Attention-based Part Assembly for 3D Volumetric Shape Modeling

Chengzhi Wu[1] Junwei Zheng[1] Julius Pfrommer[2,3] Jürgen Beyerer[1,2,3]

[1]Karlsruhe Institute of Technology, Germany [2]Fraunhofer IOSB, Germany
[3]Fraunhofer Center for Machine Learning, Germany

{chengzhi.wu, junwei.zheng}@kit.edu, {julius.pfrommer, juergen.beyerer}@iosb.fraunhofer.de

Abstract

Modeling a 3D volumetric shape as an assembly of decomposed shape parts is much more challenging, but semantically more valuable than direct reconstruction from a full shape representation. The neural network needs to implicitly learn part relations coherently, which is typically performed by dedicated network layers that can generate transformation matrices for each part. In this paper, we propose a VoxAttention network architecture for attention-based part assembly. We further propose a variant of using channel-wise part attention and show the advantages of this approach. Experimental results show that our method outperforms most state-of-the-art methods for the part relation-aware 3D shape modeling task.

1. Introduction

Modeling of 3D shapes has been a central research topic in the computer graphics domain for decades. However, although great progress has been made since the seminal work of ShapeNet [5], most existing 3D modeling methods only focus on full shape modeling and are part relation-oblivious [3, 15, 41]. Since no semantic information is used in those methods, details of the 3D shapes, especially those of small volume yet exquisite parts, are often badly generated in a blurred or shattered manner.

Recently, part-based 3D shape modeling is attracting increasing research interest since it generates better part details and gives more insight into part relations. Those methods usually consist of two steps, shape parts generation and part assembly. Generating shape parts, similar to generating a full shape, is easy to achieve with 3D generative neural networks (either AE, VAE, GAN, or their combinations). However, how to encode and learn part relations for assembling the full shape is still an open question. Current state-of-the-art methods either encode all parts together and then decode it to a full shape [36, 44], or use simple regression layers to get part transformation matrices [20]. However, they only focus on how to integrate all part feature information during the learning, but neglect to maintain their independence at the same time.

In this paper, we propose a part-based attention network to learn part relations from their features in different network layers, including the vector representation, feature maps in the generator, or even the final decoded output. The part dimension will be preserved throughout the training. The attention mechanism, which originated from Transformer [35], now is widely used in the computer vision domain and has proven its effectiveness in various frameworks including iGPT [7], ViT [11], IPT [6], etc. However, it is mostly applied to 2D data, *i.e.*, images currently. For 3D data, most attention-based works are on the point cloud data representation [13, 16, 48], 3D volumetric data have been rarely explored. Like that there are several different levels of attention on 2D data which include pixel-based (smallest input entity), patch-based (enlarged the perception field), and part-based (with semantic information), for 3D volumetric data, we can also apply the attention to levels of voxel-based, patch-based, or part-based. In our case, since we are interested in the part relations, we use part-based attention for the 3D shape modeling task.

The proposed VoxAttention framework first learns an autoencoder to encode the full shape into coherent yet mutual orthogonal part latent representations, and decode them into semantic parts. Secondly, transformation matrices for those decoded parts are learned by applying attention to their feature information in different layers to assemble them. Our experimental results demonstrate that the proposed method achieves excellent performance both qualitatively and quantitatively on the 3D shape modeling task.

Our main contributions include:

- A part-based attention neural network to learn semantic part relations for better 3d shape assembly.
- An optional channel-wise attention strategy on top of the normal part attention model for feature learning.
- An additional attention consistency loss to prevent the network from mode collapse when multiple feature layers are used for computing the part relations.

2. Related Work

2.1. Generative Networks for 3D Shape Modeling

With the development of deep learning techniques, there has been a surge of research interest in deep generative models in recent years. Since the seminal work of ShapeNet [5], which is a large 3D shape model dataset that is publicly available, lots of research has been conducted in building 3D generative models for modeling 3D shapes. Brock *et al*. [3] use a 3D VAE neural network to encode and decode 3D shapes. Wu *et al*. use a 3D-GAN [41] to generate 3D shapes from latent vectors. Additional image information has also been used in various methods. TL-network [15] first applies an autoencoder on 3D shapes, then forces the image encoder to learn a similar latent representation in between, while SwitchVAE [40] uses a switch scheme to allow both branches updates coherently. Wu *et al*. [43] decrease the uncertainty of 3D voxel grids by predicting the next best viewing angle. View images have also been used to compute an additional loss in [33]. 3D-R2N2 [10] uses a recurrent neural network to encode multi-view information for 3D shape modeling. Octree-based methods [31,38] have been proposed for more computationally efficient modeling. Apart from the volumetric data representation, 3D generative modeling methods have also been proposed on other data representations, *e.g*., point clouds [1, 30, 34, 45], surface meshes [19,37,39], or even implicit fields [9,25,29].

2.2. Assembly-based 3D Shape Modeling

Compared to generating a full 3D shape directly, generating shape semantic parts separately and assembling them is more valuable since it considers the structural decomposition of 3D shapes and the connections between shape parts. Li *et al*. [21] propose the first deep structure-aware modeling framework GRASS for 3D shapes by employing a recursive neural network. StructureNet [26] also uses a tree-based architecture to generate shape parts in sequences by extending the binary tree to N-ary tree. 3D-PRNN [50] and SCORES [49] use a similar method but only shape primitives are generated. StructureEdit [27] further provides a framework for easy editing. With additional structure landmarks, Balashova [2] adds a structure detector in combination with the generator for better shape modeling.

Apart from the above methods that are mostly structure tree-based, there are lots of methods that learn part relations directly. COALESCE [47] assembles shape components by learning to synthesize connections. G2LGAN [36] sends all the generated parts into another autoencoder to decode out a full shape. SAGNet [44] uses two branches for encoding the part and structure information separately. For methods that learn spatial transformations for the parts directly, CompoNet [32] learns the transformations from latent representations of 3D point clouds, but its output shapes

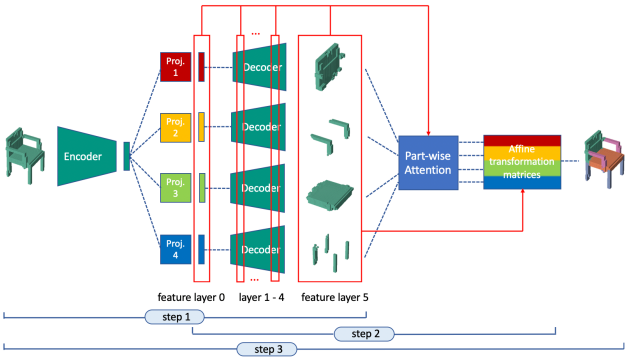


Figure 1. Pipeline of proposed VoxAttention framework. Single or multiple feature layers from the decoder are used to learn relative relations for shape parts with part-based attention.

are often disconnected. PQ-Net [42] employs a Seq2Seq generative network to reconstruct 3D shapes part by part with gated recurrent units. Dubrovina *et al*. [12] propose a decomposer-composer network to learn a factorized embedding space. A more recent work of PAGENet [20] learns part transformations only from decoded outputs with a special augmented dataset. All the above methods merge the part feature information together at a certain step. We instead propose to keep the part dimension intact and use an attention-based method to learn part relations.

2.3. Attention in Computer Vision

Attention-based methods have almost dominated the natural language processing (NLP) domain after the far-reaching model Transformer [35] was proposed. In recent years, researchers have applied the attention mechanism to computer vision tasks and achieved great success. DERT [4] applies attention to flattened feature maps that are convoluted from a CNN for object detection. To achieve better self-supervised learning on images, iGPT [7] masks some pixels out and uses an attention-based network for generative pre-training. ViT [11] cuts an input image into patches and learns over those flattened patches with a transformer encoder for classification. IPT [6] does pre-training on multiple tasks, also with a transformer encoder. A good survey on visual transformers is given in [17]. However, most current visual transformers only apply attention to the 2D image data. For 3D data, the researches that have adopted the attention mechanism are mostly on point clouds [13,16,48]. There are also works that voxelize the whole point cloud space to use an attention-based method on the voxel level for better 3D detection [18, 24]. However, to our knowledge, there are few works that apply attention directly to shapes of 3D volumetric data, especially with regard to the semantic part dimension.

3. Methodology

3.1. Overall Framework

Methods dealing with the part assembly task mostly follow a same routine which consists of two steps of generating shape parts and learning part relations [20, 36]. Our VoxAttention framework also follows this pattern but with an additional fine-tuning step added afterward as illustrated in Figure 1. An encoder-decoder architecture is used first to learn part generation. Then the latent representation, feature maps in each decoder layer, and decoded output of each part are used for part-based self-attention to learn part relative relations. After applying the learned affine transformation matrices, a full 3D shape is finally assembled. For each step, different losses are used for the training while different metrics are used to evaluate the learning performance.

3.2. Part Generator

The first step trains an autoencoder for 3D shape parts reconstruction. Although we want to obtain feature information in a part-wise manner, the information should still keep related implicitly in the latent space. Using separate autoencoders for each part as in [20] will not work since they have no information communication. The part feature information needs to be learned jointly. Here, to generate parts separately yet coherently, a shared autoencoder is used to reconstruct each part. We adopt the method in [12] to achieve the decomposition of shape latent representations. Mutual orthogonal projection matrices are learned for each part respectively. Denote N_p as the number of parts a shape category contains (*e.g.*, a chair contains four parts), the projection matrices $P_i (i = 1, \ldots, N_p)$ satisfy the properties of (i) projection property: $P_i^2 = P_i$, (ii) mutual orthogonal: $P_i P_j = 0 \ (i \neq j)$, and (iii) forming an identity matrix together: $\sum_{i=1}^{N_p} P_i = I$. A partition of the identity (PI) loss L_{PI} is used to measure the deviation of the learned projection matrices $P_1, \ldots, P_{N_p}$ from the optimal projection:

$$L_{\text{PI}} = \sum_{i=1}^{N_p} ||P_i^2 - P_i||_F^2 + \sum_{i,j=1, i\neq j}^{N_p} ||P_i P_j||_F^2 + || \sum_{i=1}^{N_p} P_i - I||_F^2 \quad (1)$$

For the part reconstruction loss L_{part}, same as [3, 40], a modified binary cross entropy loss is used by introducing a hyper-parameter γ, which weights the relative importance of false positives against false negatives.

$$L_{\text{part}} = -2(\gamma t \log(o) + (1-\gamma)(1-t)\log(1-o)) \quad (2)$$

where t is the target value in $\{0, 1\}$ and o is the output of the network in $(0, 1)$ at each output element. Adding weights to different loss terms, the total loss in step 1 is defined as:

$$L_{\text{s1}} = \omega_{\text{PI}} L_{\text{PI}} + \omega_{\text{part}} L_{\text{part}} \quad (3)$$

For evaluation, part mIoU is used as the metric for step 1.

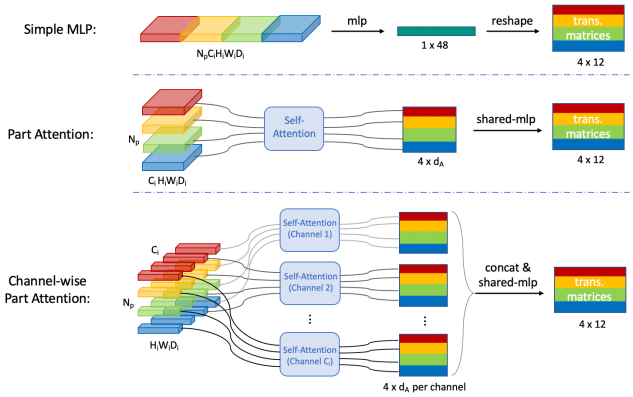


Figure 2. An illustrative comparison figure of three different methods, using an example of $N_p = 4$ and only latent features from a single ith feature layer is used as input.

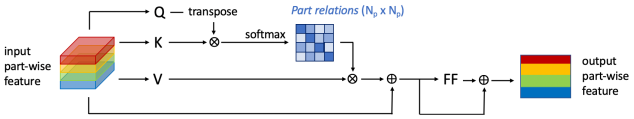


Figure 3. One attention block used for learning part relations. Q, K, and V mean query, key, and value respectively. FF means feed-forward network.

3.3. Learning Transformation Matrices

The second step is the key for part assembly. A simple idea is to apply multi-layer perceptron (MLP) directly on the decoded parts and latent representations. We instead propose a part attention-based method for learning better part relations. Figure 2 illustrates how three methods process the input features differently. d_A stands for the embedding dimension for attention blocks if they are used.

Assuming the shape has N_p parts, in the ith feature layer of the decoder, it has C_i feature channels and the feature maps are of a resolution of $H_i \times W_i \times D_i$. When this single feature layer is used in this step, ignoring the batch size dimension, the input feature is a 5-dimensional tensor of $N_p \times C_i \times H_i \times W_i \times D_i$. **Simple MLP** method reshapes the input feature into a 1-dimensional vector of $(N_p C_i H_i W_i D_i)$ and applies MLP on it directly. The output vector is then reshaped to the shape of transformation matrices. **Part Attention** method reshapes the input feature into a 2-dimensional tensor of $N_p \times (C_i H_i W_i D_i)$ and applies self-attention along the part dimension. An attention block is illustrated in Figure 3. With part relations learned during the attention operation, parts information is cross-communicated thus more comprehensive part-wise features are learned. A shared MLP is subsequently applied on all parts to learn their respective transformation matrices. **Channel-wise Part Attention** method further preserves the feature channel dimension and reshapes the input into a 3-

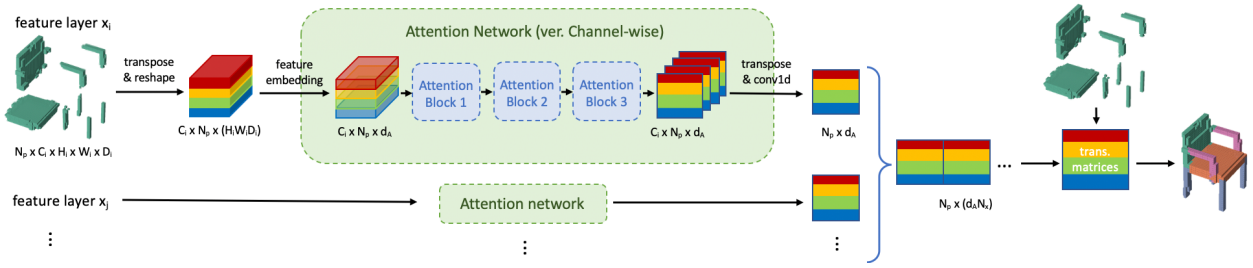


Figure 4. A detailed overview of proposed attention network architecture (channel-wise part attention version). Multiple feature layers are used as input. N_x stands for the total number of input feature layers.

dimensional tensor of $N_p \times C_i \times (H_i W_i D_i)$. For each channel, an attention block is applied to learn separate part features. All output features are subsequently concatenated and a shared MLP is applied to learn part transformation matrices.

In the actual application, since the original full feature dimension is too large to use directly, in order to reduce the feature dimension and keep it consistent for all input feature layers, we first embed the original features to the attention embedding dimension d_A before performing the attention operation. To be more specific, the input feature is embedded into $N_p \times d_A$ for the normal part attention case, while into $C_i \times N_p \times d_A$ for the channel-wise part attention case. After the embedding, three consecutive attention blocks are applied in our actual framework. Additionally, multiple feature layers can be used as input. Figure 4 gives an overview of the whole pipeline for learning transformation matrices, using an example of the channel-wise part attention version and with multiple feature layers as input. Each feature layer is firstly reshaped and embedded into a fixed-dimension vector part-and-channel-wise, followed by three consecutive attention blocks. The outputs are then concatenated and a shared MLP is applied to learn respective part transformation matrices. See more details of the network architectures in subsection 4.1.

Note that the channel-wise strategy is not applicable to all attention-based modules. Take our task as an example, if different part features are decoded from non-shared decoders, *i.e.*, multiple decoders that have a same structure but are separately trained and not parameter-shared as in [20], the channel-wise strategy is not feasible since the feature channels of a same channel index from different part decoders are irrelevant. In our case, a parameter-shared decoder is used for all different parts. This makes the channel-wise part attention feasible since part feature information is still implicitly related to each other on each feature channel in every feature layer.

In step 2, the transformation loss L_{trans} is an $L2$ loss defined between the predicted and the ground truth transformation matrix parameters, summed over all N_p parts. Additionally, if multiple feature layers are used as input for

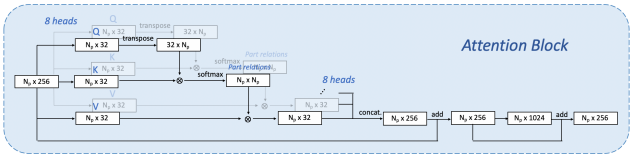


Figure 5. Detailed network design of an attention block.

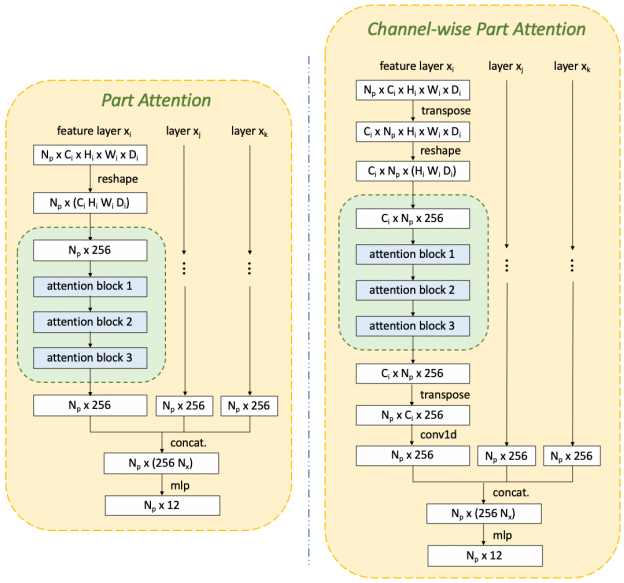


Figure 6. Detailed network design of proposed network architectures. Both part attention and channel-wise part attention models are presented. N_p stands for the number of parts of the input category. N_x stands for the total number of input feature layers.

the attention network, an optional attention consistency loss L_{AC} may be added. The attention consistency loss is defined as the $L2$ difference between the outputs of the attention network of all feature layers before concatenating them. Adding weights to different loss terms, the total loss in step 2 is defined as:

$$L_{s2} = \omega_{\text{trans}} L_{\text{trans}} + \omega_{\text{AC}} L_{\text{AC}} \tag{4}$$

For evaluation, transformation matrix MSE and full shape mIoU are used as the evaluation metrics for step 2.

3.4. Fine-tuning

Our network is a complex one and end-to-end training does not work well. Thus we divide our training into several steps like many other related works [20, 36]. In step 1, only weights in the encoder-decoder are trained. In step 2, the weights in encoder-decoder are frozen and only weights in the part-based attention network are trained. In step 3, all weights are fine-tuned with additional training.

The fine-tuning step uses all the losses above, with an additional full shape reconstruction L_{shape} loss added. L_{shape} uses a same formula as L_{part} but computes over the whole shape. To achieve better fine-tuning, weights are used for all terms. The total loss in step 3 is defined as:

$$L_{s3} = \omega_{\text{PI}} L_{\text{PI}} + \omega_{\text{part}} L_{\text{part}} + \omega_{\text{trans}} L_{\text{trans}} + \omega_{\text{AC}} L_{\text{AC}} + \omega_{\text{shape}} L_{\text{shape}} \quad (5)$$

For evaluation, all the above evaluation metrics, part mIoU, transformation matrices MSE, and shape mIoU are used.

4. Experiments

4.1. Dataset and Network Details

In our experiments, we use the ShapeNet Part dataset [46], which is a subset of the ShapeNet dataset [5] with part annotations. Shapes are firstly converted into voxel grids of resolution 32^3 using binvox [28], semantic labels are then assigned to each voxel based on part annotations. For shape part ground truth, a part dataset is generated by enlarging and centering all shape parts in the 32^3 volumetric space. The inverse transformation matrices ground truth is also computed and saved in this step. The dataset is split into the training set and the test set with a ratio of $80\%/20\%$. We use the chair category as the main demonstration category in this paper. See more results on other categories in the supplementary material.

In the autoencoder, we use convolution kernels of $4 \times 4 \times 4$ like [20, 41] to avoid unnecessary output padding for deconvolution layers. Given input 3D shapes with a resolution of 32^3, our encoder consists of four 3D convolution layers with kernel sizes of $4 \times 4 \times 4$ and strides of 2. Batch normalization and LeakyReLU layers are inserted between convolution layers. After latent representation decomposition, a shared decoder is applied to all part representations. The decoder simply mirrors the encoder except that a *Sigmoid* nonlinearity is used in the last layer. The final outputs are the decoded parts that have been enlarged and centered in the volumetric space with a resolution of 32^3.

Based on the defined decoder, we have six feature layers that may be used. We number the latent representation as feature layer 0, the decoded parts as feature layer 5, and the feature maps in each decoder layer as feature layer 1-4. For example, feature layer 3 has a feature map size of 8^3 with a channel number of 128 in our case. More details

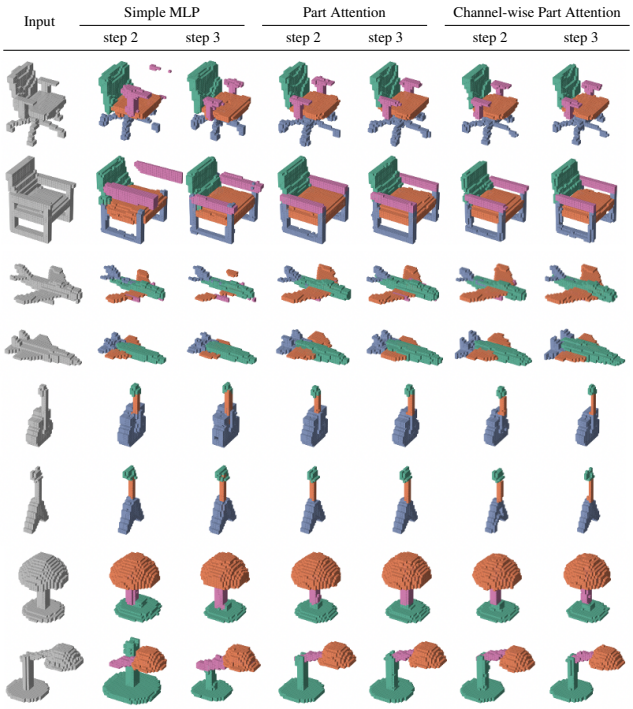


Figure 7. Reconstruction results of our attention-based methods in comparison with applying simple MLP. Results before and after the fine-tuning are both presented.

are given in the encoder-decoder architecture table in the supplementary materials. Note that layer 0 and layer 1 are different when channel-wise part attention is applied. Layer 0 has a feature size of 256 with 1 feature channel, while layer 1 has a feature size of 1^3 with 256 feature channels. In the actual experiments, layer 1 and layer 2 are of too large feature channel size thus consuming extremely large GPU memory when the channel-wise strategy is applied. Hence they are less used in the following experiments.

Figure 5 gives a detailed network design of an attention block. It consists of an 8-heads attention network and a feed-forward network. Figure 6 gives the detailed network design of our proposed attention models. Both normal part attention and channel-wise part attention models are presented. Optional attention consistency loss L_{AC} may be applied between the latent representations of different feature layers after the attention network (green frame box).

4.2. Training Parameters

In step 1, we set $\gamma = 0.6$ for L_{part}. The weights for loss terms are set as $\omega_{\text{PI}} = \omega_{\text{part}} = 1$. We use Adam optimizer with an initial learning rate of 0.001, the decay ratio is 0.8 with a decay step of every 50 epochs. The whole step 1 training takes 250 epochs.

In step 2, training settings are different for the baseline case and the attention-based case. Attention-based net-

Model	Step	Part mIoU ↑					Trans MSE ↓	Shape mIoU ↑
		Back	Seat	Leg	Armrest	Mean		
Simple MLP	1,2	71.6%	73.2%	70.1%	61.2%	72.8%	38.5	63.8% ± 0.1%
	3	71.5%	72.8%	69.8%	60.4%	72.5%	42.6	71.8%
Part Attention	1,2	71.6%	73.2%	70.1%	61.2%	72.8%	32.0	70.9% ± 2.9%
	3	71.6%	73.2%	70.1%	61.3%	72.8%	**31.4**	76.5%
Channel-wise Part Attention	1,2	71.6%	73.2%	70.1%	61.2%	72.8%	36.9	75.8% ± 2.1%
	3	71.7%	73.2%	70.1%	61.3%	72.8%	36.2	**78.4%**

Table 1. Quantitative results from different network models on the chair category. Note that for more precise comparison, we use an encoder of the same weights from step 1 for our models, thus their part mIoU before fine-tuning are identical.

Model	Shape mIoU ↑		
	Chair	Airplane	Lamp
Pix2Mesh [37]	39.6%	42.0%	32.3%
3DCNN [3]	52.6%	50.6%	36.2%
3D-R2N2 [10]	55.0%	56.1%	42.1%
OccNet [25]	50.1%	57.1%	37.1%
IM-Net [9]	52.2%	55.4%	29.6%
D2IM-Net [22]	56.1%	55.8%	42.1%
PQ-Net [42]	67.3%	-	41.3%
CompSM [12]	64.4%	-	-
Simple MLP	71.8%	71.6%	65.4%
Ours (Part Attention)	76.5%	**78.2%**	71.6%
Ours (C-wise Part Attention)	**78.4%**	74.7%	**72.6%**

Table 2. Quantitative shape reconstruction results compared to other methods. Evaluated with the shape mIoU metric on multiple categories.

works are usually more delicate and need more time to converge. For the baseline result of applying simple MLP, we use a learning rate of 0.001 with a decay ratio of 0.8 and a decay step of every 100 epochs. The training takes 500 epochs. For attention-based methods, the weights for loss terms are set as $\omega_{\text{trans}} = \omega_{\text{AC}} = 1$ if L_{AC} is applied. We use a learning rate of 0.0001 without decay and a training epochs number of 1500. The feature embedding dimension $d_A = 256$. Like Transformers [35], a multi-head attention structure with 8 heads is used in each attention block.

For step 3, the initial learning rate has to be very small for fine-tuning since the final learning rate in step 1 is already of a very small number. Here, we use a learning rate of 0.00001 with a decay step of 250 epochs. The total training takes 500 epochs. $\gamma = 0.6$ is used for the full shape binary cross entropy loss L_{shape}. For the loss weights, we set $\omega_{\text{PI}} = 1, \omega_{\text{part}} = 1, \omega_{\text{trans}} = 10, \omega_{\text{shape}} = 10$, and $\omega_{\text{AC}} = 1$ if it is applied. See the supplementary materials for an ablation study on hyper-parameter selection.

4.3. Qualitative and Quantitative Results

Figure 7 gives some shape reconstruction results from unlabeled test 3D shapes. Parts are firstly generated and subsequently assembled with the transformation matrices learned by the attention network. Baseline results of using simple MLP are also given. Results with or without the fine-tuning in step 3 are both given. From the visualization

Model	Symmetry score ↑				
	back	seat	leg	armrest	full shape
GT	0.91	0.95	0.85	0.81	0.96
3D-GAN [41]	0.71	0.76	0.40	0.16	0.70
G2L-GAN [36]	**0.93**	**0.94**	0.74	0.64	**0.91**
PAGENet [20]	0.88	0.90	0.68	0.64	0.85
Simple MLP	0.91	**0.94**	0.83	0.79	0.89
Ours (Part Attention)	0.92	**0.94**	**0.84**	**0.82**	**0.91**
Ours (C-wise Part Attention)	0.92	**0.94**	**0.84**	**0.82**	0.90

Table 3. Symmetry score of our proposed method compared to others on the chair category. Note that it is possible that the symmetry score is better than the ground truth.

results, we can observe that applying simple MLP directly produces modest results yet the shape parts are wrongly scaled and connected in detail. It even fails to correctly reconstruct small-volume parts in some cases. On the other hand, our attention-based methods, both normal part attention and channel-wise part attention, produce more part-coherent reconstruction results on all categories, even for small-volume parts. More visualization results are given in the supplementary materials.

Quantitative results on metrics of part mIoU, shape mIoU, and transformation MSE are presented in Table 1. All the evaluations are performed on the test dataset. Both results before and after the fine-tuning are presented. Larger mIoU means better reconstruction results for both parts and full shape, while smaller transformation MSE means better transformation matrices are learned. From Table 1, we can clearly observe that compared to applying simple MLP, our attention-based method gets better numerical results on both transformation MSE and shape mIoU metrics in both steps. The simple MLP method even sacrifices part mIoU and transformation MSE performance for better shape mIoU performance in step 3. The normal part attention model gets the best transformation MSE performance, which means the visualized assembly results from it are more structurally correct. And the channel-wise part attention model gets the best shape mIoU performance, which means it focuses more on the overall reconstruction. This is actually consistent with our qualitative results.

The quantitative comparison results with other methods on the shape mIoU metric and the symmetry metric are presented in Table 2 and Table 3 respectively. The symmetry score is defined as the percentage of the shape voxels that got matched with their reflection given a vertical symmetry plane. Table 3 shows that our method is capable of generating more symmetric parts even for small volume parts like chair legs or armrests.

Additionally, to quantitatively compare with methods that work on other 3D data representations, *e.g.*, primitive shape-based and point clouds, we follow a similar scheme in SAGNet [44] to firstly convert 3D volumetric data to point clouds and then use the metrics proposed in [1]. In

Model	JSD ↓	MMD-CD ↓	MMD-EMD ↓	COV-CD ↑	COV-EMD ↑
G2L [36]	0.0357	0.0034	0.0682	0.837	0.834
SAGNet [44]	0.0342	0.0024	0.0608	0.751	0.743
GRASS [21]	0.0374	0.0030	0.0744	0.460	0.445
StructureNet [26]	0.0477	0.0097	0.1524	0.297	0.317
BAE-Net [8]	0.7380	0.0057	0.3210	0.090	0.050
MRGAN [14]	0.2460	0.0021	0.1660	0.670	0.230
EDITVAE [23]	0.0310	0.0017	0.1010	0.450	0.390
Ours (Part Attention)	**0.0292**	0.0018	**0.0412**	**1.000**	**1.000**
Ours (C-wise Part Attention)	0.0295	**0.0016**	0.0451	**1.000**	**1.000**

Table 4. Quantitative evaluation for shape modeling on the chair category. For JSD and MMD, the smaller the better. For COV, the larger the better. Since our method is reconstruction-based, we achieve 1.0 on COV metric.

(a) Attention maps in the first attention block.

(b) Attention maps in the last attention block.

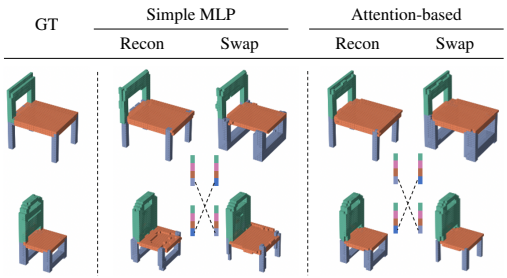
Figure 8. Learned attention maps in different attention blocks. Lighter yellow means higher correlation, darker blue means lower correlation.

Table 4, the results are presented using several metrics for 3D shape sets, including Jensen-Shannon Divergence (JSD), Coverage (COV), and Minimum Matching Distance (MMD). The latter two metrics are calculated using both the Chamfer Distance (CD) and Earth Mover's Distance (EMD) for measuring the distance between shapes.

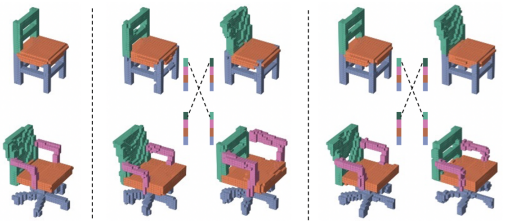
All the above results from the attention-based models are using feature layers 0/3/5 as the input. For the normal part attention mode, L_{AC} is applied; for the channel-wise part attention mode, L_{AC} is not applied. See an ablation study in subsection 4.5 for reasons of using this choice.

4.4. Learned Part Relations

Learned Attention Maps. The attention maps, i.e., the part correlation matrices, is the key to the success of our proposed method. Figure 8(a) and Figure 8(b) give some typical attention maps in different attention heads from the first attention block and the last attention block, respectively. Lighter yellow means higher part correlation, while darker blue means lower part correlation. From the figures, we can observe that in the first attention block the module mostly focuses on learning the correlation of one part to one part, or sometimes to two parts, in the attention heads (feature maps are mostly composed of light yellow or dark blue, only few greenish squares). Meanwhile in the attention heads from the last attention block, the module mostly



(a) Chair leg swapping



(b) Chair back swapping

Figure 9. Part assembly results after swapping certain part representations in the latent space.

focuses on learning the correlation of one part to three or four parts (feature maps are mostly composed of various greenish squares, only few dark blue squares). This means our network architecture learns more part correlation information going deeper through the attention blocks.

Swap. To demonstrate that our attention-based network is able to learn good part relations, we conducted experiments of swapping shape parts in the latent space. Figure 9(a) and Figure 9(b) give a demo of swapping the chair legs or the chair back respectively. The simple MLP method fails to transform the swapped parts correctly according to the size of other parts. Swapping one part may even cause incorrect changes in other parts, *e.g.*, seat in Figure 9(a) and armrest in Figure 9(b). Our attention-based method successfully overcomes all those problems. Swapped parts are correctly aligned with other parts without harming them.

Mix. Figure 10 gives some part assembly results by using random parts from the shape category to compose new shapes. From the figure, we can observe that for the newly composed shapes, the transformation matrices of different parts are learned coherently for the assembly.

Interpolation. Additionally, we also give a demo of interpolating between chair pairs in the latent space in Figure 11. The transition between the same parts in the pairs is smooth, *e.g.*, part thickness, or the morphing chair legs.

4.5. Ablation Study on Attention Models

After conducting a more detailed ablation study on the VoxAttention models with different architecture choices, in Table 5, we report quantitative results of using different feature layers or combinations of them as input for both normal

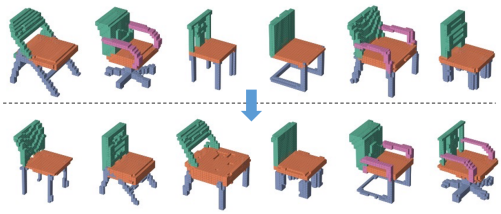


Figure 10. Part assembly results of mixing random parts from different input shapes to generate new shapes.

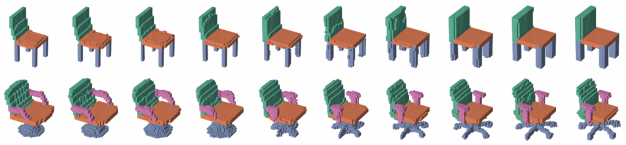


Figure 11. Interpolation results between chair pairs.

part attention case and channel-wise part attention case. For settings with multiple feature layers used as attention input, results with or without the additional L_{AC} are both given.

In step 2, the full shape reconstruction loss is not applied. Hence for the shape mIoU metric, a certain fluctuation (max. $\pm$ 5%) in the performance is observed in most cases. This is actually quite reasonable: a small difference in transformation matrices may cause a big difference in shape mIoU. For example, if the transformed chair back is moved one voxel length ahead, its transformation matrix will not make big difference, but the shape mIoU will change a lot since the shape parts are usually only several voxels thick. In this case, in Table 5, instead of smoothing the curve with a large smoothing parameter, we give a mean value of shape mIoU with a standard deviation using the numerical results in the last 100 epochs.

The following conclusions can be drawn from Table 5:

- For the settings that are identical to the network, their performances are similar to each other. For example, normal part attention with input layer 0, normal part attention with input layer 1, and channel-wise part attention with input layer 0.
- When a single feature layer is used as input, for normal part attention models, using front feature layers gets a smaller transformation MSE. However, for channel-wise part attention models, using rear-end feature layers may get a smaller transformation MSE since the network needs to balance the trade-off between feature size and feature channel number.
- When multiple feature layers are used as input in normal part attention models, L_{AC} must be applied. Otherwise, the model collapses with a large transformation MSE and a bad shape mIoU.
- When multiple feature layers are used as input in the channel-wise part attention models, L_{AC} is not necessary anymore when only one middle feature layer (layer index 1-4) is used.
- Transformation MSE indicates the shape mIoU to some degree. But they do not have an absolute strong correlation. Also, better shape mIoU does not ensure better visualization results since the full shape reconstruction loss is not introduced in this step. Multiple layer input usually yields better visualization results in shape details compared to single layer input.

Feature layer(s)	Apply L_{AC}	Part Attention		Channel-wise Part Attention	
		trans MSE ↓	shape mIoU ↑	trans MSE ↓	shape mIoU ↑
0	-	**31.5**	70.5% ± 4.4%	**31.8**	69.5% ± 3.6%
1	-	**31.5**	70.4% ± 3.1%	88.4	53.2% ± 3.1%
2	-	32.3	**72.9% ± 1.7%**	37.9	70.3% ± 0.8%
3	-	33.0	71.1% ± 2.2%	35.2	**73.3% ± 1.5%**
4	-	37.3	65.9% ± 2.3%	33.9	67.2% ± 3.2%
5	-	42.1	66.4% ± 2.6%	41.9	62.5% ± 5.1%
02	no	39.3	64.5% ± 5.1%	55.4	55.7% ± 2.2%
	yes	32.2	70.0% ± 2.2%	33.5	71.7% ± 1.2%
03	no	60.8	39.4% ± 3.6%	35.0	69.9% ± 3.3%
	yes	32.4	69.0% ± 4.7%	33.3	68.1% ± 2.3%
05	no	354.1	16.9% ± 1.6%	34.9	70.2% ± 2.9%
	yes	33.3	68.0% ± 3.8%	35.2	69.7% ± 2.8%
023	no	99.4	35.6% ± 1.7%	76.3	45.7% ± 4.2%
	yes	32.0	68.8% ± 2.9%	34.5	69.2% ± 1.9%
035	no	232.0	24.5% ± 1.2%	36.9	**75.8% ± 2.1%**
	yes	32.0	**70.9% ± 2.9%**	33.2	70.1% ± 2.1%
045	no	211.9	15.0% ± 1.7%	34.8	68.8% ± 3.2%
	yes	32.6	69.4% ± 1.8%	32.1	70.6% ± 2.3%
0345	no	286.3	24.9% ± 1.0%	91.9	31.8% ± 3.2%
	yes	31.3	70.4% ± 2.8%	**31.3**	71.7% ± 1.7%
012345	no	390.7	22.7% ± 1.2%	-	-
	yes	**31.1**	70.4% ± 3.9%	-	-

Table 5. Quantitative evaluation results in step 2 on the chair category with different experimental settings. Numbers in the first column are the layer index of feature layers used as input.

Balancing the trade-off between model size, metrics results, and visualization qualities, we use feature layers 0/3/5 as the input for most results in the previous subsections. With this choice, for the normal part attention mode, L_{AC} is applied; for the channel-wise part attention mode, L_{AC} is not applied. More intuitive evaluation curves along the training are given in the supplementary material.

5. Conclusions

In this paper, a novel attention-based part assembly method has been proposed for 3D shape modeling. Both qualitative and quantitative results demonstrate that the proposed method achieves better performance on this task compared to other state-of-the-art methods. The channel-wise strategy and the additional attention consistency loss also contribute to the good results. For future possible topics, we would like to try patch-based self-attention to see if the semantic information and the part relations can be learned in a self-supervised manner. Applying attention-based methods on other 3D data representations, *e.g.*, point clouds or meshes, to learn semantic information would also be an interesting research direction.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Learning representations and generative models for 3d point clouds. In *ICML*, 2018. 2, 6

[2] Elena Balashova, Vivek Singh, Jiangping Wang, Brian Teixeira, Terrence Chen, and Thomas A. Funkhouser. Structure-aware shape synthesis. *2018 International Conference on 3D Vision (3DV)*, pages 140–149, 2018. 2

[3] Andrew Brock, Theodore Lim, James M. Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. *ArXiv*, abs/1608.04236, 2016. 1, 2, 3, 6

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *ArXiv*, abs/2005.12872, 2020. 2

[5] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, L. Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *ArXiv*, abs/1512.03012, 2015. 1, 2, 5

[6] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12294–12305, 2021. 1, 2

[7] Mark Chen, Alec Radford, Jeff Wu, Heewoo Jun, Prafulla Dhariwal, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, 2020. 1, 2

[8] Zhiqin Chen, K. Yin, Matthew Fisher, Siddhartha Chaudhuri, and Hao Zhang. Bae-net: Branched autoencoder for shape co-segmentation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8489–8498, 2019. 7

[9] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5932–5941, 2019. 2, 6

[10] Christopher Bongsoo Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, 2016. 2, 6

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021. 1, 2

[12] Anastasia Dubrovina, F. Xia, Panos Achlioptas, Mira Shalah, and Leonidas J. Guibas. Composite shape modeling via latent space factorization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8139–8148, 2019. 2, 3, 6

[13] Nico Engel, Vasileios Belagiannis, and Klaus C. J. Dietmayer. Point transformer. *IEEE Access*, 9:134826–134840, 2021. 1, 2

[14] Rinon Gal, Amit H. Bermano, Hao Zhang, and Daniel Cohen-Or. Mrgan: Multi-rooted 3d shape representation learning with unsupervised part disentanglement. *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 2039–2048, 2021. 7

[15] Rohit Girdhar, David F. Fouhey, Mikel D. Rodriguez, and Abhinav Kumar Gupta. Learning a predictable and generative vector representation for objects. *ArXiv*, abs/1603.08637, 2016. 1, 2

[16] Meng-Hao Guo, Junxiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph Robert Martin, and Shimin Hu. Pct: Point cloud transformer. *Comput. Vis. Media*, 7:187–199, 2021. 1, 2

[17] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, Zhaohui Yang, Yiman Zhang, and Dacheng Tao. A survey on visual transformer. *ArXiv*, abs/2012.12556, 2020. 2

[18] Qingdong He, Zhengning Wang, Hao Zeng, Yi Zeng, Shuaicheng Liu, and Bing Zeng. Svga-net: Sparse voxel-graph attention network for 3d object detection from point clouds. *ArXiv*, abs/2006.04043, 2020. 2

[19] Paul Henderson, Vagia Tsiminaki, and Christoph H. Lampert. Leveraging 2d data to learn textured 3d mesh generation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7495–7504, 2020. 2

[20] Jun Li, Chengjie Niu, and Kai Xu. Learning part generation and assembly for structure-aware shape synthesis. In *AAAI*, 2020. 1, 2, 3, 4, 5, 6

[21] Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao Zhang, and Leonidas J. Guibas. Grass: Generative recursive autoencoders for shape structures. *ArXiv*, abs/1705.02090, 2017. 2, 7

[22] Manyi Li and Hao Zhang. D2im-net: Learning detail disentangled implicit fields from single images. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10241–10250, 2021. 6

[23] Shidi Li, Miaomiao Liu, and Christian J. Walder. Editvae: Unsupervised part-aware controllable 3d point cloud shape generation. *ArXiv*, abs/2110.06679, 2021. 7

[24] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3144–3153, 2021. 2

[25] Lars M. Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4455–4465, 2019. 2, 6

[26] Kaichun Mo, Paul Guerrero, L. Yi, Hao Su, Peter Wonka, Niloy Jyoti Mitra, and Leonidas J. Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. *ACM Trans. Graph.*, 38:242:1–242:19, 2019. 2, 7

[27] Kaichun Mo, Paul Guerrero, L. Yi, Hao Su, Peter Wonka, Niloy Jyoti Mitra, and Leonidas J. Guibas. Structedit: Learning structural shape variations. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8856–8865, 2020. 2

[28] Fakir S. Nooruddin and Greg Turk. Simplification and repair of polygonal models using volumetric techniques. *IEEE Trans. Vis. Comput. Graph.*, 9:191–205, 2003. 5

[29] Jeong Joon Park, Peter R. Florence, Julian Straub, Richard A. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 165–174, 2019. 2

[30] Albert Pumarola, Stefan Popov, Francesc Moreno-Noguer, and Vittorio Ferrari. C-flow: Conditional generative flow models for images and 3d point clouds. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7946–7955, 2020. 2

[31] Gernot Riegler, Ali O. Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6620–6629, 2017. 2

[32] Nadav Schor, Oren Katzir, Hao Zhang, and Daniel Cohen-Or. Componet: Learning to generate the unseen by part synthesis and composition. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8758–8767, 2019. 2

[33] Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Multi-view consistency as supervisory signal for learning shape and pose prediction. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2897–2905, 2018. 2

[34] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized generative models for 3d point clouds via graph convolution. In *ICLR*, 2019. 2

[35] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017. 1, 2, 6

[36] H. Wang, Nadav Schor, Ruizhen Hu, Haibin Huang, Daniel Cohen-Or, and Hui Huang. Global-to-local generative model for 3d shapes. *ACM Transactions on Graphics (TOG)*, 37:1 – 10, 2018. 1, 2, 3, 5, 6, 7

[37] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, W. Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. *ArXiv*, abs/1804.01654, 2018. 2, 6

[38] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions On Graphics (TOG)*, 36(4):1–11, 2017. 2

[39] Chao Wen, Yinda Zhang, Zhuwen Li, and Yanwei Fu. Pixel2mesh++: Multi-view 3d mesh generation via deformation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1042–1051, 2019. 2

[40] Chengzhi Wu, Julius Pfrommer, Mingyuan Zhou, and Jürgen Beyerer. Generative-contrastive learning for self-supervised latent representations of 3d shapes from multi-modal euclidean input. *36th Conference on Artificial Intelligence AAAI Workshop*, 2022. 2, 3

[41] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Joshua B. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *NIPS*, 2016. 1, 2, 5, 6

[42] Rundi Wu, Yixin Zhuang, Kai Xu, Hao Zhang, and Baoquan Chen. Pq-net: A generative part seq2seq network for 3d shapes. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 826–835, 2020. 2, 6

[43] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, 2015. 2

[44] Zhijie Wu, Xiang Wang, Di Lin, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Structure-aware generative network for 3d-shape modeling. *ACM Trans. Graph.*, 38:91:1–91:14, 2019. 1, 2, 6, 7

[45] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge J. Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4540–4549, 2019. 2

[46] L. Yi, Vladimir G. Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas J. Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (TOG)*, 35:1 – 12, 2016. 5

[47] K. Yin, Zhiqin Chen, Siddhartha Chaudhuri, Matthew Fisher, Vladimir G. Kim, and Hao Zhang. Coalesce: Component assembly by learning to synthesize connections. *2020 International Conference on 3D Vision (3DV)*, pages 61–70, 2020. 2

[48] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H. S. Torr, and Vladlen Koltun. Point transformer. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16239–16248, 2021. 1, 2

[49] Chenyang Zhu, Kai Xu, Siddhartha Chaudhuri, Renjiao Yi, and Hao Zhang. Scores: Shape composition with recursive substructure priors. *ArXiv*, abs/1809.05398, 2018. 2

[50] Chuhang Zou, Ersin Yumer, Jimei Yang, Duygu Ceylan, and Derek Hoiem. 3d-prnn: Generating shape primitives with recurrent neural networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 900–909, 2017. 2