Reconstruction-Aware Prior Distillation for Semi-supervised Point Cloud Completion

Zhaoxin Fan\(^1\), Yulin He\(^1\), Zhicheng Wang\(^3\), Kejian Wu\(^3\), Hongyan Liu\(^2\), Jun He\(^1\)

\(^1\)Renmin University of China
\(^2\)Tsinghua University
\(^3\)Nreal

fanzhaoxin,yolanehe,hejun@ruc.edu.cn, hyliu@tsinghua.edu.cn, zcwang,kejian@nreal.ai

Abstract

Real-world sensors often produce incomplete, irregular, and noisy point clouds, making point cloud completion increasingly important. However, most existing completion methods rely on large paired datasets for training, which is labor-intensive. This paper proposes RaPD, a novel semi-supervised point cloud completion method that reduces the need for paired datasets. RaPD utilizes a two-stage training scheme, where a deep semantic prior is learned in stage 1 from unpaired complete and incomplete point clouds, and a semi-supervised prior distillation process is introduced in stage 2 to train a completion network using only a small number of paired samples. Additionally, a self-supervised completion module is introduced to improve performance using unpaired incomplete point clouds. Experiments on multiple datasets show that RaPD outperforms previous methods in both homologous and heterologous scenarios.

1 Introduction

Point cloud can represent the 3D geometric details of a scene or an object well with high resolution, hence attracting extensive research interests and becoming increasingly important in many applications, such as autonomous driving, robotics, and augmented reality. However, point clouds scanned from real-world sensors are always incomplete, irregular, and noisy, making many algorithms designed for point cloud-based applications, such as 3D object detection [Zhu et al., 2020], unreliable. Therefore, point cloud completion, aiming at transforming a partially observed incomplete point cloud into its complete counterpart, becomes more and more popular in recent years. PCN [Yuan et al., 2018] is a pioneering work for deep learning-based point cloud completion, which utilizes PointNet [Qi et al., 2017] to learn global feature of the incomplete point cloud. Then, pointwise features of the complete point cloud are folded [Yang et al., 2018] from the global feature to generate the final complete point cloud. Due to its disability in extracting local feature, PCN fails to recover the geometric details of the object. Therefore, following works [Liu et al., 2020; Pan, 2020; Wang et al., 2020a; Xie et al., 2020; Yan et al., 2022; Yu et al., 2023; Cai et al., 2022; Wang et al., 2022] provide better completion results by preserving observed geometric details from the incomplete point shape using local features [Pan et al., 2021]. Though existing methods have achieved very promising performance, most of them require taking a large number of well-aligned complete-incomplete point cloud pairs to train their models, which is labor exhausted. In some wild scenarios, annotating the data is nearly impossible. Therefore, a question arises: can we train a point cloud completion model using less paired data?

The answer is absolutely yes. In this paper, we propose a novel deep learning model named RaPD for semi-supervised point cloud completion. RaPD achieves the point cloud completion goal by utilizing a small number of paired point cloud samples compared to previous methods. Hence, it is much more annotation friendly and gains a significant reduction in annotation cost. The intuition behind our method is that though paired samples are hard to collect, unpaired point clouds are easy to access. For example, manually designed...
clean complete point clouds from shape analysis datasets like ShapeNet [Chang et al., 2015], and noisy incomplete point clouds from autonomous driving datasets like KITTI [Geiger et al., 2013]. The semantic prior of a category of objects consisting in these unpaired point clouds is very available and we should take full advantage of it. To this end, we propose a two-stage training scheme for RaPD to achieve the goal. Fig. 1 illustrates the main idea of our method.

In training stage 1, a reconstruction-aware pretraining process is introduced. The insight behind this design is that we believe that completion-dependent features and object-shape-dependent features can be decoupled and obtained by reconstructing a point cloud using different deep learning-based models. To this end, two auto-encoders are trained for complete and incomplete point clouds separately. The one trained using unpaired complete point clouds would characterize the completion-dependent features into its decoder and the latent global code \( Z_c \), while the other one trained using unpaired incomplete point clouds would characterize the object-shape-dependent features into its encoder and latent global code \( Z_i \). The above-mentioned encoder, decoder, and latent code \( Z_c \) are regarded as the semantic prior to the next training stage.

Then, in training stage 2, we distill the learned semantic prior into a point cloud completion network to enable semi-supervised training. Specifically, weights of the above-learned encoder and decoder are adopted as a good initialization. Then, object-shape-dependent features and completion-dependent features are utilized as a balanced signal to guide the network training process. Thereby, taking advantages of this kind of prior, the network can learn the ability of competing a point cloud very well utilizing only a small number of paired training samples (10% or below than full-supervised methods in this paper). What’s more, to make usage of a large amount of unpaired incomplete point clouds, we further propose a self-supervised learning module. This module would benefit from a novel category-level latent prior and an adversarial training strategy, which further increases the performance of RaPD.

To the best of our knowledge, we are the first work to design semi-supervised models for the point cloud completion task, to verify the effectiveness of our method, extensive experiments are conducted on several widely used datasets. Experimental results show that our method can achieve excellent performance in both homologous and heterologous scenarios. When only a small number of paired training samples are available, RaPD achieves state-of-the-art performance. And our method even achieves comparable performance to fully-supervised methods. Our contribution can be summarized as: 1) We propose the first semi-supervised point cloud completion method RaPD, which achieves state-of-the-art performance in the data shortage setting. 2) We propose a two-stage training scheme, which consists of a reconstruction-aware pretraining process and a semi-supervised prior distillation process, enabling effective semi-supervised training. 3) We conduct extensive experiments to verify the effectiveness of our method in both homologous and heterologous scenarios. Contributions of each key design are also well-studied.

2 Related Work

2.1 Point Cloud Completion

Point cloud completion has obtained significant development in recent years. It plays an important role in autonomous driving, robotic grasping, and computer graphics. PCN [Yuan et al., 2018] pioneers the work of deep learning-based point cloud completion. It proposes a simple encoder-decoder network for dense and complete point set generation. Since only global features are utilized, PCN is not good at recovering the geometric details. Therefore, methods such as [Sarmad et al., 2019; Wang et al., 2020a; Tchapmi et al., 2019; Huang et al., 2020b] are proposed to learn local features better. For example, Wang et al. [Wang et al., 2020a] propose a coarse-to-fine strategy to synthesize the detailed object shapes by considering the local details of partial input with the global shape information together. There are also some methods [Wang et al., 2020b; Zhang et al., 2020; Xie et al., 2020] propose to learn better semantics for generating more complicated and cleaner complete point clouds. For example, SoftPoolNet [Wang et al., 2020b] proposes a soft pooling operation to adaptively extract better semantics features for point cloud generation. To learn finer details, VRCNet [Pan et al., 2021] proposes a variational framework that consists of a probabilistic modeling module and a relational enhancement module to improve performance. Though promising, these methods are limited in that they all require a large number of paired complete-incomplete point cloud pairs for training, which is cost unfriendly. To tackle the issue, Zhang et al. [Zhang et al., 2021] propose to use GAN-Inversion [Xia et al., 2021] for unsupervised point cloud completion. Though unpaired samples are no longer required, time-consuming inference time optimization is needed during testing in [Zhang et al., 2021], making it in-efficient. Besides, in [Zhang et al., 2021], one model only works for a particular category, further limiting its broader application. Most similar to our work is [Chen et al., 2019], which also use an encoder-decoder structure for point cloud completion. However, firstly, our approach is semi-supervised, whereas [Chen et al., 2019] is not. Secondly, we use a discriminator on incomplete point clouds with the help of a degradation module, while the [Chen et al., 2019] uses a GAN for mapping two spaces on the completion point cloud. Thirdly, we propose a semisupervised prior distillation technique that distills different features as prior, which is not discussed in it [Chen et al., 2019]. Finally, we address the case when pair-data is limited, which is not discussed in [Chen et al., 2019].

In contrast to all previous works, we propose a novel reconstruction-aware prior distillation method named RaPD for semi-supervised point cloud completion. To tackle the data shortage problem, RaPD takes full advantage of the semantic prior hidden in unpaired point clouds and distills the learned prior into the final completion network. Our method achieves superior performances than both previous supervised and unsupervised methods only taking a small number of paired point clouds for training.
2.2 Semi-supervised Learning

Semi-supervised learning is a mixed research task between supervised and unsupervised tasks [Chapelle et al., 2009; Zhu, 2005]. It targets predicting more accurately with the aid of unlabeled data than supervised learning that uses only labeled data [Yang et al., 2021]. Semi-supervised methods can be divided into generative methods [Denton et al., 2016; Dai et al., 2017], consistency regularization methods [Rasmus et al., 2015; Sajjadi et al., 2016], graph-based methods [Iscen et al., 2019; Chen et al., 2020], pseudo-labelling methods [Grandvalet and Bengio, 2004; Lee and others, 2013], and hybrid methods [Berthelot et al., 2019b; Berthelot et al., 2019a]. With the development of deep learning, semi-supervised learning has been deployed into a wide range of downstream tasks. For example, Tang et al. [Tang et al., 2016] propose a semi-supervised object detection method using visual and semantic knowledge transfer. Zhou et al. [Zhou et al., 2021] introduce a recurrent multi-scale feature modulation approach for monocular depth estimation. Quali et al. [Ouali et al., 2020] present a method that utilizes cross-consistency training for semi-supervised semantic segmentation. In the field of point cloud processing, Huang et al. [Huang et al., 2020a] propose a fast and robust semi-supervised approach for point cloud registration. Zhao et al. [Zhao et al., 2020] propose a self-ensembling technique in their work for semi-supervised 3D object detection. Cheng et al. [Cheng et al., 2021] propose to divide a point cloud into superpoints and build superpoint graphs for semi-supervised point cloud segmentation. Though semi-supervised learning has attracted massive attention, to our knowledge, there is no existing method tailored for point cloud completion.

In this paper, we propose RaPD, which is tailored for the point cloud completion task. In our work, discriminative semantic prior hidden in unpaired point clouds are distillated and utilized to achieve our semi-supervised training goal.

3 Method

3.1 Overview

Problem Statement: We aim at training a deep learning model that takes an incomplete point cloud $P_i$ as input, it can output $P_i$’s complete counterpart $P_c$. In this paper, in order to reduce the dependence on paired complete-incomplete samples, we study the semi-supervised point cloud completion task. In our setting, there exists two training datasets $D_{paired} = \{(P^1_i, P^1_c), (P^2_i, P^2_c), \ldots, (P^K_i, P^K_c)\}$ and $D_{unpaired} = \{P^1_i, P^2_i, \ldots, P^N_i, P^1_c, P^2_c, \ldots, P^M_c\}$, where $K, N$ and $M$ are number of point clouds. Point clouds in $D_{paired}$ are well-paired that can be utilized for supervised training, while point clouds in $D_{unpaired}$ are unpaired samples with great diversity. A constrain between the two datasets is that $K < N(M)$, Which means $D_{paired}$ is small scale. The deep learning model should be trained utilizing $D_{paired}$ and $D_{unpaired}$. The expectation is that the final trained model can do point cloud completion excellently even though only a small number of training samples are labeled (paired).

Framework of the Approach: To solve the semi-supervised point cloud completion problem, we propose a novel method named RaPD. Fig. 2 illustrates the framework. RaPD adopts a two-stage training scheme. In the first stage, we conduct reconstruction-aware pretraining to learn the semantic prior...
from \(D_{\text{unpaired}}\). Specifically, two auto-encoders are trained using the complete point clouds and incomplete point clouds, respectively. After trained, network weights and embedded latent codes of point clouds are regarded as the semantic prior, which would be exploited in the next stage. Then, in the second stage, we conduct semi-supervised prior distillation. This stage consists of a prior distillation module and a self-supervised completion module, both of which would take \(D_{\text{unpaired}}\) and \(D_{\text{paired}}\) as training data and jointly learn an excellent model for the final point cloud completion task.

### 3.2 Stage 1: Reconstruction-Aware Pretraining

Since it is very hard to annotate complete-incomplete point cloud pairs in real-world scenarios, only a small number of paired point clouds (\(K\) in \(D_{\text{paired}}\)) can be leveraged to train a model in the semi-supervised training setting. However, we argue that though paired data is rare, there are abundant unpaired point clouds that can be leveraged. For complete point clouds, there exist human-designed object datasets such as ShapeNet [Chang et al., 2015]. For incomplete point clouds, we can extract samples from the wild datasets such as KITTI [Geiger et al., 2013]. They consist of \(D_{\text{unpaired}}\). We note here that the semantic characteristics in these unpaired point clouds are very valuable and should be exploited to benefit the semi-supervised point cloud completion task. Therefore, we propose to learn the semantic prior from them. In fact, there are two kinds of characteristics that would benefit our task: the completeness information of an object and the shape information of an object. We term the two characteristics as completion-dependent features and object-shape-dependent features respectively. And we find that reconstruction-aware pretraining can learn them effectively.

**Learning completion-dependent features:** To learn the completion-dependent features, we train an auto-encoder using the \(M\) complete point clouds in \(D_{\text{unpaired}}\). As shown in Fig. 2, the auto-encoder consists of an encoder and a decoder. Taking a complete point cloud \(P_c\) as input, the encoder would embed it into a latent code \(Z_c\). Then, the decoder would take \(Z_c\) as input and recover the shape of \(P_c\), termed \(\hat{P}_c\). After training, the latent code \(Z_c\) would contain the completeness information of \(P_c\) since \(\hat{P}_c\) is recovered only utilizing \(Z_c\), the predicting of which is learned from massive complete point clouds without disturbed by any incomplete noise. Besides, we find that the decoder \(D_c\) works for recovering a completion point cloud, the network weights of the decoder would also contain rich implicit completeness-related information. Therefore, the decoder and \(Z_c\) together consist of the completion-dependent features. And we regard the latent code \(Z_c\) as the semantic prior.

**Learning Object-Shape-Dependent Features:** The object-shape-dependent features are learned in the similar way. The difference is that the auto-encoder is trained by using the \(N\) incomplete point clouds in \(D_{\text{unpaired}}\). Similarly, after training, the encoder \(E_i\) and latent code \(Z_i\) of the input point cloud would be very expressive in reflecting the shape of the object in an implicit way, i.e., they are actually the object-shape-dependent features that we expect to learn. It is also very straightforward to regard the two as part of the semantic prior. However, since \(Z_i\) is generated by \(E_i\), they are essentially redundant. Therefore, we only take the encoder \(E_i\) as the semantic prior. In another word, the encoder \(E_i\) is used to encode the input point cloud into a compact and low-dimensional feature representation, which contains the object-shape-dependent features. These features capture the overall shape information of the object, which we believe is crucial for accurate 3D point cloud completion.

### 3.3 Stage 2: Semi-supervised Prior Distillation

In this stage, we focus on training an encoder-decoder network to achieve the final point cloud completion goal taking \(D_{\text{paired}}\), \(D_{\text{unpaired}}\) and the semantic prior as training data. To do this, we design two modules. The first module is the distillation module, which mainly utilizes \(D_{\text{paired}}\) as input and focuses on distilling information from the semantic prior to better train the model. The second module is the self-supervised completion module, which mainly makes usage of the incomplete point clouds in \(D_{\text{unpaired}}\) to further benefit the network.

**Prior Distillation:** The distillation module quantifies the semantic prior into two categories. The first category is named weights distillation. In particular, we build the encoder-decoder network architecture to be the same as the auto-encoders in training stage 1. And we use the weights of the trained \(E_i\) and \(D_c\) as the initialization of the encoder \(E\) and decoder \(D\). Since weights of \(E_i\) and \(D_c\) are equipped with the ability of representing the shape-related information and completeness-related information, such a kind of distillation would flow very valuable prior information from training stage 1 to training stage 2. Besides, since \(E_i\) and \(D_c\) are already well capable of point cloud reconstruction, initiating the network using them would help the model coverage faster, further increasing the robustness of the model and avoiding the network from falling into local-minimal.

Besides distilling model weights, we also propose a features distillation step. Specifically, one observation is that since we initialize the decoder using \(D_c\), if the latent code \(Z\) was the same as or at least be very similar to \(Z_c\), the decoder can nearly perfectly output a complete point cloud. The reason is natural that \(D_c\) is already trained for recovering a complete point cloud in stage 1 and can perform well if the latent code is sampled from the feature space of \(Z_c\). Therefore, to promote the model to learn such a kind of prior, we encourage the distribution of \(Z\) to be close to that of \(Z_c\) during training. To achieve so, we adopt the Kullback-Leibler divergence as a loss function:

\[
L_{z,\text{paired}} = D_{\text{kl}}(Z_c || Z)
\]

Then, for point cloud in \(D_{\text{paired}}\), we use the chamfer distance loss as the point cloud similarity loss.

\[
L_{\text{cd,paired}} = \frac{1}{rN} \sum_{x \in P_1} \min_{y \in P_2} \|x - y\|_2^2 + \frac{1}{rN} \sum_{y \in P_2} \min_{x \in P_1} \|y - x\|_2^2
\]

where \(\gamma\) is a balance term.
<table>
<thead>
<tr>
<th>Methods</th>
<th>CD F1-score</th>
<th>CD F1-score</th>
<th>CD F1-score</th>
<th>CD F1-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>TopNet [Tchapmi et al., 2019]</td>
<td>20.73 0.186</td>
<td>16.49 0.220</td>
<td>14.37 0.236</td>
<td>13.62 0.251</td>
</tr>
<tr>
<td>PCN [Yuan et al., 2018]</td>
<td>20.80 0.199</td>
<td>16.16 0.229</td>
<td>12.52 0.272</td>
<td>10.99 0.299</td>
</tr>
<tr>
<td>VRCNet [Pan et al., 2021]</td>
<td>30.65 0.214</td>
<td>20.36 0.252</td>
<td>13.36 0.319</td>
<td>9.38 0.415</td>
</tr>
<tr>
<td>RaPD</td>
<td>15.19 0.263</td>
<td>12.80 0.287</td>
<td>10.90 0.315</td>
<td>10.63 0.325</td>
</tr>
<tr>
<td>RaPD*</td>
<td>12.12 0.313</td>
<td>11.07 0.319</td>
<td>10.07 0.328</td>
<td>9.64 0.335</td>
</tr>
</tbody>
</table>

Table 1: Comparison on the MVP dataset under the semi-supervised training setting. CD is loss is multiplied by $10^4$. 1% to 10% are the percentages of paired point clouds in the training set that are used for training.

<table>
<thead>
<tr>
<th>Methods</th>
<th>CD F1-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>fully supervised</td>
<td>PCN [Yuan et al., 2018]</td>
</tr>
<tr>
<td>TopNet [Tchapmi et al., 2019]</td>
<td>10.11 0.308</td>
</tr>
<tr>
<td>MSN [Liu et al., 2020]</td>
<td>7.90 0.432</td>
</tr>
<tr>
<td>ECG [Pan, 2020]</td>
<td>6.64 0.476</td>
</tr>
<tr>
<td>Wang et al. [Wang et al., 2020a]</td>
<td>7.25 0.434</td>
</tr>
<tr>
<td>VRCNet [Pan et al., 2021]</td>
<td>5.96 0.499</td>
</tr>
<tr>
<td>semi-supervised</td>
<td>RaPD*</td>
</tr>
<tr>
<td></td>
<td>RaPD</td>
</tr>
</tbody>
</table>

Table 2: Comparison with fully-supervised methods on the MVP dataset. CD loss is multiplied by $10^4$. RaPD and PaPD* are trained using 10% of paired training data, while all other methods are training using all paired training data.

**Self-supervised completion:** During training, beyond using $D_{paired}$, we also take the incomplete point clouds in $D_{unpaired}$ as training data. To take advantage of them, we propose a self-training strategy. Specifically, for point cloud $P_i$ in $D_{unpaired}$, it is first fed into the network to predict a point cloud $\hat{P}_i$. Then, since there is no ground truth for $P_i$, we can only make use of the input $P_i$ as a training signal. To do so, we should first degrade $P_n$ to $P_i$, termed $\tilde{P}_i$. For the degradation, we adopt the K-Mask Degradation Module in [Zhang et al., 2021]. Then, the loss function for this signal is set as the chamfer distance loss too:

$$L_{cd,unpaired} = \frac{1}{\gamma N} \sum_{x \in P_i} \min_{y \in \tilde{P}_i} ||x - y||_2^2$$

$$+ \frac{1}{\gamma N} \sum_{y \in \tilde{P}_i} \min_{x \in P_i} ||y - x||_2^2 (3)$$

$L_{cd,unpaired}$ would encourage $\tilde{P}_i$ to be as similar as its incomplete counterpart $P_i$ as possible during training. However, $L_{cd,unpaired}$ only supervises the similarity in a low-level manner, which is not powerful enough for the completion task. To further benefit the encoder-decoder network, we add a discriminator $D$ to guide the training process. The discriminator plays the role of encouraging $\tilde{P}_i$ to be as similar as $P_i$ as possible. It would predict a probability for each of its input, where close to 0 means it regards the point cloud as a fake incomplete one, while close to 1 means it regards the point cloud as a real incomplete one. In this way, the network would be forced to generate a more realistic $\tilde{P}_i$. Hence, the high-level shape semantics between $P_i$ and $\tilde{P}_i$ would also be aligned. The loss function for the discriminator is:

$$L_d = (D(P_i) - 1)^2 + (D(\tilde{P}_i))^2 (4)$$

$$L_g = (D(\hat{P}_i) - 1)^2 (5)$$

Though straightforward, only using $L_{cd,unpaired}$ would cause over-fitting since there is no signal account for the completeness of $P_n$. To encourage $P_n$ to also be as complete as possible, we use the following loss to make the distribution of the latent code $Z$ fall into the completeness-related space:

$$L_{z,unpaired} = D_{kl}(\bar{Z}_c||Z) (6)$$

where $\bar{Z}_c$ is the category-level mean latent code of all complete training point clouds in training stage 1. This loss can also be regarded as distilling a part of semantic prior from stage 1, further taking advantage of $D_{unpaired}$.

For training stage 1. We train two auto-encoders to formulate the reconstruction-aware pretraining. The loss functions of training both networks are chamfer distance loss. For training stage 2, the loss function for the encoder-decoder network is the weighted sum of each component of the training process:

$$L = \lambda_1 L_{z,paired} + \lambda_2 L_{z,unpaired} + \lambda_3 L_{cd,paired} + \lambda_4 L_{cd,unpaired} + \lambda_5 L_g (7)$$

where $\lambda_1$ to $\lambda_5$ are balance terms. For the discriminator, it is optimized using $L_d$ independently.

4 Experiment

4.1 Implementation Details

To verify the superiority of RaPD, we conduct experiments on three widely used public datasets: MVP [Pan et al., 2021], CRN [Wang et al., 2020a] and KITTI [Geiger et al., 2013]. We set our encoder-decoder network architecture to be the
same as PCN [Yuan et al., 2018] for a fair comparison. The architecture of the discriminator follows the basic architecture of [Li et al., 2019]. The evaluation metrics are chamfer distance (CD) and the F1 score, following [Pan et al., 2021]. More details please refer to the SuppMat.

4.2 Main Results

Results on MVP dataset: We first compare our method with PCN [Yuan et al., 2018], TopNet [Tchapmi et al., 2019] and VRCNet [Pan et al., 2021] on the MVP dataset. Since there is no existing semi-supervised point cloud completion method, for a fair comparison, we re-train their model using the same number of paired point clouds to report their performance under our semi-supervised training setting. We train all models on the MVP training set and test them on the MVP testing set. Therefore, the training and testing datasets are homologous. Table 1 and Fig. 3 show the results. RaPD* is a version of RaPD that discard the self-supervised completion module during training. It can be seen that with the increase of paired training data, performances of all models are promoted. However, no matter whether we adopt the self-supervised completion module, our method outperforms all existing methods by a large margin. These results greatly demonstrate the advantage of our idea of distilling semantic prior from large-scale unpaired data over existing supervised methods. The performance gain of RaPD over RaPD* also demonstrates the superiority of the self-supervised completion module in improving the performance. We find when using 10% data, RaPD is outperformed by VRCNet, that is because RaPD is designed for small number of labeled data. Here 10% data reaches around 6000 paired point clouds, which breaks the small data rule in practice, so it would benefit VRCNet more. But note, in real world scenarios, it is hard to collect so many paired data. Some visualization results are shown in Fig. 4. From the visualization, it can be seen that RaPD can obviously recover better geometric details than PCN and RaPD*. For example, the leg of the complete chair predicted by RaPD is obviously more clear and more slender than the one predicted by PCN.

We also compare RaPD with existing state-of-the-art supervised methods. All compared methods are training using all the paired data in the training set while RaPD and RaPD* are only trained using 10% of paired data. The result is shown in Table 2. It can be seen that though we only use 10% of paired training data, our method can achieve comparable performance with fully-supervised methods. Note that RaPD even outperforms PCN and TopNet. As mentioned before, RaPD shares the same network architecture with PCN. However, it outperforms PCN using only a small number of data, which further demonstrates the superiority of our semi-supervised training scheme.

Results on CRN: As mentioned before, the training data and testing data of training stage 1 and training stage 2 are homologous when we evaluate our method on the MVP dataset. To verify the performance of our method in the heterologous situation, we conduct experiments on CRN dataset. Specifically, we conduct the reconstruction-aware pretraining on the MVP training set and conduct the semi-supervised prior distillation on the CRN training set, and test the trained model on the CRN testing set. The results are shown in Table 3.

4.3 Ablation Study

In this section, we conduct extensive ablation study to verify the effectiveness of our key designs. All experiments are
are beneficial to our completion network. Since any of them is removed, the performance will significantly drop. When both of them are removed, the performance drop is even more serious. In a world, it can be found that both kinds of distillation play a significant role in ensuring the final completion performance. The absence of any of them would lead to a sharp performance decrease.

**Impact of Self-Supervised Completion:** In our work, we introduce a self-supervised completion module to make use of the abundant unpaired incomplete point clouds in training stage 2. The performance difference between the RaPD* and RaPD has demonstrated that this module is very important for performance gain since it helps the model learn more information from a large scale of incomplete unpaired point clouds in an effective way. In this module, we use a degradation method to get a degraded point cloud for loss computation. We find that the choice of the degradation method we use is one of the key components that may affect the performance. In Table 6, we study the impact of several different degradation methods. It can be found that the K-Mask method is the best choice. The reason may be that K-Mask can better preserve the geometric details of point cloud after degradation. What’s more, we use a discriminator for adversarial training to further increase the realism of the predicted point cloud. In Table 5, we illustrate the performance of discarding the discriminator and the corresponding adversarial loss. It can be found that without them, the performance drops a lot. Therefore, it is necessary to adopt the discriminator during training.

**Table 3:** Comparison on the CRN dataset under the semi-supervised training setting. CD loss is multiplied by $10^4$. 1% to 10% are the percentages of paired point clouds in the training set that are used for training.

<table>
<thead>
<tr>
<th></th>
<th>1%</th>
<th>2%</th>
<th>5%</th>
<th>10%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CD F1-score</td>
<td>CD F1-score</td>
<td>CD F1-score</td>
<td>CD F1-score</td>
</tr>
<tr>
<td>TopNet [Tchapmi et al., 2019]</td>
<td>39.91 0.125</td>
<td>27.77 0.157</td>
<td>22.05 0.171</td>
<td>18.86 0.193</td>
</tr>
<tr>
<td>PCN [Yuan et al., 2018]</td>
<td>36.76 0.130</td>
<td>25.65 0.165</td>
<td>18.34 0.195</td>
<td>15.50 0.215</td>
</tr>
<tr>
<td>VRCNet [Pan et al., 2021]</td>
<td>52.28 0.115</td>
<td>32.24 0.137</td>
<td>18.56 0.215</td>
<td>13.13 0.274</td>
</tr>
<tr>
<td>RaPD*</td>
<td>26.35 0.182</td>
<td>20.92 0.206</td>
<td>15.98 0.229</td>
<td>14.21 0.246</td>
</tr>
<tr>
<td>RaPD</td>
<td>17.57 0.229</td>
<td>15.70 0.235</td>
<td>13.96 0.244</td>
<td>12.47 0.272</td>
</tr>
</tbody>
</table>

**Table 4:** Ablation study: impact of prior distillation.

<table>
<thead>
<tr>
<th></th>
<th>RaPD*</th>
<th>RaPD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CD F1-score</td>
<td>CD F1-score</td>
</tr>
<tr>
<td>w/o features distillation</td>
<td>11.61 0.299</td>
<td>43.86 0.109</td>
</tr>
<tr>
<td>w/o decoder distillation</td>
<td>11.95 0.283</td>
<td>10.74 0.321</td>
</tr>
<tr>
<td>w/o encoder distillation</td>
<td>11.88 0.292</td>
<td>10.77 0.315</td>
</tr>
<tr>
<td>w/o weights distillation</td>
<td>12.52 0.272</td>
<td>11.66 0.309</td>
</tr>
<tr>
<td>full model</td>
<td><strong>10.90 0.315</strong></td>
<td><strong>10.07 0.328</strong></td>
</tr>
</tbody>
</table>

**Table 5:** Ablation study: impact of the discriminator.

<table>
<thead>
<tr>
<th></th>
<th>RaPD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CD F1-score</td>
</tr>
<tr>
<td>w/o discriminator</td>
<td>11.83 0.307</td>
</tr>
<tr>
<td>full model</td>
<td><strong>10.07 0.328</strong></td>
</tr>
</tbody>
</table>

**Table 6:** Ablation study: impact of degradation methods.

---

impact of Prior Distillation: In our work, we present two kinds of prior distillation for the final completion network: the weights distillation (including encoder weights distillation and decoder weights distillation) and the features distillation. The two kinds of distillations are both one of the most critical factors for the success of our semi-supervised point cloud completion network. Table 4 shows the impact of them. For features distillation, when it is discarded, the performance drops a lot, which demonstrates that it is very necessary to encourage the distribution of latent code of the input incomplete point cloud to be similar to that of the large scale unpaired complete point clouds. Then, we find that both the encoder weights distillation and decoder weights distillation conducted on the MVP dataset. 5% paired training samples are used for training. More ablation study can be found in the SuppMat.

**Impact of Prior Distillation:** In our work, we present two kinds of prior distillation for the final completion network: the weights distillation (including encoder weights distillation and decoder weights distillation) and the features distillation. The two kinds of distillations are both one of the most critical factors for the success of our semi-supervised point cloud completion network. Table 4 shows the impact of them. For features distillation, when it is discarded, the performance drops a lot, which demonstrates that it is very necessary to encourage the distribution of latent code of the input incomplete point cloud to be similar to that of the large scale unpaired complete point clouds. Then, we find that both the encoder weights distillation and decoder weights distillation...
5 Conclusion

Point cloud completion is an important but challenging task for many real-world applications. To reduce the dependency on paired data when training a point cloud completion model, this paper proposes a novel semi-supervised method named RaPD. A two-stage training scheme is introduced in RaPD. The first stage learns powerful semantic prior through reconstruction-aware pretraining, while the second stage learns the final completion deep model by prior distillation and self-supervised completion learning. Experiments on several widely used datasets have demonstrated the superiority of RaPD over previous supervised methods. In the future, we will research efficient unsupervised point cloud completion to further ease the training requirement.

Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant No. 2020YFB2104101 and the National Natural Science Foundation of China (NSFC) under Grant Nos. 62172421, and 62072459.

Contribution Statement

Zhaoxin Fan and Yulin He contributed equally to this work. Jun He is the corresponding author.

References


