Supplementary Material of Open-set Semantic Segmentation for Point Clouds via Adversarial Prototype Framework

We expand on the techniques of the adversarial prototype framework (APF) in the supplementary material, including the training algorithm, architectures of the networks in the feature adversarial module (FAM), details of the unseen class selection, and additional experimental results.

1. Training algorithm

As stated in the submitted text, we firstly train the feature extraction module (FEM) with prototypical constraint module (PCM) to acquire coarse point features, as shown in Algorithm 1.

Then we fix the feature extractor in FEM for the training stability of GAN and train the FAM and PCM in a joint scheme, as shown in Algorithm 2.

Algorithm 1: Train FEM and PCM jointly.

```
   Input: Original point clouds $\mathbf{X}$.
   Output: Extracted features $\mathbf{F}$ and prototypes $\mathbf{P}$.
1  Randomly initialize the parametric prototype set $\mathbf{P}$;
2  $t_1 \leftarrow 0$;
3  while $t_1 < e_1$ do
4      Utilize the feature extractor $\mathcal{F}$ to extract point features $\mathbf{F} = \mathcal{F}(\mathbf{X})$;
5      Compute the prototypical constraint $\mathcal{L}_{PCM}^{t_1}$;
6      Update the parametric prototypes by
           $$\mathbf{P}^{t_1+1} = \mathbf{P}^{t_1} - \mu^{t_1} \cdot \frac{\partial \mathcal{L}_{PCM}^{t_1}}{\partial \mathbf{P}^{t_1}};$$
7      Update the parameters in the feature extractor by
           $$\theta_{\mathcal{F}}^{t_1+1} = \theta_{\mathcal{F}}^{t_1} - \mu^{t_1} \cdot \frac{\partial \mathcal{L}_{PCM}^{t_1}}{\partial \theta_{\mathcal{F}}^{t_1}};$$
8      $t_1 \leftarrow t_1 + 1$;
9  end
10 Output the extracted features $\mathbf{F} = \mathcal{F}(\mathbf{X})$ and learned prototypes $\mathbf{P}$.
```

2. Architecture of the networks in FAM

The feature adversarial module consists of three parts: a generator, a discriminator, and an adversarial mapper. The generative adversarial networks are employed to synthesize unseen-class features. The adversarial mapper is designed

Algorithm 2: Train PCM and FAM jointly.

```
   Input: Original point clouds $\mathbf{X}$ and gaussian noises $n = \{n_i\}_{i=1}^{N_c}$.
   Output: Refined features $\hat{\mathbf{F}}$ and prototypes $\mathbf{P}$.
1  Randomly initialize the parametric prototype set $\mathbf{P}$;
2  $t_2 \leftarrow 0$;
3  Load the pretrained feature extractor $\mathcal{F}$ and freeze its parameters $\theta_{\mathcal{F}}$;
4  Extract point features $\mathbf{F} = \mathcal{F}(\mathbf{X}) = \{f_i\}_{i=1}^{N_c}$;
5  while $t_2 < e_2$ do
6      Update the discriminator parameters $\theta_D$ by ascending the stochastic gradient:
           $$\nabla_{\theta_D} \frac{1}{N_c} \sum_{i=1}^{N_c} \left[\log D(f_i) + \log(1 - D(G(n_i)))\right];$$
7      Update the generator parameters $\theta_G$ by ascending the stochastic gradient:
           $$\nabla_{\theta_G} \left[\frac{1}{N_c} \sum_{i=1}^{N_c} \log(D(G(n_i)))\right] + \lambda_{adv} \cdot \mathrm{Adv}(f^s, \mathbf{P});$$
8      Update the adversarial mapper parameters $\theta_M$ by descending the stochastic gradient:
           $$\nabla_{\theta_M} \left[\frac{1}{N_c} \sum_{i=1}^{N_c} \mathcal{L}_{PCM}\right] - \lambda_{adv} \cdot \mathrm{Adv}(f^s, \mathbf{P});$$
9      $t_2 \leftarrow t_2 + 1$;
10 end
11 Output the refined features $\hat{\mathbf{F}} = M(\mathbf{F})$ and learned prototypes $\mathbf{P}$.
```

Method	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
PointNet [8]	88.8	97.3	69.8	0.1	3.9	46.3	10.8	59.0	52.6	5.9	40.3	26.4	33.2
TangentConv [9]	90.5	97.7	74.0	0.0	20.7	39.0	31.3	77.5	69.4	57.3	38.5	48.8	39.8
PointWeb [12]	92.0	98.5	79.4	0.0	21.1	59.7	34.8	76.3	88.3	46.9	69.3	64.9	52.5
MinkowskiNet [6]	91.8	98.7	86.2	0.0	34.1	48.9	62.4	81.6	89.8	47.2	74.9	74.4	58.6
KPConv [10]	92.8	97.3	82.4	0.0	23.9	58.0	69.0	81.5	91.0	75.4	75.3	66.7	58.9
Point Transformer [13]	94.0	98.5	86.3	0.0	38.0	63.4	74.3	89.1	82.4	74.3	80.2	76.0	59.3

Table 1. Segmentation results on S3DIS dataset by 6 methods.

to map the features into a novel feature space for further refinement. And the architectures of these three networks are shown in Fig. 1 and described below :

- The generator has a three-layer perceptron architecture. It contains three fully-connected layers and uses ReLU (Rectified Linear Unit) as the non-linear activation function.

- The discriminator also has a three-layer perceptron architecture which is same with the generator, except that the last activation function is replaced with Sigmoid for outputting the probability.

- The adversarial mapper contains two fully-connected layers and two auxiliary batch normalization (ABN) layers [5], with ReLU being the non-linear activation function.

3. Details of the unseen class selection

We follow the setting of the existing work [4] to choose $\{other\text{-}vehicle\}$ as the unseen class on the SemanticKITTI dataset [2]. And considering that we are the first attempt to conduct open-set 3D semantic segmentation on the indoor S3DIS dataset [1], we select unseen classes according to the following principle:

- The segmentation accuracy of the unseen-class object should be appropriate. Theoretically, the segmentation performance of the backbone model could be regarded as the upper bound for the comparative closed-set evaluation. For instance, as seen from Table 1, most of the methods only achieve 0% IoU (Intersection over Union) on the $\{beam\}$. Thus, it is meaningless to choose $\{beam\}$ as the unseen class, because the model could not effectively segment the points of $\{beam\}$ whether their corresponding labels are provided or not.

In accordance to the principle mentioned above and the segmentation results in Table 1, we choose $\{window, sofa\}$ as the unseen classes on the S3DIS dataset. We also conduct several ablation studies in the submitted text with other appropriate classes chosen as the unseen classes.

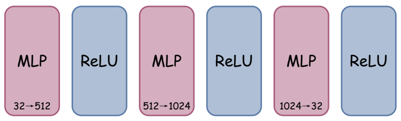
4. Additional experimental results

Effect of prototype initialization strategy. We randomly initialize the prototype set in the training procedure, as seen in Algorithm 1 and Algorithm 2. To investigate the effect of prototype initialization strategy, we conduct a contrast experiment by initializing the prototype set with all elements being zero. The experimental results are reported in Table 2. As seen from this table, different prototype initialization strategies (random or fixed) do not affect the performance of the model evidently, mainly because the parametric prototypes are learned under the prototypical constraint which takes all dimensions into consideration so that the prototypes could learn sufficiently discriminative information regardless of the initial values.

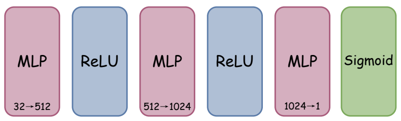
Effect of backbone. To verify the flexibility of our proposed framework, we change a different backbone to extract features. Same with the ablation studies of the effect of unseen classes in the main paper, we only compare the performance of the closed-set backbone, REAL [4] and APF

Initialization strategy	AUROC	AUPR	mIoU_c
Random initialization	90.3	30.9	69.0
Fixed initialization	90.0	31.6	69.3

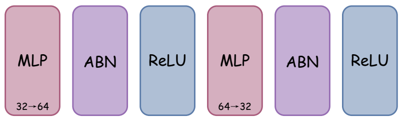
Table 2. Ablation studies of prototype initialization strategy.



(a) Architecture of the generator.



(b) Architecture of the discriminator.



(c) Architecture of the adversarial mapper.

Figure 1. Architecture of the networks in FAM.

	AUROC	AUPR	mIoU_c
Closed-set ST	-	-	**69.7**
ST + REAL [4]	86.7	28.1	69.5
ST + APF	**89.8**	**31.3**	69.2

Table 3. Ablation studies of backbone. ST denotes Stratified Transformer [7] for brevity. The best results are in bold in each metric.

here. The results are reported in Table 3, which indicate that APF is applicable to an arbitrary backbone and still achieves the best performance in AUROC and AUPR with a slight sacrifice in mIoU_c.

Visualization of the refined features. We use t-SNE [11] to reduce the dimension of the refined features from S3DIS dataset and the visualization result is shown in Fig. 2. As seen from this figure, the real unseen-class features (circles colorized in dark blue) and synthetic unseen-class features (circles colorized in light blue) are located in the center of the feature space. This is consistent with the revealed finding stated in [3] that the unseen-class samples usually aggregate in the center of the feature space.

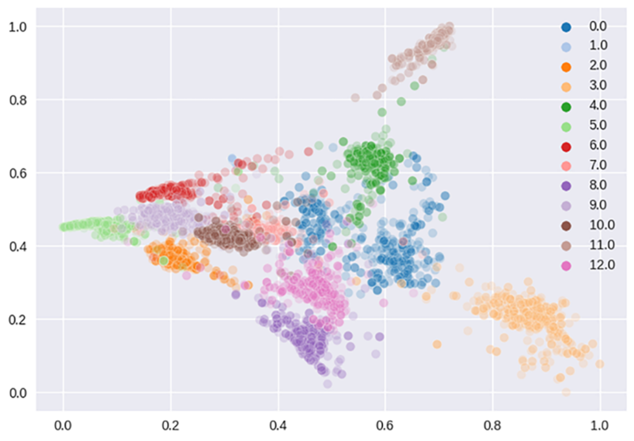


Figure 2. Visualization of the refined features. Class 0 represents the unseen-class samples, class 1 represents the synthetic samples, and class 2-12 represent the seen-class samples.

References

[1] Iro Armeni, Ozan Sener, Amir Roshan Zamir, Helen Jiang, Ioannis K. Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *CVPR*, pages 1534–1543, 2016. 2

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, C. Stachniss, and Juergen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *ICCV*, pages 9296–9306, 2019. 2

[3] Jun Cen, Peng Yun, Junhao Cai, Michael Yu Wang, and Ming Liu. Deep metric learning for open world semantic segmentation. In *ICCV*, pages 15313–15322, 2021. 3

[4] Jun Cen, Peng Yun, Shiwei Zhang, Junhao Cai, Di Luan, Michael Yu Wang, Meilin Liu, and Mingqian Tang. Open-world semantic segmentation for lidar point clouds. In *ECCV*, 2022. 2, 3

[5] Guangyao Chen, Peixi Peng, Xiangqian Wang, and Yonghong Tian. Adversarial reciprocal points learning for open set recognition. *IEEE TPAMI*, 44:8065–8081, 2022. 2

[6] Christopher Bongsoo Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, pages 3070–3079, 2019. 2

[7] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. Stratified transformer for 3d point cloud segmentation. In *CVPR*, pages 8490–8499, 2022. 3

[8] C. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 77–85, 2017. 2

[9] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *CVPR*, pages 3887–3896, 2018. 2

[10] Hugues Thomas, C. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, pages 6410–6419, 2019. 2

[11] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008. 3

[12] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *CVPR*, pages 5560–5568, 2019. 2

[13] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H. S. Torr, and Vladlen Koltun. Point transformer. In *ICCV*, pages 16239–16248, 2021. 2