PointGrow: Autoregressively Learned Point Cloud Generation with Self-Attention

Yongbin Sun[1], Yue Wang[1], Ziwei Liu[2]*, Joshua E Siegel[3] & Sanjay E Sarma[1]

1. Massachusetts Institute of Technology 2. The Chinese University of Hong Kong
3. Michigan State University

Abstract

Generating 3D point clouds is challenging yet highly desired. This work presents a novel autoregressive model, PointGrow*, which can generate diverse and realistic point cloud samples from scratch or conditioned on semantic contexts. This model operates recurrently, with each point sampled according to a conditional distribution given its previously-generated points, allowing inter-point correlations to be well-exploited and 3D shape generative processes to be better interpreted. Since point cloud object shapes are typically encoded by long-range dependencies, we augment our model with dedicated self-attention modules to capture such relations. Extensive evaluations show that* PointGrow *achieves satisfying performance on both unconditional and conditional point cloud generation tasks, with respect to realism and diversity. Several important applications, such as unsupervised feature learning and shape arithmetic operations, are also demonstrated.*

1. Introduction

Recently 3D generative model has attracted enormous research interests because it directly promotes the development of emerging applications, such as virtual/augmented reality [44, 46] and self-driving cars. For example, it is capable of completing the LIDAR scans that might suffer from occlusion issues [63]. Therefore, intelligent systems that are able to automatically generate realistic and diverse 3D shapes are highly desired.

3D shapes are usually represented as triangle meshes or point clouds due to their light-weight nature and simple form. Such shape representations are flexible for rendering, but impose a problem when applying computer vision techniques for processing, because most standard operations are designed based on regular grid-based formats (e.g. image), while meshes and point clouds are fundamentally irregular: vertex or point positions are continuously distributed in the space, and any permutation of their face or point ordering does not change the spatial distribution. Therefore, one major line of 3D research discretizes a continuous shape representation onto a 3D grid [5, 30, 57, 58]. But such volume-based representations consume significant memory and introduce quantization artifacts, making it difficult to generate high-res 3D shapes and retain fine-grained surface details. Therefore, it is more desirable to design a framework specific to raw shape representations, rather than relying upon intermediate representations. In this paper, we choose to represent shapes using point clouds, because they comprise the output of most existing 3D sensing technologies, showing more application values but with less complexity.

The vast majority of existing learning-based works for 3D point clouds generation rely on two types of distance metric between point sets, Chamfer Distance (CD) and Earth Mover's Distance (EMD) [12], to handle the irregularity problem of point clouds. Serving as the loss function, these distance metric penalizes the dissimilarity between the generated and ground truth point sets, and help deep generative models learn to produce visually-plausible 3D shapes for various types of inputs [14, 28, 2, 13, 21, 62, 17]. However, the generative process is hard to interpret due to the intrinsic limitation of *inter-set* distance metric.

In this work, we seek to explore a different approach to better *understand* and *interpret* the point cloud generative process. We begin by observing that the points constituting a 3D shape have correlations. For example, most man-made shapes are symmetric, such as the four legs of a table; also, points are distributed in a correlated way to form different parts of a shape (e.g. to produce an airplane, some points form its wings, while others have to form its body structure). To explicitly learn and utilize such *inter-point* correlations for shape generation, we adopt a probabilistic approach to jointly model the spatial distribution of all the points of a shape in an n dimensional space, where n is the number of points in a point cloud. Underpinned by a joint point distribution reflecting the underlying inter-point correlation

*corresponding author.

in the data, a high joint probability value should correspond to the point set distribution of a plausible 3D shape, while a low value should indicate an implausible one. In this way, the point cloud generation process can be cast as a sampling process in an n dimensional point space.

Furthermore, since a joint probability can be decomposed by chain rule as the product of a series of conditional probabilities, the joint point set probability can be expressed by a series of conditional point probabilities, where each point is conditioned on its previously generated ones. This property naturally enables us to visualize the shape generative process and interpret inter-point correlations in a *point-by-point* manner during the point sampling process, as shown in Figure 1.

To this end, we propose an autoregressive framework dubbed *PointGrow* to generate every point recurrently. Specifically, *PointGrow* estimates a conditional distribution of the point under consideration given all its preceding points. However, the irregularity of point clouds imposes difficulties when aggregating meaningful information from a given point set, especially when such information is contained by distant points. Therefore, we further propose two point cloud-based self-attention modules to dynamically aggregate long-range dependencies from available points. Our experiments show that those two modules can improve information flow between points, and successfully capture meaningful semantic information.

The contributions of our work are summarized as below:

- We propose a novel autogressive model, *PointGrow*, for point cloud generation, which models the joint 3D spatial distribution in a *point-by-point* manner. PointGrow has two appealing properties: 1) it is capable of generating diverse and realistic 3D clouds, 2) it constitutes an interpretable 3D shape generative process.

- Two self-attention modules are carefully designed to capture long-range dependencies and semantic correlations between points, facilitating the generation of plausible part configurations within 3D objects.

- Besides point cloud generation, our framework also enables several important applications, such as diverse shape completions, unsupervised feature learning and shape arithmetic operations.

2. Related Work

While 3D data processing and generation has a long history, here, we only discuss the directly related work of using deep networks to analyze 3D shapes, autoregressive networks, and self-attention.

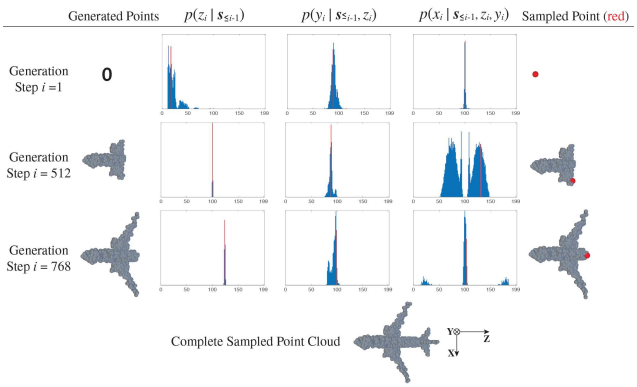

Figure 1. The point cloud generation process in *PointGrow* (best viewed in color). Given $i-1$ generated points, our model estimates a conditional distribution of coordinate z_i, indicated as $p(z_i|\mathbf{s}_{\leq i-1})$, and then samples a value (indicated as a red bar) according to this distribution. 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 $\{x_i, y_i, z_i\}$. Note that x_{512} shows a symmetric wing shaped conditional distribution.

2.1. Shape Analysis

Volumetric Methods. 3D shape recognition and generation has been studied using 3D voxel grids [5, 10, 30, 57, 58, 47]. Voxelization often produces a sparsely-occupied 3D grid, which limits resolution and introduces quantization artifacts. Recent frameworks have been proposed to reduce spatial complexity of volumetric shape representations, [16, 18, 39, 45, 49, 55], though these generally suffer from high computation costs.

Mesh-Based Methods. 3D meshes are a lightweight approach for geometric modelling via a set of vertices and triangular or quad primitives. Recent work has extended standard convolutions to mesh surfaces for aggregating and propagating local features [4, 6, 29, 61]. Relevant work reconstructing 3D shapes as 3D meshes is found in [54, 2, 41, 22, 25, 33, 34, 20, 48].

Point Cloud-Based Methods. PointNet [36] is the pioneering work in applying deep neural nets to point sets, using a symmetric function to aggregate feature vectors for all points in a permutation-invariant manner. PointNet's successors explore ways to accumulate local information in the spatial [37, 43, 24, 35] and embedded feature domains [56, 27, 50] to achieve high performance. To address point cloud generative tasks, [12] introduced two symmetric distance metrics, CD and EMD, to measure the distance between two point sets. These metrics are order-invariant, which makes them suitable as loss function operated directly on point clouds. By taking advantage of these metrics, models have been proposed to address point cloud synthesis problems under different settings [14, 28, 2, 13, 21, 62, 60, 17, 63]. However, existing

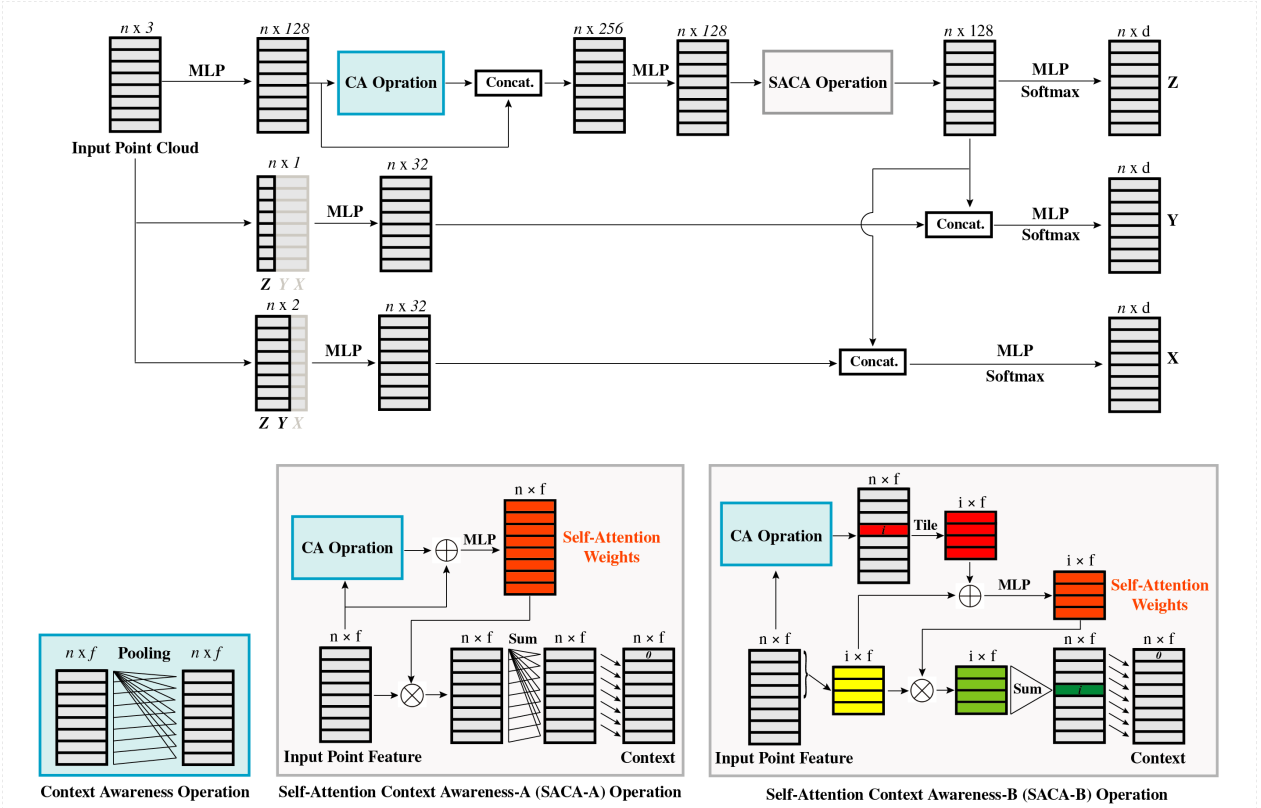

Figure 2. The proposed model architecture and context awareness operations to estimate conditional distributions of point coordinates. $\otimes$: element-wise production, $\oplus$: concatenation.

generative approaches focus on measuring inter-set dissimilarity, and the inter-point relationship within a point set is not well understood.

2.2. Autoregressive Networks

Autoregressive networks model current values as a function of their own previous values, and have been adopted to model the joint distribution of image pixels [31, 52, 40] and audio samples [51]. The joint distribution is cast as a product of conditional distributions, and each condition distribution is modeled using a deep neural network that takes as input previously generated values and outputs a distribution for the value currently under consideration. But it is not trivial to adapt autoregressive frame to point cloud due to the irregularity problem of point clouds.

2.3. Self-Attention

Attention is a flexible mechanism to capture information in a self-adaptive manner such that accumulated important information is weighted highly. It improves performance in tasks including image recognition [26, 11] and natural language processing [9, 53, 3]. Recently, self-attention has been adopted into generative tasks, such as image generation [64]. In our experiments, we demonstrate that self-attention modules can be extended to process unordered point sets and capture inter-point correlations.

3. PointGrow

This section introduces the formulation and implementation of *PointGrow* (and its conditional version), a point-by-point generative model for 3D point clouds.

Unconditional PointGrow. A point cloud, $\mathbf{S}$, that consists of n points is defined as $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n\}$, with its i^{th} point $\mathbf{s}_i = \{x_i, y_i, z_i\}$ 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:

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

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

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

where each coordinate is conditioned on available coordinates. To facilitate the generation process, we sort training points according to their z coordinates to encourage a shape to be generated mainly along its primary axis during testing (like 3D printing), hoping that semantic information can be better captured to produce consistent shapes (e.g. a rear car body to be generated has to match an existing front).

But since our model is sampling-based, the generated coordinates are not strictly larger than their previous ones and processing modules (introduced later) should be invariant to point permutation. Here, we model the conditional probability distribution of each coordinate using a deep neural network. Prior art [31] shows that a softmax discrete distribution is more flexible than a continuous one to model any arbitrary distribution. So we discrete point coordinates by scaling them into the range [0, 1] and quantizing them to d uniformly distributed values. Note that different from many existing voxel-based methods generating tensors of size d^3 and operating on 3D volumes, our model outputs tensors of $3 \times n \times d$ (usually much smaller than d^3 considering the resolution of shape volumes) and operates directly on sparse point representation. We set $n = 1024$ and $d = 200$ as a trade-off between generative performance and quantization artifacts. Larger n and d can be used to achieve better visual results but at the cost of slower performance.

Context Awareness Operation. Context awareness improves model inference. For example, in [36] and [56], 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 context-aware features for all sets of generated available points in the point cloud generation process, as illustrated in Figure 2 (bottom left). Each row of resultant context-aware 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 accumulates point features in a fixed manner via pooling. Improving this, we propose two learning-based operations to determine the weights for aggregating point features self-adaptively. We define these as Self-Attention Context Awareness (SACA) operations, and the weights as self-attention weights.

Figure 2 shows the first SACA operation, SACA-A. To allow each input point feature to understand its importance in context and later determine its weight, we associate it with its context-aware feature obtained after a CA module. The combined feature vector is then passed into a Multi-Layer Perception (MLP) to learn self-attention weights. Given an $n \times f$ point feature matrix, $\mathbf{F}$, with its i^{th} row, $\mathbf{f}_i$, representing the feature vector of the i^{th} point, we compute the i^{th} self-attention weight vector, $\mathbf{w}_i$, as below:

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

where $Mean\{\cdot\}$ is mean pooling, $\oplus$ is concatenation, and $MLP(\cdot)$ is a sequence of fully connected layers. The self-attention weights encode information about context changes due to each newly generated point, and are unique to that point. Next, we conduct element-wise multiplication between input point features and self-attention weights to obtain weighted features, which are then accumulated sequentially to generate corresponding context features. The process to calculate the i^{th} context feature, $\mathbf{c}_i$, is summarized 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 for the i^{th} 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, indicating no a-priori context knowledge is available.

Figure 2 also shows the other SACA operation, SACA-B. SACA-B differs from SACA-A in the way to compute and apply self-attention weights. In SACA-B, the i^{th} context-aware feature after CA operation is shared by all the first i point features to obtain self-attention weights, which are used to compute $\mathbf{c}_i$. This process is described 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. Their differences are highlighted in Eq. (4) and (5).

In Figure 3, we visualize the attention fields during generative processes by visualizing Euclidean distances between the context feature of a query point and the point features before the SACA operation of its previously generated points.

Model Architecture. Figure 2 top shows the proposed network 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. Note that the input points in the latter two cases are masked so that the network cannot see information not-yet-generated. During training, points are available to compute all the context features, thus coordinate distributions can be estimated in parallel. During the generative phase, the point coordinates are sampled according to the estimated softmax probability distributions. This occurs sequentially, since each sampled coordinate needs to be fed as input back into the network, as shown in Figure 1. Our proposed autoregressive architecture models point coordinate distribution categorically, thus a simple cross-entropy loss is sufficient to handle the shape learning process efficiently without requiring the use of a computationally-intensive set-to-set distance loss function.

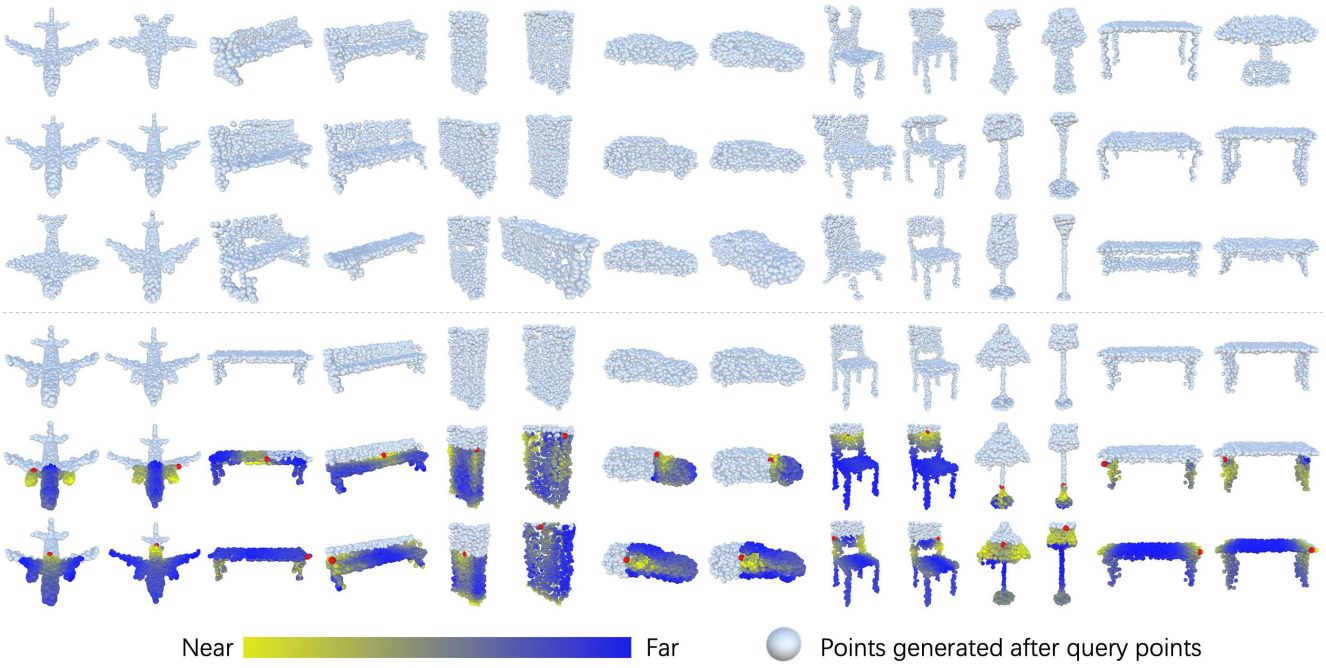

Figure 3. Visualize generated shapes by *PointGrow* (for each two-column shape set, Left: SACA-A; Right: SACA-B). The bottom part additionally shows self-attention fields, measured by Euclidean distance in feature space, for query locations (red points). Note that the attention modules capture not only spatial but also semantic correlations. For example, when generating wing structure points, the model attends more on other available wing structure points, and some are spatially far but semantically close.

Conditional PointGrow. Given a condition or embedding vector, $\mathbf{h}$, we intend 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:

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

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

The additional condition, $\mathbf{h}$, affects the coordinate distributions by adding biases and potential constrains 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 non-linear 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 imposes geometric constraints.

4. Experiments

Datasets. We evaluated our framework on the ShapeNet [7] CAD dataset. We used a subset consisting of 17,687 models across 7 categories: airplanes, cars, tables, chairs, benches, cabinets and lamps. To generate corresponding point clouds, we sampled 10,000 points uniformly from each mesh file, and then used *farthest point sampling* to select 1,024 points representing the shape. Each category follows a split ratio 0.9/0.1 to separate training from testing sets. ModelNet40 [58] and PASCAL3D+ [59] are used for additional analysis and demonstration.

4.1. Unconditional point cloud generation

We first evaluate *unconditional PointGrow* by addressing the following questions: (1) Can the model generate visually plausible 3D shapes in an interpretable manner? (2) Can the model generate diverse shapes? (3) Is the proposed self-attention module important for shape generation? (4) Does the model learn meaningful feature representation?

(1) Generative Process Interpretability and Shape Quality. We start evaluation by showing qualitative results of generated point clouds of *unconditional PointGrow*, shown in Figure 3. Fine-detailed structures can be observed from the generated shapes, such as the jet engine of airplanes and legs of furniture (e.g. table and chair). In the bottom part of Figure 3, we additionally show attention fields when generating corresponding query points in the process. It can be observed that the model focuses on different regions when generating points for different parts, and the focused areas are usually semantically related no matter their spatial distances. For example, when generating airplane wing points, the model aggregates structural knowledge from available wing part points; when generating points close to table legs, other previously generated leg points contribute most; when

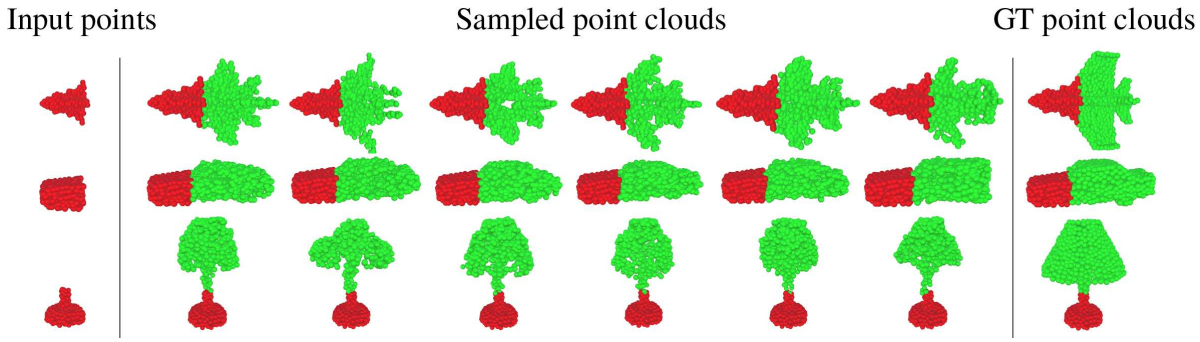

Figure 4. Shape completion results generated by *PointGrow*.

generating torchiere shade points for the lamp, the model considers both existing torchiere shade and base areas for a proper structural match. We also show a sampled generative process for an airplane in Figure 1. Note that the model produces different conditional distribution for points at different parts (e.g. in the second row of Figure 1, the network model outputs a roughly symmetric distribution along the X axis, describing the airplane's wings). From the above investigation, the interpretability of the proposed generative process is demonstrated. In this experiment, we train our model for each category separately, because an unconditional model lacks knowledge about the target shape when sampling from scratch. In later experiments, we show that when a categorical condition is given, it is possible to train the model across multiple shape categories and generate plausible shapes.

Next, we quantitatively evaluate the quality of generated shapes. The negative log-likelihood is commonly used to evaluate autoregressive models for image and audio generation [31, 51]. However, we observed inconsistency between this value and the visual quality of generated 3D shapes. This is validated by comparing 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 5, 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-likelihood values, it gives less visually plausible results (i.e. airplanes lose wings and cars lose rear ends).

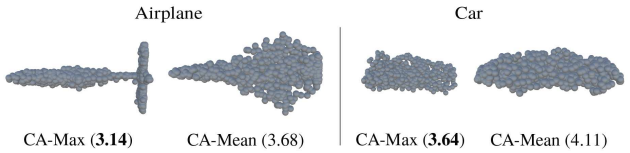

Figure 5. Negative log-likelihood for CA-Max and CA-Mean baselines on ShapeNet airplane and car testing sets.

To actually evaluate the generated shape quality, we argue that if generated 3D shapes contain consistent semantic features as real shapes, a classification model trained on real shapes should perform well on generated ones, and vice versa. Therefore, after training on ShapeNet sets, we generate 300 point clouds per category (2,100 in total for 7 categories), and conduct two classification tasks: one training on original ShapeNet training sets and testing on generated shapes, the other training on generated shapes and testing on original ShapeNet testing sets. Here, PointNet [36], a widely-uesd model, is used as the point cloud shape classifier. We implement another two GAN-based competing methods and report classification results in Table 1, together with model complexity using number of model parameters. 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.

Methods	SG	GS	# parameters
3D-GAN [57]	82.7	83.4	22.53M
Latent-GAN [1]	81.6	82.7	15.78M
Baseline (CA-Max)	71.9	83.4	0.23M
Baseline (CA-Mean)	82.1	84.4	0.23M
Ours (SACA-A)	**90.3**	91.8	0.29M
Ours (SACA-B)	89.4	**91.9**	0.25M

Table 1. Classification accuracy using PointNet [36]. **SG:** Training on ShapeNet sets and testing on generated shapes; **GS:** Training on generated shapes and testing on ShapeNet sets.

(2) Shape Diversity. To demonstrate *PointGrow* can generate diverse shapes, we conducted a shape completion task. Given an initial set of points, our model is capable of completing shapes in multiple ways. Figure 4 visualizes examples. The input points are sampled from ShapeNet testing sets, which are not seen during the training process. The shapes generated by our model are different from the original ground truth point clouds, but still 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, and in future work we will investigate how to complete shapes when partial point clouds are given from any directions.

(3) Ablation Study on Self-Attention Module. We conduct an ablation study to investigate the importance of the Self-Attention module in shape generation process. To quantitatively evaluate generated shapes and inspired by Fréchet Inception Distance (FID) [19, 64], a metric widely used to measure the visual quality of generated images, we provide a similar metric, "PointNet Distance" (PND), to

measure the geometric quality of generated point clouds. PND follows the same assumption and formula as FID, except that the features from a point cloud is obtained from the global feature of a PointNet model pre-trained on ModelNet40 with a global feature dimension of 128. Three model architectures are considered: without the second CA module (No CA), CA-Mean baseline, and SACA-A. We generate 200 point clouds per category for each model architecture. The PND is computed for each category and reported in Table 2. We observe a large performance improvement by gathering context information with self-attention supported.

	airplane	car	table	chair	bench	cabinet	lamp	Avg.
No CA	54.08	58.30	219.54	181.18	104.38	318.89	165.52	157.41
CA-Mean	7.05	4.33	12.83	27.47	6.09	20.67	88.14	23.79
SACA-A	**1.92**	**0.53**	**2.40**	**4.61**	**1.24**	**3.67**	**8.34**	**3.24**

Table 2. The PointNet Distance (PND) for each category. A lower score signifies a better model.

Methods	SVM	SL
SPH [23]	68.2	-
LFD [8]	75.2	-
T-L Network [15]	74.4	-
VConv-DAE [42]	75.5	-
3D-GAN [57]	83.3	-
Latent-GAN-EMD [1]	84.0	-
Latent-GAN-CD [1]	84.5	-
MTN [14]	-	**86.4**
Ours (SACA-A)	**85.8**	86.3
Ours (SACA-B)	84.4	85.3

Table 3. The comparison on classification accuracy between our models and other unsupervised methods on ModelNet40 dataset using linear SVM classifier and single layer classifier (SL).

(4) Unsupervised Feature Learning. To prove the model actually learns meaningful representations, we extract learned features and use them for classification tasks. We obtain the feature vector of a shape by applying different types of "symmetric" functions as illustrated in [36] (i.e. min, max and mean pooling) on features of each layer before the SACA operation, and concatenate them all. Following [57], we pre-train our model on 7 categories from the ShapeNet dataset, and then use this model to extract feature vectors for both training and testing shapes from the ModelNet40 dataset. We experimented with both linear SVM and single layer classifiers, following the same settings as [57] and [14], respectively. We report our best results in Table 3. The SACA-A model achieves the best performance using SVM classifier, and performs slightly worse than MTN [14] using single layer classifier.

4.2. Conditional Point Cloud Generation

To evaluate *conditional PointGrow*, we answer the following questions: (1) Can the model be trained across multiple categories? (2) Can the model generate plausible shapes for image conditions?

(1) Conditioned on Category Label. We first experiment with category-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, enabling the model to be trained across multiple categories. Figure 6 shows generated shape examples. 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.

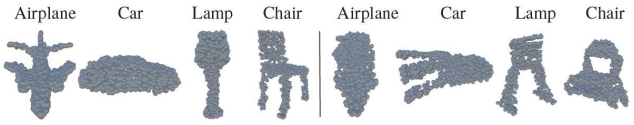

Figure 6. Generated point clouds of *PointGrow* conditioned on different one-hot categorical labels. Right part shows failure cases.

(2) Conditioned on 2D Image. Next, we experiment with image conditions for point cloud generation. Image conditions add additional constrains to the point cloud generation process such that the geometric structures of sampled shapes match their 2D projections. In our experiments, we obtain an image condition vector through an image encoder, and optimize it together with the rest model components from scratch. The model is trained on synthetic ShapeNet dataset, and one out of 24 views of a shape (provided by [10]) is selected as the image condition input. The trained model is also tested on foreground objects of real images from the PASCAL3D+ dataset to prove its generalizability. The PASCAL3D+ dataset is challenging because the images are captured in real environments. Testing examples are shown on Figure 7 upper left.

We quantitatively evaluate the conditional generation results in terms of mean Intersection-over-Union (mIoU) [10] and point-wise 3D Euclidean distance [28]. Here we obtain ground truth points as voxel centers of 3D volumes from [10], and only consider shapes containing more than 500 occupied voxels, with 500 of them uniformly sampled to describe the shape. To compensate for the sampling randomness of *PointGrow*, we align generated points to their nearest voxels within a neighborhood of 2-voxel radius. When calculating surface-to-surface 3D Euclidean distance metric, we further remove interior points with 26 non-empty neighbors for fair comparison when selecting ground truth points. As shown in Table 4 and Table 5, *PointGrow* achieves above-par performance on conditional 3D shape generation.

Further, we demonstrate that intermediate 3D shapes can be generated from linearly interpolated embedding vectors of image pairs (Figure 7 bottom), and compositive shapes can be generated by applying arithmetic on embedded image condition vectors (Figure 7 upper right).

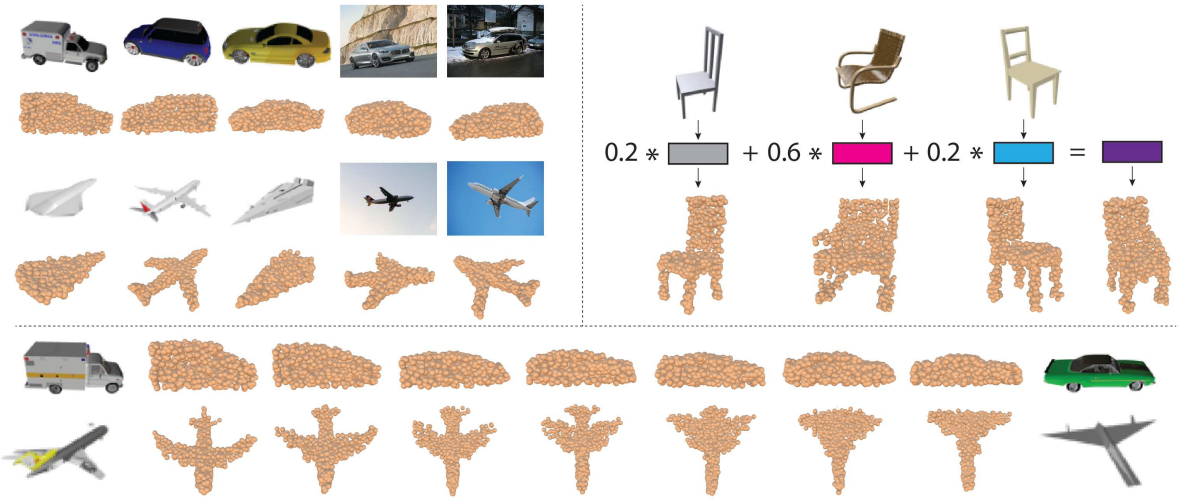

Figure 7. **Upper left:** Generated point clouds conditioned on synthetic testing images from ShapeNet (first 4 columns) and real images from PASCAL3D+ (last 2 columns). **Upper right:** Examples of image condition arithmetic for chairs. **Bottom:** Examples of image condition linear interpolation for cars. Condition vectors from leftmost and rightmost images are used as endpoints for the shape interpolation.

	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
PointSetGen	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**

Table 4. Conditional generation evaluation by per-category IoU on 13 ShapeNet categories. We compare our results with 3D-R2N2 ([10]) and PointSetGen [12].

	airplane	bench	cabinet	car	chair	monitor	lamp	speaker	firearm	couch	table	cellphone	watercraft	Avg.
3D-R2N2 (1 view)	3.207	3.350	1.636	1.808	2.759	3.235	8.400	2.652	4.798	2.725	3.118	2.202	3.592	3.345
3D-R2N2 (5 views)	2.399	2.323	1.420	1.664	1.854	2.088	5.698	2.487	4.193	2.306	2.128	1.874	3.210	2.588
PointSetGen (1 view)	1.301	1.814	2.463	1.800	1.887	1.919	2.347	3.215	1.316	2.592	1.874	1.516	1.715	1.982
Lin et al. (1 view)	1.294	1.757	1.814	1.446	1.886	2.142	2.635	2.371	1.289	1.917	1.689	1.939	1.813	1.846
MRTNet (1 view)	0.976	**1.438**	1.774	1.395	**1.650**	**1.815**	**1.944**	2.165	**1.029**	1.768	**1.570**	1.346	1.394	1.559
Ours (1 view)	**0.615**	1.726	**1.201**	**0.416**	1.775	1.937	2.235	**1.998**	1.300	**1.405**	1.974	**0.765**	**0.865**	**1.401**
3D-R2N2 (1 view)	2.879	3.697	2.817	3.238	4.207	4.283	9.722	4.335	2.996	3.628	4.208	3.314	4.007	4.102
3D-R2N2 (5 views)	2.391	2.603	2.619	3.146	3.080	2.953	7.331	4.203	2.447	3.196	3.134	2.734	3.614	3.342
PointSetGen (1 view)	1.488	1.983	2.444	2.053	2.355	2.334	2.212	2.788	1.358	2.784	2.229	1.989	1.877	2.146
Lin et al. (1 view)	1.541	1.487	**1.072**	1.061	2.041	**1.440**	4.459	**1.706**	1.510	1.423	1.620	1.198	1.550	1.701
MRTNet (1 view)	0.920	**1.326**	1.602	1.303	**1.603**	1.901	2.089	2.121	**1.028**	1.756	**1.570**	1.332	1.490	1.529
Ours (1 view)	**0.914**	1.531	1.564	**0.732**	1.857	1.789	**1.732**	2.020	1.166	**1.381**	1.744	**0.869**	**1.070**	**1.413**

Table 5. Single-image shape inference results. Top part: pred → GT errors; Bottom part: GT → pred errors, both scaled by 100. We compare our results with 3D-R2N2 [10], PointSetGen [12], Lin et al. [28] and MRTNet [14].

5. Discussion and Conclusion

This work studies the problem of point cloud generation. Unlike previous work, which minimizes set-to-set distances for generative learning, our model builds upon an autoregressive architecture and exploits point-to-point relations during generation, allowing the generative process to be better understood and interpreted. To further capture long-range dependencies in a self-adaptive manner and address the irregularity of point cloud data, two self-attention models are integrated within our framework. Extensive experiments validates the efficacy of this approach across a wide range of tasks.

Though our model generates visually-plausible 3D shapes, it faces two potential limitations. Firstly, due to the iterative property intrinsic to autoregressive models, the model scales poorly when generating large point sets. Recent work [38, 32] has accelerated autoregression-based audio generation. Similar techniques are also applicable here (*e.g.* generating clouds in a hierarchical rather than sequential manner). Secondly, our model only generates a point cloud along its primary axis as determined in training. This does not hinder generation performance if the shape is sampled from scratch, but limits its applicability to applications like shape completion. Generating clouds more flexibly will be an important topic for further research.

References

[1] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas. Representation learning and adversarial generation of 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2017. 6, 7

[2] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas. Learning representations and generative models for 3d point clouds. 2018. 1, 2

[3] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 3

[4] D. Boscaini, J. Masci, E. Rodolà, and M. Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 3189–3197, 2016. 2

[5] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236*, 2016. 1, 2

[6] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013. 2

[7] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 5

[8] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, 2003. 7

[9] J. Cheng, L. Dong, and M. Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016. 3

[10] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, 2016. 2, 7, 8

[11] M. Denil, L. Bazzani, H. Larochelle, and N. de Freitas. Learning where to attend with deep architectures for image tracking. *Neural computation*, 24(8):2151–2184, 2012. 3

[12] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, volume 2, page 6, 2017. 1, 2, 8

[13] M. Gadelha, S. Maji, and R. Wang. Shape generation using spatially partitioned point clouds. *arXiv preprint arXiv:1707.06267*, 2017. 1, 2

[14] M. Gadelha, R. Wang, and S. Maji. Multiresolution tree networks for 3d point cloud processing. *arXiv preprint arXiv:1807.03520*, 2018. 1, 2, 7, 8

[15] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, 2016. 7

[16] B. Graham and L. van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017. 2

[17] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. Atlasnet: A papier-m\^ ach\'e approach to learning 3d surface generation. *arXiv preprint arXiv:1802.05384*, 2018. 1, 2

[18] C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3d object reconstruction. In *3D Vision (3DV), 2017 International Conference on*, pages 412–420. IEEE, 2017. 2

[19] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017. 6

[20] D. Jack, J. K. Pontes, S. Sridharan, C. Fookes, S. Shirazi, F. Maire, and A. Eriksson. Learning free-form deformations for 3d object reconstruction. *arXiv preprint arXiv:1803.10932*, 2018. 2

[21] L. Jiang, S. Shi, X. Qi, and J. Jia. Gal: Geometric adversarial loss for single-view 3d-object reconstruction. In *European Conference on Computer Vision*, pages 820–834. Springer, Cham, 2018. 1, 2

[22] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. Learning category-specific mesh reconstruction from image collections. *arXiv preprint arXiv:1803.07549*, 2018. 2

[23] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3 d shape descriptors. In *Symposium on geometry processing*, 2003. 7

[24] R. Klokov and V. Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *ICCV*, 2017. 2

[25] C. Kong, C.-H. Lin, and S. Lucey. Using locally corresponding cad models for dense 3d reconstructions from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, 2017. 2

[26] H. Larochelle and G. E. Hinton. Learning to combine foveal glimpses with a third-order boltzmann machine. In *Advances in neural information processing systems*, pages 1243–1251, 2010. 3

[27] Y. Li, R. Bu, M. Sun, and B. Chen. Pointcnn. *arXiv preprint arXiv:1801.07791*, 2018. 2

[28] C.-H. Lin, C. Kong, and S. Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. *arXiv preprint arXiv:1706.07036*, 2017. 1, 2, 7, 8

[29] J. Masci, D. Boscaini, M. Bronstein, and P. Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 37–45, 2015. 2

[30] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 922–928. IEEE, 2015. 1, 2

[31] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016. 3, 4, 6

[32] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. *arXiv preprint arXiv:1711.10433*, 2017. 8

[33] J. K. Pontes, C. Kong, A. Eriksson, C. Fookes, S. Sridharan, and S. Lucey. Compact model representation for 3d reconstruction. *arXiv preprint arXiv:1707.07360*, 2017. 2

[34] J. K. Pontes, C. Kong, S. Sridharan, S. Lucey, A. Eriksson, and C. Fookes. Image2mesh: A learning framework for single image 3d reconstruction. *arXiv preprint arXiv:1711.10669*, 2017. 2

[35] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. *arXiv preprint arXiv:1711.08488*, 2017. 2

[36] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CVPR*, 2017. 2, 4, 6, 7

[37] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017. 2

[38] P. Ramachandran, T. L. Paine, P. Khorrami, M. Babaeizadeh, S. Chang, Y. Zhang, M. A. Hasegawa-Johnson, R. H. Campbell, and T. S. Huang. Fast generation for convolutional autoregressive models. *arXiv preprint arXiv:1704.06001*, 2017. 8

[39] G. Riegler, A. O. Ulusoy, and A. Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 3, 2017. 2

[40] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017. 3

[41] S. Sengupta, A. Kanazawa, C. D. Castillo, and D. Jacobs. Sfsnet: Learning shape, reflectance and illuminance of faces in the wild. *arXiv preprint arXiv:1712.01261*, 2, 2017. 2

[42] A. Sharma, O. Grau, and M. Fritz. Vconv-dae: Deep volumetric shape learning without object labels. In *ECCV*, 2016. 7

[43] Y. Shen, C. Feng, Y. Yang, and D. Tian. Neighbors do help: Deeply exploiting local structures of point clouds. *arXiv preprint arXiv:1712.06760*, 2017. 2

[44] J. D. Stets, Y. Sun, W. Corning, and S. W. Greenwald. Visualization and labeling of point clouds in virtual reality. In *SIGGRAPH Asia 2017 Posters*, 2017. 1

[45] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, and J. Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2530–2539, 2018. 2

[46] Y. Sun, S. N. R. Kantareddy, R. Bhattacharyya, and S. E. Sarm. X-vision: An augmented vision tool with real-time sensing ability in tagged environments. *arXiv preprint arXiv:1806.00567*, 2018. 1

[47] Y. Sun, Z. Liu, Y. Wang, and S. E. Sarma. Im2avatar: Colorful 3d reconstruction from a single image. *arXiv preprint arXiv:1804.06375*, 2018. 2

[48] Q. Tan, L. Gao, Y.-K. Lai, and S. Xia. Variational autoencoders for deforming 3d mesh models. *arXiv preprint arXiv:1709.04307*, 2017. 2

[49] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *ICCV*, 2017. 2

[50] G. Te, W. Hu, Z. Guo, and A. Zheng. Rgcnn: Regularized graph cnn for point cloud segmentation. *arXiv preprint arXiv:1806.02952*, 2018. 2

[51] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. In *SSW*, 2016. 3, 6

[52] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelcnn decoders. In *NIPS*, 2016. 3

[53] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. 3

[54] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. *arXiv preprint arXiv:1804.01654*, 2018. 2

[55] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):72, 2017. 2

[56] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 2, 4

[57] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016. 1, 2, 6, 7

[58] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 1, 2, 5

[59] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *WACV*, 2014. 5

[60] Y. Yang, C. Feng, Y. Shen, and D. Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 3, 2018. 2

[61] L. Yi, H. Su, X. Guo, and L. J. Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *CVPR*, pages 6584–6592, 2017. 2

[62] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018. 1, 2

[63] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert. Pcn: Point completion network. In *2018 International Conference on 3D Vision (3DV)*, pages 728–737. IEEE, 2018. 1, 2

[64] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018. 3, 6