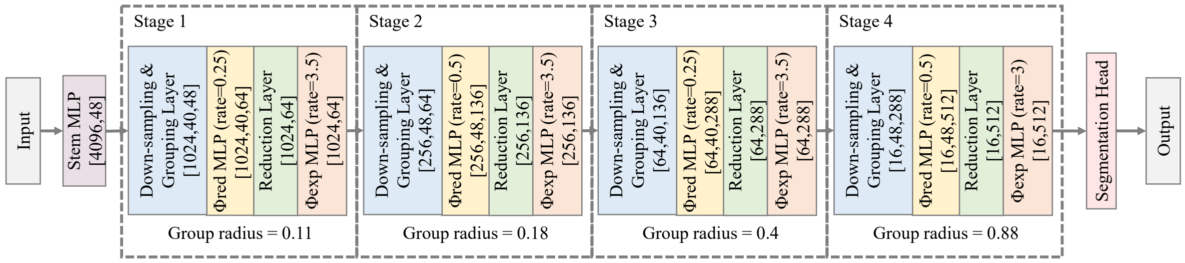
[18] Zaiwei Zhang, Bo Sun, Haitao Yang, and Qixing Huang. H3dnet: 3d object detection using hybrid geometric primitives. In *European Conference on Computer Vision*, pages 311–329. Springer, 2020. 3

[19] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5565–5573, 2019. 2

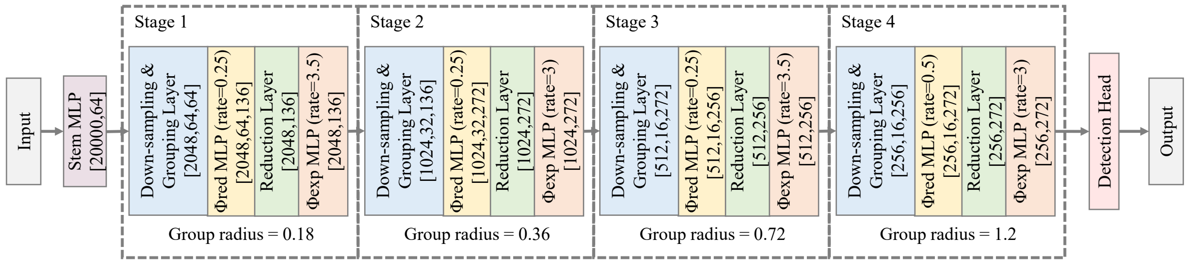
[20] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16259–16268, 2021. 2


Input
Stem MLP [1024,56]
Stage 1
Down-sampling & Grouping Layer [512,40,56]
Φred MLP (rate=0.25) [512,40,112]
Reduction Layer [512,112]
Φexp MLP (rate=3.5) [512,112]
Group radius = 0.1
Stage 2
Down-sampling & Grouping Layer [256,32,112]
Φred MLP (rate=0.25) [256,32,288]
Reduction Layer [256,288]
Φexp MLP (rate=3.5) [256,288]
Group radius = 0.18
Stage 3
Down-sampling & Grouping Layer [64,32,288]
Φred MLP (rate=0.5) [64,32,288]
Reduction Layer [64,288]
Φexp MLP (rate=3.5) [64,288]
Group radius = 0.4
Stage 4
Down-sampling & Grouping Layer [16,32,288]
Φred MLP (rate=0.5) [16,32,256]
Reduction Layer [16,256]
Φexp MLP (rate=3.5) [16,256]
Group radius = 0.44
Classification Head
Output

(a) Point classification

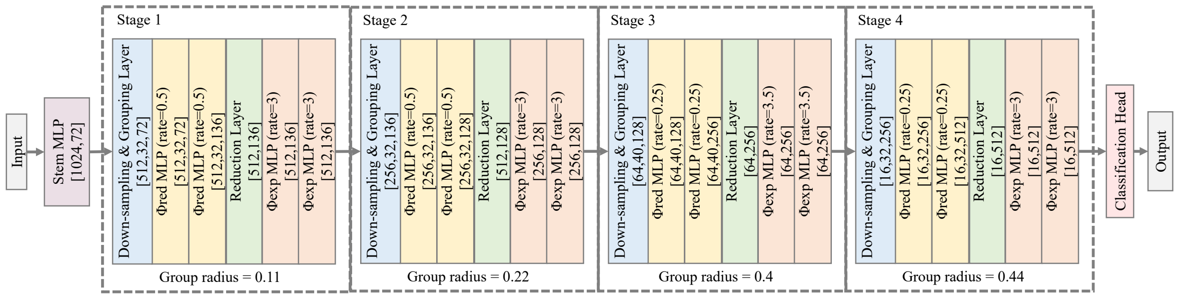

Input
Stem MLP [4096,48]
Stage 1
Down-sampling & Grouping Layer [1024,40,48]
Φred MLP (rate=0.25) [1024,40,64]
Reduction Layer [1024,64]
Φexp MLP (rate=3.5) [1024,64]
Group radius = 0.11
Stage 2
Down-sampling & Grouping Layer [256,48,64]
Φred MLP (rate=0.5) [256,48,136]
Reduction Layer [256,136]
Φexp MLP (rate=3.5) [256,136]
Group radius = 0.18
Stage 3
Down-sampling & Grouping Layer [64,40,136]
Φred MLP (rate=0.25) [64,40,288]
Reduction Layer [64,288]
Φexp MLP (rate=3.5) [64,288]
Group radius = 0.4
Stage 4
Down-sampling & Grouping Layer [16,48,288]
Φred MLP (rate=0.5) [16,48,512]
Reduction Layer [16,512]
Φexp MLP (rate=3) [16,512]
Group radius = 0.88
Segmentation Head
Output

(b) Point segmentation


Input
Stem MLP [20000,64]
Stage 1
Down-sampling & Grouping Layer [2048,64,64]
Φred MLP (rate=0.25) [2048,64,136]
Reduction Layer [2048,136]
Φexp MLP (rate=3.5) [2048,136]
Group radius = 0.18
Stage 2
Down-sampling & Grouping Layer [1024,32,136]
Φred MLP (rate=0.25) [1024,32,272]
Reduction Layer [1024,272]
Φexp MLP (rate=3) [1024,272]
Group radius = 0.36
Stage 3
Down-sampling & Grouping Layer [512,16,272]
Φred MLP (rate=0.25) [512,16,256]
Reduction Layer [512,256]
Φexp MLP (rate=3.5) [512,256]
Group radius = 0.72
Stage 4
Down-sampling & Grouping Layer [256,16,256]
Φred MLP (rate=0.5) [256,16,272]
Reduction Layer [256,272]
Φexp MLP (rate=3) [256,272]
Group radius = 1.2
Detection Head
Output

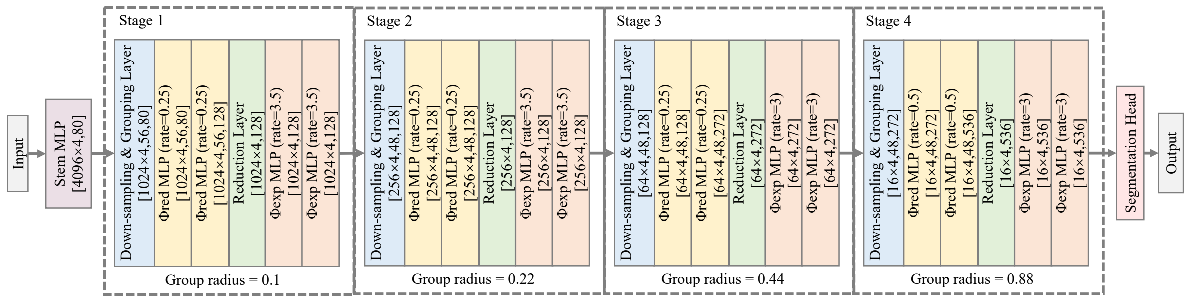
(c) Point detection

Figure 2: Searched architectures of CO-Net (base) for point classification, segmentation and detection.



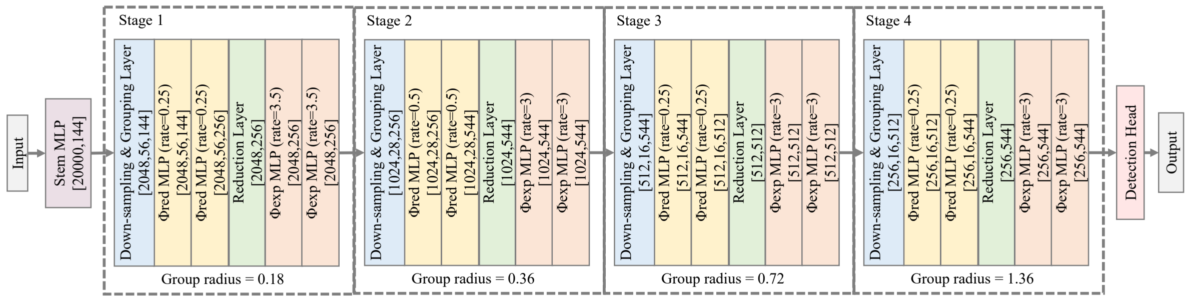
Input
Stem MLP [1024,72]
Stage 1
Down-sampling & Grouping Layer [512,32,72]
Φred MLP (rate=0.5) [512,32,72]
Φred MLP (rate=0.5) [512,32,136]
Reduction Layer [512,136]
Φexp MLP (rate=3) [512,136]
Φexp MLP (rate=3) [512,136]
Group radius = 0.11
Stage 2
Down-sampling & Grouping Layer [256,32,136]
Φred MLP (rate=0.5) [256,32,136]
Φred MLP (rate=0.5) [256,32,128]
Reduction Layer [512,128]
Φexp MLP (rate=3) [256,128]
Φexp MLP (rate=3) [256,128]
Group radius = 0.22
Stage 3
Down-sampling & Grouping Layer [64,40,128]
Φred MLP (rate=0.25) [64,40,128]
Φred MLP (rate=0.25) [64,40,256]
Reduction Layer [64,256]
Φexp MLP (rate=3.5) [64,256]
Φexp MLP (rate=3.5) [64,256]
Group radius = 0.4
Stage 4
Down-sampling & Grouping Layer [16,32,256]
Φred MLP (rate=0.25) [16,32,256]
Φred MLP (rate=0.25) [16,32,512]
Reduction Layer [16,512]
Φexp MLP (rate=3) [16,512]
Φexp MLP (rate=3) [16,512]
Group radius = 0.44
Classification Head
Output

(a) Point classification



Input
Stem MLP [4096×4,80]
Stage 1
Down-sampling & Grouping Layer [1024×4,56,80]
Φred MLP (rate=0.25) [1024×4,56,80]
Φred MLP (rate=0.25) [1024×4,56,128]
Reduction Layer [1024×4,128]
Φexp MLP (rate=3.5) [1024×4,128]
Φexp MLP (rate=3.5) [1024×4,128]
Group radius = 0.1
Stage 2
Down-sampling & Grouping Layer [256×4,48,128]
Φred MLP (rate=0.25) [256×4,48,128]
Φred MLP (rate=0.25) [256×4,48,128]
Reduction Layer [256×4,128]
Φexp MLP (rate=3.5) [256×4,128]
Φexp MLP (rate=3.5) [256×4,128]
Group radius = 0.22
Stage 3
Down-sampling & Grouping Layer [64×4,48,128]
Φred MLP (rate=0.25) [64×4,48,128]
Φred MLP (rate=0.25) [64×4,48,272]
Reduction Layer [64×4,272]
Φexp MLP (rate=3) [64×4,272]
Φexp MLP (rate=3) [64×4,272]
Group radius = 0.44
Stage 4
Down-sampling & Grouping Layer [16×4,48,272]
Φred MLP (rate=0.5) [16×4,48,272]
Φred MLP (rate=0.5) [16×4,48,536]
Reduction Layer [16×4,536]
Φexp MLP (rate=3) [16×4,536]
Φexp MLP (rate=3) [16×4,536]
Group radius = 0.88
Segmentation Head
Output

(b) Point segmentation



Input
Stem MLP [20000,144]
Stage 1
Down-sampling & Grouping Layer [2048,56,144]
Φred MLP (rate=0.25) [2048,56,144]
Φred MLP (rate=0.25) [2048,56,256]
Reduction Layer [2048,256]
Φexp MLP (rate=3.5) [2048,256]
Φexp MLP (rate=3.5) [2048,256]
Group radius = 0.18
Stage 2
Down-sampling & Grouping Layer [1024,28,256]
Φred MLP (rate=0.5) [1024,28,256]
Φred MLP (rate=0.5) [1024,28,544]
Reduction Layer [1024,544]
Φexp MLP (rate=3) [1024,544]
Φexp MLP (rate=3) [1024,544]
Group radius = 0.36
Stage 3
Down-sampling & Grouping Layer [512,16,544]
Φred MLP (rate=0.25) [512,16,544]
Φred MLP (rate=0.25) [512,16,512]
Reduction Layer [512,512]
Φexp MLP (rate=3) [512,512]
Φexp MLP (rate=3) [512,512]
Group radius = 0.72
Stage 4
Down-sampling & Grouping Layer [256,16,512]
Φred MLP (rate=0.25) [256,16,512]
Φred MLP (rate=0.25) [256,16,544]
Reduction Layer [256,544]
Φexp MLP (rate=3) [256,544]
Φexp MLP (rate=3) [256,544]
Group radius = 1.36
Detection Head
Output

(c) Point detection

Figure 3: Searched architectures of CO-Net (large) for point classification, segmentation and detection.