A Closer Look at Invariances in Self-supervised Learning for 3D Vision —*Supplementary Material*—

Lanxiao Li[0000−0003−3267−2525] and Michael Heizmann[0000−0001−9339−2055]

Institute of Industrial Information Technology, Karlsruhe Institute of Technology, Karlsruhe, Germany
{lanxiao.li, michael.heizmann}@kit.edu

Abstract. This document contains more technical details and experimental results which complement our main paper.

A Data Augmentation

Data augmentations play an important role in successful self-supervised learning. In Tab. 1, we summarize the applied approaches under our unified framework. Note that most augmentations are only used for specific input formats, as our framework supports various data formats and network structures.

B Fine-tuning Details

B.1 VoteNet

Network Structure. VoteNet [8] consists of a PointNet++ backbone [10], a voting module and a detection module. The backbone encodes 20,000 input points into 1024 feature points with 256 channels, which are further centralized by the voting module. The detection module applies bounding box regression based on the centralized features. The voting module and detection module follow the structure of PointNet [9], which is implemented as a shared MLP.
Detection on SUN RGB-D. The training setup follows the original publication [8]. The SUN RGB-D dataset [11] contains ∼10K RGB-D images. Following the official train/val split, we use ∼5K samples for the fine-tuning and evaluate the performance on the rest. We train the model with batch size 8 on a single NVIDIA 2080ti. We use an ADAM optimizer with an initial learning rate of 0.001 to train the network for 180 epochs. We decay the learning rate by 10 after 80, 120, and 160 epochs, respectively. As data augmentation, the point clouds are randomly scaled, horizontally flipped, and rotated around the vertical axis. The mean average precision (mAP) is calculated over 10 representative classes. For a fair comparison with previous works, we use the v1 annotation instead of v2. We also apply a warm-up with 10 epochs, where only the voting and detection module are updated, to stabilize the fine-tuning.

Augmentation	Target	Parameters
Random Crop	Raw	crop ratio in Uniform[0.7, 0.9]
Box Erase	Crop	erase ratio in Uniform[0.2, 0.4]
Left-right Flip	D,I,P,V	probability of 0.5
Front-back Flip	P,V	probability of 0.5
Scale	P,V	scaling factor in Uniform[0.7, 1.3]
Rotation: x-axis	P,V	angle in Uniform$[-\pi/6, \pi/6]$
Rotation: y-axis	P,V	angle in Uniform$[-\pi/6, \pi/6]$
Rotation: z-axis	P,V	angle in Uniform$[-\pi/3, \pi/3]$
Rotation: principle point	D,I	angle in Uniform$[-\pi/6, \pi/6]$
Set Zeros	D	20% percent of pixels
Sub-sample	P	randomly sample 20000 points
Translation	V	distance in Uniform[-0.1m, 0.1m]
Gaussian Blur	I,V	probability of 0.5, kernel size 5
Gray Scale	I,V	probability of 0.2
Color Jitter	I,V	probability of 0.8

Table 1: Data augmentations used in the pre-training. The column 'target' describes the data to which the augmentations are applied. Raw: raw input of our unified pipeline. Crop: a random crop from the raw input. D: depth map. I: color image. P: point cloud. V: voxel. We assume that D, I, P and V are converted from a Crop.

Detection on ScanNet. The ScanNet consists of ∼1500 multi-view point clouds. We follow the official train/val split and use ∼1.2K scans for training and ∼300 for validation. We use the v2 annotation for a fair comparison with previous works. The mAP is calculated over 18 representative classes. Other configurations follow the SUN RGB-D training.

B.2 2.5D-VoteNet

Network Structure. Similar to VoteNet, a 2.5D-VoteNet consists of a 2D CNN backbone, a voting module, and a detection module. The backbone encodes the input depth map into a feature map with 8 times smaller scale and 256 channels. Meanwhile, the depth map is scaled to the same resolution as the feature map. The scaled depth map is then converted into a point cloud, to which the feature map is registered. Then, the point features are processed by the voting and detection module.

Detection on SUN RGB-D. Most configurations follow the VoteNet finetuning. The main difference lies in the data augmentation due to different input data. For 2.5D-VoteNet, the depth maps are randomly resized, horizontally flipped, and rotated around the principle point, following the original publication [7]. Also, we apply warm-up to only train the voting and detection modules in the first 10 epochs. For evaluation, the input size is fixed to 416×544.

B.3 Sparse Residual U-Net for Segmentation

Network Structure. Since the voxel encoder used in our work is a U-shaped network with the same input and output resolution, we simply modify the last convolution layer to fit the number of classes and remove the last batch norm and activation function for fine-tuning. Other layers are initialized with pre-trained weights. Also, note that the voxel encoder supports both color (3 input channels) and geometric (1 input channel) inputs. For semantic segmentation, the color variant is used.
Segmentation on S3DIS. We follow the configurations in [4]. Specifically, we use the Area 5 of S3DIS dataset [1] for validation. With this split, 199 point clouds are used for training and 67 for validation. We use the SGD optimizer with an initial learning rate 0.1 and batch size 6. We train the model for 60K iterations and use polynomial learning rate decay with power 0.9 and step size 2000. The voxel size is fixed to 5 *cm* in the fine-tuning. Also, standard data augmentations are applied *e.g.*, random rotation, scaling, translation, and color jitter. The mean intersection over union (mIoU) is calculated over 13 classes.
Segmentation on ScanNet. We follow the official train/val split and use ∼1.2K scans for training and ∼300 for validation. We use batch size 7 and train the model for 120K iterations. Other configurations follow the S3DIS training.

C More Experimental Results

C.1 Transfer on Synthetic Data

Network	**Input**	**From Scratch**	**Pre-trained**
PointNet++	Point Clouds	88.6	90.4
Sparse Residual U-Net	Voxels	88.1	89.2

Table 2: Classification accuracy on ModelNet40 validation set.

In our main paper, we report the performance of the pre-trained encoders (*e.g.*, PointNet++, sparse residual U-Net) on object detection and semantic segmentation tasks. The pre-training and fine-tuning use both the real-world data from depth sensors. In this experiment, we investigate whether the pre-trained features can generalize on synthetic data. To this end, we pre-train the point cloud and voxel encoder with DPCo and DVCo respectively, and fine-tune the encoders on ModelNet40 dataset [12], which consists of ∼9.8K and ∼2.4K CAD models for training and validation, respectively. Note that the PointNet++ and sparse residual U-Net contain up-sampling layers to increase the resolution. For the classification task, we skip the up-sampling layers and aggregate global features at the lowest scale level.

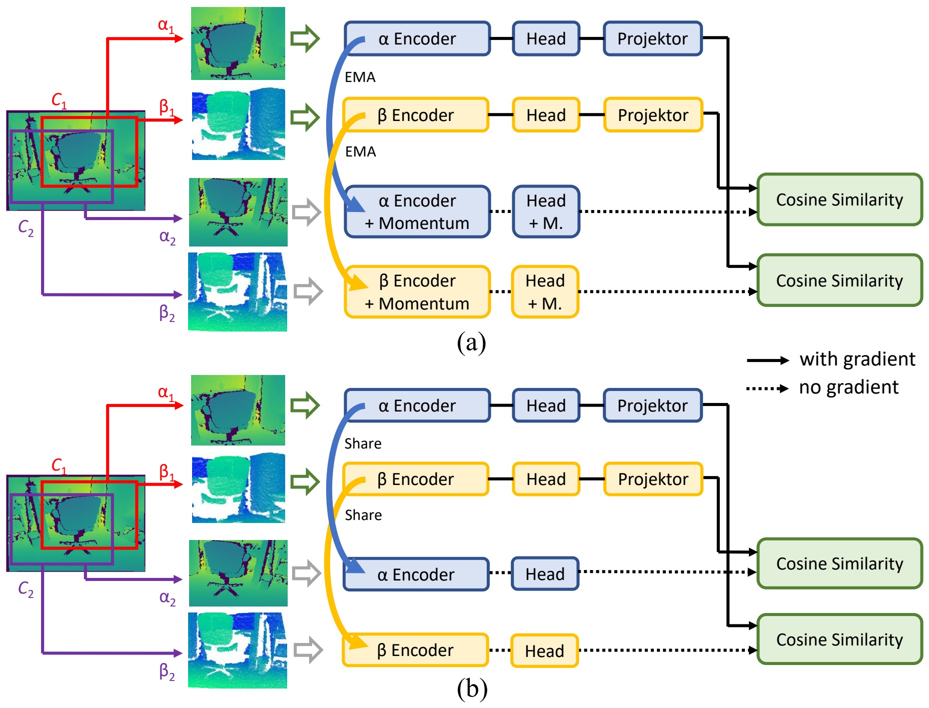

Fig. 1: Non-contrastive self-supervised learning methods for 3D vision. (a) framework following BYOL. (b) framework following SimSiam.

As shown in Tab. 2, the pre-trained weights generalize well on synthetic data. The classification accuracy with point clouds and voxels is improved by 1.8% and 1.1% (absolute), respectively. Note that the baseline performance is lower than the original PointNet++ paper [10], as our network configuration is specialized for complex indoor scenes, which might be not suitable for simple single-object synthetic data in ModelNet40.

C.2 Non-contrastive Methods

As introduced in the main paper, our pipeline uses a contrastive loss consisting of a local and a global sub-loss. The global sub-loss follows the well-known MoCo [2, 6] contrastive scheme. In this experiment, we investigate whether our idea of jointly pre-training a 2D and a 3D encoder works with non-contrastive methods *e.g.* SimSiam [3] and BYOL [5].

In comparison to MoCo, BYOL applies the cosine similarity loss instead of the contrastive loss. Also, it uses a projector to break the symmetry and prevent the mode collapse. SimSiam further simplifies the BYOL pipeline by removing the momentum encoder and sharing weights of the Siamese networks. To address the mode collapse issue, SimSiam stops the gradient from back-propagation in one of the Siamese networks. Since SimSiam and BYOL are originally applied for Siamese networks with the same structure and input formats, their pipelines are modified in this work. As shown in Fig. 1, we still only calculate the losses with features from different input formats, following the unified framework in our main paper.

We jointly pre-train a depth map and a point cloud encoder using these non-contrastive frameworks and fine-tune the point cloud encoder on the 3D object detection task. For simplicity, we omit the local correspondence in this experiment. Besides the aforementioned two methods, we also test an even simpler end-to-end pipeline, which pre-trains the two encoders and projection heads end-to-end and optimizes the cross-format similarity directly. It can be interpreted as SimSiam without projectors and stop-gradient (see Fig. 1 (b)).

Pre-training	**SUN RGB-D**		**ScanNet**	
	AP25	**AP50**	**AP25**	**AP50**
From Scatch	58.4	33.3	60.0	37.6
BYOL	58.2	32.5	62.7	40.2
SimSiam	58.6	33.6	62.4	39.9
End-to-End	58.5	34.0	62.5	39.7
DPCo (MoCo)	**59.4**	**34.9**	**63.8**	**41.0**

Table 3: VoteNet fine-tuning results on point cloud object detection task. Only the global correspondence is used in pre-training.

The results in Tab. 3 show that all pre-training methods significantly improve the detection quality on ScanNet benchmark. However, the contrastive method DPCo, which follows the idea of MoCoV2, achieves better results than non-contrastive methods. We believe that it's because the pre-training data are extracted from continuous RGB-D videos and contain a lot of similar (hard) samples. To optimize the contrastive loss, the encoders must maximize the similarity of positive pairs and the dissimilarity with negative samples. On the contrary, the non-contrastive methods don't take into account the dissimilarity and cannot benefit from hard samples. Moreover, the non-contrastive methods either bring marginal improvement or degrade the performance on SUN RGB-D dataset. It's probably because SUN RGB-D benchmark is more challenging as it contains more noisy data and requires oriented bounding box predictions instead of axis-aligned ones. Another interesting result of this experiment is that the simple end-to-end method shows good performance in fine-tuning. Note that this method leads to mode collapse in the case of Siamese networks. To address this issue, previous works introduce *e.g.* momentum encoders, memory banks, unsymmetrical projectors, stop-gradients [2, 3, 5, 6]. Since we use two different encoders in this work, this problem is avoided without the bells and whistles.

References

1. Armeni, I., Sax, S., Zamir, A.R., Savarese, S.: Joint 2d-3d-semantic data for indoor scene understanding. CoRR **abs/1702.01105** (2017)
2. Chen, X., Fan, H., Girshick, R.B., He, K.: Improved baselines with momentum contrastive learning. CoRR **abs/2003.04297** (2020), https://arxiv.org/abs/2003.04297
3. Chen, X., He, K.: Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 15750–15758 (June 2021)
4. Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3075–3084 (June 2019)
5. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., Piot, B., kavukcuoglu, k., Munos, R., Valko, M.: Bootstrap your own latent - a new approach to self-supervised learning. In: Advances in Neural Information Processing Systems. vol. 33, pp. 21271–21284 (2020)
6. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020)
7. Li, L., Heizmann, M.: 2.5D-VoteNet: Depth map based 3d object detection for real-time applications. In: Britisch Machine Vision Conference (BMVC) (2021)
8. Qi, C.R., Litany, O., He, K., Guibas, L.J.: Deep Hough voting for 3d object detection in point clouds. In: The IEEE International Conference on Computer Vision (ICCV). pp. 9277–9286 (October 2019)
9. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep learning on point sets for 3d classification and segmentation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 652–660 (July 2017)
10. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 5099–5108. Curran Associates, Inc. (2017)
11. Song, S., Lichtenberg, S.P., Xiao, J.: SUN RGB-D: A RGB-D scene understanding benchmark suite. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 567–576 (June 2015)
12. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015)