PointContrast: Unsupervised Pre-training for 3D Point Cloud Understanding (Supplementary Material)

Saining Xie[1], Jiatao Gu[1], Demi Guo*, Charles R. Qi*,
Leonidas Guibas[2*], and Or Litany[2*]

[1] Facebook AI Research
[2] Stanford University

A Visualization of the SR-UNet Architecture

Here we show the SR-UNet architecture that is used as a shared backbone in our paper for both the pre-training and the fine-tuning phases. This U-Net architecture was originally proposed in [1] for ScanNet semantic segmentation.

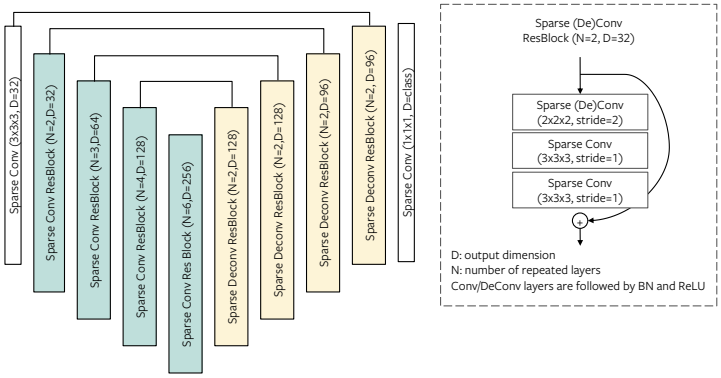

Fig. 1: SR-UNet architecture we used as a shared backbone network for pre-training and fine-tuning tasks. For segmentation and detection tasks, both the encoder and decoder weights are fine-tuned; for classification downstream tasks, only the encoder network is kept and fine-tuned.

* Work done while at Facebook AI Research.

B Visualization of the ScanNet Point Cloud Pair Dataset

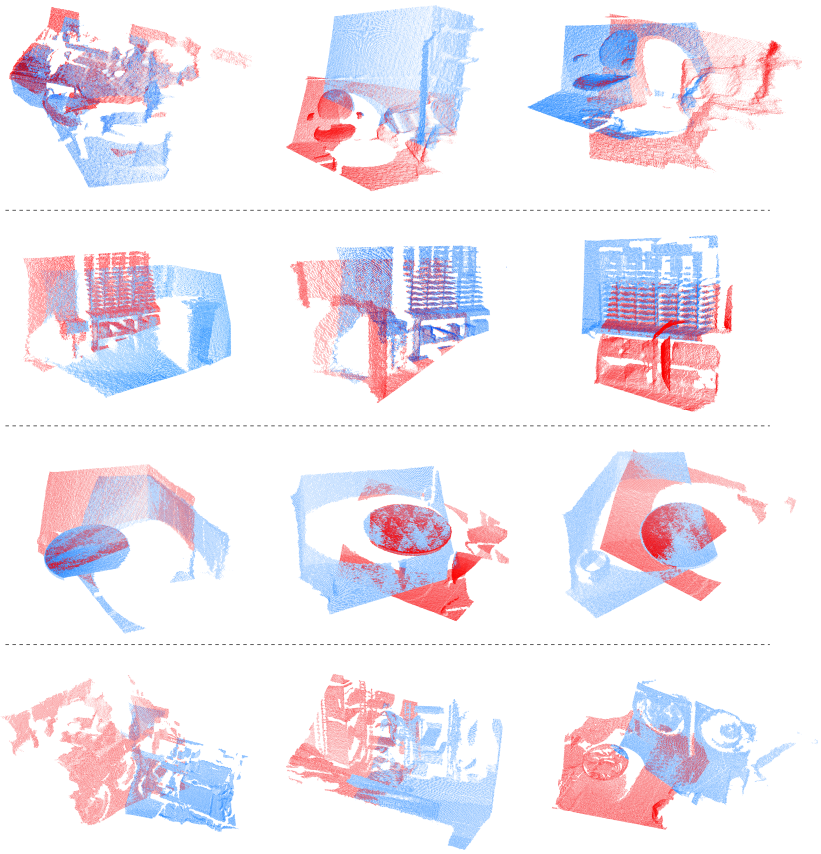

Fig. 2: Visualization of the ScanNet point cloud pair dataset used for pre-training. Each row is a randomly sampled scene. Each column is a different pair of point clouds sampled from the same scene. Different colors are corresponding to two different views (partial scans). At least 30% of the points are overlapped between two views.

C ShapeNet Supervised Training Details

We use a sparse ResNet network that has an identical structure to the encoder part of the SR-UNet in Appendix A. We use Adam optimizer, and add standard data augmentations including rotation, scaling and translation, following [9, 10]. We perform a grid search over learning rate, weight decay, voxel size (for sparse convolution), and number of input points. The best performing model configuration is: learning rate 0.004, voxel size 0.01, weight decay 1e-5, batch size 512 and

2048 input points. The 85.4% accuracy is to our knowledge the best results that has been reported on this SHREC benchmark split. We use 8 Titan-V100 GPU with data parallelism to train the model. We train the model for 200 epochs and the training takes around 8 hours.

D Details on PointContrast Pre-training

D.1 Details on Transformations

The transformations $\mathbf{T}_1$ and $\mathbf{T}_2$ applied to two views $\mathbf{x}^1$ and $\mathbf{x}^2$ in our experiments involves a random rotation (0 to 360°) along an arbitrary axis (applied independently to both views). We apply scale augmentation to both views (0.8× to 1.2× of the input scale). We have experimented with other augmentations such as translation, point coordinate jittering, and point dropout and did not find noticeable difference in fine-tuning performances.

D.2 Details on Loss Functions

For the hardest-contrastive loss, the positive sample size is 1024 and the hardest negative sample size is 256. More details can be found in [2]. For the PointInfoNCE loss we provide a detailed PyTorch-like pseudo code (and explanatory comments) in Algorithm 1.

Algorithm 1 Pseudocode of PointInfoNCE Loss implementation.

```
# f_v1, f_v2: features for matched points (in a minibatch) between view 1 and view 2: NxC
# NN: shared backbone network
# t: temperature
# Ns: subsampling size for point features.

f_v1,inds = random.choice(f_v1, Ns, dim=0) # subsample view 1 point features
f_v2 = f_v2[inds,:] # subsample view 2 point features

logits = torch.mm(f_v1, f_v2.transpose(1, 0)) # Ns by Ns
labels = torch.arange(Ns) # for k-th row, the positive sample is at the k-th position.
loss = CrossEntropyLoss(logits/t, labels)

# SGD update: shared backbone network
loss.backward()
update(NN.params)
```

`mm`: matrix multiplication;

E S3DIS Segmentation Experimental Details

Here we provide training details for S3DIS semantic segmentation task. We use the widely adopted Area 5 Test (Fold 1) split for training and testing. For all the PointContrast variants (Training from scratch, Hardest-contrastive Pretrained,

and PointInfoNCE Pretrained) we use the same hyperparameter settings. Specifically we train the model with 8 V100 GPUs with data parallelism for 10,000 iterations. Batch size is 48. Batch normalization is applied independently on each GPU. We use SGD+momentum optimizer with an initial learning rate 0.8. We use Polynomial LR scheduler with a power factor of 0.9. Weight decay is 0.0001 and voxel size is 0.05 (5cm). We use the same data augmentation techniques in [1] such as color hue/saturation augmentation and jittering, as well as scale augmentations (0.9× to 1.1×). In Table 1 we show detailed per-category performance breakdown for our models and previous approaches.

Method	ceiling	floor	wall	beam	clmn	windw	door	chair	table	bkcase	sofa	board	clutter	mIOU	mAcc
PointNet [9]	88.80	97.33	69.80	0.05	3.92	46.26	10.76	52.61	58.93	40.28	5.85	26.38	33.22	41.09	48.98
SegCloud [13]	90.06	96.05	69.86	0.00	18.37	38.35	23.12	75.89	70.40	58.42	40.88	12.96	41.60	48.92	57.35
TangentConv [12]	90.47	97.66	73.99	0.0	20.66	38.98	31.34	77.49	69.43	57.27	38.54	48.78	39.79	52.8	60.7
3D RNN [14]	95.2	98.6	77.4	0.8	9.83	52.7	27.9	76.8	78.3	58.6	27.4	39.1	51.0	53.4	71.3
PointCNN [6]	92.31	98.24	79.41	0.00	17.60	22.77	62.09	80.59	74.39	66.67	31.67	62.05	56.74	57.26	63.86
SuperpointGraph [5]	89.35	96.87	78.12	0.0	42.81	48.93	61.58	84.66	75.41	69.84	52.60	2.1	52.22	58.04	66.5
MinkowskiNet20 [1]	91.55	98.49	84.99	0.8	26.47	46.18	55.82	88.99	80.52	71.74	48.29	62.98	57.72	62.60	69.62
MinkowskiNet32 [1]	91.75	98.71	86.19	0.0	34.06	48.90	62.44	89.82	81.57	**74.88**	47.21	74.44	58.57	65.35	71.71
PntContrast(Scratch)	91.47	98.56	84.08	0.00	33.03	56.88	63.94	90.11	81.67	72.46	76.45	77.89	59.63	68.17	75.45
PntContrast(Hardest-Ctr)	**94.82**	**98.72**	**86.06**	0.00	42.84	**58.00**	**73.72**	**91.73**	**82.38**	74.74	74.58	81.42	**62.66**	**70.90**	**77.00**
PntContrast(Pnt-InfoNCE)	93.26	98.67	85.56	0.11	**45.90**	54.41	67.87	91.56	80.09	74.66	**78.20**	**81.49**	62.32	70.32	76.94

Table 1: **Stanford Area 5 Test (Fold 1).** Per-category IOU performance.

F Synthia4D Segmentation Experimental Details

Here we provide training details for Synthia4D semantic segmentation task. As mentioned in the main paper, we only use 3D sparse convnet without any temporal aggregation mechanisms such as 4D kernels and temporal CRF. For all the PointContrast variants (Training from scratch, Hardest-contrastive Pretrained, and PointInfoNCE Pretrained) we use the same hyperparameter settings, and those are mostly identical the S3DIS experiments. Specifically we train the model with 8 V100 GPUs with data parallelism for 15,000 iterations. Batch size is 72. Batch normalization is applied independently on each GPU. We use SGDmomentum optimizer with an initial learning rate 0.8. We use Polynomial LR scheduler with a power factor of 0.9. Weight decay is 0.0001 and voxel size is 0.05 (5cm). We also use the same data augmentation techniques in [1] in color space and point coordinate space. In Table 2 we show detailed per-category performance breakdown for our models and results reported in [1].

G ScanNet Segmentation Experimental Details

For ScanNet segmentation task, we train the model with 8 V100 GPUs with data parallelism for 15,000 iterations. Batch size is 48. We use SGD+momentum

Method	Bldn	Road	Sdwlk	Fence	Vegittn	Pole	Car	T. Sign	Pedstrn	Bicycl	Lane	T. Light	mIoU	mAcc
MinkNet20 + TA [1]	88.096	97.790	78.329	87.088	96.540	97.486	94.203	78.831	92.489	0.000	46.407	67.071	77.03	89.198
4D MinkNet32 + TS-CRF [1]	89.694	98.632	86.893	87.801	98.776	97.284	94.039	80.292	92.300	0.000	49.299	69.060	78.67	90.51
PntContrast(Train from scratch)	92.237	98.619	90.217	86.863	99.346	96.848	95.085	75.526	88.596	0.000	72.173	62.060	79.797	91.492
PntContrast(Hardest-Contrastive)	**92.518**	**99.040**	93.309	87.331	99.384	97.500	**96.174**	81.627	92.007	0.000	**80.257**	71.764	82.576	93.650
PntContrast(PointInfoNCE)	92.238	99.006	**93.993**	**87.368**	**99.657**	**97.755**	95.648	**83.446**	**93.279**	0.000	79.002	**76.364**	**83.146**	**93.707**

Table 2: **Synthia4D segmentation test results** Per-category IOU performance.

optimizer with an initial learning rate 0.8. We use Polynomial LR scheduler with a power factor of 0.9. Weight decay is 0.0001 and voxel size is 0.025 (2.5cm). We also use the same data augmentation techniques in [1] in color space and point coordinate space. In Table 2 we show detailed per-category performance breakdown for our models and results reported in [1].

Method	bath	bed	bookshelf	cab	chair	counter	curtain	desk	door	floor	other	pic	ref	shower	sink	sofa	tab	toilet	wall	wind	mIoU	mAcc
Scratch	84.866	96.412	64.271	80.928	90.910	**85.954**	73.890	61.767	59.935	80.753	30.896	**68.571**	62.109	75.252	54.603	67.236	90.084	**68.720**	85.936	60.279	72.169	80.718
Hardest-Ct	85.773	96.442	**66.883**	**81.525**	**91.829**	84.708	74.030	65.149	62.835	**82.645**	32.113	66.451	64.408	77.507	53.049	**69.762**	94.269	67.934	**88.601**	60.190	73.305	81.025
PntInfoNCE	**86.540**	**96.456**	66.630	81.294	91.301	84.281	**77.393**	**68.031**	**65.168**	81.600	**33.530**	66.037	**67.639**	**77.803**	**56.853**	69.398	**95.202**	68.329	88.303	**60.924**	**74.136**	**81.623**

Table 3: **ScanNet segmentation results on val set** Per-category IOU performance.

H ScanNet and SUN RGB-D Detection Details

For the 3D object detection experiments, we mostly follow the configurations in VoteNet [7] framework after switching in the SR-UNet backbone architecture. We train the model on 1 GPU, with batch size 64 for SUN RGB-D and 32 for ScanNet. Learning rate is 0.001 and we use Adam optimizer. The input points are subsampled before voxelization, we use 20000 points for SUN RGB-D and 40000 points for ScanNet. The voxel size is 2.5cm for ScanNet and 5cm for SUN RGB-D. In Table 7 we show more results reported by previous methods. In Table 4 and Table 5, we show per-category AP performance for PointContrast models agains training from scratch results, under AP@0.5 metric.

Methods	bed	table	sofa	chair	toil	desk	dress	night	book	bath	mAP@0.5
Train from Scratch	47.8	19.6	48.1	54.6	60.0	6.3	15.8	27.3	5.4	32.1	31.7
Hardest-Contrastive	**52.0**	**20.1**	**52.3**	**55.8**	**60.0**	**7.5**	14.7	36.8	**10.0**	35.6	34.5
PointInfoNCE	50.5	19.4	51.8	54.9	57.4	**7.5**	**16.2**	**37.0**	5.9	**47.6**	**34.8**

Table 4: **SUN RGB-D detection results** Per-category AP@0.5 performance.

Methods	cabinet	bed	chair	sofa	table	door	wind	bkshlf	pic	cntr	desk	curtain	refrig	shower	toilet	sink	bath	garbage	mAP@0.5
Train from Scratch	9.9	70.5	70.0	60.5	43.4	**21.8**	10.5	33.3	0.8	15.4	33.3	26.6	**39.3**	9.7	74.7	23.7	75.8	18.1	35.4
Hardest-Contrastive	10.5	68.4	**75.6**	59.1	43.1	19.6	9.6	**35.0**	**2.1**	15.6	**34.3**	**32.8**	37.8	13.6	76.9	**28.8**	**82.4**	25.8	37.3
PointInfoNCE	**13.1**	**74.7**	75.4	**61.3**	**44.8**	19.8	**12.9**	32.0	0.9	**21.9**	31.9	27.0	32.6	**17.5**	**87.4**	23.2	80.8	**26.7**	**38.0**

Table 5: **ScanNet detection results** Per-category AP@0.5 performance.

I PointContrast vs FCGF for low- and high-level tasks

We take the best performing FCGF model released in [2] that achieves a high registration feature matching recall (FMR) of: 0.958. However, this model does not perform well for S3DIS segmentation. On the other hand, the PointContrast model that performs best for segmentation achieves a lower FMR when applied to the registration task. We conclude that low-level tasks and high-level tasks in 3D might require different design choices.

methods	Registration FMR	S3DIS mIoU
FCGF[1]	**0.958**	63.06
PointContrast	0.912	**70.90**

Table 6: **FCGF vs PointContrast.** FCGF achieves a much higher registration feature matching recall, while PointContrast achieves higher mIoU for segmentation.

methods	input	mAP@0.5	mAP@0.25
DSS [11, 4]	Geo+RGB	6.8	15.2
MRCNN 2D-3D [3, 4]	Geo+RGB	10.5	17.3
F-PointNet [8, 4]	Geo+RGB	10.8	19.8
GSPN [15]	Geo+RGB	17.7	30.6
3D-SIS [4]	Geo+RGB (5 Views)	22.5	40.2
VoteNet [7]	Geo+Height	33.5	58.6
Training from scratch	Geo	35.4	56.7
PointContrast(Hardest-Contrastive)	Geo	37.3	**59.2**
PointContrast(PointInfoNCE)	Geo	**38.0**	58.5

Table 7: **3D object detection results on ScanNet dataset.** More methods in comparison.

References

1. Choy, C., Gwak, J., Savarese, S.: 4D spatio-temporal convnets: Minkowski convolutional neural networks. In: CVPR (2019)
2. Choy, C., Park, J., Koltun, V.: Fully convolutional geometric features. In: ICCV (2019)

3. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: ICCV (2017)
4. Hou, J., Dai, A., Nießner, M.: 3D-SIS: 3d semantic instance segmentation of rgb-d scans. In: CVPR (2019)
5. Landrieu, L., Simonovsky, M.: Large-scale point cloud semantic segmentation with superpoint graphs. In: CVPR (2018)
6. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: Convolution on x-transformed points. In: NeurIPS (2018)
7. Qi, C.R., Litany, O., He, K., Guibas, L.J.: Deep hough voting for 3d object detection in point clouds. ICCV (2019)
8. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3d object detection from rgb-d data. In: CVPR (2018)
9. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. CVPR (2017)
10. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. NeurIPS (2017)
11. Song, S., Xiao, J.: Deep sliding shapes for amodal 3d object detection in rgb-d images. In: CVPR (2016)
12. Tatarchenko, M., Park, J., Koltun, V., Zhou, Q.Y.: Tangent convolutions for dense prediction in 3d. In: CVPR (2018)
13. Tchapmi, L., Choy, C., Armeni, I., Gwak, J., Savarese, S.: Segcloud: Semantic segmentation of 3d point clouds. In: 3DV (2017)
14. Ye, X., Li, J., Huang, H., Du, L., Zhang, X.: 3d recurrent neural networks with context fusion for point cloud semantic segmentation. In: ECCV (2018)
15. Yi, L., Zhao, W., Wang, H., Sung, M., Guibas, L.: Gspn: Generative shape proposal network for 3d instance segmentation in point cloud. In: CVPR (2019)