Learning Representations and Generative Models for 3D Point Clouds

Panos Achlioptas[1] Olga Diamanti[1] Ioannis Mitliagkas[2] Leonidas Guibas[1]

Abstract

Three-dimensional geometric data offer an excellent domain for studying representation learning and generative modeling. In this paper, we look at geometric data represented as point clouds. We introduce a deep AutoEncoder (AE) network with state-of-the-art reconstruction quality and generalization ability. The learned representations outperform existing methods on 3D recognition tasks and enable shape editing via simple algebraic manipulations, such as semantic part editing, shape analogies and shape interpolation, as well as shape completion. We perform a thorough study of different generative models including GANs operating on the raw point clouds, significantly improved GANs trained in the fixed latent space of our AEs, and Gaussian Mixture Models (GMMs). To quantitatively evaluate generative models we introduce measures of sample fidelity and diversity based on matchings between sets of point clouds. Interestingly, our evaluation of generalization, fidelity and diversity reveals that GMMs trained in the latent space of our AEs yield the best results overall.

1. Introduction

Three-dimensional (3D) representations of real-life objects are a core tool for vision, robotics, medicine, augmented reality and virtual reality applications. Recent attempts to encode 3D geometry for use in deep learning include view-based projections, volumetric grids and graphs. In this work, we focus on the representation of 3D point clouds. Point clouds are becoming increasingly popular as a homogeneous, expressive and compact representation of surface-based geometry, with the ability to represent geometric details while taking up little space. Point clouds are particularly amenable to simple geometric operations and are a standard 3D acquisition format used by range-scanning devices like LiDARs, the Kinect or iPhone's face ID feature.

All the aforementioned encodings, while effective in their target tasks (e.g. rendering or acquisition), are hard to manipulate directly in their raw form. For example, naïvely interpolating between two cars in any of those representations does not yield a representation of an "intermediate" car. Furthermore, these representations are not well suited for the design of generative models via classical statistical methods. Using them to edit and design new objects involves the construction and manipulation of custom, object-specific parametric models, that link the semantics to the representation. This process requires significant expertise and effort.

Deep learning brings the promise of a *data-driven approach*. In domains where data is plentiful, deep learning tools have eliminated the need for hand-crafting features and models. Architectures like AutoEncoders (AEs) (Rumelhart et al., 1988; Kingma & Welling, 2013) and Generative Adversarial Networks (GANs) (Goodfellow et al., 2014; Radford et al.; Che et al., 2016) are successful at learning data representations and generating realistic samples from complex underlying distributions. However, an issue with GAN-based generative pipelines is that training them is notoriously hard and unstable (Salimans et al., 2016). In addition, and perhaps more importantly, *there is no universally accepted method for the evaluation of generative models*.

In this paper, we explore the use of deep architectures for *learning representations* and introduce the first deep *generative models* for *point clouds*. Only a handful of deep architectures tailored to 3D point clouds exist in the literature, and their focus is elsewhere: they either aim at classification and segmentation (Qi et al., 2016a; 2017), or use point clouds *only* as an intermediate or output representation (Kalogerakis et al., 2016; Fan et al., 2016). Our specific contributions are:

- A new AE architecture for point clouds—inspired by recent architectures used for classification (Qi et al., 2016a)—that can learn compact representations with (i) good reconstruction quality on unseen samples; (ii) good classification quality via simple methods (SVM), outperforming the state of the art (Wu et al., 2016); (iii) the capacity for meaningful semantic operations,

[1]Department of Computer Science, Stanford University, USA [2]MILA, Department of Computer Science and Operations Research, University of Montréal, Canada. Correspondence to: Panos Achlioptas <optas@cs.stanford.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

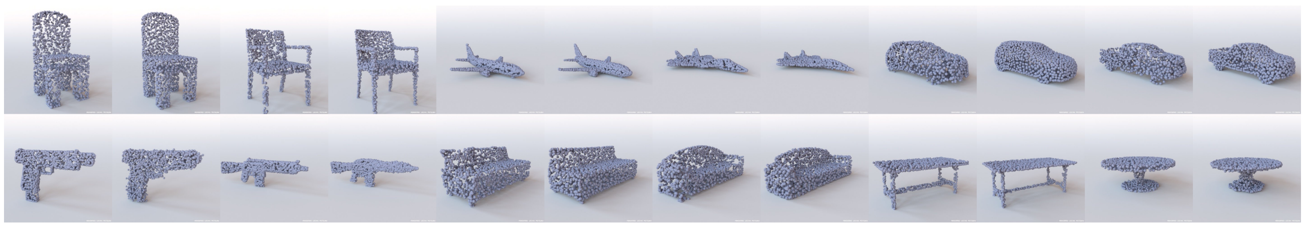


Figure 1. Reconstructions of unseen shapes from the test split of the input data. The leftmost image of each pair shows the ground truth shape, the rightmost the shape produced after encoding and decoding using a *class-specific* AE-EMD.

interpolations and shape-completion.

- The first set of deep generative models for point clouds, able to synthesize point clouds with (i) measurably high fidelity to, and (ii) good coverage of both the training and the held-out data. One workflow that we propose is to first train an AE to learn a latent representation and then train a generative model in that fixed latent space. The GANs trained in the latent space, dubbed here *l-GANs*, are easier to train than raw GANs and achieve superior reconstruction and better coverage of the data distribution. Multi-class GANs perform almost on par with class-specific GANs when trained in the latent space.

- A study of various old and new point cloud metrics, in terms of their applicability (i) as reconstruction objectives for learning good representations; (ii) for the evaluation of generated samples. We find that a commonly used metric, Chamfer distance, fails to identify certain pathological cases.

- Fidelity and coverage metrics for generative models, based on an optimal matching between two different collections of point clouds. Our coverage metric can identify parts of the data distribution that are completely missed by the generative model, something that diversity metrics based on cardinality might fail to capture (Arora & Zhang, 2017).

The rest of this paper is organized as follows: Section 2 outlines some background for the basic building blocks of our work. Section 3 introduces our metrics for the evaluation of generative point cloud pipelines. Section 4 discusses our architectures for latent representation learning and generation. In Section 5, we perform comprehensive experiments evaluating all of our models both quantitatively and qualitatively. Further results can be found in the supplementary material. Last, the code for all our models is publicly available[1].

2. Background

In this section we give the necessary background on point clouds, their metrics and the fundamental building blocks that we will use in the rest of the paper.

2.1. Point clouds

Definition A point cloud represents a geometric shape—typically its surface—as a set of 3D locations in a Euclidean coordinate frame. In 3D, these locations are defined by their x, y, z coordinates. Thus, the point cloud representation of an object or scene is a $N \times 3$ matrix, where N is the number of points, referred to as the point cloud resolution.

Point clouds as an input modality present a unique set of challenges when building a network architecture. As an example, the convolution operator—now ubiquitous in image-processing pipelines—requires the input signal to be defined on top of an underlying grid-like structure. Such a structure is not available in raw point clouds, which renders them significantly more difficult to encode than images or voxel grids. Recent classification work on point clouds (PointNet (Qi et al., 2016a)) bypasses this issue by avoiding convolutions involving groups of points. Another related issue with point clouds as a representation is that they are permutation invariant: any reordering of the rows of the point cloud matrix yields a point cloud that represents the same shape. This property complicates comparisons between two point sets which is needed to define a reconstruction loss. It also creates the need for making the encoded feature permutation invariant.

Metrics Two permutation-invariant metrics for comparing unordered point sets have been proposed in the literature (Fan et al., 2016). On the one hand, the *Earth Mover's* distance (EMD) (Rubner et al., 2000) is the solution of a transportation problem which attempts to transform one set to the other. For two equally sized subsets $S_1 \subseteq R^3, S_2 \subseteq R^3$, their EMD is defined by

$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \to S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2$$

where ϕ is a bijection. As a loss, EMD is differentiable almost everywhere. On the other hand, the *Chamfer* (pseudo)-distance (CD) measures the squared distance between each point in one set to its nearest neighbor in the other set:

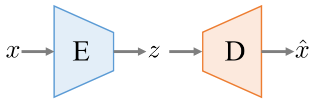
$$d_{CH}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2.$$

[1] http://github.com/optas/latent_3d_points

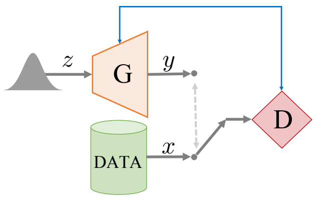
CD is differentiable and compared to EMD more efficient to compute.

2.2. Fundamental building blocks

Autoencoders One of the main deep-learning components we use in this paper is the *AutoEncoder* (AE, inset), which is an architecture that learns to reproduce its input. AEs can be especially useful, when they contain a narrow *bottleneck layer* between input and output. Upon successful training, this layer provides a low-dimensional representation, or *code*, for each data point. The Encoder (E) learns to compress a data point $\boldsymbol{x}$ into its latent representation, $\boldsymbol{z}$. The Decoder (D) can then produce a reconstruction $\hat{\boldsymbol{x}}$, of $\boldsymbol{x}$, from its encoded version $\boldsymbol{z}$.



Generative Adversarial Networks In this paper we also work with Generative Adversarial Networks (GANs), which are state-of-the-art generative models. The basic architecture (inset) is based on a adversarial game between a *generator* (G) and a *discriminator* (D). The generator aims to synthesize samples that look indistinguishable from real data (drawn from $\boldsymbol{x} \sim p_{\text{data}}$) by passing a randomly drawn sample from a simple distribution $\boldsymbol{z} \sim p_z$ through the generator function. The discriminator is tasked with distinguishing synthesized from real samples.



Gaussian Mixture Model A GMM is a probabilistic model for representing a population whose distribution is assumed to be multimodal Gaussian, i.e. comprising of multiple subpopulations, where each subpopulation follows a Gaussian distribution. Assuming the number of subpopulations is known, the GMM parameters (means and variances of the Gaussians) can be learned from random samples, using the Expectation-Maximization (EM) algorithm (Dempster et al., 1977). Once fitted, the GMM can be used to sample novel synthetic samples.

3. Evaluation Metrics for Generative Models

An important component of this work is the introduction of measures that enable comparisons between two sets of points clouds A and B. These metrics are useful for assessing the degree to which point clouds, synthesized or reconstructed, represent the same population as a held-out test set. Our three measures are described below.

JSD The Jensen-Shannon Divergence between marginal distributions defined in the Euclidean 3D space. Assuming point cloud data that are axis-aligned and a canonical voxel grid in the ambient space; one can measure the degree to which point clouds of A tend to occupy similar locations as those of B. To that end, we count the number of points lying within each voxel across *all* point clouds of A, and correspondingly for B and report the JSD between the obtained empirical distributions (P_A, P_B):

$$JSD(P_A \parallel P_B) = \frac{1}{2}D(P_A \parallel M) + \frac{1}{2}D(P_B \parallel M)$$

where $M = \frac{1}{2}(P_A + P_B)$ and $D(\cdot \parallel \cdot)$ the KL-divergence between the two distributions (Kullback & Leibler, 1951).

Coverage For each point cloud in A we first find its closest neighbor in B. Coverage is measured as the *fraction* of the point clouds in B that were matched to point clouds in A. Closeness can be computed using either the CD or EMD point-set distance of Section 2, thus yielding two different metrics, COV-CD and COV-EMD. A high coverage score indicates that most of B is roughly represented within A.

Minimum Matching Distance (MMD) Coverage does not indicate exactly *how well* the covered examples (point-clouds) are represented in set A; matched examples need not be close. We need a way to measure the *fidelity* of A with respect to B. To this end, we match every point cloud of B to the one in A with the minimum distance (MMD) and report the average of distances in the matching. Either point-set distance can be used, yielding MMD-CD and MMD-EMD. Since MMD relies directly on the distances of the matching, it correlates well with how faithful (with respect to B) elements of A are.

Discussion The complementary nature of MMD and Coverage directly follows from their definitions. The set of point clouds A captures all modes of B with good fidelity when MMD is small *and* Coverage is large. JSD is fundamentally different. First, it evaluates the similarity between A and B in coarser way, via marginal statistics. Second and contrary to the other two metrics, it requires pre-aligned data, but is also computationally friendlier. We have found and show experimentally that it correlates well with the MMD, which makes it an efficient alternative for e.g. model-selection, where one needs to perform multiple comparisons between sets of point clouds.

4. Models for Representation and Generation

In this section we describe the architectures of our neural networks starting from an autoencoder. Next, we introduce a GAN that works directly with 3D point cloud data, as well as a decoupled approach which first trains an AE and

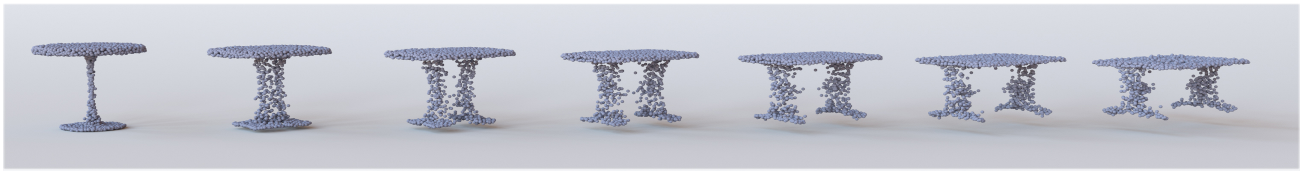


Figure 2. Interpolating between different point clouds, using our latent space representation. More examples for furniture and *human-form* objects (Bogo et al., 2017) are demonstrated in the supplementary material in Figures 3 and 6, respectively.

then trains a minimal GAN in the AE's latent space. We conclude with a similar but even simpler solution that relies on classical Gaussian mixtures models.

4.1. Learning representations of 3D point clouds

The input to our AE network is a point cloud with 2048 points (2048×3 matrix), representing a 3D shape. The encoder architecture follows the design principle of (Qi et al., 2016a): 1-D convolutional layers with kernel size 1 and an increasing number of features; this approach encodes every point *independently*. A "symmetric", permutation-invariant function (e.g. a max pool) is placed after the convolutions to produce a joint representation. In our implementation we use 5 1-D convolutional layers, each followed by a ReLU (Nair & Hinton, 2010) and a batch-normalization layer (Ioffe & Szegedy, 2015). The output of the last convolutional layer is passed to a feature-wise maximum to produce a k-dimensional vector which is the basis for our latent space. Our decoder transforms the latent vector using 3 fully connected layers, the first two having ReLUs, to produce a 2048×3 output. For a permutation invariant objective, we explore both the EMD approximation and the CD (Section 2) as our structural losses; this yields two distinct AE models, referred to as AE-EMD and AE-CD. To regularize the AEs we considered various bottleneck sizes, the use of drop-out and on-the-fly augmentations by randomly-rotating the point clouds. The effect of these choices is showcased in the supplementary material (Section 1) along with the detailed training/architecture parameters. In the remainder of the paper, unless otherwise stated, we use an AE with a 128-dimensional bottleneck layer.

4.2. Generative models for Point Clouds

Raw point cloud GAN (r-GAN) Our first GAN operates on the raw 2048×3 point set input. The architecture of the discriminator is identical to the AE (modulo the filter-sizes and number of neurons), without any batch-norm and with leaky ReLUs (Maas et al., 2013) instead or ReLUs. The output of the last fully connected layer is fed into a sigmoid neuron. The generator takes as input a Gaussian noise vector and maps it to a 2048×3 output via 5 FC-ReLU layers.

Latent-space GAN (l-GAN) For our l-GAN, instead of operating on the raw point cloud input, we pass the data through a pre-trained autoencoder, which is trained separately for each object class with the EMD (or CD) loss function. Both the generator and the discriminator of the l-GAN then operate on the bottleneck variables of the AE. Once the training of GAN is over, we convert a code learned by the generator into a point cloud by using the AE's decoder. Our chosen architecture for the l-GAN, which was used throughout our experiments, is *significantly* simpler than the one of the r-GAN. Specifically, an MLP generator of a single hidden layer coupled with an MLP discriminator of two hidden layers suffice to produce measurably good and realistic results.

Gaussian mixture model In addition to the l-GANs, we also fit a family of Gaussian Mixture Models (GMMs) on the latent spaces learned by our AEs. We experimented with various numbers of Gaussian components and diagonal or full covariance matrices. The GMMs can be turned into point cloud generators by first sampling the fitted distribution and then using the AE's decoder, similarly to the l-GANs.

5. Experimental Evaluation

In this section we experimentally establish the validity of our proposed evaluation metrics and highlight the merits of the AE-representation (Section 5.1) and the generative models (Section 5.2). In all experiments in the main paper, we use shapes from the ShapeNet repository (Chang et al., 2015), that are axis aligned and centered into the unit sphere. To convert these shapes (meshes) to point clouds we uniformly sample their faces in proportion to their area. Unless otherwise stated, we train models with point clouds from a single object class and work with train/validation/test sets of an 85%-5%-10% split. When reporting JSD measurements we use a 28^3 regular voxel grid to compute the statistics.

5.1. Representational power of the AE

We begin with demonstrating the merits of the proposed AE. First we report its generalization ability as measured using the MMD-CD and MMD-EMD metrics. Next, we utilize its latent codes to do semantically meaningful operations. Finally, we use the latent representation to train SVM classifiers and report the attained classification scores.

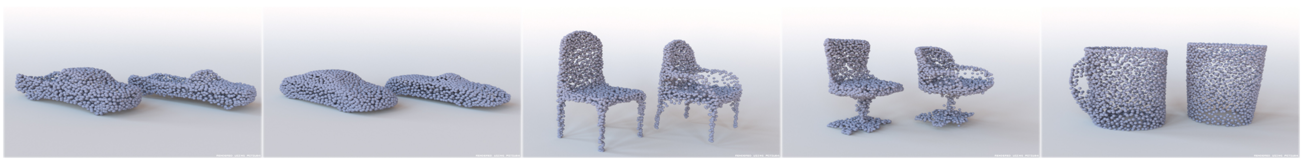


Figure 3. Editing parts in point clouds using simple additive algebra on the AE latent space. Left to right: tuning the appearance of cars towards the shape of convertibles, adding armrests to chairs, removing handle from mug. Note that the height of chairs with armrests is on average 13% shorter than of chairs without one; which is reflected also in these results.

Generalization ability. Our AEs are able to reconstruct unseen shapes with quality almost as good as that of the shapes that were used for training. In Fig. 1 we use our AEs to encode unseen samples from the *test* split (the left of each pair of images) and then decode them and compare them visually to the input (the right image). To support our visuals quantitatively, in Table 1 we report the MMD-CD and MMD-EMD between reconstructed point clouds and their corresponding ground-truth in the train and test datasets of the chair object class. The generalization gap under our metrics is small; to give a sense of scale for our reported numbers, note that the MMD is 0.0003 and 0.033 under the CD and EMD respectively between two versions of the test set that only differ by the randomness introduced in the point cloud sampling. Similar conclusions regarding the generalization ability of the AE can be made based on the reconstruction loss attained for each dataset (train or test) which is shown in Fig. 1 of the supplementary material.

AE loss	MMD-CD		MMD-EMD	
	Train	Test	Train	Test
CD	**0.0004**	**0.0012**	0.068	0.075
EMD	0.0005	0.0013	**0.042**	**0.052**

Table 1. Generalization of AEs as captured by MMD. Measurements for reconstructions on the training and test splits for an AE trained with either the CD or EMD loss and data of the chair class; Note how the MMD favors the AE that was trained with the same loss as the one used by the MMD to make the matching.

Latent space and linearity. Another argument against under/over-fitting can be made by showing that the learned representation is amenable to intuitive and semantically rich operations. As it is shown in several recent works, well trained neural-nets learn a latent representation where additive linear algebra works to that purpose (Mikolov et al., 2013; Tasse & Dodgson, 2016). First, in Fig. 2 we show linear interpolations, in the latent space, between the left and right-most geometries. Similarly, in Fig. 3 we alter the input geometry (left) by adding, in latent space, the mean vector of geometries with a certain characteristic (e.g., convertible cars or cups without handles). Additional operations (e.g. shape analogies) are also possible, but due to space limitations we illustrate and provide the details in the supplementary material (Section 2) instead. These results attest to the smoothness of the learned space but also highlight the intrinsic capacity of point clouds to be smoothly morphed.

Shape completions. Our proposed AE architecture can be used to tackle the problem of shape completion with minimal adaptation. Concretely, instead of feeding and reconstructing the same point cloud, we can feed the network with an *incomplete* version of its expected output. Given proper training data, our network learns to complete severely partial point clouds. Due to space limitations we give the exact details of our approach in the supplementary material (Section 4) and demonstrate some achieved completions in Fig. 4 of the main paper.

Classification. Our final evaluation for the AE's design and efficacy is done by using the learned latent codes as features for classification. For this experiment to be meaningful, we train an AE across all different shape categories: using 57,000 models from 55 categories of man-made objects. Exclusively for this experiment, we use a bottleneck of 512 dimensions and apply random rotations to the input point clouds along the gravity axis. To obtain features for an input 3D shape, we feed its point cloud into the AE and extract the bottleneck activation vector. This vector is then classified by a linear SVM trained on the de-facto 3D classification benchmark of ModelNet (Wu et al., 2015). Table 2 shows comparative results. Remarkably, in the ModelNet10 dataset, which includes classes (chairs, beds etc.) that are populous in ShapeNet, our simple AE significantly outperforms the state of the art (Wu et al., 2016) which instead uses several layers of a GAN to derive a 7168-long feature. In Fig. 8 of the supplementary material we include the confusion matrix of the classifier evaluated on our latent codes on ModelNet40 – the confusion happens between particularly similar geometries: a dresser vs. a nightstand or a flower-pot vs. a plant. The nuanced details that distinguish these objects may be hard to learn without stronger supervision.

	A	B	C	D	E	ours EMD	ours CD
MN10	79.8	79.9	-	80.5	91.0	**95.4**	**95.4**
MN40	68.2	75.5	74.4	75.5	83.3	84.0	**84.5**

Table 2. Classification performance (in %) on ModelNet10/40. Comparing to A: SPH (Kazhdan et al., 2003), B: LFD (Chen et al., 2003), C: T-L-Net (Girdhar et al., 2016), D: VConv-DAE (Sharma et al., 2016), E: 3D-GAN (Wu et al., 2016).

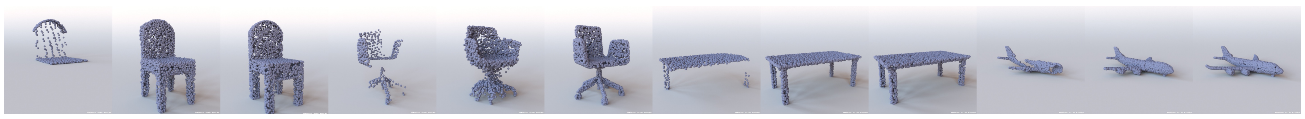


Figure 4. Point cloud *completions* of a network trained with partial and complete (input/output) point clouds and the EMD loss. Each triplet shows the partial input from the test split (left-most), followed by the network's output (middle) and the complete ground-truth (right-most).

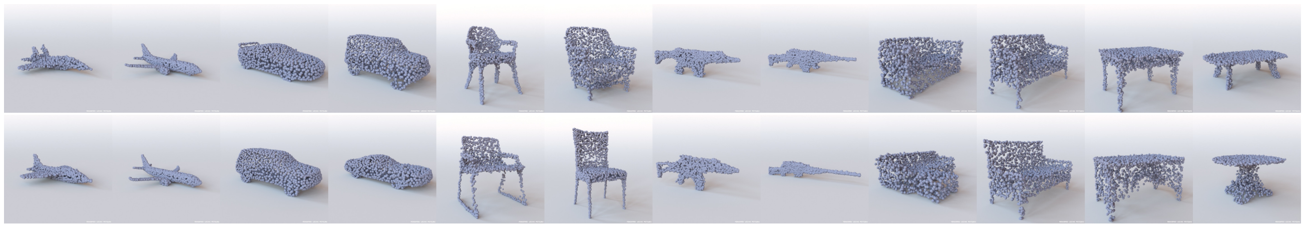


Figure 5. Synthetic point clouds generated by samples produced with l-GAN (top) and 32-component GMM (bottom), both trained on the latent space of an AE using the EMD loss.

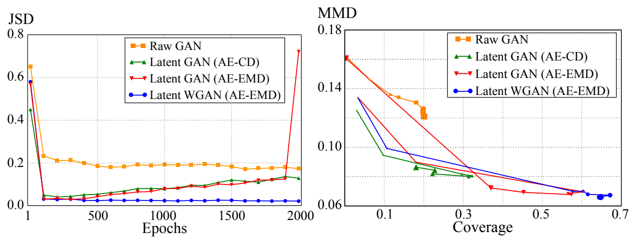


Figure 6. Learning behavior of the GANs, in terms of coverage / fidelity to the ground truth **test** dataset. Left – the JSD distance between the ground truth test set and synthetic datasets generated by the GANs at various epochs of training. Right – EMD based MMD/Coverage: curve markers indicate epochs 1, 10, 100, 200, 400, 1000, 1500, 2000, with *larger symbols denoting later epochs.*

5.2. Evaluating the generative models

Having established the quality of our AE, we now demonstrate the merits and shortcomings of our generative pipelines and establish one more successful application for the AE's learned representation. First, we conduct a comparison between our generative models followed by a comparison between our latent GMM generator and the state-of-the-art 3D voxel generator. Next, we describe how Chamfer distance can yield misleading results in certain pathological cases that r-GANs tends to produce. Finally, we show the benefit of working with a pre-trained latent representation in multi-class generators.

Comparison of our different generative models For this study, we train five generators with point clouds of the *chair* category. First, we establish two AEs trained with the CD or EMD loss respectively—referred to as AE-CD and AE-EMD and train an l-GAN in each latent space with the non-saturating loss of Goodfellow et al. (2014). In the space learned by the AE-EMD we train two additional models: an identical (architecture-wise) l-GAN that utilizes the Wasserstein objective with gradient-penalty (Gulrajani et al., 2017) and a family of GMMs with a different number of means and structures of covariances. We also train an r-GAN directly on the point cloud data.

Fig. 6 shows the JSD (left) and the MMD and Coverage (right) between the produced synthetic datasets and the held-out *test* data for the GAN-based models, as training proceeds. Note that the r-GAN struggles to provide good coverage and good fidelity of the test set; which alludes to the well-established fact that end-to-end GANs are generally difficult to train. The l-GAN (AE-CD) performs better in terms of fidelity with much less training, but its coverage remains low. Switching to an EMD-based AE for the representation and otherwise using the same latent GAN architecture (l-GAN, AE-EMD), yields a dramatic improvement in coverage and fidelity. Both l-GANs though suffer from the known issue of mode collapse: half-way through training, first coverage starts dropping with fidelity still at good levels, which implies that they are overfitting a small subset of the data. Later on, this is followed by a more catastrophic collapse, with coverage dropping as low as 0.5%. Switching to a latent WGAN largely eliminates this collapse, as expected.

In Table 3, we report measurements for all generators based on the epoch (or underlying GMM parameters) that has minimal JSD between the generated samples and the validation set. To reduce the sampling bias of these measurements each generator produces a set of synthetic samples that is $3\times$ the population of the comparative set (test or validation) and repeat the process 3 times and report the averages. The GMM selected by this process has 32 Gaussians and a full covariance. As shown in Fig. 10 of the supplementary material, GMMs with full covariances perform much better than those that have diagonal structure and $\sim$20 Gaussians suffice for good results. Last, the first row of Table 3 shows a baseline model that memorizes a random subset of the training data of the same size as the other generated sets.

Discussion. The results of Table 3 agree with the trends shown in Fig. 6 and further verify the superiority of the latent-based approaches and the relative gains of using an AE-EMD vs. an AE-CD. Moreover they demonstrate that a simple GMM can achieve results of comparable quality to a latent WGAN. Lastly, it is worth noting how the GMM has achieved similar fidelity as that of the perfect/memorized chairs and with almost as good coverage. Table 8 of the supplementary shows the same performance-based conclusions when our metrics are evaluated on the *train* split.

Model	Type	JSD	MMD-CD	MMD-EMD	COV-EMD	COV-CD
A	MEM	0.017	0.0018	0.063	78.6	79.4
B	RAW	0.176	0.0020	0.123	19.0	52.3
C	CD	0.048	0.0020	0.079	32.2	59.4
D	EMD	0.030	0.0023	0.069	57.1	59.3
E	EMD	0.022	0.0019	0.066	66.9	67.6
F	GMM	**0.020**	**0.0018**	**0.065**	**67.4**	**68.9**

Table 3. Evaluating 5 generators on the *test* split of the chair dataset on epochs/models selected via minimal JSD on the validation-split. We report: A: sampling-based memorization baseline, B: r-GAN, C: l-GAN (AE-CD), D: l-GAN (AE-EMD) , E: l-WGAN (AE-EMD), F: GMM (AE-EMD).

Chamfer's blindness, r-GAN's hedging. An interesting observation regarding r-GAN can be made in Table 3. The JSD and the EMD based metrics strongly favor the latent-approaches, while the Chamfer-based ones are much less discriminative. To decipher this discrepancy we did an extensive qualitative inspection of the r-GAN samples and found many cases of point clouds that were over-populated in locations, that on average, *most* chairs have mass. This hedging of the r-GAN is particularly hard for Chamfer to penalize since one of its two summands can become significantly small and the other can be only moderately big by the presence of a few sparsely placed points in the non-populated locations. Figure 7 highlights this point. For a ground-truth point cloud we retrieve its nearest neighbor, under the CD, in synthetically generated sets produced by the r-GAN and the l-GAN and in-image numbers report their CD and EMD distances from it. Notice how the CD fails to distinguish the inferiority of the r-GAN samples while the EMD establishes it. This blindness of the CD metric to only partially good matches, has the additional side-effect that the CD-based coverage is consistently bigger than the EMD-based one.

Comparisons to voxel generators. Generative models for other 3D modalities, like voxels, have been recently proposed (Wu et al., 2016). One interesting question is: if point clouds are our target modality, does it make sense to use voxel generators and then convert to point clouds? This experiment answers this question in the negative. First, we make a comparison using a latent GMM which is trained in conjunction with an AE-EMD. Secondly, we build an AE which operates with *voxels* and fit a GMM in the corresponding latent space. In both cases, we use 32 Gaussians and a full covariance matrix for these GMMs. To use our point-based metrics, we convert the output of (Wu et al., 2016) and our voxel-based GMM into meshes which we sample to generate point clouds. To do this conversion we use the marching-cubes (Lewiner et al., 2003) algorithm with an isovalue of 0.1 for the former method (per authors' suggestions) and 0.5 for our voxel-AE. We also constrain each mesh to be a single connected component as the vast majority of ground-truth data are.

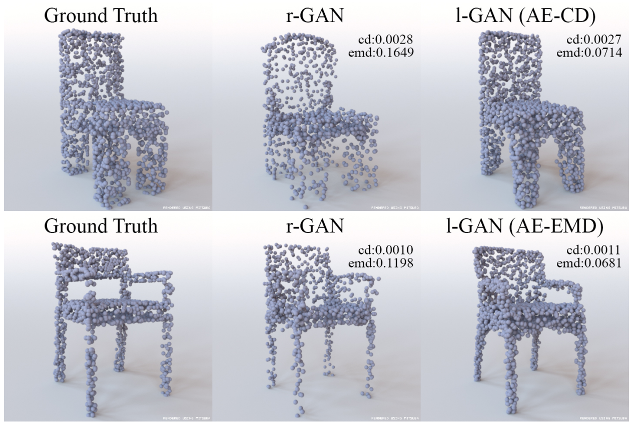


Figure 7. The CD distance is less faithful than EMD to visual quality of synthetic results; here, it favors r-GAN results, due to the overly high density of points in the seat part of the synthesized point sets.

Class	Fidelity		Coverage	
	A	Ours	A	Ours
car	0.059	**0.041**	28.6	**65.3**
rifle	0.051	**0.045**	69.0	**74.8**
sofa	0.077	**0.055**	52.5	**66.6**
table	0.103	**0.061**	18.3	**71.1**

Table 4. Fidelity (MMD-EMD) and coverage (COV-EMD) comparison between A: Wu et al. (2016) and our GMM generative model on the *test* split of each class. Note that Wu et al. uses *all* models of each class for training contrary to our generators.

Table 4 reveals how our point-based GMM trained with a class specific AE-EMD fares against (Wu et al., 2016) on four object classes for which the authors have made their (also class-specific) models publicly [2] available. Our

[2] `http://github.com/zck119/3dgan-release`

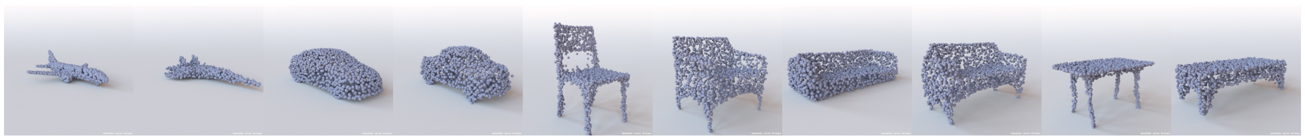


Figure 8. Synthetic point clouds produced with l-WGANs trained in the latent space of an AE-EMD trained on a *multi-class* dataset.

approach is consistently better, with a coverage boost that can be as large as $4\times$ and an almost $2\times$ improved fidelity (case of table). This is despite the fact that (Wu et al., 2016) uses *a*ll models of each class for training, contrary to our generators that never had access to the underlying test split.

Table 5 reveals the performance achieved by pre-training a *voxel*-based AE for the chair class. Observe how by working with a voxel-based latent space, aside of making comparisons more direct to (Wu et al., 2016) (e.g. we both convert output voxels to meshes), we also establish significant gains in terms of coverage and fidelity.

	MMD-CD	MMD-EMD	COV-CD	COV-EMD
A	0.0046	0.091	19.6	22.4
Ours	**0.0025**	**0.072**	**60.3**	**64.8**

Table 5. MMD and Coverage metrics evaluated on the output of voxel-based methods at resolution 64^3, matched against the chair *test* set, using the same protocol as in Table3. Comparing: A: "raw" 64^3-voxel GAN (Wu et al., 2016) and a latent 64^3-voxel GMM.

Qualitative results In Fig. 5, we show some synthetic results produced by our l-GANs and the 32-component GMM. We notice high quality results from either model. The shapes corresponding to the 32 means of the Gaussian components can be found in the supplementary material (Fig. 12), as well as results using the r-GAN (Fig. 4).

Multi-class generators Finally, we compare between class specific and class agnostic generators. In Table 6 we report the MMD-CD for l-WGANs trained in the space of either a dedicated (per-class) AE-EMD or with an AE-EMD trained with all listed object classes. It turns out that the l-WGANs produce perform similar results in either space. Qualitative comparison (Fig. 8) also reveals that by using a multi-class AE-EMD we do not sacrifice much in terms of visual quality compared to the dedicated AEs.

	airplane	*car*	*chair*	*sofa*	*table*	average	multi-class
Tr	0.0004	0.0006	0.0015	0.0011	0.0013	**0.0010**	0.0011
Te	0.0006	0.0007	0.0019	0.0014	0.0017	**0.0013**	0.0014

Table 6. MMD-CD measurements for l-WGANs trained on the latent spaces of dedicated (left 5 columns) and multi-class EMD-AEs (right column). Also shown is the weighted average of the per-class values, using the number of train (Tr) resp. test (Te) examples of each class as weights. All l-WGANs use the model parameter resulted by 2000 epochs of training.

6. Related Work

Recently, deep learning architectures for view-based projections (Su et al., 2015; Wei et al., 2016; Kalogerakis et al., 2016), volumetric grids (Qi et al., 2016b; Wu et al., 2015; Hegde & Zadeh, 2016) and graphs (Bruna et al., 2013; Henaff et al., 2015; Defferrard et al., 2016; Yi et al., 2016) have appeared in the 3D machine learning literature.

A few recent works ((Wu et al., 2016), (Wang et al., 2016), (Girdhar et al., 2016), (Brock et al., 2016), (Maimaitimin et al., 2017), (Zhu et al., 2016)) have explored generative and discriminative representations for geometry. They operate on different modalities, typically voxel grids or view-based image projections. To the best of our knowledge, our work is the first to study such representations for point clouds.

Training Gaussian mixture models (GMM) in the latent space of an autoencoder is closely related to VAEs (Kingma & Welling, 2013). One documented issue with VAEs is over-regularization: the regularization term associated with the prior, is often so strong that reconstruction quality suffers (Bowman et al., 2015; Sønderby et al., 2016; Kingma et al., 2016; Dilokthanakul et al., 2016). The literature contains methods that start only with a reconstruction penalty and slowly increase the weight of the regularizer. An alternative approach is based on adversarial autoencoders (Makhzani et al., 2015) which use a GAN to implicitly regularize the latent space of an AE.

7. Conclusion

We presented a novel set of architectures for 3D point cloud representation learning and generation. Our results show good generalization to unseen data and our representations encode meaningful semantics. In particular our generative models are able to produce faithful samples and cover most of the ground truth distribution. Interestingly, our extensive experiments show that the best generative model for point clouds is a GMM trained in the fixed latent space of an AE. While this might not be a universal result, it suggests that simple classic tools should not be dismissed. A thorough investigation on the conditions under which simple latent GMMs are as powerful as adversarially trained models would be of significant interest.

Acknowledgements

The authors wish to thank all the anonymous reviewers for their insightful comments and suggestions. Lin Shao and Matthias Nießner for their help with the shape-completions and Fei Xia for his suggestions on the evaluation metrics. Last but not least, they wish to acknowledge the support of NSF grants NSF IIS-1528025 and DMS-1546206, ONR MURI grant N00014-13-1-0341, a Google Focused Research award, and a gift from Amazon Web Services for Machine Learning Research.

References

Arora, S. and Zhang, Y. Do gans actually learn the distribution? an empirical study. *CoRR*, abs/1706.08224, 2017.

Bogo, F., Romero, J., Pons-Moll, G., and Black, M. J. Dynamic FAUST: Registering human bodies in motion. In *IEEE CVPR*, 2017.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. Generating sentences from a continuous space. *CoRR*, abs/1511.06349, 2015.

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

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. Spectral networks and locally connected networks on graphs. *CoRR*, abs/1312.6203, 2013.

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

Che, T., Li, Y., Jacob, A. P., Bengio, Y., and Li, W. Mode regularized generative adversarial networks. *CoRR*, abs/1612.02136, 2016.

Chen, D.-Y., Tian, X.-P., Shen, Y.-T., and Ouhyoung, M. On Visual Similarity Based 3D Model Retrieval. *Computer Graphics Forum*, 2003.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, 2016.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1), 1977.

Dilokthanakul, N., Mediano, P. A., Garnelo, M., Lee, M. C., Salimbeni, H., Arulkumaran, K., and Shanahan, M. Deep unsupervised clustering with gaussian mixture variational autoencoders. *CoRR*, abs/1611.02648, 2016.

Fan, H., Su, H., and Guibas, L. J. A point set generation network for 3d object reconstruction from a single image. *CoRR*, abs/1612.00603, 2016.

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

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *NIPS*, 2014.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.

Hegde, V. and Zadeh, R. Fusionnet: 3d object classification using multiple data representations. *CoRR*, abs/1607.05695, 2016.

Henaff, M., Bruna, J., and LeCun, Y. Deep convolutional networks on graph-structured data. *CoRR*, abs/1506.05163, 2015.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Kalogerakis, E., Averkiou, M., Maji, S., and Chaudhuri, S. 3d shape segmentation with projective convolutional networks. *CoRR*, abs/1612.02808, 2016.

Kazhdan, M., Funkhouser, T., and Rusinkiewicz, S. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *ACM SGP*, 2003.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

Kingma, D. P., Salimans, T., and Welling, M. Improving variational inference with inverse autoregressive flow. *CoRR*, abs/1606.04934, 2016.

Kullback, S. and Leibler, R. A. On information and sufficiency. *Annals of Mathematical Statistics*, 1951.

Lewiner, T., Lopes, H., Vieira, A. W., and Tavares, G. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of Graphics Tools*, 2003.

Maas, A. L., Hannun, A. Y., and Ng, A. Y. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013.

Maimaitimin, M., Watanabe, K., and Maeyama, S. Stacked convolutional auto-encoders for surface recognition based on 3d point cloud data. *Artificial Life and Robotics*, 2017.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B. Adversarial autoencoders. *CoRR*, abs/1511.05644, 2015.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CoRR*, abs/1612.00593, 2016a.

Qi, C. R., Su, H., Nießner, M., Dai, A., Yan, M., and Guibas, L. J. Volumetric and multi-view cnns for object classification on 3d data. In *IEEE CVPR*, 2016b.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *CoRR*, 2017.

Radford, A., Metz, L., and Chintala, S. *CoRR*, abs/1511.06434.

Rubner, Y., Tomasi, C., and Guibas, L. J. The earth mover's distance as a metric for image retrieval. *IJCV*, 2000.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *Cognitive modeling*, 5, 1988.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. In *NIPS*, 2016.

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

Sønderby, C. K., Raiko, T., Maaløe, L., Sønderby, S. K., and Winther, O. How to train deep variational autoencoders and probabilistic ladder networks. *CoRR*, abs/1602.02282, 2016.

Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. G. Multi-view convolutional neural networks for 3d shape recognition. In *2015 IEEE ICCV*, 2015.

Tasse, F. P. and Dodgson, N. Shape2vec: Semantic-based descriptors for 3d shapes, sketches and images. *ACM Trans. Graph.*, 2016.

Wang, Y., Xie, Z., Xu, K., Dou, Y., and Lei, Y. An efficient and effective convolutional auto-encoder extreme learning machine network for 3d feature learning. *Neurocomputing*, 174, 2016.

Wei, L., Huang, Q., Ceylan, D., Vouga, E., and Li, H. Dense human body correspondences using convolutional networks. In *IEEE CVPR*, 2016.

Wu, J., Zhang, C., Xue, T., Freeman, B., and Tenenbaum, J. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *NIPS*. 2016.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In *IEEE CVPR*, 2015.

Yi, L., Su, H., Guo, X., and Guibas, L. J. Syncspeccnn: Synchronized spectral CNN for 3d shape segmentation. *CoRR*, abs/1612.00606, 2016.

Zhu, Z., Wang, X., Bai, S., Yao, C., and Bai, X. Deep learning representation using autoencoder for 3d shape retrieval. *Neurocomputing*, 2016.