POINTGROW: AUTOREGRESSIVELY LEARNED POINT CLOUD GENERATION WITH SELF-ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

A point cloud is an agile 3D representation, efficiently modeling an object's surface geometry. However, these surface-centric properties also pose challenges on designing tools to recognize and synthesize point clouds. This work presents a novel autoregressive model, *PointGrow*, which generates realistic point cloud samples from scratch or conditioned from given semantic contexts. Our model operates recurrently, with each point sampled according to a conditional distribution given its previously-generated points. Since point cloud object shapes are typically encoded by long-range interpoint dependencies, we augment our model with dedicated self-attention modules to capture these relations. Extensive evaluation demonstrates that *PointGrow* achieves satisfying performance on both unconditional and conditional point cloud generation tasks, with respect to fidelity, diversity and semantic preservation. Further, *conditional PointGrow* learns a smooth manifold of given images where 3D shape interpolation and arithmetic calculation can be performed inside.

1 INTRODUCTION

3D visual understanding (Bogo et al. (2016); Zuffi et al. (2018)) is at the core of next-generation vision systems. Specifically, point clouds, agile 3D representations, have emerged as indispensable sensory data in applications including indoor navigation (Díaz Vilariño et al. (2016)), immersive technology (Stets et al. (2017); Sun et al. (2018a)) and autonomous driving (Yue et al. (2018)). There is growing interest in integrating deep learning into point cloud processing (Klokov & Lempitsky (2017); Lin et al. (2017a); Achlioptas et al. (2017b); Kurenkov et al. (2018); Yu et al. (2018); Xie et al. (2018)). With the expressive power brought by modern deep models, unprecedented accuracy has been achieved on high-level point cloud related tasks including classification, detection and segmentation (Qi et al. (2017a;b); Wang et al. (2018); Li et al. (2018); You et al. (2018)). Yet, existing point cloud research focuses primarily on developing effective discriminative models (Xie et al. (2018); Shen et al. (2018)), rather than generative models.

This paper investigates the synthesis and processing of point clouds, presenting a novel generative model called *PointGrow*. We propose an autoregressive architecture (Oord et al. (2016); Van Den Oord et al. (2016)) to accommodate the surface-centric nature of point clouds, generating every single point recurrently. Within each step, *PointGrow* estimates a conditional distribution of the point under consideration given all its preceding points, as illustrated in Figure 1. This approach easily handles the irregularity of point clouds, and encodes diverse local structures relative to point distance-based methods (Fan et al. (2017); Achlioptas et al. (2017b)).

However, to generate realistic point cloud samples, we also need long-range part configurations to be plausible. We therefore introduce two self-attention modules (Lin et al. (2017b); Velickovic et al. (2017); Zhang et al. (2018)) in the context of point cloud to capture these long-range relations. Each dedicated self-attention module learns to dynamically aggregate long-range information during the point generation process. In addition, our *conditional PointGrow* learns a smooth manifold of given images where interpolation and arithmetic calculation can be performed on image embeddings.

Compared to prior art, *PointGrow* has appealing properties:

- Unlike traditional 3D generative models that rely on local regularities on grids (Wu et al. (2016); Choy et al. (2016); Yang et al. (2017); Sun et al. (2018b)), *PointGrow* builds upon

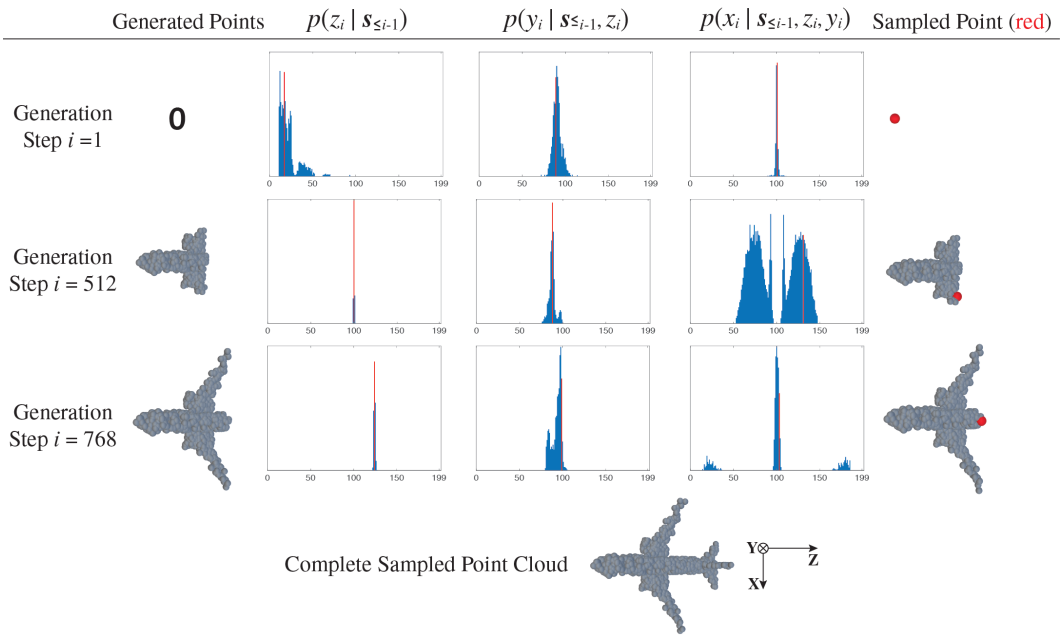

Figure 1: The point cloud generation process in *PointGrow* (best viewed in color). Given $i-1$ generated points, our model first estimates a conditional distribution of z_i, indicated as $p(z_i|\mathbf{s}_{\leq i-1})$, and then samples a value (indicated as a red bar) according to it. The process is repeated to sample y_i and x_i with previously sampled coordinates as additional conditions. The i^{th} point (red point in the last column) is obtained as $\{x_i, y_i, z_i\}$. When $i = 1$, $p(z_i|\mathbf{s}_{\leq i-1}) = p(z_i)$.

autoregressive architecture that is inherently suitable for modeling point clouds, which are irregular and surface-centric.

- Our proposed self-attention module successfully captures the long-range dependencies between points, helping to generate plausible part configurations within 3D objects.
- *PointGrow*, as a generative model, enables effective unsupervised feature learning, which is useful for recognition tasks, especially in the low-data regime.

Extensive evaluations demonstrate that *PointGrow* can generate realistic and diverse point cloud samples with high resolution, on both unconditional and conditional point cloud generation tasks.

2 POINTGROW

In this section, we introduce the formulation and implementation of *PointGrow*, a new generative model for point cloud, which generates 3D shapes in a point-by-point manner.

Unconditional PointGrow. A point cloud, $\mathbf{S}$, that consists of n points is defined as $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n\}$, and the i^{th} point is expressed as $\mathbf{s}_i = \{x_i, y_i, z_i\}$ in 3D space. Our goal is to assign a probability $p(\mathbf{S})$ to each point cloud. We do so by factorizing the joint probability of $\mathbf{S}$ as a product of conditional probabilities over all its points:

$$p(\mathbf{S}) = \prod_{i=1}^{n} p(\mathbf{s}_i|\mathbf{s}_1, ..., \mathbf{s}_{i-1}) = \prod_{i=1}^{n} p(\mathbf{s}_i|\mathbf{s}_{\leq i-1}) \tag{1}$$

The value $p(\mathbf{s}_i|\mathbf{s}_{\leq i-1})$ is the probability of the i^{th} point $\mathbf{s}_i$ given all its previous points, and can be computed as the joint probability over its coordinates:

$$p(\mathbf{s}_i|\mathbf{s}_{\leq i-1}) = p(z_i|\mathbf{s}_{\leq i-1}) \cdot p(y_i|\mathbf{s}_{\leq i-1}, z_i) \cdot p(x_i|\mathbf{s}_{\leq i-1}, z_i, y_i) \tag{2}$$

, where each coordinate is conditioned on all the previously generated coordinates. To facilitate the point cloud generation process, we sort points in the order of z, y and x, which forces a shape to be generated in a "plane-sweep" manner along its primary axis (z axis). Following Oord et al. (2016) and Van Den Oord et al. (2016), we model the conditional probability distribution of each coordinate using a deep neural network. Prior art shows that a softmax discrete distribution works

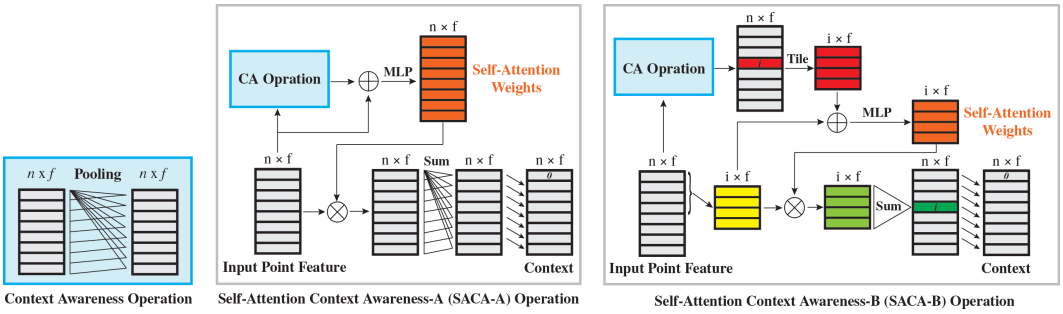

Figure 2: Context related operations. $\otimes$: element-wise production, $\oplus$: concatenation.

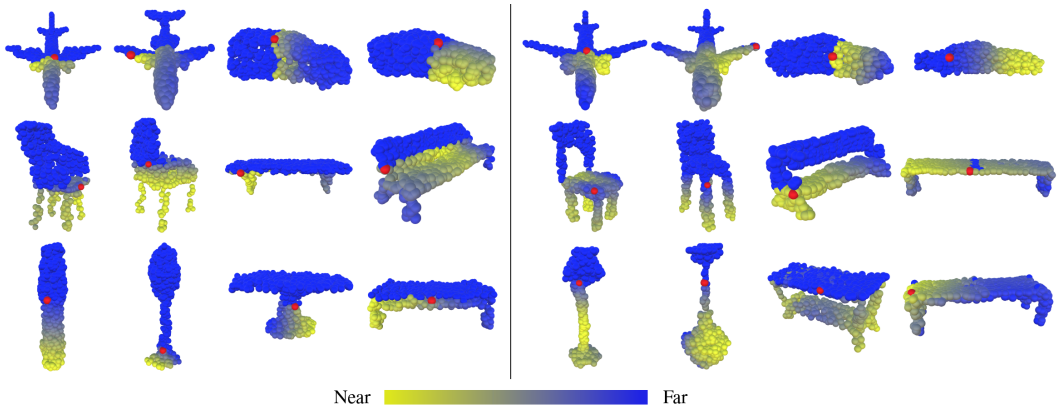

Figure 3: The self-attention fields in *PointGrow* for query locations (red points), visualized as Euclidean distances between the context feature of a query point and the features of its accessible points. The distances of inaccessible points (generated after the query point) are considered to be infinitely far. **Left:** SACA-A; **Right:** SACA-B.

better than mixture models, even though the data are implicitly continuous. To obtain discrete point coordinates, we scale all point clouds to fall within the range [0, 1], and quantize their coordinates to uniformly distributed values. We use 200 values as a trade-off between generative performance and minimizing quantization artifacts. Other advantages of adopting discrete coordinates include (1) simplified implementation, (2) improved flexibility to approximate any arbitrary distribution, and (3) it prevent generating distribution mass outside of the range, which is common for continuous cases.

Context Awareness Operation. Context awareness improves model inference. For example, in Qi et al. (2017a) and Wang et al. (2018), a global feature is obtained by applying max pooling along each feature dimension, and then used to provide context information for solving semantic segmentation tasks. Similarly, we obtain "semi-global" features for all sets of available points in the point cloud generation process, as illustrated in Figure 2 (left). Each row of the resultant features aggregates the context information of all the previously generated points dynamically by fetching and averaging. This Context Awareness (CA) operation is implemented as a plug-in module in our model, and mean pooling is used in our experiments.

Self-Attention Context Awareness Operation. The CA operation summarizes point features in a fixed way via pooling. For the same purpose, we propose two alternative learning-based operations to determine the weights for aggregating point features. We define them as Self-Attention Context Awareness (SACA) operations, and the weights as self-attention weights.

The first SACA operation, SACA-A, is shown in the middle of Figure 2. To generate self-attention weights, the SACA-A first associates local and "semi-global" information by concatenating input and "semi-global" features after CA operation, and then passes them to a Multi-Layer Perception (MLP). Formally, given a $n \times f$ point feature matrix, $\mathbf{F}$, with its i^{th} row, $\mathbf{f}_i$, representing the feature vector of the i^{th} point for $1 \leq i \leq n$, we compute the i^{th} self-attention weight vector, $\mathbf{w}_i$, as below:

$$\mathbf{w}_i = MLP(\underset{1 \leq j \leq i}{Mean}\{\mathbf{f}_j\} \oplus \mathbf{f}_i) \tag{3}$$

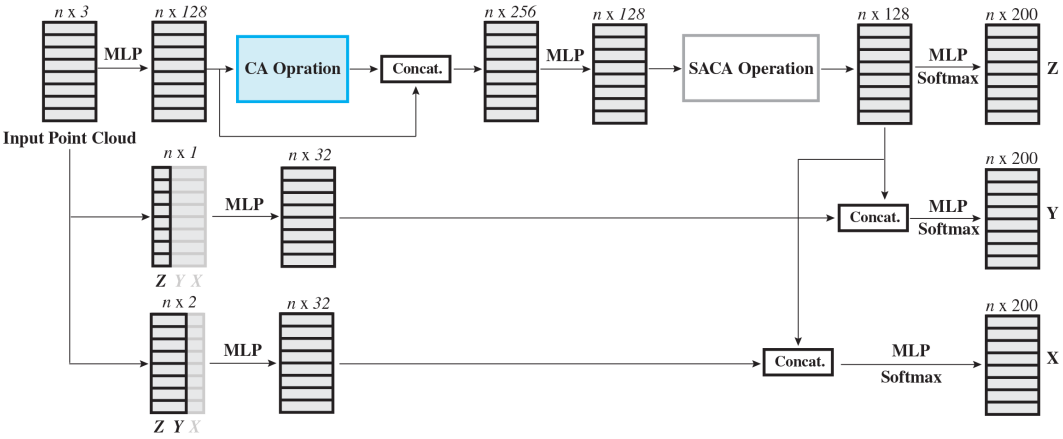

Figure 4: The proposed model architecture to estimate conditional distributions of point coordinates.

, where $Mean\{\cdot\}$ is mean pooling, $\oplus$ is concatenation, and $MLP(\cdot)$ is a sequence of fully connected layers. The self-attention weight encodes information about the context change due to each newly generated point, and is unique to that point. We then conduct element-wise multiplication between input point features and self-attention weights to obtain weighted features, which are accumulated sequentially to generate corresponding context features. The process to calculate the i^{th} context feature, $\mathbf{c}_i$, can be expressed as:

$$\mathbf{c}_i = \sum_{m=1}^{i} \mathbf{f}_m \otimes \mathbf{w}_m = \sum_{m=1}^{i} \mathbf{f}_m \otimes MLP(\underset{1 \le j \le \mathbf{m}}{Mean}\{\mathbf{f}_j\} \oplus \mathbf{f}_m) \tag{4}$$

, where $\otimes$ is element-wise multiplication. Finally, we shift context features downward by one row, because when estimating the coordinate distribution for point, $\mathbf{s}_i$, only its previous points, $\mathbf{s}_{\le i-1}$, are available. A zero vector of the same size is attached to the beginning as the initial context feature, since no previous point exists when computing features for the first point.

Figure 2 (right) shows the other SACA operation, SACA-B. SACA-B is similar to SACA-A, except the way to compute and apply self-attention weights. In SACA-B, the i^{th} "semi-global" feature after CA operation is shared by the first i point features to obtain self-attention weights, which are then used to compute $\mathbf{c}_i$. This process can be described mathematically as:

$$\mathbf{c}_i = \sum_{m=1}^{i} \mathbf{f}_m \otimes \mathbf{w}_m = \sum_{m=1}^{i} \mathbf{f}_m \otimes MLP(\underset{1 \le j \le \mathbf{i}}{Mean}\{\mathbf{f}_j\} \oplus \mathbf{f}_m) \tag{5}$$

Compared to SACA-A, SACA-B self-attention weights encode the importance of each point feature under a common context, as highlighted in Eq. (4) and (5).

Learning happens only in MLP for both operations. In Figure 3, we plot the attention maps, which visualize Euclidean distances between the context feature of a selected point and the point features of its accessible points before SACA operation.

Model Architecture. Figure 4 shows the proposed network model to output conditional coordinate distributions. The top, middle and bottom branches model $p(z_i|\mathbf{s}_{\le i-1})$, $p(y_i|\mathbf{s}_{\le i-1}, z_i)$ and $p(x_i|\mathbf{s}_{\le i-1}, z_i, y_i)$, respectively, for $i = 1, ..., n$. The point coordinates are sampled according to the estimated softmax probability distributions. Note that the input points in the latter two cases are masked accordingly so that the network cannot see information that has not been generated. During the training phase, points are available to compute all the context features, thus coordinate distributions can be estimated in parallel. However, the point cloud generation is a sequential procedure, since each sampled coordinate needs to be fed as input back into the network, as demonstrated in Figure 1.

Conditional PointGrow. Given a condition or embedding vector, $\mathbf{h}$, we hope to generate a shape satisfying the latent meaning of $\mathbf{h}$. To achieve this, Eq. (1) and (2) are adapted to Eq. (6) and (7), respectively, as below:

$$p(\mathbf{S}) = \prod_{i=1}^{n} p(\mathbf{s}_i|\mathbf{s}_1, ..., \mathbf{s}_{i-1}, \mathbf{h}) = \prod_{i=1}^{n} p(\mathbf{s}_i|\mathbf{s}_{\le i-1}, \mathbf{h}) \tag{6}$$

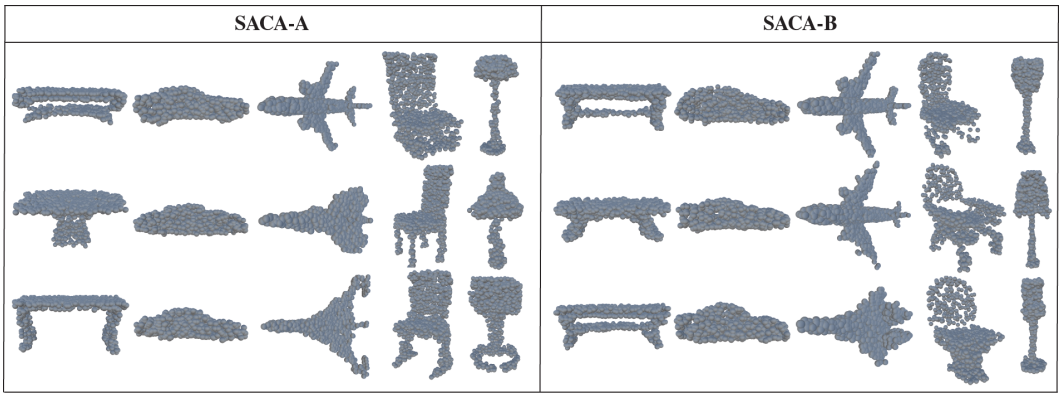

Figure 5: Generated point clouds for different categories from scratch by unconditional *PointGrow*. From left to right for each set: table, car, airplane, chair and lamp.

$$p(\mathbf{s}_i|\mathbf{s}_{\leq i-1}, \mathbf{h}) = p(z_i|\mathbf{s}_{\leq i-1}, \mathbf{h}) \cdot p(y_i|\mathbf{s}_{\leq i-1}, z_i, \mathbf{h}) \cdot p(x_i|\mathbf{s}_{\leq i-1}, z_i, y_i, \mathbf{h}) \tag{7}$$

The additional condition, $\mathbf{h}$, affects the coordinate distributions by adding biases in the generative process. We implement this by changing the operation between adjacent fully-connected layers from $\mathbf{x}^{i+1} = f(\mathbf{W}\mathbf{x}^i)$ to $\mathbf{x}^{i+1} = f(\mathbf{W}\mathbf{x}^i + \mathbf{H}\mathbf{h})$, where $\mathbf{x}^{i+1}$ and $\mathbf{x}^i$ are feature vectors in the $i + 1^{th}$ and i^{th} layer, respectively, $\mathbf{W}$ is a weight matrix, $\mathbf{H}$ is a matrix that transforms $\mathbf{h}$ into a vector with the same dimension as $\mathbf{W}\mathbf{x}^i$, and $f(\cdot)$ is a nonlinear activation function. In this paper, we experimented with $\mathbf{h}$ as an one-hot categorical vector which adds class dependent bias, and an high-dimensional embedding vector of a 2D image which adds geometric constraint.

3 Experiments

Datasets. We evaluated the proposed framework on ShapeNet dataset (Chang et al. (2015)), which is a collection of CAD models. We used a subset consisting of 17,687 models across 7 categories. To generate corresponding point clouds, we first sample 10,000 points uniformly from each mesh, and then use *farthest point sampling* to select 1,024 points among them representing the shape. Each category follows a split ratio of 0.9/0.1 to separate training and testing sets. ModelNet40 (Wu et al. (2015)) and PASCAL3D+ (Xiang et al. (2014)), are also used for further analysis. ModelNet40 contains CAD models from 40 categories, and we obtain their point clouds from Qi et al. (2017a). PASCAL3D+ is composed of PASCAL 2012 detection images augmented with 3D CAD model alignment, and used to demonstrate the generalization ability of conditional *PointGrow*.

3.1 Unconditional point cloud generation

Figure 5 shows point clouds generated by unconditional *PointGrow*. Since an unconditional model lacks knowledge about the shape category to generate, we train separate models for each category. Figure 1 demonstrates point cloud generation for an airplane category. Note that no semantic information of discrete coordinates (*i.e.*scattered points in point cloud) is provided during training, but the predicted distribution turns out to be categorically representative. (*e.g.*in the second row, the network model outputs a roughly symmetric distribution along X axis, which describes the wings' shape of an airplane.) The autoregressive architecture in *PointGrow* is capable of abstracting high-level semantics even from unaligned point cloud samples.

Evaluation on Fidelity and Diversity. The negative log-likelihood is commonly used to evaluate autoregressive models for image and audio generation (Oord et al. (2016); Van Den Oord et al. (2016)). However, we observed inconsistency between its value and the visual quality of point cloud generation. It is validated by the comparison of two baseline models: *CA-Mean* and *CA-Max*, where the SACA operation is replaced with the CA operation implemented by mean and max pooling, respectively. In Figure 6 (left), we report negative log-likelihoods in *bits per coordinate* on ShapeNet testing sets of airplane and car categories, and visualize their representative results. Despite CA-Max shows lower negative log-likelihoods values, it gives less visually plausible results (i.e. airplanes lose wings and cars lose rear ends).

Table 1: Classification accuracy using PointNet (Qi et al. (2017a)). **SG:** Train on ShapeNet training sets, and test on generated shapes; **GS:** Train on generated shapes, and test on ShapeNet testing sets.

Methods	SG	GS
3D-GAN	82.7	83.4
Latent-GAN-CD	81.6	82.7
Baseline (CA-Max)	71.9	83.4
Baseline (CA-Mean)	82.1	84.4
Ours (SACA-A)	**90.3**	91.8
Ours (SACA-B)	89.4	**91.9**

Table 2: The comparison on classification accuracy between our models and other unsupervised methods on ModelNet40 dataset. All the methods train a linear SVM on the high-dimensional representations obtained from unsupervised training.

Methods	Accuracy
SPH (Kazhdan et al. (2003))	68.2
LFD (Chen et al. (2003))	75.2
T-L Network (Girdhar et al. (2016))	74.4
VConv-DAE (Sharma et al. (2016))	75.5
3D-GAN (Wu et al. (2016))	83.3
Latent-GAN-EMD (Achlioptas et al. (2017a))	84.0
Latent-GAN-CD (Achlioptas et al. (2017a))	84.5
Ours (SACA-A)	**85.8**
Ours (SACA-B)	84.4

To faithfully evaluate the generation quality, we conduct user study *w.r.t.*two aspects, fidelity and diversity, among CA-Max, CA-Mean and *PointGrow* (implemented with SACA-A). We randomly select 10 generated airplane and car point clouds from each method. To calculate the fidelity score, we ask the user to score 0, 0.5 or 1.0 for each shape, and take the average of them. The diversity score is obtained by asking the user to scale from 0.1 to 1.0 with an interval of 0.1 about the generated shape diversity within each method. 8 subjects without computer vision background participated in this test. We observe that (1) CA-Mean is more favored than CA-Max, and (2) our *PointGrow* receives the highest preference on both fidelity and diversity.

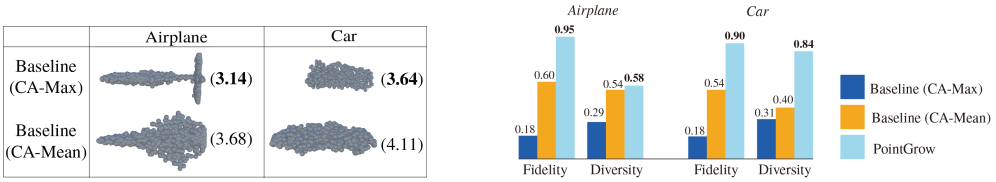

Figure 6: **Left:** Negative log-likelihood for CA-Max and CA-Mean baselines on ShapeNet airplane and car testing sets. **Right:** User study results on fidelity and diversity of the generated point clouds.

Evaluation on Semantics Preserving. After generating point clouds, we perform classification as a measure of semantics preserving. Original meshes are used for training while the generated point clouds are used for testing. More specifically, after training on ShapeNet training sets, we generated 300 point clouds per category (2,100 in total for 7 categories), and conducted two classification tasks: one is training on original ShapeNet training sets, and testing on generated shapes; the other is training on generated shapes, and testing on original ShapeNet testing sets. PointNet (Qi et al. (2017a)), a widely-uesd model, was chosen as the point cloud classifier. We implement two GAN-based competing methods, 3D-GAN (Wu et al. (2016)) and latent-GAN (Achlioptas et al. (2017a)), to sample different shapes for each category, and also include CA-Max and CA-Mean for comparison. The results are reported in Table 1. Note that the CA-Mean baseline achieves comparable performance against both GAN-based competing methods. In the first classification task, our SACA-A model outperforms existing models by a relatively large margin, while in the second task, SACA-A and SACA-B models show similar performance.

Unsupervised Feature Learning. We next evaluate the learned point feature representations of the proposed framework, using them as features for classification. We obtain the feature representation of a shape by applying different types of "symmetric" functions as illustrated in Qi et al. (2017a) (i.e. min, max and mean pooling) on features of each layer before the SACA operation, and concatenate them all. Following Wu et al. (2016), we first pre-train our model on 7 categories from the ShapeNet dataset, and then use the model to extract feature vectors for both training and testing shapes of ModelNet40 dataset. A linear SVM is used for feature classification. We report our best results in Table 2. SACA-A model achieves the best performance, and SACA-B model performs slightly worse than Latent-GAN-CD (Achlioptas et al. (2017a) uses 57,000 models from 55 categories of ShapeNet dataset for pre-training, while we use 17,687 models across 7 categories).

Shape Completion. Our model can also perform shape completion. Given an initial set of points, our model is capable of completing shapes multiple ways. Figure 7 visualizes example predictions. The input points are sampled from ShapeNet testing sets, which are not seen during the training

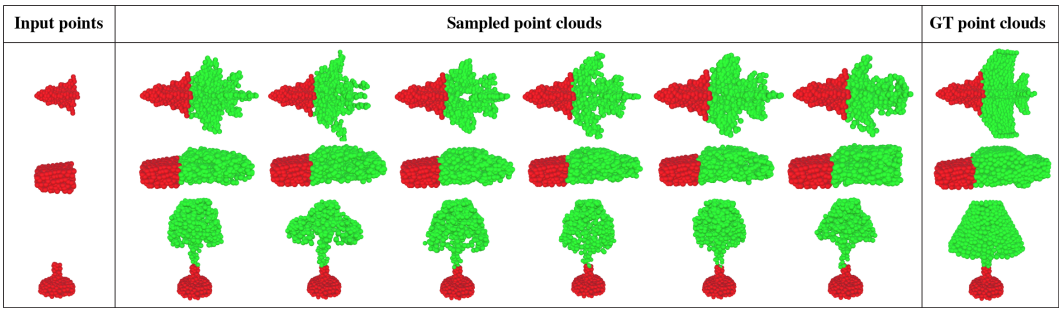

Figure 7: Shape completion results generated by *PointGrow*.

process. The shapes generated by our model are different from the original ground truth point clouds, but look plausible. A current limitation of our model is that it works only when the input point set is given as the beginning part of a shape along its primary axis, since our model is designed and trained to generate point clouds along that direction. More investigation is required to complete shapes when partial point clouds are given from other directions.

3.2 Conditional Point Cloud Generation

SACA-A is used to demonstrate *conditional PointGrow* here owing to its high performance.

Conditioned on Category Label. We first experiment with class-conditional modelling of point clouds, given an one-hot vector $\mathbf{h}$ with its nonzero element h_i indicating the i^{th} shape category. The one-hot condition provides categorical knowledge to guide the shape generation process. We train the proposed model across multiple categories, and plausible point clouds for desired shape categories can be sampled (as shown in Figure 8). Failure cases are also observed: generated shapes present interwoven geometric properties from other shape types. For example, the airplane misses wings and generates a car-like body; the lamp and the car develop chair leg structures.

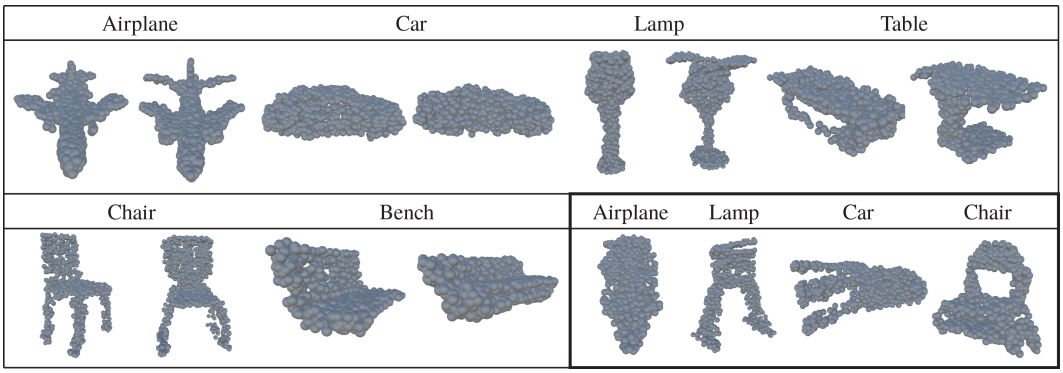

Figure 8: Examples of successfully generated point clouds by *PointGrow* conditioned on one-hot categorical vectors. Bottom right shows failure cases.

Conditioned on 2D Image. Next, we experiment with image conditions for point cloud generation. Image conditions apply constraints constrains to the point cloud generation process because the geometric structures of sampled shapes should match their 2D projections. In our experiments, we obtain an image condition vector by adopting the image encoder in Sun et al. (2018b) to generate a feature vector of 512 elements, and optimize it along with the rest of the model components. The model is trained on synthetic ShapeNet dataset, and one out of 24 views of a shape (provided by Choy et al. (2016)) is randomly selected at each training step as the image condition input. The trained model is also tested on real images from the PASCAL3D+ dataset to prove its generalizability. For each input, we removed the background, and cropped the image so that the target object is centered. The PASCAL3D+ dataset is challenging because the images are captured in real environments, and contain noisier visual signals which are not seen during the training process. We show ShapeNet testing images and PASCAL3D+ real images together with their sampled point cloud results on Figure 9 upper left.

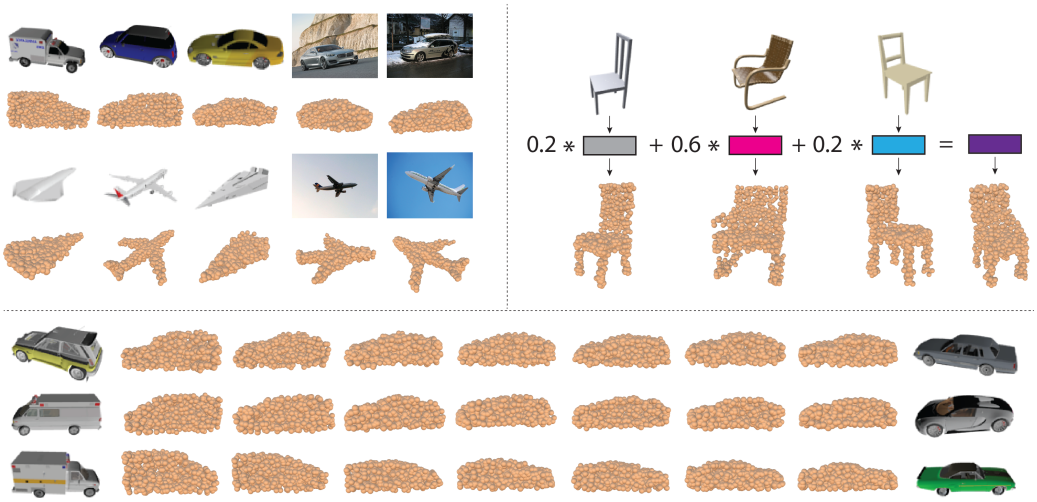

Figure 9: **Upper left:** Generated point clouds conditioned on synthetic testing images from ShapeNet (first 4 columns) and real images from PASCAL3D+ (last 2 columns). **Upper right:** Examples of image condition arithmetic for chairs. **Bottom:** Examples of image condition linear interpolation for cars. Condition vectors from leftmost and rightmost images are used as endpoints for the shape interpolation.

We further quantitatively evaluate the conditional generation results by calculating mean Intersection-over-Union (mIoU) with ground truth volumes. Here we only consider 3D volumes containing more than 500 occupied voxels, and using furthest point sampling method to select 500 out of them to describe the shape. To compensate for the sampling randomness of *PointGrow*, we slightly align generated points to their nearest voxels within a neighborhood of 2-voxel radius. As shown in Table 3, *PointGrow* achieves above-par performance on conditional 3D shape generation.

Table 3: Conditional generation evaluation by per-category IoU on 13 ShapeNet categories. We compare our results with 3D-R2N2 (Choy et al. (2016)) and PointOutNet (Fan et al. (2017)).

	airplane	bench	cabinet	car	chair	monitor	lamp	speaker	firearm	couch	table	cellphone	watercraft	Avg.
3D-R2N2 (1 view)	0.513	0.421	0.716	0.798	0.466	0.468	0.381	0.662	0.544	0.628	0.513	0.661	0.513	0.560
3D-R2N2 (5 views)	0.561	0.527	**0.772**	0.836	**0.550**	0.565	0.421	0.717	0.600	0.706	0.580	**0.754**	0.610	0.631
PointOutNet	0.601	0.550	0.771	0.831	0.544	0.552	0.462	**0.737**	0.604	**0.708**	**0.606**	0.749	0.611	0.640
Ours	**0.742**	**0.629**	0.675	**0.839**	0.537	**0.567**	**0.560**	0.569	**0.674**	0.676	0.590	0.729	**0.737**	**0.656**

3.3 Learned Image Condition Manifold

Image Condition Interpolation. Linearly interpolated embedding vectors of image pairs can be conditioned upon to generate linearly interpolated geometrical shapes. As shown on Figure 9 bottom, walking over the embedded image condition space gives smooth transitions between different geometrical properties of different types of cars.

Image Condition Arithmetic. Another interesting way to impose the image conditions is to perform arithmetic on them. We demonstrate this by combining embedded image condition vectors with different weights. Examples of this kind are shown on Figure 9 upper right. Note that the generated final shapes contain geometrical features shown in generated shapes of their operands.

4 Conclusion

In this work, we propose *PointGrow*, a new generative model that can synthesize realistic and diverse point cloud with high resolution. Unlike previous works that rely on local regularities to synthesize 3D shapes, our *PointGrow* builds upon autoregressive architecture to encode the diverse surface information of point cloud. To further capture the long-range dependencies between points, two dedicated self-attention modules are designed and carefully integrated into our framework. *PointGrow* as a generative model also enables effective unsupervised feature learning, which is extremely useful for low-data recognition tasks. Finally, we show that *PointGrow* learns a smooth image condition manifold where 3D shape interpolation and arithmetic calculation can be performed inside.

REFERENCES

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Representation learning and adversarial generation of 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2017a.

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2017b.

Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *ECCV*, pp. 561–578, 2016.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pp. 223–232. Wiley Online Library, 2003.

Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, pp. 628–644, 2016.

Lucía Díaz Vilariño, Pawel Boguslawski, Kourosh Khoshelham, Henrique Lorenzo, and Lamine Mahdjoubi. Indoor navigation from point clouds: 3d modelling and obstacle detection. International Society for Photogrammetry and Remote Sensing, 2016.

Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, volume 2, pp. 6, 2017.

Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, pp. 484–499, 2016.

Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3 d shape descriptors. In *Symposium on geometry processing*, volume 6, pp. 156–164, 2003.

Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *ICCV*, pp. 863–872, 2017.

Andrey Kurenkov, Jingwei Ji, Animesh Garg, Viraj Mehta, JunYoung Gwak, Christopher Choy, and Silvio Savarese. Deformnet: Free-form deformation network for 3d shape reconstruction from a single image. In *WACV*, pp. 858–866, 2018.

Yangyan Li, Rui Bu, Mingchao Sun, and Baoquan Chen. Pointcnn. *arXiv preprint arXiv:1801.07791*, 2018.

Chen-Hsuan Lin, Chen Kong, and Simon Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. *arXiv preprint arXiv:1706.07036*, 2017a.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017b.

Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CVPR*, 1(2):4, 2017a.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, pp. 5099–5108, 2017b.

Abhishek Sharma, Oliver Grau, and Mario Fritz. Vconv-dae: Deep volumetric shape learning without object labels. In *ECCV*, pp. 236–250, 2016.

Yiru Shen, Chen Feng, Yaoqing Yang, and Dong Tian. Mining point cloud local structures by kernel correlation and graph pooling. In *CVPR*, volume 4, 2018.

Jonathan Dyssel Stets, Yongbin Sun, Wiley Corning, and Scott W Greenwald. Visualization and labeling of point clouds in virtual reality. In *SIGGRAPH Asia 2017 Posters*, pp. 31, 2017.

Yongbin Sun, Sai Nithin R Kantareddy, Rahul Bhattacharyya, and Sanjay E Sarm. X-vision: An augmented vision tool with real-time sensing ability in tagged environments. *arXiv preprint arXiv:1806.00567*, 2018a.

Yongbin Sun, Ziwei Liu, Yue Wang, and Sanjay E Sarma. Im2avatar: Colorful 3d reconstruction from a single image. *arXiv preprint arXiv:1804.06375*, 2018b.

Aäron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *SSW*, pp. 125, 2016.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018.

Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *NIPS*, pp. 82–90, 2016.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, pp. 1912–1920, 2015.

Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *WACV*, pp. 75–82, 2014.

Saining Xie, Sainan Liu, Zeyu Chen, and Zhuowen Tu. Attentional shapecontextnet for point cloud recognition. In *CVPR*, pp. 4606–4615, 2018.

Bo Yang, Hongkai Wen, Sen Wang, Ronald Clark, Andrew Markham, and Niki Trigoni. 3d object reconstruction from a single depth view with adversarial learning. *arXiv preprint arXiv:1708.07969*, 2017.

Haoxuan You, Yifan Feng, Rongrong Ji, and Yue Gao. Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition. *arXiv preprint arXiv:1808.07659*, 2018.

Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *CVPR*, pp. 2790–2799, 2018.

Xiangyu Yue, Bichen Wu, Sanjit A Seshia, Kurt Keutzer, and Alberto L Sangiovanni-Vincentelli. A lidar point cloud generator: from a virtual world to autonomous driving. In *ICMR*, pp. 458–464, 2018.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.

Silvia Zuffi, Angjoo Kanazawa, and Michael J Black. Lions and tigers and bears: Capturing non-rigid, 3d, articulated shape from images. In *CVPR*, pp. 3955–3963, 2018.