MRGAN: Multi-Rooted 3D Shape Representation Learning with Unsupervised Part Disentanglement

Rinon Gal[1] Amit Bermano[1] Hao Zhang[2] Daniel Cohen-Or[1]
[1]Tel-Aviv University [2]Simon Fraser University

Abstract

We introduce MRGAN, multi-rooted GAN*, the first generative adversarial network to learn a* part-disentangled *3D shape representation* without *any part supervision. The network fuses multiple branches of tree-structured graph convolution layers which produce point clouds in a controllable manner. Specifically, each branch learns to grow a different shape part, offering control over the shape generation at the part level. Our network encourages disentangled generation of semantic parts via two key ingredients: a* root-mixing *training strategy which helps decorrelate the different branches to facilitate disentanglement, and a set of loss terms designed with part disentanglement and shape semantics in mind. Of these, a novel convexity loss incentivizes the generation of parts that are more convex, as semantic parts tend to be. In addition, a root-dropping loss further ensures that each root seeds a single part, preventing the degeneration or over-growth of the point-producing branches. We evaluate the performance of our network on a number of 3D shape classes, and offer qualitative and quantitative comparisons to previous works and baseline approaches. We demonstrate the controllability offered by our part-disentangled representation through two applications for shape modeling: part mixing and individual part variation,* without *receiving segmented shapes as input.*

1. Introduction

Generative adversarial networks (GANs) [12] play a prominent role in modern deep learning, owing to their applicability to a multitude of tasks. In the realm of geometric deep learning, these tasks range from 3D shape editing [1, 43] and completion [44] to novel view synthesis [31], shape-to-shape translations [46], and even 3D scene synthesis [47]. However, despite the progresses made, the expressive power of GANs still leaves more to be desired. In particular, there is often a lack of control over the attributes of the generated objects, since these attributes are all entangled in the latent space of the networks. When considering 3D generative models, a promising means to attain controllability is to learn a model with *semantic part disentanglement*. Part disentanglement promotes treating a shape as a *composable structure* of parts, rather than an inseparable whole, a view which has been shown to boost the generative diversity of shapes [35]. An intriguing question is whether a part-disentangled GAN can be trained for a shape class without segmented shapes to provide supervision. A true structural understanding of the shape class may very well result from such an *unsupervised* part disentanglement.

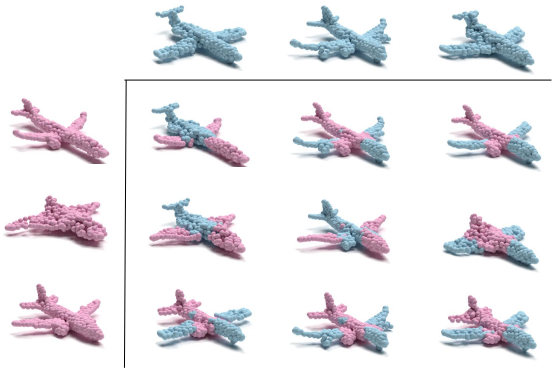

Figure 1. Our network learns a *part-disentangled* 3D shape representation without any part supervision. The part disentanglement enables a new editing operation in latent space: by mixing a pair of latent codes from two source shapes, we can generate seamless shapes that share parts from both. Each (single colored) shape on the left and top was generated from a single latent vector. The 3×3 mixed color shapes were generated using the latent vector from top row for half of the parts, and latent vector from left column for the rest. All parts share a color with their origin shape.

For image generation, multiple conditional models have attempted to assert controls by injecting additional supervision [15, 27, 34], such as class labels [2] and segmentation masks [25]. These models have managed to achieve varying degrees of disentanglement, yet they are limited by the need to acquire labeled data, which can be prohibitively expensive, especially for 3D models. More recently, several works have shifted their attention to unsupervised disentanglement. Utilizing novel architectures designed with this goal in mind, they are able to obtain control over stylistic features [18], pose information [31], the number of gener-

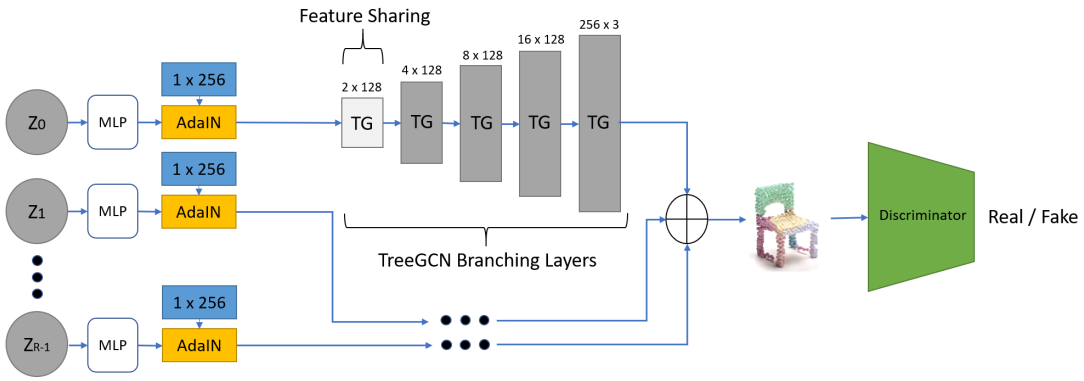

Figure 2. Network architecture of MRGAN, our multi-root generative adversarial network for 3D shapes. For each of the R roots, a latent vector is drawn from a normal distribution and applied to a learned constant (blue) via an AdaIN layer, in a similar manner to StyleGAN. The roots are grown together into a single object using TreeGCN layers, with limited feature sharing at the lowest layer.

ated objects [32], or general modalities in the data [37]. All of these works, however, focus on image-space disentanglment, not part-level controls for 3D shapes.

In this paper, we address this gap in the realm of 3D point cloud generation, by introducing MRGAN, a *multi-rooted* GAN which learns a *part-disentangled* 3D shape representation *without* any part-based shape supervision. Specifically, our model is split into several different generative paths, each responsible for producing a single shape part, before being assembled into a complete shape; see Figure 2. Each path begins from a learned root constant and a perturbation using a sampled latent vector, in a similar fashion to recent style based architectures such as StyleGAN [18], and leads into a tree-structured graph convolution network which produces a shape part as a point cloud.

The roots are encouraged to grow into meaningful shape parts with no repetitions via a collection of losses. In addition to an adversarial loss, we develop a novel *distance-to-hull* based loss by training and applying an auxiliary scoring network to encourage the generation of parts that are more convex, as semantic parts tend to be. We further introduce a *root-dropping* loss, wherein we take the complete shape and subtract all the points emanating from a single root. Building on the intuition that a shape devoid of one of its meaningful parts (e.g., an airplane without one of the wings) is no longer recognizable as a real shape, we elect to penalize the generator if the partial shape is still recognizable as a real object by the discriminator. Lastly, we employ a triplet loss between the object parts and an identity regularization term [31] applied to each path, prompting stronger localization and disentanglement of the parts.

On top of the network losses, a key feature of our approach is a *root mixing* training strategy, where we utilize various combinations of latent code inputs for the different roots to train the model. This strategy not only helps decorrelate the point-producing branches for disentanglement, but also prepares the trained MRGAN to exert control over individual parts during inference, as the network already faced various part mixing scenarios during training.

In summary, our main contributions are four-fold:

- A multi-path, unconditional GAN which is, to the best of our knowledge, the first to achieve *unsupervised* part disentanglement for 3D shapes.
- A differentiable approach to estimating and encouraging convexity of point-cloud shapes.
- An intuitive point removal loss, i.e., the root-dropping loss, that encourages part individuality.
- A multi-scenario training strategy via root mixing.

These contributions combine to form a network which learns a part-disentangled 3D shape representation. Most importantly, such a representation is learned without part-based shape supervision, for the first time.

We evaluate MRGAN on several shape classes, and compare qualitatively and quantitatively to prior works and baselines, demonstrating favorable performance in terms of plausibility and diversity of the generated samples. The sampling of an individual latent vector for each root allows for natural control over different parts. We demonstrate such controllability through two novel applications for shape modeling: part mixing and individual part variation, *without* segmented shapes as input.

2. Related work

Part-aware shape generation. Structure- or part-aware shape modeling has been a prominent topic in computer graphics, where new shapes are synthesized by composing parts from a collection of *segmented* shapes [28]. More recently, attention has been shifting towards learning generative models of 3D structures [4]. In particular, the proliferation of deep learning has led to an extensive collection of part-based representations, ranging from

boxes [23, 39], convexes [5, 7], and super quadratics [33], to implicit indicators [6], learned primitives [8] and deformable meshes [11].

For 3D shape generation, some recent works [9, 35] have resorted to Spatial Transformer Networks [16], while others adopted recursive [23] or graph neural networks [29]. The generative models used include GANs [42], variational autoencoders (VAEs) [11, 22, 35], or a combination thereof [23], while the networks were all trained with collections of segmented shapes or structural representations such as part hierarchies [23, 30] or graphs [29].

The key difference between our work and all of these prior works is that we learn a part-disentangled 3D shape representation without any part supervision. Unsupervised learning of semantic parts often must rely on auxiliary part priors such as compactness [6] and convexity [41]. In BSP-Net, Chen et al. [5] obtain unsupervised part-aware shape reconstruction using a union of convexes assembled via binary space partitioning. However, their network does not generate shapes like a GAN nor enable individual control over the generation of the convexes.

Unsupervised disentanglement. Learning disentangled generative models aims to gain control over individual factors of variation in the generated samples while leaving the other factors unchanged [26]. Unsupervised disentanglement learns such controls without any prior knowledge regarding the factors of variation to disentangle. While recent work suggests that such solutions are indeed fundamentally impossible without using some inductive bias on the model or the data [26], we are content with the commonly approached subset of the task — learning a disentangled representation without the use of explicit, labeled data. Multiple works have tackled this task [14, 19], offering ways to disentangle specific properties [31, 32] or sets of properties divided by scale [18]. Some works are designed to encourage control over a pre-determined property, e.g., object pose [31], while others are more general, proposing control over modalities in the data [37] without prior knowledge of what form these modalities may take.

Our method falls firmly into the camp of networks designed for disentangling over pre-determined, yet latent, properties, i.e., the geometric and structural properties of shape parts. These are clearly distinctive from scale-dependent properties such as the stylistic features learned by StyleGAN [18]. Furthermore, whereas many of the prior works address only global properties of the resulting objects, e.g., pose and class identity, our method allows for control over more localized traits.

Neural Point Cloud Generation. Compared to 3D shape representations such as meshes and voxels, there has been relatively thin literature on neural generation of point clouds, perhaps owning to their irregular and unordered nature. The earliest fully generative models have appeared as recently as Achlioptas et al. [1]. Their approach, however, had difficulty in exploiting localized features [40], prompting more recent works to consider hierarchical sampling operations [21, 45] and graph convolutions [38, 40] which allows for the same, while maintaining permutational invariance. Unlike initial graph convolutional methods [20], these generative works had to contend with a non-constant graph structure, with deeper layers containing more vertices and a different connectivity structure. Their solutions involved dynamic generation of adjacency matrices [40] or a tree-structured approach in the case of TreeGAN [38], where graph connectivity lies between points and their ancestors in previous layers, rather than their same-layer neighbours. Both of these graph-based works displayed a natural *clustering* of the generated points, with TreeGAN even demonstrating a connection between the shared ancestry of points and their relative spatial positions in the final cloud.

Our work takes this natural clustering a step further by directly encouraging the clusters to model meaningful parts. Importantly, whereas these learned models are fully entangled, with no control over the point clusters, we split the network structure into multiple, independently controlled trees. This allows us to achieve more than just a natural clustering, but a disentangled representation with individual control over the shape and number of generated parts.

In a concurrent work, Li et al. [24] present a method for point cloud generation that allows for similar part-level editing, without the need to train on part-level annotations. This level of control, however, relies on manual segmentation provided by a user *after* training. Our method, meanwhile, learns part-level decompositions without *any* supervision.

3. Method

Given a set of objects belonging to a single class, our goal is to learn a latent representation and a generator that can map vectors sampled from this representation into novel instances of the object class. Furthermore, we want the latent representation to be disentangled in parts, such that we can modify only a single part of the generated object, but leave the remainder untouched. We achieve this disentanglement via a forest-like set of tree-convolution paths, as well as a set of network losses and training strategies, designed to encourage separation with no explicit part labels.

3.1. Multi-Rooted Tree-Convolutional GAN

Inspired by the natural ability of graph convolutional networks (GCNs) to generate 3D point clouds composed of localized clusters [38], we elect to employ such a network - TreeGAN - as the backbone of our generator.

The standard graph convolutional approach, first described by [20], considers the entire graph at each layer, combining features of a static number of points with fixed

connectivity. The tree-structured approach instead considers points according to their *depth* in a tree.

Tree-GCN for point cloud generation. Our tree begins with a root and progressively grows into a set of vertices through the application of convolution and branching operations. These operations increase the depth of the tree by creating a set of child nodes for each leaf in the current tree. Each layer, then, is responsible for extending the depth of the tree and calculating the features of the newly created leaves. In this approach, connections to neighbours in a graph are replaced with connections to a node's ancestors in the tree, such that feature propagation is given by:

$$v_i^{l+1} = \sigma\left(W_{i,i}^l v_i + \sum_{v_j \in A(v_i)} W_{i,j}^l v_j + b^l\right), \quad (1)$$

where v_i is the i-th node in the graph, $A(v)$ are the ancestors of node v, v^l denotes the set of features of node v at layer l, W^l and b^l are learned weights and bias terms for the l-th layer, and σ is an activation function. *Branching* is performed by an additional multiplication:

$$v_{i,j}^{l+1} = B_{i,j}^{l+1} v_i^{l+1}, \quad (2)$$

where $v_{i,j}^{l+1}$ is the j-th child of vertex i at layer l + 1, and $B_{i,j}^{l+1}$ is a learned branching-weight term.

This framework lends itself well to the traditional generative approach of increased resolution at deeper layers, and the common ancestors have been shown to drive points on the same branch to display similar properties (e.g., have similar spatial coordinates). However, it does not allow for individual control over these clusters, nor does it make an effort to ensure that the clusters are meaningful or avoid overlaps between the clusters.

Multi-rooted GAN. To encourage the latent space to display the desired part disentanglement, we make the separation of parts *explicit* in our generator architecture, breaking the generative path into *multiple* parallel networks with *limited information sharing*, as shown in Figure 2.

Specifically, we begin with R root constants, each of dimension 1×256. For each root, we sample a latent vector $z \in \mathcal{N}(0,1)^{96}$, pass it through an MLP network, and apply the output to the root through an AdaIN layer, following the same concepts shown in StyleGAN [18]. Each root is then fed as the initial point into the TreeGCN framework, growing into an independent part.

To ensure that the resulting object maintains coherence, e.g., the different parts border each other at the right places, we allow some information to propagate between parts at the lowest layer, via *feature sharing*. The feature vector of each point is concatenated with a vector of features max-pooled over all trees and passed through an MLP in order to reduce it back to the original dimension.

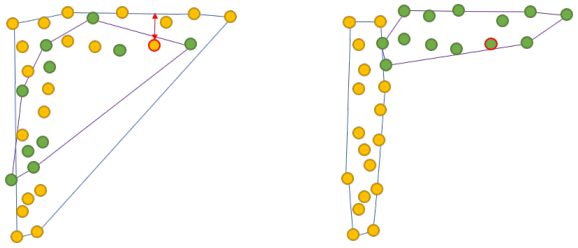

Figure 3. Intuition behind our convexity loss. Consider points representing the seat and a leg of a chair that are split in two ways: randomly on the left and semantically on the right. On the left, points in neither class exhibit convexity, hence they tend to lie further away from the boundary of the convex hull of their class.

The points for each part are given by the leaves of the lowest level of the corresponding tree. After generating all the individual parts, their points are concatenated, resulting in a point cloud as the network output.

Our discriminator follows the architecture of [1]. The full architecture details and an illustration of the feature sharing layers are provided in the supplementary material.

3.2. Network losses

While our multi-path architecture offers a clear way to influence each tree separately, there is no guarantee that the trees will converge into unique, meaningful parts. Indeed, applying only a discriminative loss to the generated cloud results in poor localization and redundancy between parts. In order to encourage a meaningful separation, we apply a set of additional losses, described in the following.

Differentiable convexity loss. We draw inspiration from previous works that eschew the traditional semantic partitioning, but choose instead to recognize a shape as composed from multiple approximately convex parts [5, 10, 17]. While not strictly semantic, these weakly convex parts are still strongly associated with semantic parts [7, 41]. However, known measures for weak convexity are generally not differentiable or highly expensive to compute.

We side-step both issues by defining a simple *differentiable* measure that correlates with convexity. Given a cloud of points, we measure the *average distance* from the points to the nearest hyperplane of the convex hull of the points. It is expected that a point cloud which encompasses a convex part will have relatively more of its points on or near that convex hull, leading to a relatively small *distance-to-hull*, while the opposite is true for a cloud encompassing several shape parts; see Figure 3 for an illustration of the contrast.

We train an auxiliary network to predict the distance-to-hull measure for a given point cloud. The network follows the same form as the discriminator, and uses an L_2 metric as the optimization objective. Complete architecture and training details are provided in the supplementary material.

The convexity loss for the generator is given by applying

the distance-to-hull prediction network to the points generated from each root separately, and taking the mean over the results: $L_h = \frac{1}{R}\sum_{i=1}^{R} F\left(G_i\left(z\right)\right)$, where $F(p)$ is the result of applying the auxiliary network to a set of points and $G_i\left(z\right)$ is the set of points generated from the i-th root.

Root dropping loss. An unfortunate side effect of the global operations used to side-step the cloud permutation concerns is that our discriminator becomes less sensitive to repetitions amongst different parts. So long as a good approximation of the object surface can be achieved, parts may find themselves overlapping partially or entirely with each other. As our aim is for the parts to become distinct, we discourage this behaviour via a root-dropping loss.

This loss is calculated by taking the difference of the discriminator's score on the full shape and the score returned for the shape without the points originating at the dropped root, normalized by the full-shape score, as per:

$$L_{rd} = \frac{1}{R}\sum_{i=1}^{R}\frac{D\left(G(z)\right) - D\left(G(z)\backslash G_i(z)\right)}{D\left(G(z)\right)}, \qquad (3)$$

where $D(p)$ is the discriminator score for a set of points and $G(z)\backslash G_i(z)$ is the set of generated points excluding those originating at the i-th root.

If the trees display considerable overlap, the discriminator returns a similar score with and without the repeated root. Lack of repeats, meanwhile, results in the discriminator being significantly more confident that the part-less shape is a fake, and thus the loss is decreased.

Triplet loss and reconstruction loss. While neither is unique to our work, we employ two additional losses.

The first is a triplet loss [36], employed between generated points, where their source root index is used as a class label. This loss is intended to encourage the points to better localize. We denote the triplet loss by L_t.

Lastly, we employ an identity regularization term similar to the one proposed by HoloGAN [31]. This loss is calculated by appending an additional head to the discriminator, one that is meant to reconstruct the latent vectors that were initially provided to the network. The loss term is the L_2 distance between the original sampled latent codes and their reconstruction. The aim of this loss is to discourage the network from ignoring any of the inputs and instead abusing the feature-sharing layer to dictate their appearance. We denote this reconstruction loss by L_{rec}.

For an adversarial loss, we utilize the Wasserstein GAN loss with a gradient penalty [13].

In total, our loss term is given by:

$$L_{total} = L_{WGAN} + \lambda_h \cdot L_h + \lambda_{rd} \cdot L_{rd} + \lambda_t \cdot L_t + \lambda_{rec} \cdot L_{rec}, \qquad (4)$$

where we used $\lambda_h = \lambda_t = \lambda_{rec} = 1.0$ and $\lambda_{rd} = 0.1$ throughout our experiments.

3.3. Network Training and Root-Mixing

Root-mixing training strategy. We add additional regularization to our network in the form of a root-mixing training strategy, where we utilize different combinations of latent code inputs for the different network roots, according to the following uniformly sampled strategies:

- Sample a single $z \in \mathcal{N}\left(0,1\right)^{96}$ and feed it in all roots.
- Sample two latent vectors $z_1, z_2 \in \mathcal{N}\left(0,1\right)^{96}$, feed z_1 to a random half of the roots and z_2 to the other half.
- Sample two latent vectors $z_1, z_2 \in \mathcal{N}\left(0,1\right)^{96}$, feed z_1 into a randomly chosen root and z_2 into the rest.
- Sample a random latent code for each root.

In doing this, we are drawing lessons from StyleGAN, where Karras et al. [18] show that adding a degree of latent code mixing between different scales during training helps improve disentanglement and the quality of generated outputs. However, their best results are achieved when employing mixing only half the time. We stipulate that a degree of non-mixed inputs helps the network stability and convergence as the network need deal only with fewer factors of variation. We have designed our strategy to reflect that.

Training and implementation details. We trained our network on the ShapeNet [3] dataset, using the chair, table and airplane object classes. For the chair data set we allow the network to make use of 6 roots, while for both the table and airplane data sets we make use of 5 roots.

For all data sets we train the network for 500 epochs using an Adam Optimizer with a learning rate of $1e-4$, $\beta_1 = 0$ and $\beta_2 = 0.99$. We perform 8 optimization steps on the discriminator for every step of the generator. Our latent codes were sampled from a normal distribution $\mathcal{N}\left(0,1\right)$. Our branching operations were of sizes: [2, 2, 2, 2, 16] leading to a total of 256 points generated from each root.

4. Experiments

Shape generation. Figure 4 shows a collection of generated shapes from the airplane, table and chair classes, with points colored according to the path they belong to. Note the comparison to TreeGAN's generated shapes, whose parts are not so cleanly divided between the branches.

Controllable editing. Figure 5 shows the results of editing a generated shape by changing the latent code provided to one root at a time, including part-code interpolation, showing that our latent space is continuous and well behaved. Figures 1 and 6 demonstrate the mixing of latent codes between two shapes from the same object class. By utilizing the latent codes from one shape for half the

Figure 4. Generated samples from the airplane, table, and chair classes. Last row shows results by a vanilla TreeGAN [38] implementation for comparison. For each class, all points originating in a single root share the same color. We use 5 roots for airplanes and tables and 6 roots for chairs. As TreeGAN is grown from a single root, we colored the points according to their ancestor at the 8-point level.

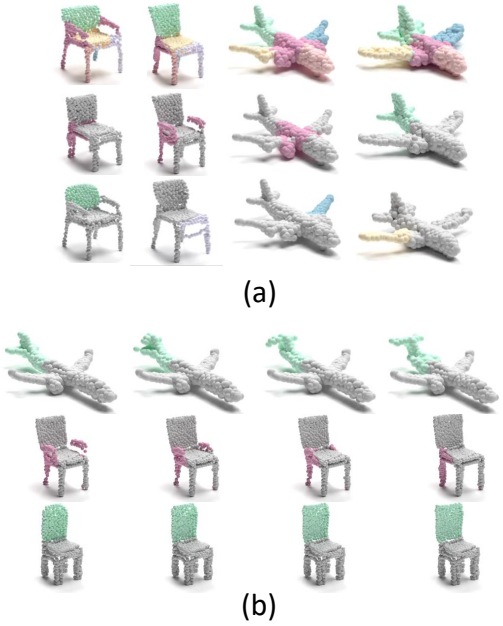

Figure 5. Single root modification experiment results. (a) Top row: original shapes, parts colored by path of origin. Bottom rows: a single (colored) root is randomly changed, while keeping all others unaffected (grey). (b) Interpolation of individual part latent codes. For each row we smoothly modify the latent code of the colored part, and leave the remainder unchanged (grey).

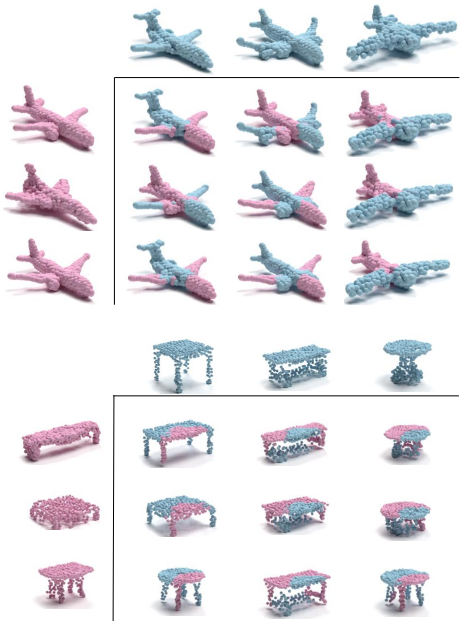

Figure 6. Part mixing. For each class, shapes on the left and top were generated from a single latent vector each. The remaining shapes were generated using the latent vector from top row for half of the roots, and latent vector from left column for the rest.

roots and the latent codes from the other for the remainder, we generate a new shape that combines traits from both. As can be observed when mixing tables of various heights, the model successfully generates a single, coherent shape, while a direct part combination could cause detachment.

Disentanglement quality and ablation. Intuitively, we wish to measure the quality of disentanglement in our network by modifying the latent code for a single part and measuring the change it undergoes when compared to the rest of the points. However, many visually distinct modifications do not require many of the points in the root to change (adding an engine to a wing only requires the movement of a small subset of points, for example). Additionally, even single root modifications result in minor global changes in the shape, which occur in order to preserve coherence. These two effects combine to drown out the effects of single root changes when observing traditional distance metrics.

As such, we propose to measure instead the fraction of points in each part that undergo a *meaningful change*, i.e., one that is greater than the mean movement of points be-

Model	Modified part	Unmodified parts	Ratio ↑
Chair	0.096	0.017	5.36
Table	0.080	0.029	2.71
Airplane	0.122	0.026	4.66
Airplane - Mix + GAN	0.107	0.050	2.13
+ Hull-distance loss	0.109	0.048	2.23
+ Triplet loss	0.123	0.050	2.46
+ Reconstruction loss	0.122	0.026	4.66

Table 1. Quantitative disentanglement evaluation and loss-level ablation results. We show the fraction of points that undergo a meaningful change when modifying a single part, for both points within the modified part and outside it. A larger ratio indicates improved disentanglement. Top: Full-model disentanglement scores for different object classes. Bottom: Ablation results, adding losses one-by-one. Results without any mixing or feature sharing are not provided as they do not generate coherent shapes.

tween pairs of randomly sampled shapes, calculated over 2,500 pairs. We also use this metric to conduct an ablation study of our proposed losses and mixing techniques.

The results are given in Table 1. Our ablation study examined the importance of our proposed components, and resulted in three key observations: the reconstruction loss is of considerable importance in maintaining disentanglement, and any attempt to discard the feature sharing layer or the root-mixing training strategy lead to the network failing to converge or to generated shapes losing coherence.We provide a qualitative overview of the proposed meaningful distance metric, a standard distance metric, additional ablation scenarios, an analysis of disentanglement using a mutual-information based metric and a visualization of the effects of all remaining losses in the supplementary materials.

Mixing strategies. We evaluated multiple mixing strategies utilizing the same disentanglement-quality metrics. For strategies dominated by the scenario where a single latent code is fed into all network roots, the network quickly learns that the inputs are correlated and thus disregards the signal from all but a single root. The resulting model shows no part-level control. For strategies dominated by the scenario where each root is fed a different latent code, the network fails to converge. Strategies which balance the above scenarios in addition to at least one other scenario all successfully converge to disentangled models with similar quantitative scores, indicating that the fine details of the chosen strategy are less crucial, so long as a strategy is chosen that provides a gradient between these two extremes. A table containing our experimental strategies and their scores is provided in the supplementary material.

Varying number of roots. We further evaluated the performance of our network when trained with different numbers of roots. All experiments were performed using the airplane class while keeping other network parameters (i.e. relative loss term weights and mixing strategies) constant. The results are shown in Table 2. As can be expected, the number of roots serves as a strong inductive bias. Choosing a number of roots which is in line with the expected number of meaningful shape parts can help ensure that the network has the capacity to match a root to each part without overly segmenting them. As the number of roots increases, some of them are delegated to constant parts which display very little variation when their latent code is changed. As such, both the the modified and unmodified part scores decrease as we add more roots, but their ratio remains low.

Number of roots	Modified part	Unmodified parts	Ratio ↑
4	0.180	0.044	4.05
5	0.122	0.026	**4.66**
6	0.089	0.023	3.78
8	0.0.78	0.019	3.96
10	0.076	0.018	4.08
15	0.073	0.017	4.15

Table 2. Quantitative disentanglement ablation results for varying the number of object roots using the metrics of Table 1. All experiments were performed using the airplane class and the full model. The best results are achieved when the number of roots is consistent with the expected number of meaningful object parts.

We further investigated whether results could be improved by empowering the network to generate shapes with differing number of parts. Rather than setting a fixed number of roots a-priori, we chose a range of possible values and allowed the network to randomly sample the number used for each generated shape. For these experiments, we utilized the chair class, where samples display a greater variation in the number of parts, than in other classes. During training, for each generated chair, we uniformly sampled a number of roots $\in \{6, 7, 8\}$ and proceeded to generate a novel shape using the chosen number.

Networks trained in this manner allowed for an additional measure of control over the generated result simply by choosing the number of utilized roots at inference time. For example, a choice of 8 roots would result in a swivel chair, while 6 roots would generate a standard 4-legged sample. However, part identities were no longer consistent between samples with varying numbers of roots. The same root that grew into a leg in the 6-root case could instead grow into part of the seat when adding a 7th root. Indeed, adding a root would force a change in the shape of the entire chair, thus breaking disentanglement and our entire ability to mix shapes with different forms. Qualitative results for this experiment are given in supplementary material.

Quantitative comparisons. Ideally, we would like to compare the representational power of our network to other unsupervised, part-disentangled models. However, as we are unaware of such prior works, we employ a baseline

Class	Model	JSD ↓	MMD-CD ↓	MMD-EMD ↓	COV-CD ↑	COV-EMD ↑
Chair	BAE-Net + CompoNet	0.738	0.0057	0.321	9	5
	MRGAN (ours)	**0.246**	**0.0021**	**0.166**	**67**	**23**
Airplane	BAE-Net + CompoNet	0.866	0.0011	0.358	27	4
	MRGAN (ours)	**0.243**	**0.0006**	**0.114**	**75**	**21**
Table	BAE-Net + CompoNet	0.690	0.0036	0.275	25	17
	MRGAN (ours)	**0.287**	**0.0020**	**0.155**	**78**	**31**

Table 3. Comparison to an unsupervised part cloud generative baseline. Our model outperforms the baseline on all metrics of [1], for all classes. Note in particular the gap in coverage metrics, indicating that MRGAN better represents the intra-class variety of shapes.

Class	Coverage metric	r-GAN dense	r-GAN conv	Valsesia et al no up.	Valsesia et al up.	TreeGAN	Ours
Chair	CD ↑	33	23	26	30	58	**67**
	EMD ↑	13	4	20	26	**30**	23
Airplane	CD ↑	31	26	24	31	61	**75**
	EMD ↑	9	7	13	14	20	**21**
Table	CD ↑	-	-	-	-	71	**78**
	EMD ↑	-	-	-	-	**48**	31

Table 4. Coverage comparisons to previous point cloud generators, using the metrics of [1]. We use the values reported by [38] where applicable. The best results for each metric are in bold. Previous works did not provide evaluation metrics on the table class. For the sake of completeness, we provide coverage metrics for tables using our own network and a TreeGAN model trained by us.

composed of two leading models. As a point cloud generator, we utilize CompoNet [35], a part-aware network which represents the state of the art in terms of generative diversity. CompoNet, however, requires supervision in the form of pre-segmented shapes. To overcome this hurdle, we supplement it with BAE-NET [6], a network trained to provide unsupervised co-segmentation of point clouds. We compare the quality of our generated shapes to those assembled by CompoNet when trained on the parts provided by BAE-NET. Results are provided in Table 3. Our model outperforms the baseline on all metrics, showing our ability to represent a richer variety of plausible shapes.

Finally, we compare our results to previous point cloud generators. Coverage results are shown in Table 4, demonstrating once again that MRGAN is capable of generating more diverse shapes. This mirrors the observations of Schor et al. [35], which observe that part-based models can lead to an increase in the diversity of generated content. A full comparison, including an analysis of the quality of our generated parts, can be found in the supplementary materials.

5. Conclusions

We presented MRGAN, a multi-rooted generative architecture that addresses, for the first time, the challenging task of learning a part-disentangled 3D representation in an unsupervised manner. In the absence of any semantic part labels, our work is grounded by the strong inductive bias, or priors, built into the model. These include weak convexity, the root-dropping loss, and even the tree architecture itself, since it naturally tends to a part-like subdivision, as indicated by prior work. Our work shows that, as in the image domain, providing the network with such a strong inductive bias can help overcome the need for detailed labeling. This in turn allows us to train on unlabeled sets and still achieve part-level modifications at inference time.

In order for the network to achieve disentanglement at test time, a degree of root, i.e., part-level, mixing is also crucial at training time. This mirrors the results of StyleGAN [18], where employing mixing through different scales was shown to assist in attribute separability.

We emphasize that the part-level editing shown in Figures 1, 5 and 6 differs from conventional part assembly [28] in two ways. First, part assembly always operates on *pre-segmented* shapes, while our editing/mixing inputs latent codes, *never shape parts*. Second, while the shape parts can undergo a variety of deformations during assembly, such deformations are typically performed only to achieve *local* part alignment and fitting. In contrast, MRGAN arrives at a *globally coherent* and well-connected shape in an end-to-end manner. As the different roots are allowed to affect each other, a point cloud part grown from the same latent code may develop into different shapes according to *non-local* contexts and these shape changes are not confined to any pre-defined deformation space. This is especially apparent in the first row of the table-mixing example in Figure 6.

While MRGAN outperforms the unsupervised baseline (Table 3) and successfully learns a part-disentangled representation with controllable part-level manipulation, it comes at a cost to traditional quality metrics when compared with entangled representations and supervised assembly. We hope to address this limitation in future work.

Finally, our multi-path architecture may offer further advantages. With the parts borne from different paths, a natural extension could be to allow each path to generate a different number of points, helping overcome the drop in resolution for parts that cover a large volume. Taking this further, hierarchical representations, encoding symmetry and other part relations may be explored.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning Representations and Generative Models for 3D Point Clouds. In *ICLR*, 2018. 1, 3, 4, 8

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. *arXiv:1809.11096 [cs, stat]*, Feb. 2019. arXiv: 1809.11096. 1

[3] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 5

[4] Siddhartha Chaudhuri, Daniel Ritchie, Jiajun Wu, Kai Xu, and Hao Zhang. Learning generative models of 3d structures. *Computer Graphics Forum (Eurographics STAR)*, 2020. 2

[5] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. BSP-NET: Generating compact meshes via binary space partitioning. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020. 3, 4

[6] Zhiqin Chen, Kangxue Yin, Matt Fisher, Siddhartha Chaudhuri, and Hao Zhang. BAE-NET: Branched autoencoder for shape co-segmentation. In *IEEE Int. Conf. on Computer Vision (ICCV)*, 2019. 3, 8

[7] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. Cvxnet: Learnable convex decomposition. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2020. 3, 4

[8] Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. Learning elementary structures for 3d shape generation and matching. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 7435–7445, 2019. 3

[9] Anastasia Dubrovina, Fei Xia, Panos Achlioptas, Mira Shalah, Raphael Groscot, and Leonidas Guibas. Composite Shape Modeling via Latent Space Factorization. In *ICCV*, 2019. 3

[10] Matheus Gadelha, Aruni RoyChowdhury, Gopal Sharma, Evangelos Kalogerakis, Liangliang Cao, Erik Learned-Miller, Rui Wang, and Subhransu Maji. Label-Efficient Learning on Point Clouds using Approximate Convex Decompositions. *arXiv:2003.13834 [cs]*, Mar. 2020. arXiv: 2003.13834. 4

[11] Lin Gao, Jie Yang, Tong Wu, Hongbo Fu, Yu-Kun Lai, and Hao Zhang. SDM-NET: Deep generative network for structured deformable mesh. *ACM Trans. on Graphics*, 38(6):Article 243, 2019. 3

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. 1

[13] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017. 5

[14] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. Beta-VAE: Learning basic visual concepts with a constrained variational framework. In *ICLR*, page 13, 2017. 3

[15] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 1

[16] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015. 3

[17] Oliver Van Kaick, Noa Fish, Yanir Kleiman, Shmuel Asafi, and Daniel Cohen-OR. Shape Segmentation by Approximate Convexity Analysis. *ACM Transactions on Graphics*, 34(1):1–11, Dec. 2014. 4

[18] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019. 1, 2, 3, 4, 5, 8

[19] Hyunjik Kim and Andriy Mnih. Disentangling by Factorising. *arXiv:1802.05983 [cs, stat]*, July 2019. arXiv: 1802.05983. 3

[20] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017. 3

[21] Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabas Poczos, and Ruslan Salakhutdinov. Point cloud gan. *arXiv preprint arXiv:1810.05795*, 2018. 3

[22] Jun Li, Chengjie Niu, and Kai Xu. Learning part generation and assembly for structure-aware shape synthesis. *CoRR*, abs/1906.06693, 2019. 3

[23] Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao Zhang, and Leonidas Guibas. GRASS: Generative recursive autoencoders for shape structures. *ACM Transactions on Graphics*, 36(4):Article 52, 2017. 3

[24] Ruihui Li, Xianzhi Li, Ka-Hei Hui, and Chi-Wing Fu. Sp-gan: Sphere-guided 3d shape generation and manipulation. *ACM Trans. Graph.*, 40(4), July 2021. 3

[25] Xiaodan Liang, Hao Zhang, Liang Lin, and Eric Xing. Generative Semantic Manipulation with Mask-Contrasting GAN. In *ECCV*, pages 574–590, 2018. 1

[26] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. *arXiv:1811.12359 [cs, stat]*, June 2019. arXiv: 1811.12359. 3

[27] Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. *arXiv:1411.1784 [cs, stat]*, Nov. 2014. arXiv: 1411.1784. 1

[28] Niloy Mitra, Michael Wand, Hao (Richard) Zhang, Daniel Cohen-Or, Vladimir Kim, and Qi-Xing Huang. Structure-aware shape processing. In *SIGGRAPH Asia 2013 Courses*,

SA '13, pages 1–20, Hong Kong, Hong Kong, Nov. 2013. Association for Computing Machinery. 2, 8

[29] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. *ACM Transactions on Graphics (TOG), Siggraph Asia 2019*, 38(6):Article 242, 2019. 3

[30] Kaichun Mo, He Wang, Xinchen Yan, and Leonidas J. Guibas. PT2PC: Learning to generate 3d point cloud shapes from part tree conditions. In *ECCV*, 2020. 3

[31] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7588–7597, 2019. 1, 2, 3, 5

[32] Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy Mitra. BlockGAN: Learning 3D Object-aware Scene Representations from Unlabelled Images. *arXiv:2002.08988 [cs]*, Feb. 2020. arXiv: 2002.08988. 2, 3

[33] Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. Superquadrics revisited: Learning 3d shape parsing beyond cuboids. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10344–10353, 2019. 3

[34] Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Scribbler: Controlling deep image synthesis with sketch and color. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5400–5409, 2017. 1

[35] Nadav Schor, Oren Katzir, Hao Zhang, and Daniel Cohen-Or. CompoNet: Learning to Generate the Unseen by Part Synthesis and Composition. In *ICCV*, pages 8758–8767, Seoul, Korea (South), 2019. 1, 3, 8

[36] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, June 2015. arXiv: 1503.03832. 5

[37] Omry Sendik, Dani Lischinski, and Daniel Cohen-Or. Unsupervised multi-modal Styled Content Generation. *arXiv:2001.03640 [cs]*, Apr. 2020. arXiv: 2001.03640. 2, 3

[38] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3D Point Cloud Generative Adversarial Network Based on Tree Structured Graph Convolutions. In *ICCV*, 2019. 3, 6, 8

[39] Shubham Tulsiani, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2635–2643, 2017. 3

[40] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized generative models for 3d point clouds via graph convolution. In *International Conference on Learning Representations (ICLR) 2019*, 2019. 3

[41] Oliver van Kaick, Noa Fish, Yanir Kleiman, Shmuel Asafi, and Daniel Cohen-Or. Shape segmentation by approximate convexity analysis. *ACM Trans. on Graphics*, 34(1):4:1–4:11, 2014. 3, 4

[42] Hao 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*, 37(6):1–10, Jan. 2019. 3

[43] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems (NIPS)*, pages 82–90, 2016. 1

[44] Bo Yang, Stefano Rosa, Andrew Markham, Niki Trigoni, and Hongkai Wen. Dense 3d object reconstruction from a single depth view. *IEEE transactions on pattern analysis and machine intelligence*, 41(12):2820–2834, 2018. 1

[45] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. *arXiv*, 2019. 3

[46] Kangxue Yin, Zhiqin Chen, Hui Huang, Daniel Cohen-Or, and Hao Zhang. LOGAN: Unpaired Shape Transform in Latent Overcomplete Space. *ACM Transactions on Graphics*, 38(6):1–13, Nov. 2019. arXiv: 1903.10170. 1

[47] Zaiwei Zhang, Zhenpei Yang, Chongyang Ma, Linjie Luo, Alexander Huth, Etienne Vouga, and Qixing Huang. Deep generative modeling for scene synthesis via hybrid representations. *CoRR*, abs/1808.02084, 2018. 1