

Point-GCC : Universal Self-supervised 3D Scene Pre-training via Geometry-Color Contrast

Supplementary Material

Anonymous Author(s)

A IMPLEMENTATION DETAILS

In this section, we provide the details and hyperparameters for pre-training and transfer learning.

A.1 Experimental details

Pre-training Details Our pre-training settings are based on the default detection configs in MMDetection3D [3]. For all settings, we use the AdamW optimizer with an initial learning rate of 0.001 and weight decay of 0.0001. More detailed training configurations are shown in Table 1.

Downstream Transferring Details For 3D object detection task, we fine-tune with the PointNet++ [7] backbone for VoteNet [6], VoteNet+FF [9] and GroupFree-3D [5] and the MinkResNet [2] backbone for TR3D [9], TR3D+FF [9] respectively. For 3D instance segmentation task, we fine-tune with the MinkResNet backbone for TD3D [4] on ScanNet and S3DIS datasets. For 3D semantic segmentation task, we fine-tune with PointNet++ backbone for PointNet++(SSG) [7]. We follow the default setting in the corresponding downstream model. More detailed training configurations are shown in Table 1.

Config	VoteNet	GroupFree-3D	PointNet++(SSG)	TR3D	TD3D
backbone	PointNet2SASSG	PointNet2SASSG	PointNet2SASSG	MinkResNet	MinkResNet
Pre-training Details					
optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
learning rate	1e-3	1e-3	1e-3	1e-3	1e-3
weight decay	1e-4	1e-4	1e-4	1e-4	1e-4
training epochs	400	400	400	200	200
lr scheduler	cosine	cosine	cosine	cosine	cosine
batch size	4x8	4x8	4x8	4x4	4x4
augmentation	default	default	default	default	default
Downstream Transferring Details					
optimizer	AdamW	AdamW	Adam	AdamW	AdamW
learning rate	4e-3	3e-3	1e-3	1e-3	1e-3
weight decay	1e-2	5e-4	1e-2	1e-4	1e-4
training epochs	72	160	200	24	66
lr scheduler	step	step	cosine	step	step
batch size	8x8	8x4	16x2	16x1	6x1
augmentation	default	default	default	default	default
GPU device	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 3090	RTX 3090

Table 1: Training recipes for pretraining and downstream fine-tuning.

A.2 Implementation details of Deep Clustering

We provide a pseudo-code for Deep Clustering via Swapped Prediction training loop in Pytorch style as follows. All of the hyperparameters are the same as the previous works [1] in 2D. The temperature is set to 0.1 and the Sinkhorn regularization parameter is set to 0.05 for all runs.

```
# C: prototypes (DxK) i.e., linear layer
# feat: feature from backbone + projection head
```

```
# temp: temperature

def swapped_prediction(feats_geo, feats_color):

    # normalize prototypes
    with torch.no_grad():
        C = normalize(C, dim=0, p=2)

    # normalize features
    norm_geo = normalize(feats_geo, dim=-1, p=2)
    norm_color = normalize(feats_color, dim=-1, p=2)

    # compute scores
    scores_geo = mm(norm_geo, C)
    scores_color = mm(norm_color, C)

    # compute assignments
    with torch.no_grad():
        q_geo = sinkhorn(scores_geo)
        q_color = sinkhorn(scores_color)

    # convert scores to probabilities
    p_geo = Softmax(scores_geo / temp)
    p_color = Softmax(scores_color / temp)

    loss = - 0.5 * mean(q_geo * log(p_color) + q_color
    * log(p_geo))

# Sinkhorn-Knopp
def sinkhorn(scores, eps=0.05, niters=3):
    Q = exp(scores / eps).T
    Q /= sum(Q)
    K, B = Q.shape
    u, r, c = zeros(K), ones(K) / K, ones(B) / B
    for _ in range(niters):
        u = sum(Q, dim=1)
        Q *= (r / u).unsqueeze(1)
        Q *= (c / sum(Q, dim=0)).unsqueeze(0)
    return (Q / sum(Q, dim=0, keepdim=True)).T
```

B ADDITIONAL EXPERIMENTS

B.1 Pseudo-label Classes

We compare the results with different pseudo-label classes in deep clustering to analyze the effect of cluster distribution. The results in table 2 show that more pseudo-label classes degrade the performance of downstream tasks. We guess that finer-grained labels in

the ScanNet dataset, such as matching to ScanNet200 [8], show more apparent characteristics of long-tailed data, contrary to our assumption of equal cluster distribution because we are agnostic to the ground truth labels in pre-training.

Pseudo-label Classes	Object Detection	
	AP ₂₅	AP ₅₀
20	65.3	44.1
40	64.3	43.7
200	63.9	42.6

Table 2: Ablation study of the pseudo-label classes in Deep Clustering.

B.2 Error elimination

To obtain statistically significant results, we train our models five times and evaluate each trained model five times independently on erratic downstream tasks. Table 3 shows the best and the average value (in brackets).

Method	ScanNetV2		SUN RGB-D		S3DIS	
	AP ₂₅	AP ₅₀	AP ₂₅	AP ₅₀	AP ₂₅	AP ₅₀
VoteNet+FF [9]	-	-	64.5 (63.7)	39.2 (38.1)	-	-
+ Point-GCC	-	-	<u>64.9</u> (64.2)	<u>41.3</u> (40.6)	-	-
GroupFree-3D [5]	66.3 (65.7)	47.8 (47.7)	-	-	-	-
+ Point-GCC	<u>68.1</u> (67.3)	<u>49.2</u> (48.7)	-	-	-	-
TR3D [9]	72.9 (72.0)	59.3 (57.4)	67.1 (66.3)	50.4 (49.6)	74.5 (72.1)	51.7 (47.6)
+ Point-GCC	<u>73.1</u> (72.2)	<u>59.6</u> (57.2)	<u>67.7</u> (66.1)	<u>51.0</u> (49.9)	74.9(72.6)	53.2(50.9)
+ Point-GCC [†]	-	-	-	-	<u>75.1</u> (73.5)	<u>56.7</u> (54.4)
TR3D+FF [9]	-	-	69.4 (68.7)	53.4 (52.4)	-	-
+ Point-GCC	-	-	<u>69.7</u> (69.0)	<u>54.0</u> (53.3)	-	-

Table 3: 3D Object detection results on ScanNetV2 validation set. The main result is the best value, and the number within the bracket is the average value across 25 trials following previous works [5, 9]. [†] means with extra training dataset ScanNetV2.

REFERENCES

- [1] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2020. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). <https://proceedings.neurips.cc/paper/2020/hash/70feb62b69f16e0238f741fab228fec2-Abstract.html>
- [2] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 2019. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3075–3084.
- [3] MMDetection3D Contributors. 2020. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. <https://github.com/open-mmlab/mmdetection3d>.
- [4] Maksim Kolodiaznyi, Danila Rukhovich, Anna Vorontsova, and Anton Konushin. 2023. Top-Down Beats Bottom-Up in 3D Instance Segmentation. *CoRR* abs/2302.02871 (2023). <https://doi.org/10.48550/arXiv.2302.02871> arXiv:2302.02871
- [5] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. 2021. Group-Free 3D Object Detection via Transformers. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*. IEEE, 2929–2938. <https://doi.org/10.1109/ICCV48922.2021.00294>
- [6] Charles R. Qi, Or Litany, Kaiming He, and Leonidas J. Guibas. 2019. Deep Hough Voting for 3D Object Detection in Point Clouds. In *Int. Conf. Comput. Vis. (ICCV)*. IEEE, 9276–9285.

- [7] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Adv. Neural Inform. Process. Syst. (NIPS)*. 5099–5108.
- [8] Dávid Rozenberszki, Or Litany, and Angela Dai. 2022. Language-Grounded Indoor 3D Semantic Segmentation in the Wild. In *Eur. Conf. Comput. Vis. (ECCV)*.
- [9] Danila Rukhovich, Anna Vorontsova, and Anton Konushin. 2023. TR3D: Towards Real-Time Indoor 3D Object Detection. *CoRR* abs/2302.02858 (2023). <https://doi.org/10.48550/arXiv.2302.02858> arXiv:2302.02858