DOI: 10.1111/cgf.14663
Pacific Graphics 2022
N. Umetani, E. Vouga, and C. Wojtan
(Guest Editors)

Volume 41 (2022), Number 7

DiffusionPointLabel: Annotated Point Cloud Generation with Diffusion Model

Tingting Li[1] Yunfei Fu[2] Xiaoguang Han[†34*] Hui Liang[‡5*]
Jian Jun Zhang[1] Jian Chang[1]

[1] Bournemouth University [2] iArt.ai [3] The Chinese University of Hong Kong, Shenzhen
[4] Shenzhen Research Institute of Big Data [5] Zhengzhou University of Light Industry

Abstract

Point cloud generation aims to synthesize point clouds that do not exist in supervised dataset. Generating a point cloud with certain semantic labels remains an under-explored problem. This paper proposes a formulation called DiffusionPointLabel, which completes point-label pair generation based on a DDPM generative model (Denoising Diffusion Probabilistic Model). Specifically, we use a point cloud diffusion generative model and aggregate the intermediate features of the generator. On top of this, we propose Feature Interpreter that transforms intermediate features into semantic labels. Furthermore, we employ an uncertainty measure to filter unqualified point-label pairs for a better quality of generated point cloud dataset. Coupling these two designs enables us to automatically generate annotated point clouds, especially when supervised point-labels pairs are scarce. Our method extends the application of point cloud generation models and surpasses state-of-the-art models.

CCS Concepts

• ***Methods and Applications*** → *Point-Based Methods;*

1. Introduction

In recent years, Deep Neural Networks have dominated point cloud processing and understanding tasks such as object detection [YZK21], robot manipulation [YKH*19], depth estimation [WCG*19], and semantic segmentation [JST*21]. Though substantial progress has been made, point cloud applications based on modern deep learning suffer from practical limitations. Networks require large amounts of annotated data to sufficiently optimize all parameters.

Unfortunately, creating point cloud datasets with point labels such as semantic or instance segmentation is labour-intensive and expensive. This is because labelling a complex shape usually involves the help of a human auxiliary to rotate and look through different angles to identify an object from incomplete or occluded point cloud data. In this scenario, creating a point cloud dataset of the scale we desire is still a challenge [QYW*19].

One solution is to build generative models, [MWYG20, GBZCO21, YWZJ21, SPK19], to synthesize expressive point clouds while having control of the structure. Point clouds and point-wise semantic labels are bred from key structural points in these methods. However, due to the irregular distribution and high complexity of 3D point clouds, existing generative models often struggle with explicit structural controllability and producing realistic-looking shapes. Our approach goes beyond existing solutions in terms of explicit point-label pairs generation as it generates point-wise labels without affecting the shape generation results because the semantic information is obtained from the intermediate features of the generator.

Denoising Diffusion Probabilistic Models (DDPM) are emerging as a new class of generative models and have achieved impressive performance on point cloud generation [LH21, LKX*21]. DDPM defines a consecutive point-wise mapping between two point clouds in the diffusion process and characterizes it as a Markov chain. Our approach is motivated by the observation that the generator of the diffusion model primarily recovers the coarse structure of the point cloud at the early stage of the diffusion process while gradually enriching details at the later stage. However, the output of the generator alongside the diffusion process should be a set of independent and identically distributed random variables. Based on this observation, we assume that the discriminability of point representations of the diffusion model varies along the diffusion process.

Driven by this assumption and inspiration from [BRV*21], we investigate the intermediate features of the point cloud diffusion generative model to figure out how the discriminability changes and whether it has semantically interpretable potential. On top of that, we aggregate the intermediate features of the diffusion generator and conduct a simple Multi Layer Perceptrion (MLP), called Fea-

† Corresponding author
‡ Corresponding author

ture Interpreter, to transform the intermediate features into point-wise semantic labels. Under this design, our paradigm enables point-wise label annotation without affecting the quality of point cloud generation. Although our approach can generate plausible annotated point clouds, it still generates some unqualified point-label pairs. Following the setup of [ZLG*21] to eliminate this error, we adopt an uncertainty measurement to estimate the quality of annotated point-label pairs. Specifically, we train a committee of Feature Interpreters and compute uncertainty scores for point-label pairs via the entropy of the committee. Then uncertainty scores can then be used to filter unqualified point clouds.

The main contributions of our work can be summarized as follows.

1. Different from previous point cloud generation methods that focus on structure-aware point cloud generation or breeding point labels from key structural points, we are the first to propose a point-label pairs generation framework based on the point cloud diffusion generative model, termed as DiffusionPointLabel. Our method can simultaneously generate point clouds and the corresponding point-wise semantic labels.

2. We experimentally exhibit that the intermediate features of the point cloud diffusion generative model are interpretable at the semantic level and have the potential to help 3D understanding.

3. Experimental results demonstrate that the proposed method has great advantages in efficiency and effectiveness in obtaining annotated point clouds, especially when the supervised labelled examples are scarce, which surpasses state-of-the-art performances. The code is publicly available at: code

2. Related Works

This section briefly describes the existing research lines relevant to our work.

2.1. Learning Representations of Point Clouds

Deep representation learning has been developed for many years. Since point cloud data has irregular structure, [MS15a, WSK*15, ROUG17, MS15b] quantized the 3D space and transformed points into regular voxels so that a convolutional neural network could process 3D data. However, since the 3D point cloud is a sparse and discretely distributed representation, convolution operators are inefficient and computationally expensive for 3D data.

Qi et al. [QSMG17] proposed PointNet, a notable landmark for point-based deep learning work, which works by leveraging weight-shared multi-layered perceptrons and a point-wise max-pooling layer to learn the features of the point cloud. The max-pooling layer can address the irregular structure of the point cloud, while it may neglect local information. Subsequent works have been proposed to tackle this issue. PointNet++ [QYSG17] defined a hierarchical architecture, which is effective for capturing local features of point sets in increasing contextual scale and improved semantic segmentation performance. Its design includes a deformable convolutional kernel to adapt to the local geometry and be robust to varying densities. Following them, [LCL18, ZJFJ19, WSL*19] adopted a wider neighbourhood to enhance local region features; [WQF19, TQD*19] designed a flexible kernel-based convolution operator and [WSL*19, QLJ*17, WHH*19] regarded point clouds as undirected graphs to group points to enrich latent features.

Recently, random walks [MBST21, XZS*21] have been used for 3D model representations and achieved state-of-the-art performance. Xiang et al. [XZS*21] proposed to use shape curves to analyze point cloud features, which are initialized based on a given set of rules and heuristics. Mesika and Ben-Shabat [MBST21] presented a technique that imposes structure on the point set by multiple random walks to aggregate point features.

With the recent success of applying Transformers [VSP*17] in vision tasks, many works [ZWL*21, ZJJ*21, GCL*21, HJCX21, YTR*21] have proposed delicately designed transformer networks for point clouds. These models focused on reducing the cost of point cloud annotation. A comprehensive point cloud dataset consists of both point clouds and their corresponding labels. In this paper, we propose a pipeline that can generate a point cloud dataset of this kind.

2.2. Generative Models for 3D Point Clouds

In the past few years, plenty of works have extended the generative model to point clouds. Current point cloud generative works can be generally classified into three categories: Autoregressive-based, flow-based, and GAN-based.

PointGrow [SWL*20] is one of the notable works of **Autoregressive-based** methods. It estimates the probability of samples autoregressively based on previously generated points. However, this method is restricted to generating a fixed-dimension point cloud because it assumes a determinate order of point cloud.

GAN-based generative models explore adversarial learning to train the shape generator with the help of a discriminator. Shu et al. [SPK19] combined tree-structure and graph to perform convolution on the point cloud. It demonstrated that tree-GAN could edit point clouds on the semantic level without prior knowledge, but the precision of the label falls short of expectations. Gal et al. [GBZCO21] extended it into multi-roots version. The node of the multiple roots can generate and control different parts of a point cloud. Nevertheless, there is no clear classification boundary between different parts, which indicates that the generated point clouds do not have clear semantic definitions. Wang et al. [YWZJ21] draw inspiration from S^2-GANs and proposed using enhanced controllability and point-level label accuracy. However, the label accuracy will inevitably be affected because the semantic label of a point is inherited by the structure points. Compared with the above GAN-based approaches, we incorporate a pre-trained generator and use its intermediate features to generate point-wise labels. It drastically improves the accuracy.

For **Flow-based** generative models [KLK*20, YHH*19, KBV20, HXX*20], the basic idea is to train an invertible parameterized transformation that can characterize the distribution of samples. This transformation can output a target shape by moving points from a prior distribution at one time.

Recently, Denoising Diffusion Probabilistic Models have shown

superior performance in terms of generative fidelity and diversity in 2D dataset generation [GRS*20]. In the 3D generation domain, Luo et al. [LH21] applied a diffusion model to point clouds and achieved competitive results compared to state-of-art. Zhou et al. [ZDW21] used conditional DDPM for point cloud completion by training a point-voxel CNN. Lyu et al. [LKX*21] applied a diffusion model to point clouds completion task. They proposed an adaption network architecture for point cloud and added a denoise module. The experimental results indicate that their method enhances the precision of results. However, existing point-based generative approaches mainly focus on 3D geometry while neglecting the implicit feature information, which is complementary to 3D understanding.

More recently, SetVAE [KYLH21] learned to generate set-structured data such as point clouds with a tree structure. However, its latent variable is not explicitly trained to segregate semantic parts.

3. Methodology

Figure 1 shows the structural details of our method. Given a random latent code $z \sim \mathcal{N}(0,I)$ and a random noise of point cloud $X \in \mathbb{R}^{N\times 3} \sim \mathcal{N}(0,I)$, we aim to generate a point cloud $X \in \mathbb{R}^{N\times 3}$ and its corresponding semantic labels $SL \in \mathbb{R}^{N\times b}$, where b denotes the number of semantic label category of the point cloud. To this end, firstly we extract the intermediate features of X in different layers of the diffusion generator θ_G at certain time steps $t = \{t_i | i = 1,..,T\}$. Formally, $C_{i,j} \in \mathbb{R}^{N\times C_{out_i}}$ denotes the intermediate feature of the i-th layer at time step $t = j$. We concatenate several $C_{i,j}$ as C^*. Finally, we use a Feature Interpreter to transform C^* into point-label pairs.

We revisit the point cloud diffusion generative model in Section 3.1. In Section 3.2, we analyze how the discriminability of intermediate features of the diffusion model changes along the diffusion process. We use K-means to verify our assumption that the intermediate features of the diffusion model are interpretable at the semantic level. Then in Section 3.3, we introduce the Feature Interpreter, which transforms the intermediate features of the diffusion generator into the point-wise label. In Section 3.4, we propose assembling a set of Feature Interpreters as a committee to compute the uncertainty score of annotated point clouds which can be used to filter unqualified point-label pairs.

3.1. Denoising Diffusion Probabilistic Models for Point Clouds

Denoising Diffusion Probabilistic Models regard the process of point cloud generation as a Markov chain. The parameterized Markov chain maps noise (i.e. 3D Gaussian) to shape by recursively perturbing the input point cloud. The forward diffusion process transforms shape into noise in an unconditional way. The reverse process generates the desired shape from Gaussian noise that is conditioned on a latent variable of global shape. Both processes have a fixed time step, denoted by T. Ho et al. [HJA20] utilized variational inference to solve the parameterized diffusion process.

The Diffusion Process. We use superscript to denote the diffusion step t. The forward diffusion process q transforms shape X^0 to noise $\mathrm{X}^T \sim \mathcal{N}(0,1)$. We assume $p_{data}(X^{(0)})$ to be the distribution of the point cloud X in the ground-truth dataset. Given N points in a point cloud $\mathrm{X} = \{x_i | i = 1,..,N\} \in \mathbb{R}^{N\times 3}$, the distribution of each point in the forward diffusion process can be formulated to:

$$\begin{aligned} q(x_i^{(1:T)}|x^{(0)}) &:= \prod_{t=1}^{T} q(x_i^{(t)}|x_i^{(t-1)}), \\ \text{where } q(x^{(t)}|x^{(t-1)}) &:= \mathcal{N}(x^{(t)}; \sqrt{1-\beta_t}x^{(t-1)}, \beta_t \mathbf{I}) \end{aligned} \tag{1}$$

where the hyperparameter β_t is a fixed monotonic increasing list [HJA20],

Note that in the forward diffusion process, the point cloud sample X^0 gradually loses geometric features as the time step t increases. Eventually, when $T \to \infty$, X^T is equivalent to an isotropic Gaussian distribution.

The Reverse Process. The reverse process p is used to predict a 3D shape from a latent code z. Conversely to the forward process, points are recursively moved from a prior noise distribution $p(x_i^{(T)})$ to approximate $q(x_i^{(0)})$. We use a network, θ, to estimate the denoise movement of every point at time step t. This process can be formulated as:

$$\begin{aligned} p_\theta(x^{(0:T)}|z) &:= p(x^T) \prod_{t=1}^{T} p_\theta(x^{(t-1)}|x^{(t)}, z) \\ p_\theta\left(x^{t-1} \mid x^t, z\right) &:= \mathcal{N}\left(x^{t-1}; \mu_\theta\left(x^t, t\right), \Sigma_\theta\left(x^t, t\right)\right) \end{aligned} \tag{2}$$

where the μ_θ and Σ_θ denotes the mean value and variance of the $x^{(t-1)}$.

Since the points in a point cloud are independently sampled from a distribution, the probability of the whole point cloud is simply the product of the probability of each point:

$$\begin{aligned} q(\mathbf{X}^{(1:T)}|\mathbf{X}^0) &= \prod_{i=1}^{N} q(x_i^{(1:T)}|x_i^{(0)}) \\ p_\theta(\mathbf{X}^{(0:T)}|z) &= \prod_{i=1}^{N} p_\theta(x_i^{(0:T)}|z) \end{aligned} \tag{3}$$

Training. Training objective can be simplified by optimizing the variational bound on negative log-likelihood:

$$\begin{aligned} \mathcal{L} = &\mathbb{E}_q[\sum_{t=2}^{T} D_{KL}(q(\mathbf{X}^{(t-1)}|\mathbf{X}^{(t)}, \mathbb{X}^{(0)}) \| p_\theta(\mathbf{X}^{(t-1)}|\mathbf{X}^{(t)}, z)) \\ &- \log p_\theta(\mathbf{X}^{(0)}|\mathbf{X}^{(1)}, z) + D_{KL}(q_\phi(z|\mathbf{X}^{(0)}) \| p(z))] \end{aligned} \tag{4}$$

Ho et al. [HJA20] showed that the training objective of the diffusion model θ can be simplified in a closed-form by a parameterization trick. Let $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_1^t \alpha_i$, then the training objective becomes:

$$\mathcal{L}(\theta) := \mathbb{E}_{t, x^{(0)}, \varepsilon} \| \varepsilon - \varepsilon_\theta(\sqrt{\bar{\alpha}^{(t)}} + \sqrt{(1-\bar{\alpha}^{(t)})}\varepsilon, t) \|^2 \tag{5}$$

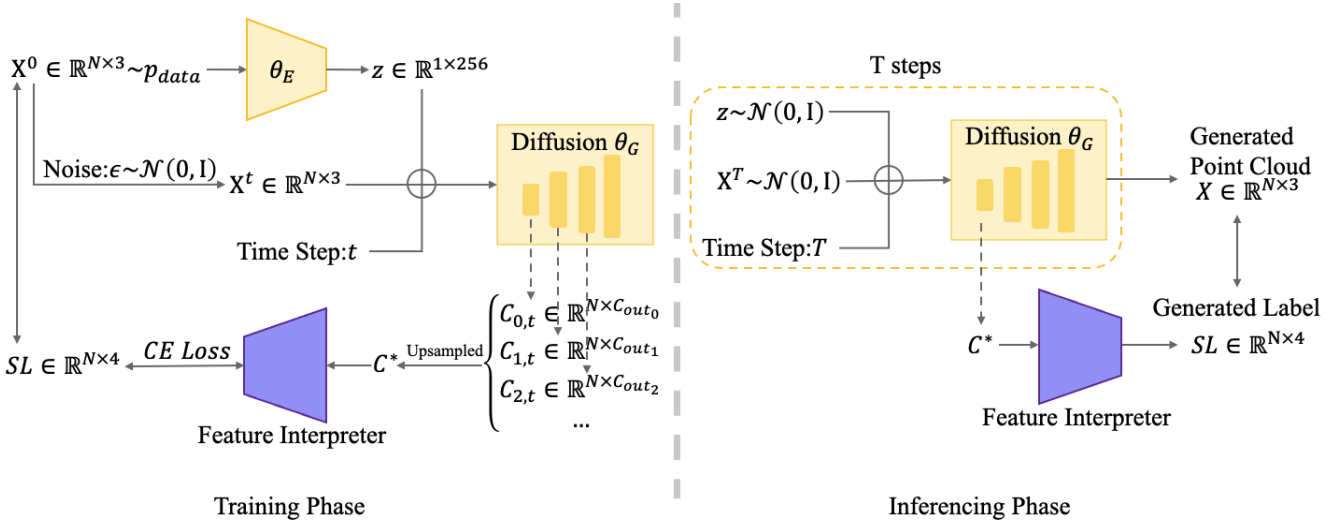

Figure 1: *Illustration of the point-label pairs generation method.*

where t is uniformly distributed between 1 and T, $\varepsilon \sim \mathcal{N}(0,I)$, ε_θ is a point cloud diffusion generative network that predicts injected noise ε at each time step. DDPM does not require CD or EMD loss during the training phase because it defines a consecutive and invertible point-wise mapping. Note that the network is non-autoregressive, so the predecessor only determines its prediction.

3.2. Intermediate Feature Analysis

The point cloud generation process based on the diffusion generative model is shown in the top row of Figure 2. As shown in Equation 5, the output of the diffusion generator at each step should be independent and identically distributed random variables. However, in the early and later stages of the diffusion process, the change tendency of the point cloud is different: coarse structure in the former and fine details in the latter. Therefore, we assume that the point representation of the diffusion generator has different discriminability alongside the diffusion process.

To prove our assumption, we use the K-means cluster to analyze the intermediate features of the diffusion generator at different time steps. In practice, we freeze a diffusion model θ and take a point cloud $X \in \mathbb{R}^{N\times3}$ and a time step t as the inputs of θ. Then we extract the intermediate features of one layer of the generator θ_G. Because the generator we used is an MLP, the intermediate features of every layer at each time step t can be denoted by $C_{i,t} \in \mathbb{R}^{N\times out_i}$. We use the K-means clustering algorithm to estimate the cluster label of each point from the intermediate features of X and visualize the results, as shown in the bottom row of Figure 2. In K-means clustering, we use the number of ground truth labels as the cluster number. For example, the airplane in Figure 2 has 4 semantic parts, and then we set the K-means cluster number to be 4. The results of K-means demonstrate that the discriminability of intermediate layer features gradually increases with decreasing time steps and is interpretable at the semantic level.

To further determine which layer of features we should extract or at which time steps, we quantitatively compute the K-means clustering results. If the K-means clustering effect is good (the cluster labels of the close points are very similar, and the cluster labels of the distant points are not the same), it means that the discriminability of this intermediate feature is very high. Otherwise, it will remain low and cannot be used for further learning. We extract and cluster the intermediate features of each layer of θ_G from time $t = 0$ to T. We compute the clustering results with the Calinski-Harabasz Index algorithm [CH74], and the result is shown in Figure 3. The Calinski-Harabasz Index algorithm is used to measure the quality of the cluster model without the ground-truth label. The Calinski-Harabaz score is defined as the ratio of separation and cohesion of clusters. The formulation is shown as:

$$s = \frac{SS_B}{k-1} \Big/ \frac{SS_W}{N-k} \tag{6}$$

where k denotes the number of clusters, N denotes the number of all data, SS_B denotes the variance between different clusters, and SS_W denotes the variance between one cluster. The higher the score, the better the clustering effect of K-means.

As shown in Figure 3, the Calinski-Harabasz Index score starts converging to a high value when $t = T/4$, which means the features are more discriminative and can be well clustered. The experiments utilize the intermediate features at $t < T/4$. It is worth mentioning that this score becomes stable at $t \to T$. The attribution of it is that the discriminability of intermediate features tends to be consistent at this time, and the point features represent the object's category or overall shape information.

3.3. Feature Interpreter

The Feature Interpreter takes the intermediate features as the input, aiming to generate explicit semantic labels for the generated point

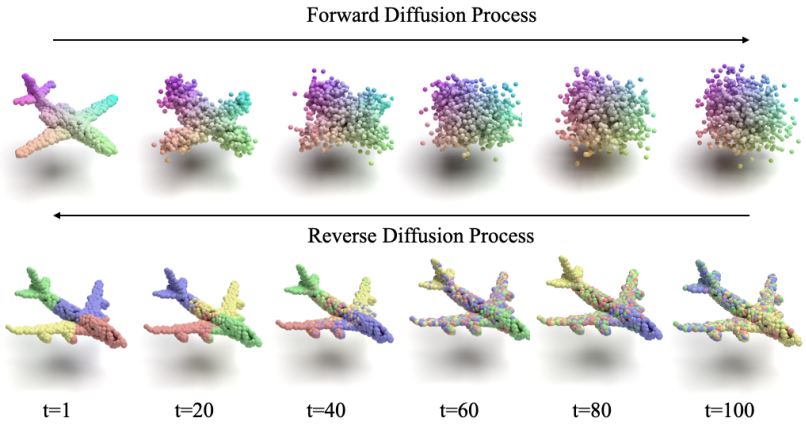

Figure 2: *Visualization of the diffusion process and corresponding K-means cluster of intermediate features of the point cloud generative model. The top row represents the states of the point cloud in the diffusion process in the time variable. The bottom row represents the results of the corresponding K-means features of the intermediate features.*

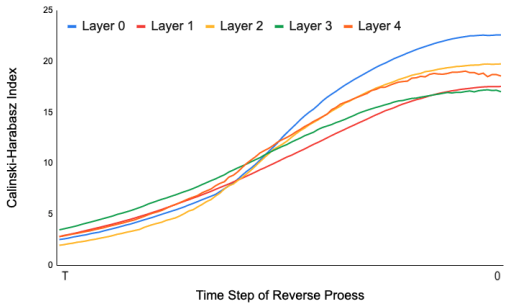

Figure 3: *Calinski-Harabasz Index of intermediate features clustering results. The score represents the quality of the cluster. Different colors represent different layers of the* θ_G

cloud. An MLP is implemented to realize the label prediction. Similar to the K-means clustering process, we freeze a diffusion model θ and take a point cloud $X \in \mathbb{R}^{N\times 3}$ and a time step t as the inputs of θ. Based on the analysis in section 3.1, we sample $C_{0,t}$ across different time steps, where $t < T/4$. Then the intermediate features $C_{0,t}$ of the θ_G are upsampled via linear interpolation and concatenated to form $C^* \in \mathbb{R}^{N\times 1024}$. In practice, we use three-layer MLPs to predict the semantic label for each point from the C^*. The Feature Interpreter is optimized by cross-entropy loss.

3.4. Uncertainty Measurement

We found that the point cloud generative models occasionally generate meaningless point clouds in experiments. Since we do not want to involve human labour in this task, we need to filter these point clouds before collecting the final dataset. Following [ZLG*21, GRS*20], we adopted the Jensen-Shannon (JS) divergence [KHY*18] to compute the uncertainty measure for each point-label pair. Specifically, we train a committee of Feature Interpreters in the same way. And then, we estimate the label likelihood $LS \in \mathbb{R}^{N\times 4}$ for the point cloud $X \in \mathbb{R}^{N\times 3}$. Formally, the uncertainty measurement is denoted by $\mathcal{JS} \in \mathbb{R}^N$. The computation can be formulated as:

$$\mathcal{JS} = H(\frac{1}{M}\sum_{i}^{M} LS_i) - \frac{1}{M}H(LS_i) \tag{7}$$

where M denotes the number of Feature Interpreters in one committee; LS_i denotes the label likelihood of the i-th Feature Interpreter for point cloud; H denotes the entropy function. We use the score of the lowest-rated 200 points of point cloud $\mathcal{JS}$ as the uncertainty score for each point cloud in the implementation. The uncertainty score can be used to filter unqualified point clouds.

4. Evaluation and Discussion

In section 4.1, we evaluate the effectiveness of our method in two aspects: a) the validity of our generated dataset; b) the effectiveness of the Feature Interpreter. In section 4.2, We discuss the discriminability of intermediate features of three popular point cloud autoencoder networks. In section 4.3, we compare our method with a close point cloud generative model CPCGAN [YWZJ21]. We conduct an ablation study in section 4.4.

Dataset and Implementation Details. We evaluate the proposed method on the ShapeNet-Partseg dataset [FSG17]. This dataset includes 16881 shapes from 16 object categories. In our evaluation, we mainly work with the "chair", "airplane", "guitar", "table", "lamp", "car" and "bag" categories.

Our method was conducted on a pre-trained point cloud diffusion generative model. We used Luo et al. [LH21] as the backbone in practice and trained the model θ according to their specifications. We trained the Feature Interpreter for 200 epochs with batch size 128, starting with a learning rate of 0.02, which decayed by 0.5 every 10 epochs. The training process is optimized by Cross-entropy loss.

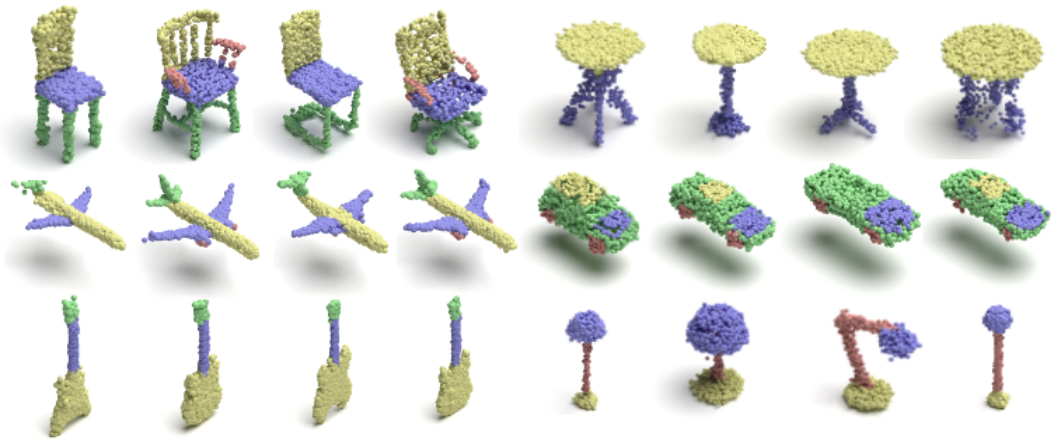

Figure 4: *Examples of generated point-label pairs.*

Evaluation Metrics and Baselines. 3D semantic segmentation is evaluated using mean Intersection over Union on point and accuracy, referred to as mIoU and mAcc respectively. We include PointNet++ [QYSG17] as the baseline that evaluates the validity of the generated dataset and the effectiveness of our method.

4.1. Validation of Generated Datasets and Feature Interpreter

In this section, we evaluate our method in two settings.

The validity of the generated dataset: We first train a network [QYSG17] for semantic segmentation using a training set of ground truth data. After training, we use this network to validate our generated dataset and the ground truth test set, respectively. When producing our dataset, we generated 10,000 point clouds for each category and filtered samples based on their uncertainty scores. Figure 5 shows the quantitative comparison of the generated dataset produced by our method. Some examples from our generated dataset are visualized in Figure 4. Our generated dataset show competitive results for most categories compared to GT dataset. The performance of our generated dataset is much lower than the GT dataset in the category of Lamp. However, the visualized results of the Lamps are plausible. We attribute this result to the fact that because the Lamps in the GT dataset are few, the segmentation network has not fully learned the accurate features of the Lamps. *Further experiment of application of the generated dataset in a few-shot scenario and more visualized examples can be seen in supplementary materials.*

The effectiveness of the Feature Interpreter: We believe that one of the future application scenarios of our method is to generate point cloud datasets for a new category. Since the cost of point cloud annotating is too high, we can use few-shot examples of point-label pairs and generate large-scale annotated point cloud datasets. Therefore, it is important to verify whether our method can generate results with high segmentation accuracy when the example samples are scarce. We conduct few-shot segmentation to verify the effectiveness of our method. We compare the evaluation

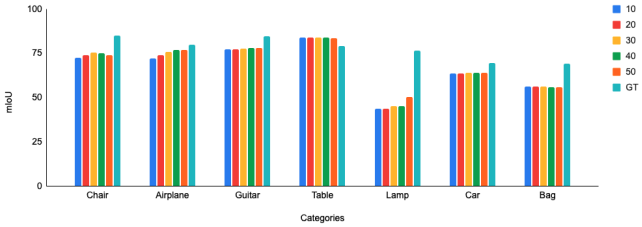

Figure 5: *Comparison between our generated datasets and ground-truth ones. Different colors denote different filter ratios.*

results with baseline [QYSG17] as shown in Table 1. Our method demonstrates comparable performance to baseline when trained on a few samples. The comparison results demonstrate that our method is capable of generating compelling point-label pairs in a few-shot setting. In this experiment, we set the train epoch as 20, the learning rate starts from 0.001, which decayed by 0.1 every 2 epochs.

4.2. Validation of Representation Effectiveness

Since the intermediate features of our analysis and learning are extracted from a diffusion generative network with an autoencoder frame, we naturally question whether these learnable intermediate features have nothing to do with the diffusion process but only benefit from the autoencoder framework. Therefore, we conduct an experiment to find out whether other methods capable of extracting intermediate features from point clouds can achieve the same effect. To the best of our knowledge, this is the first work to find out the discriminability of intermediate features in a point cloud generative model.

As in our method, we first collect and cluster latent feature spaces of existing generative models: CapsNetwork [ZBDT19], PointFlow [YHH*19], and FoldingNet [YFST18]. The cluster effect is shown in Figure 6.

Category	Model	k=1	k=3	k=5	k=10	k=16	k=32
Airplane	Baseline	20.9	47.2	29.4	43.3	59.6	64.6
	Ours	**58.1**	**62.8**	**63.9**	**64.8**	**66.0**	**67.2**
Chair	Baseline	33.9	63.8	50.0	64.8	**79.5**	**81.6**
	Ours	**66.2**	**67.9**	**72.1**	**74.7**	77.6	78.2

Table 1: *Few-shot segmentation on the ground truth dataset. k demonstrates the number of samples that be used in Training.*

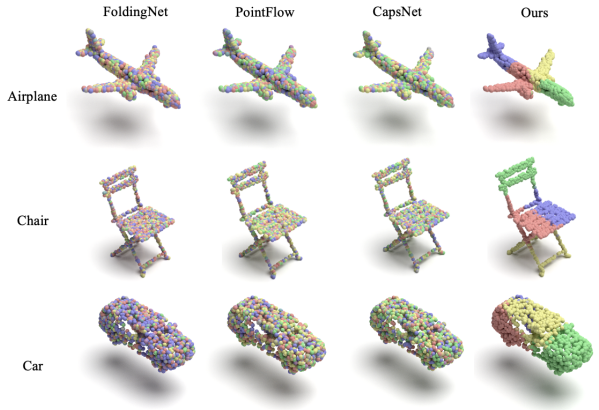

Figure 6: *Qualitative comparison of intermediate features based on a different baseline using K-means cluster.*

The comparison results answer our question: the discriminability of the intermediate features benefited from the diffusion process, and not all point cloud autoencoder networks have similar discriminability. The possible explanations could be that (a) these model use CD-Loss to optimize the parameter of the network, which calculates the overall structural similarity; (b) these models train the network in a one-shot discriminative way. Therefore, their intermediate features do not contain fine-grained information.

4.3. Comparing with Related Methods

The work most closely related to ours is CPCGAN [YWZJ21]. CPCGAN [YWZJ21] proposes a two-stage GAN to generate point clouds in a controllable structure manner and is trained on the ShapeNet-Partseg dataset as well. The first stage generator generates the key structural points and corresponding labels. The second stage generator generates the point cloud by expanding the key structural points into a complete point cloud. The semantic labels of the final point cloud are bred from the key structural points of the first stage. Because it is hard to annotate ground truth labels for generated point clouds, following their experimental setting, we train a PointNet++ model [QYSG17] for the semantic segmentation task. We use this pre-trained segmentation network to evaluate the generated point-label pairs. The quantitative comparison is shown in Table 2. From the results shown in the table, we can see that our generated point cloud (airplane and chair) outperforms their method consistently on the two evaluation metrics for both mIoU and mAcc by a large margin.

Moreover, we visualized comparison results as shown in Figure 7. Here, we pick some random point clouds generated by both methods and categorize the semantic labels by color, using the same color for the same label across models. From these results, we can see that the point clouds with semantic labels generated by our method exhibit more accurate labels, whereas [YWZJ21] tends to generate noisy semantic labels.

Class	Model	mIoU%(↑)	mAcc%(↑)
Chair	CPCGAN	57.1	83.6
	Ours	**72.0**	**86.3**
Airplane	CPCGAN	67.8	82.6
	Ours	**74.2**	**89.2**

Table 2: *Comparison of point cloud and label generation performance. mIoU and mACC are multiplied by* 10^2

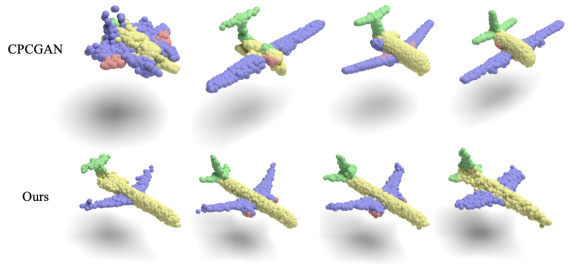

Figure 7: *Visualized results of [YWZJ21] and ours.*

4.4. Ablation Study

Intuitively, there are two deterministic factors of representation discriminability. The first is that intermediate features with the highest dimension have better discriminability because they may contain the most information. The second is that we tend to consider the features of the shallow layer because the feature of the deeper layer is closer to the estimated noise of the diffusion process, while the shallows contain abstract information, such as semantics.

Then we compare it to the following settings: a) the features of the shallow layers are upsampled to the highest dimension; b) the features of the layer that has the highest dimension.

The results are proved in Table 3. The intermediate features within the highest dimension slightly underperform than features of the shallow layers. The experimental results confirmed that the intermediate features of the shallow layer have better discriminability.

	$C_{(0,0)}$	Upsample $C_{(0,0)}$	$C_{(2,0)}$
Airplane	75.7	76.0	72.8

Table 3: *Evaluation of the different intermediate feature extraction variations for part segmentation.*

5. Conclusions

To conclude, this paper presents a paradigm called DiffusionPoint-Label, a simple and useful method for the generation of point-label pairs. A Feature Interpreter is applied to transform intermediate features of the point cloud diffusion generative model into the semantic label. Uncertainty measurement is introduced to enhance the quality of the generated point cloud dataset. We further show the effectiveness and efficiency of our method within scarce supervised labelled examples. In the future, we plan to explore more of the potential power of the point cloud generative model, such as fine-grain point-label pairs generation.

Acknowledgment The research was partially supported by the matched funded PhD scholarship of Bournemouth University, UK and Zhengzhou University of Light Industry, China. We thank our colleagues Ben Snow for his proofreading and comments to improve the manuscript.

References

[BRV*21] BARANCHUK D., RUBACHEV I., VOYNOV A., KHRULKOV V., BABENKO A.: Label-efficient semantic segmentation with diffusion models. *arXiv preprint arXiv:2112.03126* (2021).

[CH74] CALIŃSKI T., HARABASZ J.: A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods 3*, 1 (1974), 1–27.

[FSG17] FAN H., SU H., GUIBAS L. J.: A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 605–613.

[GBZCO21] GAL R., BERMANO A., ZHANG H., COHEN-OR D.: Mrgan: Multi-rooted 3d shape representation learning with unsupervised part disentanglement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 2039–2048.

[GCL*21] GUO M.-H., CAI J.-X., LIU Z.-N., MU T.-J., MARTIN R. R., HU S.-M.: Pct: Point cloud transformer. *Computational Visual Media 7*, 2 (2021), 187–199.

[GRS*20] GADELHA M., ROYCHOWDHURY A., SHARMA G., KALOGERAKIS E., CAO L., LEARNED-MILLER E., WANG R., MAJI S.: Label-efficient learning on point clouds using approximate convex decompositions. In *European Conference on Computer Vision* (2020), Springer, pp. 473–491.

[HJA20] HO J., JAIN A., ABBEEL P.: Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems 33* (2020), 6840–6851.

[HJCX21] HAN X.-F., JIN Y.-F., CHENG H.-X., XIAO G.-Q.: Dual transformer for point cloud analysis. *arXiv preprint arXiv:2104.13044* (2021).

[HXX*20] HUI L., XU R., XIE J., QIAN J., YANG J.: Progressive point cloud deconvolution generation network. In *European Conference on Computer Vision* (2020), Springer, pp. 397–413.

[JST*21] JIANG L., SHI S., TIAN Z., LAI X., LIU S., FU C.-W., JIA J.: Guided point contrastive learning for semi-supervised point cloud semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 6423–6432.

[KBV20] KLOKOV R., BOYER E., VERBEEK J.: Discrete point flow networks for efficient point cloud generation. In *European Conference on Computer Vision* (2020), Springer, pp. 694–710.

[KHY*18] KUO W., HÄNE C., YUH E., MUKHERJEE P., MALIK J.: Cost-sensitive active learning for intracranial hemorrhage detection. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2018), Springer, pp. 715–723.

[KLK*20] KIM H., LEE H., KANG W. H., LEE J. Y., KIM N. S.: Softflow: Probabilistic framework for normalizing flow on manifolds. *Advances in Neural Information Processing Systems 33* (2020), 16388–16397.

[KYLH21] KIM J., YOO J., LEE J., HONG S.: Setvae: Learning hierarchical composition for generative modeling of set-structured data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 15059–15068.

[LCL18] LI J., CHEN B. M., LEE G. H.: So-net: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 9397–9406.

[LH21] LUO S., HU W.: Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 2837–2845.

[LKX*21] LYU Z., KONG Z., XU X., PAN L., LIN D.: A conditional point diffusion-refinement paradigm for 3d point cloud completion. *arXiv preprint arXiv:2112.03530* (2021).

[MBST21] MESIKA A., BEN-SHABAT Y., TAL A.: Cloudwalker: 3d point cloud learning by random walks for shape analysis. *arXiv preprint arXiv:2112.01050* (2021).

[MS15a] MATURANA D., SCHERER S.: 3d convolutional neural networks for landing zone detection from lidar. In *2015 IEEE international conference on robotics and automation (ICRA)* (2015), IEEE, pp. 3471–3478.

[MS15b] MATURANA D., SCHERER S.: Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2015), IEEE, pp. 922–928.

[MWYG20] MO K., WANG H., YAN X., GUIBAS L.: Pt2pc: Learning to generate 3d point cloud shapes from part tree conditions. In *European Conference on Computer Vision* (2020), Springer, pp. 683–701.

[QLJ*17] QI X., LIAO R., JIA J., FIDLER S., URTASUN R.: 3d graph neural networks for rgbd semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 5199–5208.

[QSMG17] QI C. R., SU H., MO K., GUIBAS L. J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 652–660.

[QYSG17] QI C. R., YI L., SU H., GUIBAS L. J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems 30* (2017).

[QYW*19] QIN C., YOU H., WANG L., KUO C.-C. J., FU Y.: Pointdan: A multi-scale 3d domain adaption network for point cloud representation. *Advances in Neural Information Processing Systems 32* (2019).

[ROUG17] RIEGLER G., OSMAN ULUSOY A., GEIGER A.: Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 3577–3586.

[SPK19] SHU D. W., PARK S. W., KWON J.: 3d point cloud generative adversarial network based on tree structured graph convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 3859–3868.

[SWL*20] SUN Y., WANG Y., LIU Z., SIEGEL J., SARMA S.: Pointgrow: Autoregressively learned point cloud generation with self-attention. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (2020), pp. 61–70.

[TQD*19] THOMAS H., QI C. R., DESCHAUD J.-E., MARCOTEGUI B., GOULETTE F., GUIBAS L. J.: Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision* (2019), pp. 6411–6420.

[VSP*17] VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER Ł., POLOSUKHIN I.: Attention is all you need. *Advances in neural information processing systems 30* (2017).

[WCG*19] WANG Y., CHAO W.-L., GARG D., HARIHARAN B., CAMPBELL M., WEINBERGER K. Q.: Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 8445–8453.

[WHH*19] WANG L., HUANG Y., HOU Y., ZHANG S., SHAN J.: Graph attention convolution for point cloud semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 10296–10305.

[WQF19] WU W., QI Z., FUXIN L.: Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 9621–9630.

[WSK*15] 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* (2015), pp. 1912–1920.

[WSL*19] WANG Y., SUN Y., LIU Z., SARMA S. E., BRONSTEIN M. M., SOLOMON J. M.: Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog) 38*, 5 (2019), 1–12.

[XZS*21] XIANG T., ZHANG C., SONG Y., YU J., CAI W.: Walk in the cloud: Learning curves for point clouds shape analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 915–924.

[YFST18] YANG Y., FENG C., SHEN Y., TIAN D.: Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 206–215.

[YHH*19] YANG G., HUANG X., HAO Z., LIU M.-Y., BELONGIE S., HARIHARAN B.: Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 4541–4550.

[YKH*19] YAN X., KHANSARI M., HSU J., GONG Y., BAI Y., PIRK S., LEE H.: Data-efficient learning for sim-to-real robotic grasping using deep point cloud prediction networks. *arXiv preprint arXiv:1906.08989* (2019).

[YTR*21] YU X., TANG L., RAO Y., HUANG T., ZHOU J., LU J.: Point-bert: Pre-training 3d point cloud transformers with masked point modeling. *arXiv preprint arXiv:2111.14819* (2021).

[YWZJ21] YANG X., WU Y., ZHANG K., JIN C.: Cpcgan: A controllable 3d point cloud generative adversarial network with semantic label generating. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2021), vol. 35, pp. 3154–3162.

[YZK21] YIN T., ZHOU X., KRÄHENBÜHL P.: Multimodal virtual point 3d detection. *Advances in Neural Information Processing Systems 34* (2021).

[ZBDT19] ZHAO Y., BIRDAL T., DENG H., TOMBARI F.: 3d point capsule networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 1009–1018.

[ZDW21] ZHOU L., DU Y., WU J.: 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 5826–5835.

[ZJFJ19] ZHAO H., JIANG L., FU C.-W., JIA J.: Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 5565–5573.

[ZJJ*21] ZHAO H., JIANG L., JIA J., TORR P. H., KOLTUN V.: Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 16259–16268.

[ZLG*21] ZHANG Y., LING H., GAO J., YIN K., LAFLECHE J.-F., BARRIUSO A., TORRALBA A., FIDLER S.: Datasetgan: Efficient labeled data factory with minimal human effort. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 10145–10155.

[ZWL*21] ZHANG C., WAN H., LIU S., SHEN X., WU Z.: Pvt: Point-voxel transformer for 3d deep learning. *arXiv preprint arXiv:2108.06076* (2021).