TopNet: Structural Point Cloud Decoder

http://completion3d.stanford.edu

Lyne P. Tchapmi[1] Vineet Kosaraju[1] S. Hamid Rezatofighi[1,2] Ian Reid[2] Silvio Savarese[1]

[1]Stanford University, [2]The University of Adelaide, Australia

Abstract

3D point cloud generation is of great use for 3D scene modeling and understanding. Real-world 3D object point clouds can be properly described by a collection of low-level and high-level structures such as surfaces, geometric primitives, semantic parts, etc. *In fact, there exist many different representations of a 3D object point cloud as a set of point groups. Existing frameworks for point cloud generation either do not consider structure in their proposed solutions, or assume and enforce a specific structure/topology,* e.g. *a collection of manifolds or surfaces, for the generated point cloud of a 3D object. In this work, we propose a novel decoder that generates a structured point cloud without assuming any specific structure or topology on the underlying point set. Our decoder is softly constrained to generate a point cloud following a hierarchical rooted tree structure. We show that given enough capacity and allowing for redundancies, the proposed decoder is very flexible and able to learn any arbitrary grouping of points including any topology on the point set. We evaluate our decoder on the task of point cloud generation for 3D point cloud shape completion. Combined with encoders from existing frameworks, we show that our proposed decoder significantly outperforms state-of-the-art 3D point cloud completion methods on the Shapenet dataset.*

1. Introduction

Generating 3D point clouds using neural networks is increasingly studied for various applications such as 3D reconstruction [9, 13, 37], object point cloud completion [38, 13], and representation learning for point clouds [37, 1, 13]. This work focuses on the common sub-task of producing a complete 3D point cloud shape, given a feature vector representing the shape. The input feature can originate from various inputs (such as images [9], 2.5D sketches [35], point clouds [23], etc.), but the scope of evaluation in this work focuses on the task of 3D object shape completion, where the input is a partial 3D point cloud and the desired output is the ground-truth completed point cloud. The ability to infer the full shape of an object from an incomplete scan acquired by depth camera or LiDAR is indeed an important task which enables several downstream applications such as robotics manipulation [31], scene understanding for navigation [7], and virtual manipulation of completed shapes [8].

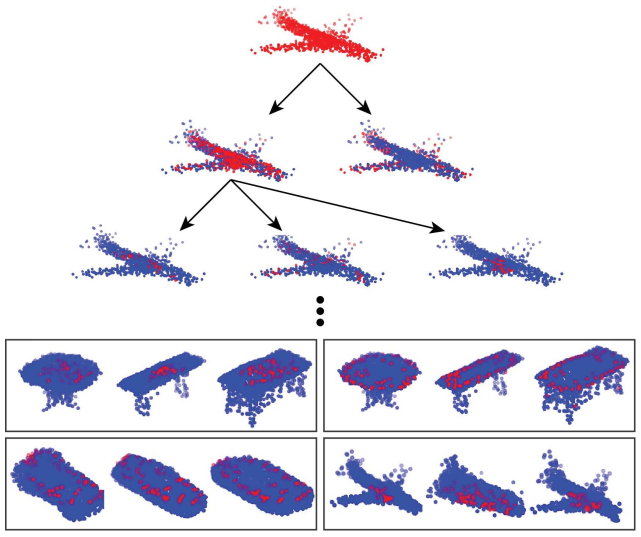


Figure 1: Our proposed decoder generates point clouds according to a tree structure where each node of the tree represents a subset of the point cloud. The embedded point cloud structure is shown by visualizing nodes in the decoder, as a collection of all its descendant points. We show selected point grouping patterns emerging from our structural decoder for several object classes.

Although moderate success has been achieved with performing shape completion on regular representations of 3D objects such as distance fields, meshes, and voxel grids [3, 6, 22, 28, 16, 14], these methods fall short due to inefficiency of such representations. Recent progresses in developing highly efficient and powerful point cloud encoders [23, 21, 24, 20, 18], have made point clouds a highly promising representation for 3D object generation and completion using neural networks. Several state-of-the-art approaches applicable to shape completion focus on point-cloud based shape completion [38, 13, 37]. The dominant paradigm in these frameworks is to use a point cloud encoder to embed the partial point cloud input [21], and design a decoder to generate a completed point cloud from the embedding of the encoder.

The key focus in most existing point cloud generation approaches is on the representation of 2D or 3D objects and also designation of a relevant point cloud based decoder for the proposed representation. The earliest method uses a parallel multilayer perceptron network and a deconvolution network to decode the latent feature generated by a 2D encoder [9] while using a permutation invariant loss such as the earth mover's ditance [26, 9] or Chamfer loss [9] to deal with the orderless nature of point clouds. However, this framework does not explicitly consider any topology or structure that naturally exists in real-world 3D objects. The recent successful approaches concentrate on the designation of decoders which generate structured point clouds [38, 13, 37]. For example in [13], it is assumed the point clouds of a 3D object lie on a 2-manifold, formally defined as a topological space that locally resembles the Euclidean plane. Therefore, the proposed decoder is enforced to learn a set of mappings or deformations from the euclidean plane to the object point cloud. Imposing these structures into the learning process may result in superior performance in generating 3D object point clouds compared to the approaches ignoring structure. However what is usually ignored is the potential impact that a specific representation for grouping the point clouds may have on a learning process that uses an unstructured (*i.e.* permutation invariant) loss. Enforcing a single specific structure during learning may not be optimal for training as the space of possible solutions is constrained.

To address this issue, we propose a more general decoder that can generate structured point clouds by implicitly modeling the point cloud structure in a rooted tree architecture. We show that given enough capacity and allowing for redundancies, a rooted tree allows us to model structure, including any topology on the point set, making our decoder a more general structured decoder. Since structure is only implicitly modeled in our decoder, our model is not bound to a pre-determined structure, and therefore can embed arbitrary structure and/or topologies on the point set, as in Fig. 1.

More specifically for the shape completion task, we embed the partial input point cloud as a feature vector or code which is used by our proposed decoder to generate a completed point cloud. Our decoder has a rooted tree structure in which the root node embeds the entire point set as the encoder feature vector, the leaf nodes are individual points in the point set, and each internal node embeds a subset of the entire point cloud made of all its descendant leaves. The set of point cloud subsets represented by the tree nodes defines the generated point cloud structure or topology. This model choice is inspired by the formal definition of topology on finite discrete point sets as detailed in Section 4.

We evaluate our proposed decoder on the Shapenet dataset and show a 34% relative improvement over the next-best performing methods for the task of point cloud shape completion[1]. Visualizations of the nodes of our tree decoder reveals various non-identical patterns learned by our decoder.

The main contributions of the paper are summarized as follows:

- We propose a novel way to model arbitrary structure/topology on a point cloud using a rooted tree in which each node embeds a defining element of the structure.
- We design a novel tree decoder network for point cloud generation and completion which generates arbitrarily structured point clouds without explicitly enforcing a specific structure.
- We show an intuitive visualization of the structure learned by our decoder by visualizing a node in the tree decoder as a set of all its descendants.
- Finally, our network sets a new state-of-the-art on object point cloud completion by more than 30% improvement over the next-best performing approach.

2. Related Works

Our work follows a long line of frameworks on shape completion which leverage various representations. We review a subset of these approaches among those leveraging neural networks.

Volumetric 3D shape completion: Earlier and some recent works on shape completion leveraging neural networks favored voxel grids, and distance fields representations [6, 15, 29, 27, 18] which are well suited for processing with 3D convolutional neural networks. These works have shown great success in the tasks of 3D Reconstruction [5, 11], shape completion [6, 14, 36, 19], and shape generation from embeddings [3]. However voxel grids require large footprints and early works operate on low dimensional grids. This issue has been addressed using sparse representations [30, 32, 33, 17, 12, 25] which makes processing voxels more efficient. However the process of voxelization still introduces a quantization effect which discards some details of the data [34] and is not suitable for representing fine-grained information. To avoid this limitation, recent works generate 3D shapes in point cloud space which is naturally able to represent fine-grained details, and

[1]The code for this project and an associated object point cloud completion benchmark with all evaluated methods are available at http://completion3d.stanford.edu.

several of these works show superior performance to voxel-based methods [38, 9, 10].

Multiresolution Point Cloud Generation with Neural Networks: A few works on point cloud generation consider or introduce a multi-resolution structure in the process of point cloud generation. Yuan *et al*. [38] generate point clouds in 2 stages where the first stage is a lower resolution point cloud and the 2nd stage is the final output. Gadelha *et al*. [10] represents a point cloud as a 1D list of spatially ordered points, and generates a point cloud through a tree network in which each branch aims at representing different resolutions of the point cloud which are connected through multiresolution convolution. These works while highly performant, constrain the network to focus on the multiresolution structure in the point cloud.

Unstructured Point Cloud Generation with Neural Networks: Fan *et al*. [9] proposed one of the earliest works in the literature addressing the task of generating 3D point clouds from an input image. They proposed an architecture made of an encoder which encodes the input into a embedding, and a decoder which generates the point cloud from the embedding. The decoder they propose is a 2 branch architecture, one multilayer perceptron (MLP) branch and one deconvolutional branch. They also introduce the Chamfer loss, a differentiable loss for point cloud comparison which we leverage in our work. However, the output generated by their method is an unstructured point cloud, while real-world object point clouds are structured and can be represented as a collection of subsets, *e.g*. surfaces and parts. Recent approaches based on imposing one of the structured representations, have shown to be superior in generating point clouds of real-world objects. We review them next.

Manifold-based Point Cloud Generation with Neural Networks: Several state-of-the-art methods on point cloud generation and completion generate structured point clouds by assuming and enforcing a 2-manifold topology on the point cloud. Groueix *et al*. [13] design a decoder that learns a manifold by learning a mapping from the Euclidean plane to the ground-truth point cloud. Yang *et al*. [37] and Yuan *et al*. [38] also learn to generate a point cloud structured as a manifold through a series of deformation (folding) operations on the Euclidean plane. While several point clouds are indeed derived from sampling manifolds, they exhibit several other structures or topological representations that can be leveraged during training. Therefore enforcing a specific structured representation may constrain the learning process by limiting the solution search space. To avoid this limitation, we propose a decoder which is able to represent arbitrary structures and topologies on a point set. The decoder has a rooted tree structure in which each node of the tree represents and generates a subset of the point cloud defining the point cloud structure. Unlike [13, 37, 38] which specifically enforce their decoder to generate a manifold, we do not enforce our decoder to generate any specific topology which increases the space of potential topologies that can be generated by the decoder. Visualization of structural patterns learned by our decoder suggests that the decoder learns patterns which are not necessarily traditional 2-manifolds but occur across several objects.

3. Background and Motivation

We first provide some background on concepts relating to object structure and topology.

2-manifolds and surface topology: Object structure is commonly modeled as a surface or 2-manifold which formally is a topological space that locally resembles the Euclidean plane. This implies that 2-manifolds have a local smoothness property. Previous works have attempted to explicitly enforce this property by learning mappings from smooth 2D grids to local surfaces of 3D objects[13, 37, 38]. While enforcing local smoothness may be helpful for learning explicitly smooth representations such as meshes, this assumption may be less relevant for point clouds due to their discrete nature which allows for various potentially more suitable non-smooth representations.

Topology on discrete point sets: Unlike 2-manifold representations, we do not leverage the local smoothness assumption due to the discrete nature of point clouds. Instead we leverage one of the multiple more general equivalent definitions of topological space on finite discrete point sets which for a set $\mathcal{S}$ is defined as a nonempty collection of subsets of $\mathcal{S}$ [2](see Section 4). Note that this definition of topology is significantly less constrained than 2-manifolds and does not impose restrictions on smoothness or point neighborhoods within the topology. Leveraging this definition allows us to design a point cloud decoder which is less constrained than previous topological point cloud decoders. Indeed, we simply design a decoder which is able to group the point cloud into subsets defining the point cloud structure. The intuition behind designing a decoder that groups a point cloud into subsets is that if a topology - defined as a collection of nonempty subsets is adequate for the learning process, then the decoder has the ability to generate the point cloud according to that topology by adequately grouping points.

Designing Topological Decoders: Given the more general definition of topology on discrete finite point sets as a collection of subsets, how can we design a decoder capable of generating collections of subsets for a set $\mathcal{S}$? One straightforward approach is to train N multilayer perceptrons to generate separate point cloud subsets and merge them into the final point cloud. This method trivially scales poorly the larger the size of the topology or structure defined on the point set. We build and improve on this basic idea and instead propose a decoder modeled as a hierarchical rooted tree in which each node of the tree represents a subset of the point cloud and the root of the tree represents the entire point cloud (Fig. 1). This rooted tree architecture has several appealing properties including its ability to represent arbitrary topologies and structure on discrete point sets. The hierarchical nature of the decoder is also a more efficient and compact representation since parts of the neural network are shared in generating overlapping subsets of the point cloud (see Section 5). Next, we present the theoretical foundation of our work starting with a formal definition of topology on discrete point sets, and we show that our proposed rooted tree structure is adequate to represent several point cloud structures including any arbitrary topology.

4. Topology Representation

In this work, we leverage a general definition of topology on point sets and propose a decoder that generates a structured point cloud. To this end, we first provide a *general definition* of a topological space on point sets. There exists several equivalent definitions of topological spaces, and for our purposes, we use the following [2]:

Definition: Assume $\mathcal{S} = \{s_1, s_2, \cdots, s_n\}$ is a set of points, where $s_i \in \mathbb{R}^d$ is a point in d dimensional space, and $\mathbb{T} = \{\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_k\}$ is a **collection of open subsets of** $\mathcal{S}$. Then, $\mathbb{T}$ is a topology on $\mathcal{S}$ if:

1. The empty set $\emptyset$ and $\mathcal{S}$ are open,
2. The intersection of any finite number of subsets of $\mathbb{T}$ is in $\mathbb{T}$, *i.e.* $\cap_{\forall i}\mathcal{S}_i \in \mathbb{T}$,
3. The union of an arbitrary number of subsets of $\mathbb{T}$ is in $\mathbb{T}$, *i.e.* $\cup_{\forall i}\mathcal{S}_i \in \mathbb{T}$,

Any finite point set can be defined as open and throughout this work we use that assumption, such that $\emptyset$, $\mathcal{S}$, and all subsets of $\mathcal{S}$ are open. From the above definition, it becomes apparent that topology on a finite discrete point set can be generally conceived as a collection of subsets of the point set. Therefore a decoder modeling topology must be able to generate or model groups of points in order to represent a topology on the point set. Among several possibilities to accomplish the task of generating a point cloud as a group of points, we choose to use a rooted tree topology for two main reasons. The first reason for our choice is that assuming enough capacity and allowing for redundancies, any topology $\mathbb{T}$ on a point set $\mathcal{S}$ can be represented as a rooted tree as shown in Proposition 1. The second reason for our choice is that any rooted tree with at least three non leaf nodes can embed at least 2 topologies (Proposition 2) meaning this representation regardless of capacity, (as long as there are at least 3 non leaf nodes) can quantitatively encode more topologies than previous works in which a single pre-determined topology is assumed. We now go into the details and proof of the two propositions above.

Proposition 1: Any topology $\mathbb{T}$ on a point set $\mathcal{S}$ can be modeled as a rooted tree structure G in which:

– Every leaf of G represents a singleton $\{s\}$ where s is an individual point $s \in \mathcal{S}$ (**P1a**)
– Each non-leaf node $G(\mathcal{S}_i)$ in the tree represents a non-empty element $\mathcal{S}_i \in \mathbb{T}$ (**P1b**)
– For any pair of nodes $G(\mathcal{S}_i)$, $G(\mathcal{S}_j)$ in G, if $G(\mathcal{S}_j)$ is a child of $G(\mathcal{S}_i)$, then $\mathcal{S}_i \subseteq \mathcal{S}_j$ (**P1c**)

Proof by existence: We want to show that for each topology $\mathbb{T}$ on a set $\mathcal{S}$, there exists at least one rooted tree structure G satisfying the conditions of Proposition 1. Lets define $\mathbb{T}^* = \mathbb{T} \cup \{\{s\} : s \in \mathcal{S}\}$. We create a graph G as follows:

– For each individual point $s \in \mathcal{S}$ and each non-empty set $\mathcal{S}_i \in \mathbb{T}$, we create a representative node $G(s)$, $G(\mathcal{S}_i)$ in G. (**C1a**)
– For each non-empty $\mathcal{S}_i, \mathcal{S}_j \in \mathbb{T}^*$, if $\mathcal{S}_i \nsubseteq \mathcal{S}_j$, we add a directed edge $E(\mathcal{S}_i, \mathcal{S}_j)$ from $\mathcal{S}_i$ to $\mathcal{S}_j$ in G (**C1b**)
– Since $\forall \mathcal{S}_i \in \mathbb{T}^*/\mathcal{S}, \mathcal{S}_i \nsubseteq \mathcal{S}, E(\mathcal{S}_i, \mathcal{S})$ is an edge in G and we designate $G(\mathcal{S})$ as the root of our graph G (**C1d**)

By **C1a**, all non-empty subsets of $\mathbb{T}$ are represented by a node in G and we now show that G satisfies the conditions in Proposition 1.

P1a proof: Let G_l be a leaf in G representing a subset $\mathcal{S}_i \subseteq \mathcal{S}$. By **C1a**, $\mathcal{S}_i$ is non-empty which means $\exists s \in \mathcal{S}, s \in \mathcal{S}_i$. Therefore $s \subseteq \mathcal{S}_i$. By contradiction, lets assume $\mathcal{S}_i$ is not a singleton. By **C1b**, $E(\mathcal{S}_i, \{s\})$ is a directed edge G which means G_l is not a leaf which is a contradiction. $\mathcal{S}_i$ must therefore be a singleton.

P1b proof: This is a direct consequence of the definition in **C1a**

P1c proof: This is a direct consequence of the converse of **C1d**

Corollary 1: The set of all leaf descendants of a given node in $G(\mathcal{S}_i)$ in G is equal to $\mathcal{S}_i$. Based on this corollary, we can visualize individual nodes in a point set's topological tree by visualizing all its descendants leaves.

Proposition 2: Given a structured point cloud, representing a set of points $\mathcal{S}$, and a rooted tree graph G with at least 3 non-leaf nodes, and the properties defined in Proposition 1, there exist more than one topology that can be represented by G.

Proof: This proposition can also be proven by example. There are two trivial topologies that can be represent in a rooted tree G of at least 3 non-leaf nodes. The first of which being $\mathbb{T} = \{\emptyset, \mathcal{S}\}$ which can be represent in G by choosing the root node and every other node in the tree to represent $\mathcal{S}$ (our representation allows for duplicate nodes). The empty set is represented implicitly. The second trivial topology that can be represented by G is $\mathbb{T} = \{\emptyset, \mathcal{S}, \mathcal{S}_1, \mathcal{S}_2\}$

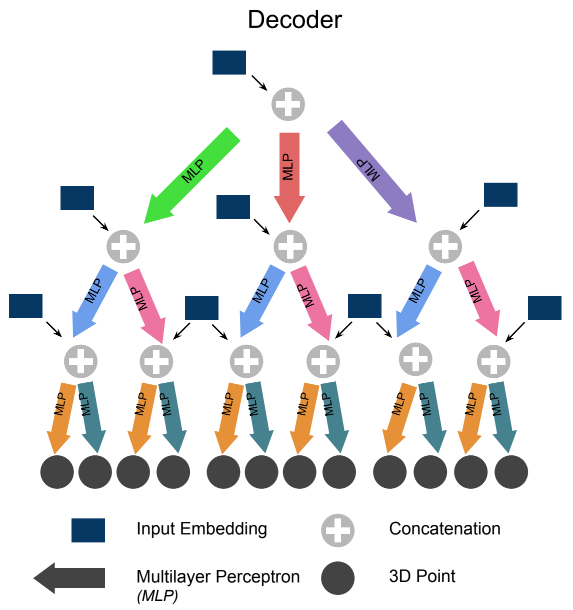


Figure 2: **Model Architecture:** Our point completion framework comprises a 2-stage point cloud encoder, and a tree-structured decoder. The arrows of the decoder are multilayer perceptron networks (MLP). Similarly colored MLP share the same parameters.

where $\mathcal{S}_2$ is the complement of $\mathcal{S}_1$ in $\mathcal{S}_1$ *i.e.* $\mathcal{S}_2 = \mathcal{S}/\mathcal{S}_1$. This topology can be represented by G by assigning the root node to represent $\mathcal{S}$ and the first 2 children nodes of G to represent $\mathcal{S}_1$ and $\mathcal{S}_2$, concluding our proof.

In summary, our propositions show that it is possible to represent structure/topology on a discrete and finite point set using a rooted tree in which every node of the tree represents an element of the topology and the edges of tree represent a subset relationship from child to parent. While this representation of topology is not the only way to represent a topology, we find it to be suitable as a basis for designing a point cloud decoder which is flexible enough to represent various structures and topology, but not too constrained as to be forced to generate a specific or any topology. The design of such a decoder and our general framework for point cloud generation and completion are presented next.

5. Structured Point Set Generation

Given that general structure and topology on discrete and finite point sets can be represented as a rooted tree, we present a novel decoder with a rooted tree structure which takes as input an embedding representing a 3D shape and generates the corresponding point cloud (Fig. 2). Our proposed decoder generates point clouds according to a tree structure where each node of the tree represents a subset of the point cloud (Fig. 1). The decoder is trained using the Chamfer distance as loss [9], within a framework that includes an encoder as a first stage.

5.1. Design of Rooted Tree Decoder

Our proposed decoder architecture has a rooted tree structure in which the root node embeds and processes the global point cloud embedding representing a 3D shape. A collection of multilayer perceptrons arranged in a tree structure are used to learn embeddings of child nodes from parent node embeddings concatenated with the global embedding at each node. The decoder architecture is illustrated in Fig. 2. Our decoder's architecture features a root node N_0 which takes the feature vector from the encoder and uses M_1 MLPs to generate M_1 feature vectors of dimension C corresponding to M_1 children node at level 1 of the tree. Subsequently , the feature vector of each node at level $i \geq 1$ of the tree is concatenated with the global feature produced by the encoder and further processed by M_{i+1} MLPs to produce M_{i+1} children features per node for the next level $i+1$. All nodes at a given level i are processed by the same M_i shared MLPs. At the last level of the tree, the feature vectors generated for each leaf node have dimension $C = 3$, and represent the individual points in the output point cloud.

Design analysis: In Section 4, we demonstrated that any topology $\mathbb{T}$ on a set $\mathcal{S}$ can be embedded in a rooted tree in which each node of the tree represents a non-empty subset of $\mathbb{T}$ or a singleton of $\mathcal{S}$. We proved this hypothesis for any arbitrary rooted tree. However, the decoder D which we propose here, is a specific rooted tree structure with the following features:

- Every node in D has at most one parent
- All nodes at the same level have the same number of children
- All leaves are at the same level.

While G is not guaranteed to meet the conditions above, it can be transformed into a new equivalent tree G' for which the conditions above are met by performing a series of node duplication on G. We provide more details in the supplementary materials. Our decoder in this case can be seen as modeling G' rather than G which implies potential duplication of nodes and points. This is acceptable since duplication of points in a point set does not change the point set overall.

5.2. Point Cloud Loss

The point cloud generated by our decoder needs to be compared against the ground-truth for learning. An ideal loss must be differentiable and invariant to the permutation of point clouds in both target $\mathcal{S}$ and ground-truth $\mathcal{S}_G$. The Chamfer distance between $\mathcal{S}, \mathcal{S}_G \subseteq \mathbb{R}^3$ proposed by Fan *et* *al.* [9] meets these two requirements and is defined as:

$$d_{CD}(\mathcal{S}, \mathcal{S}_G) = \sum_{x \in \mathcal{S}} \min_{y \in \mathcal{S}_G} \|x - y\|^2 + \sum_{y \in \mathcal{S}_G} \min_{x \in \mathcal{S}} \|x - y\|^2 \tag{1}$$

For each point, the Chamfer distance finds the nearest neighbor in the other set and computes their squared distances which are summed over both target and ground-truth sets.

6. Experiments

We now evaluate our proposed decoder both quantitatively and qualitatively on the task of 3D point cloud shape completion. A given partial input point cloud is processed through an existing point cloud encoder and the resulting embedding is processed by our decoder to generate a point cloud. The encoder used in our final model is the 2-stage PointNet-based encoder from Yuan *et al.* [38].

Dataset: We evaluate our dataset on a subset of the Shapenet dataset [4] derived from the dataset in Yuan *et al.* [38] The ground-truth in [38] was generated by uniformly sampling 16384 points on the mesh surfaces and the partial point clouds were generated by back-projecting 2.5D depth images into 3D. In our experiments we use N = 2048 for both input and ground-truth point clouds which are obtained by random subsampling/oversampling of the clouds in [38]. We keep the same train/test split as [38].

Implementation Details: Our final decoder has $L = 6$ levels and each MLP in the decoder tree generates a small node feature embedding of size $F = 8$. When generating $N = 2048$ points, the root node has 4 children and all other internal nodes in subsequent level generate 8 children, yielding a total of $N = 4 \times (8^3) = 2048$ points generated by the decoder. Each MLP in the decoder is a has 3 stages with 256, 64, and C channels respectively, where $C = 8$ for inner nodes and $C = 3$ for leaf nodes.

Training Setup: We train all models for 300 epochs, with a batch size of 32, a learning rate of 1e-2 or 5e-3 depending on stability, and Adagrad optimizer. The best model is chosen based on the validation set.

Evaluation: We evaluate our model across 8 classes from the Shapenet dataset, against state-of-the-art methods on point cloud completion. For each class, the Chamfer Distance is computed (averaged over the number of class instances). Our final metric is the mean Chamfer Distance averaged across classes. In addition, we train a fully connected decoder baseline with 4 layers of output dimensions 256, 512, 1024, and $3 \times N$. For $N = 2048$, the results of the evaluation of our method against state-of-the-art methods are shown in Table 1, with qualitative results in Figure 4. Our method significantly outperforms existing methods across all classes, and shows a 33.9% relative improvement over the next best method.

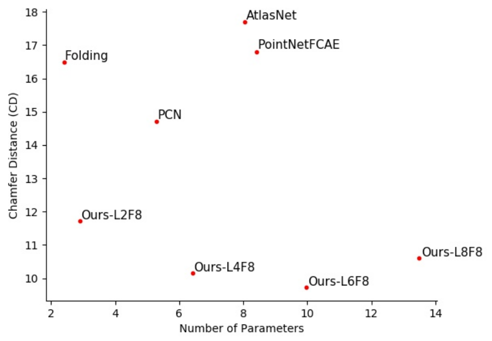


Figure 3: **Number of Parameters:** We analyze the performance of our networks instantiations as a function of its number of parameters. Across all instantiations, our network outperforms previous works. A local minimum seems to emerge from this plot. Note the Chamfer distance is reported multiplied by 10^4.

6.1. Encoder analysis

Methods proposed in previous works use a variety of encoders. The point cloud generated is dependent not only on the decoder but also on the feature generated by the encoder. In this experiment we analyze the effect of encoder choice on the performance of our decoder and that of previous decoder. The results of this analysis are tabulated in Table 2. The first encoder (A) used in previous works [13] and for our fully connected baseline is a PointNet[13]. The second encoder is proposed in [38] in which it demonstates better performance than PointNet++ [24]. We show results with these two encoders and compare against methods using each respectively. Regardless of the encoder used, our method outperforms existing methods. There is a noticeable gap in performance depending on the encoder chosen, but comparing across methods using the same encoder, our networks still shows a significant performance improvement.

6.2. Ablation studies

Design choices involved in our decoder include choosing the number of features F generated for each node embedding and the number of tree levels L. We analyze the effect of these parameters for an output cloud size $N = 2048$ by varying F in $\{8, 16, 32, 64\}$, and L in $\{2, 4, 6, 8\}$. This ablations study was used to pick the model's final number of layers and number of features. One important thing to note is that since the number of output points is fixed at 2048 in this experiment, increasing the number of levels requires decreasing the number of children per level. This operation is therefore not similar to adding a new layer in conventional networks and a deeper tree may not necessarily improve performance.

In Figure 5a, we plot the Chamfer distance as a function of the number of levels L for different values of F. For

Method / Eval.	Chamfer Distance (CD)								
	Plane	Cabinet	Car	Chair	Lamp	Couch	Table	Watercraft	Average
AtlasNet [13]	10.37	23.40	13.41	24.16	20.24	20.82	17.52	11.62	17.69
PointNetFCAE [base.]	10.31	19.07	11.83	24.69	20.31	20.09	17.57	10.50	16.80
Folding [37]	11.18	20.15	13.25	21.48	18.19	19.09	17.80	10.69	16.48
PCN [38]	8.09	18.32	10.53	19.33	18.52	16.44	16.34	10.21	14.72
Ours	**5.50**	**12.02**	**8.90**	**12.56**	**9.54**	**12.20**	**9.57**	**7.51**	**9.72**

Table 1: **Point Cloud Completion Results on ShapeNet**: Comparison of our approach against previous works for the resolution ($N \sim= 2048$). The Chamfer distance is reported multiplied by 10^4.

Method	AtlasNet-A[13]	PointNetFC-A[base.]	Folding-B[37]	PCN-B[38]	**Ours(A)**	**Ours(B)**
CD	17.69	16.80	16.48	14.72	12.97	**9.72**

Table 2: **Encoder analysis**: Comparison of our approach against previous works using different encoders. The Chamfer distance is reported multiplied by 10^4. Encoder A is a 1-stage PointNet based encoder, and encoder B is a 2-stage PointNet encoder. Our method outperforms all other methods with their respective encoders, with encoder B giving a better performance than encoder A.

$F \in \{8, 32, 64\}$ the graphs exhibit different local minima but for $F = 6$, the performance oscillates around an average value. In Figure 5b, we plot the Chamfer distance as a function of the number of features F for different values of L). The graphs for $L \in \{4, 6, 8\}$ exhibit slightly similar trend though the pattern is non-convex. The graph for $L = 2$ exhibits a very different pattern compare to the others. In all experiments, regardless of the value for L and F above, our method outperforms all previous method.

6.3. Number of parameters

The number of parameters used by each method can influence performance so we analyze the performance of our method as a function of the number of parameters for $F = 8$ and $L \in \{2, 4, 6, 8\}$. This analysis in shown in Figure 3 where the Chamfer distance is plotted as a function of the number of network parameters. Our model varies in the number of network parameters depending on design choice with some instantiations of our method having fewer or more parameters than previous methods. Regardless of the number of parameters in experiments, our network shows superior performance, which suggests that the number of parameters is not the primary reason for our performance.

6.4. Analysis of point cloud resolution

Previous works have found that the shape of most objects in Shapenet can be summarized by as few as $N = 1024$ points[23]. But for other applications such as graphics in which the ability to get accurate normals is key, generating dense accurate point clouds can be useful. We therefore analyze the performance of our network as the number of output points scales upward. We compare performance at $N = 2048, 4096, 8192, 16384$. Since the computation of the Chamfer loss scales quadratically with the number of points, we keep the ground-truth at 2048 points and only increase the output size: quantitative results are shown in Table 3. We notice that our network's performance increases with the number of output points albeit the absolute improvement is less than that of PCN[38]. As a percentage, the improvement is still significant, ranging from 9.16% to 18.46% for our method while PCN's improvement ranges from -2.61% to 20.26%. In all resolutions, our method still shows superior absolute performance.

7. Discussion and limitations

The decoder proposed in our work aims at generating structured point clouds by generating a point cloud as a collection of its subsets. A constraint of our decoder is that the tree needs to be sufficiently large: for instance, representing a topology of S whose elements are all subsets of S requires our decoder to have at least $2^{|S|}+1$ nodes, which is intractable. The capacity of the decoder therefore can limit the possible topologies learned. The proposed architecture is still a promising step towards incorporating structure in decoders as unlike previous works, we generate a structured point cloud without explicitly enforcing a specific structure which allows the network to learn arbitrary topologies.

Visualizing learned structure: By design, each node in our decoder generates a subset of S made of all its descendant leaves. We can visualize learned structure by plotting each node's descendant leaves. In Fig. 1 and the supplementary material we visualize several decoder nodes. We notice that several geometric clustering patterns emerge. Some clusterings seem geometric while others appear random but are consistent across.This can be seen as a consequence of our adopting the more general definition of topology which does not enforce the generated clustering to be smooth.

Usefulness & Redundancies: One interesting future study

Resolution (# points)	Method				
	AtlasNet [13]	PointNetFCAE [base.]	Folding [37]	PCN [38]	**Ours**
2048	17.69	16.8	16.48	14.72	**9.72**
4096	16.12	14.79	13.19	11.88	**8.83**
8192	15.32	12.57	12.95	12.19	**7.20**
16384	14.85	10.61	12.26	9.72	**6.50**

Table 3: **Resolution analysis**: Comparison of our approach against the best performing method (PCN) with varying point cloud resolution. We train each method to generate N = 2048, 4096 points. The Chamfer distance is reported, multiplied by 10^4. Our method outperforms PCN for all different resolutions, but the performance gap between the methods is higher for low-resolution point clouds.

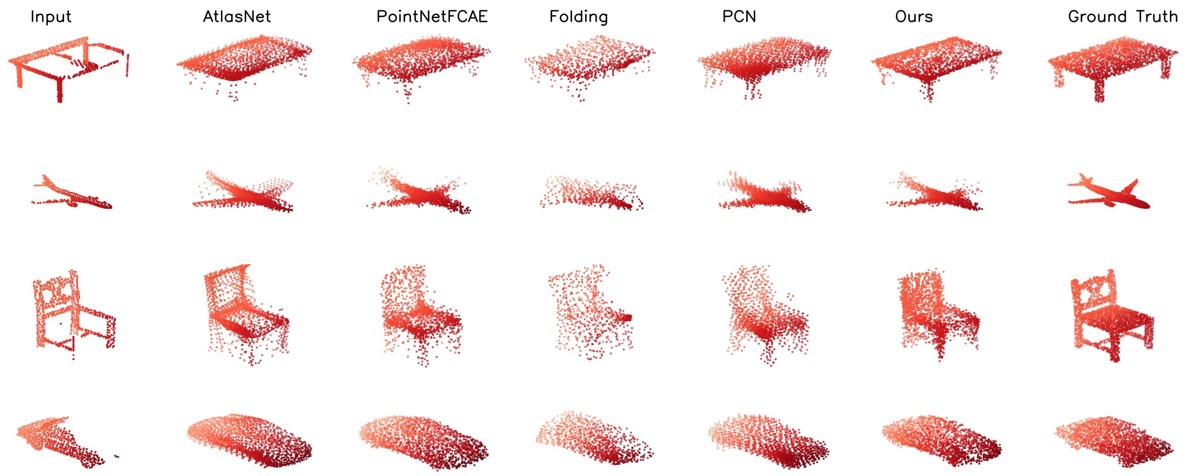
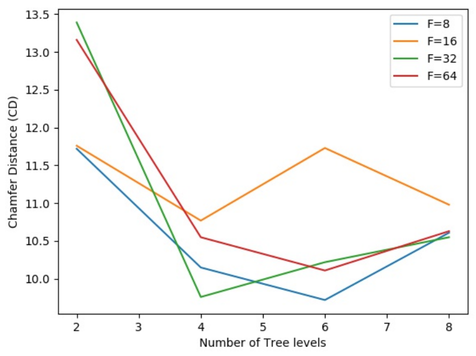
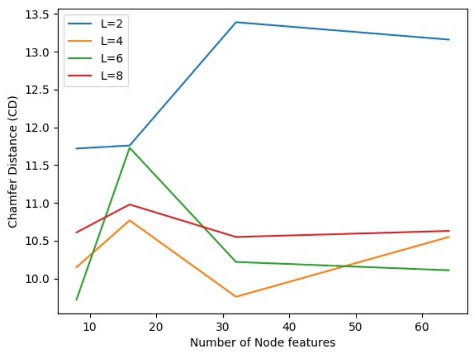


Figure 4: Point Cloud Completion Results. A partial point cloud is given as input and our method generates a completed point cloud.



(a)



(b)

Figure 5: **Ablation Experiments:** We analyze the effect of varying different parameters in our network. We vary the number of node features {8, 16, 32, 64}, and the number of tree levels {2, 4, 6, 8}, while keeping the number of outputs constants. For instance when the number of levels L=2, the number of children per level is 32-64. When L=4, the number of children per level is 4, 4, 4, 8. All instantiations of our method outperform previous works. The number of levels seem to suggest a local minimum, but the number of features does not show a noticeable pattern. The Chamfer distance is reported multiplied by 10^4.

would be to explore if the learned structure embeddings have value as representations in classification. Another future improvement of this work could be to propose a way to generate structured point clouds without redundancies.

Multiple structures: While our proposed model can embed arbitrary topologies, it is only able to embed a single topology at test time. One avenue of exploration would be to combine several such decoders and evaluate whether they all converge to the same topology for S or whether each decoder learns to embed disjoint subsets of S.

References

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

[2] M. Armstrong. *Basic Topology*. Undergraduate Texts in Mathematics. Springer, 1990. 3, 4

[3] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. Generative and discriminative voxel modeling with convolutional neural networks. *CoRR*, abs/1608.04236, 2016. 1, 2

[4] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015. 6

[5] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 2

[6] A. Dai, C. R. Qi, and M. Niener. Shape completion using 3D-encoder-predictor CNNs and shape synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2

[7] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2018. 1

[8] R. F. de Figueiredo, P. Moreno, and A. Bernardino. Automatic object shape completion from 3d point clouds for object manipulation. In *VISIGRAPP*, 2017. 1

[9] H. Fan, H. Su, and L. Guibas. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 3, 5, 6

[10] M. Gadelha, R. Wang, and S. Maji. Multiresolution tree networks for 3d point cloud processing. In *The European Conference on Computer Vision (ECCV)*, September 2018. 3

[11] R. Girdhar, D. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, 2016. 2

[12] B. Graham, M. Engelcke, and L. van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. *CVPR*, 2018. 2

[13] 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. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 3, 6, 7, 8

[14] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu. High-resolution shape completion using deep neural networks for global structure and local geometry inference. In *IEEE International Conference on Computer Vision (ICCV)*, October 2017. 1, 2

[15] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu. High-Resolution Shape Completion Using Deep Neural Networks for Global Structure and Local Geometry Inference. *IEEE International Conference on Computer Vision (ICCV)*, 2017. 2

[16] C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3d object reconstruction. In *3DV*, 2017. 1

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

[18] T. Le and Y. Duan. Pointgrid: A deep network for 3d shape understanding. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2

[19] D. Li, T. Shao, H. Wu, and K. Zhou. Shape completion from a single rgbd image. *IEEE Transactions on Visualization and Computer Graphics*, 23(7):1809–1822, July 2017. 2

[20] J. Li, B. M. Chen, and G. H. Lee. So-net: Self-organizing network for point cloud analysis. *arXiv preprint arXiv:1803.04249*, 2018. 1

[21] Y. Li, R. Bu, M. Sun, and B. Chen. Pointcnn. *CoRR*, abs/1801.07791, 2018. 1, 2

[22] O. Litany, A. Bronstein, M. Bronstein, and A. Makadia. Deformable shape completion with graph convolutional autoencoders. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1

[23] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 7

[24] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5099–5108. Curran Associates, Inc., 2017. 1, 6

[25] G. Riegler, A. O. Ulusoy, H. Bischof, and A. Geiger. Octnetfusion: Learning depth fusion from data. In *3DV*, pages 57–66. IEEE Computer Society, 2017. 2

[26] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000. 2

[27] A. Sharma, O. Grau, and M. Fritz. Vconv-dae: Deep volumetric shape learning without object labels. In G. Hua and H. Jégou, editors, *Computer Vision – ECCV 2016 Workshops*, pages 236–250, Cham, 2016. Springer International Publishing. 2

[28] A. Sinha, A. Unmesh, Q. Huang, and K. Ramani. Surfnet: Generating 3d shape surfaces using deep residual networks. In *CVPR*, pages 791–800. IEEE Computer Society, 2017. 1

[29] D. Stutz and A. Geiger. Learning 3D Shape Completion from Laser Scan Data with Weak Supervision. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[30] 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

[31] J. Varley, C. DeChant, A. Richardson, A. Nair, J. Ruales, and P. Allen. Shape completion enabled robotic grasping. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017. 1

[32] 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 (SIGGRAPH)*, 36(4), 2017. 2

[33] P.-S. Wang, C.-Y. Sun, Y. Liu, and X. Tong. Adaptive O-CNN: A Patch-based Deep Representation of 3D Shapes. *ACM Transactions on Graphics (SIGGRAPH Asia)*, 37(6), 2018. 2

[34] Z. Wang and F. Lu. Voxsegnet: Volumetric cnns for semantic part segmentation of 3d shapes. *CoRR*, abs/1809.00226, 2018. 2

[35] J. Wu, C. Zhang, X. Zhang, Z. Zhang, W. T. Freeman, and J. B. Tenenbaum. Learning shape priors for single-view 3d completion and reconstruction. *European Conference on Computer Vision (ECCV)*, 2018. 1

[36] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, June 2015. 2

[37] Y. Yang, C. Feng, Y. Shen, and D. Tian. FoldingNet: Interpretable Unsupervised Learning on 3D Point Clouds. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 3, 7, 8

[38] W. Yuan and D. Held. PCN : Point Completion Network. *International Conference on 3D Vision (3DV)*, 2018. 1, 2, 3, 6, 7, 8