

High Fidelity Semantic Shape Completion for Point Clouds using Latent Optimization

Shubham Agrawal*
Carnegie Mellon University
shubhamag@cmu.edu

Swaminathan Gurumurthy*
Carnegie Mellon University
swamig@cmu.edu

Abstract

Semantic shape completion is a challenging problem in 3D computer vision where the task is to generate a complete 3D shape using a partial 3D shape as input. We propose a learning-based approach to complete incomplete 3D shapes through generative modeling and latent manifold optimization. Our algorithm works directly on point clouds. We use an autoencoder and a GAN to learn a distribution of embeddings for point clouds of object classes. An input point cloud with missing regions is first encoded to a feature vector. The representations learnt by the GAN are then used to find the best latent vector on the manifold using a combined optimization that finds a vector in the manifold of plausible vectors that is close to the original input (both in the feature space and the output space of the decoder). Experiments show that our algorithm is capable of successfully reconstructing point clouds with large missing regions with very high fidelity without having to rely on exemplar based database retrieval.

1. Introduction

With the increasing availability of low-cost RGB-D scanners, the availability and consequently the need to process 3D data is becoming of great interest to the robotics and vision community. Voxelized representations of 3D data have been quite popular in the learning community because of the ease of generalizing convolution operations to 3D. However most 3D data, whether acquired through RGB-D scanners like Kinect, or through Structure from Motion (SfM) and stereo cameras, is in the form of point clouds. This, along with the fact that point clouds are highly memory efficient while preserving fine surface details, makes it highly desirable to extend deep-learning methods to point clouds. Point clouds have been significantly harder to incorporate into deep learning architectures due to irregular organization of points, i.e, they are not regular structures and can't be di-

rectly used with architectures that exploit regularity in the input for weight sharing. The networks proposed for point clouds need to be able to handle arbitrarily sized inputs and permutation invariance.

A common challenge when reconstructing 3D scenes is that the resulting point clouds may have large missing regions. Reconstructions using SfM may be sparse due to lack of feature points to track on the object. Similarly the point cloud generated by a range scanner may have gaps due to occlusions, limited viewing angle, and may be limited by the resolution of the sensor.

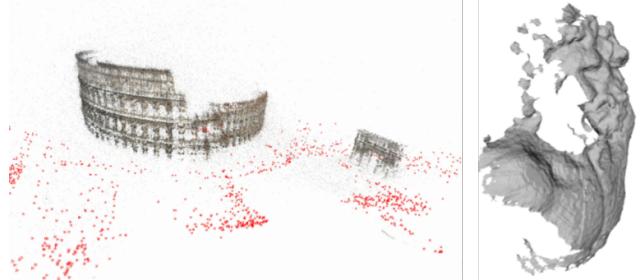


Figure 1. 3D reconstruction through techniques like Structure from Motion or RGB-D scanners often leads to incomplete shapes due to lack of feature points or occlusions respectively.

In this paper we aim to solve this challenge using a deep learning approach. We propose a framework that can take as input a point cloud with arbitrary corruption, such as large holes, entire missing regions (such as due to occlusions/viewing angle) and low resolution/ small number of points (which can also be caused by texture-less surfaces during SfM, or due to limitations in the resolution of a RGB-D sensor); and output a dense complete point cloud. Note that the novelty of our approach lies in the fact that we don't need to train on a dataset containing these corruptions and yet manage to handle them at test time.

Our main contributions are as follows :

- The first shape completion framework that works directly on point clouds, and can handle all types of point

*Both authors contributed equally

cloud noises at test time such as large holes, multiple smaller missing regions, and low density, even if trained only on complete point clouds.

- A novel algorithm that performs shape completion by performing optimization on a latent manifold learnt by a generative model, using a combination of losses that ensures reconstruction of a valid object while simultaneously fitting the available data.
- Quantitative and qualitative evaluation of our method and other baseline methods on both synthetic and real (SfM) data. We demonstrate our method is able to generalize to real data while being trained entirely on synthetic data, something that the baseline methods fail at.

2. Related Work

Deep Learning on 3D data. Common tasks on point clouds include classification, segmentation, object detection and dense labeling [16, 17, 18, 19, 24]. Incorporating point clouds into a deep-learning framework poses several challenges, due to several peculiarities such as input size and order variance, non-uniform density, and shape and scaling variance.

Previously, most deep-learning approaches for point cloud centric applications overcame these challenges by voxelizing the point clouds, which allows for the extension of ideas from 2D CNNs into the 3D space [14] [15]. However voxels are highly inefficient in terms of accuracy and fidelity of the shape represented, and the network size increases rapidly as spatial resolution is increased [41]. More importantly, point clouds are the most common and general representation for 3D data as other representations can easily be obtained from them.

Qi *et al.* [1, 2] first introduced a deep learning network, PointNet, for point cloud classification and segmentation. The network handles the arbitrary input size of point clouds by using an element wise symmetric operation, such as max pool, to encode any input into a fixed size feature vector.

Achlioptas *et al.* [5] proposed coupling a PointNet-style encoder with a decoder of fully connected layers, along with loss metric like Earth Mover’s distance (EMD) to learn representations of point clouds. They further showed that Gaussian Mixture Models (GMMs) or Generative Adversarial Networks (GANs) could be trained to directly generate the latent representations, which can be used for point cloud generation.

Yu *et al.* [7] adapt [2] for the task of upsampling point clouds. As noted by the authors in the paper, their approach is not suited for point clouds with large gaps or missing regions.

Shape Completion. Shape completion has long been a problem of interest in the graphics and vision community.

Traditional geometric methods such as [33, 32, 31] can only fill in small holes in surfaces. A lot of classical approaches relied on exemplar-based completion, where a CAD database was used to fetch similar models to reconstruct the object, which may then be deformed to match the partial input [25, 35, 34]. The vast majority of deep learning works on shape completion have relied on voxel representations due to the ease of generalizing convolution operations to 3D using 3DCNNs. Dai *et al.* [27] used 3DCNNs to predict a coarse complete shape, which was in turn used to lookup similar model from a database. These similar models were then used together with the input for a combined complete shape synthesis. Other recent methods have removed reliance on a database by directly building predictive models for the complete 3D shape, often in a coarse-to-fine manner [27, 29, 26, 30]. Another interesting angle has been the task of predicting 3D shapes from depth maps [39, 40], although these don’t address the challenge of having point clouds with uneven density and arbitrary holes.

The idea of using deep generative models has been shown to be effective in recovering missing regions in 2D images [11] [12]. Similar methods were extended to voxelized 3D shapes recently in [20], which incorporates a convolutional encoder-decoder, a GAN and an LSTM to better learn global and local structure.

There has been very little work done on learning-based methods for shape completion directly on point clouds. The authors of [5] show that their autoencoder architecture may also be trained for completing point clouds. More robust formulations of the autoencoder architecture have been proposed in [4, 6] although these works don’t address shape completion. We note that these recently proposed approaches aimed at learning more robust representations of point clouds can be seamlessly incorporated into our algorithm, as our latent-optimization is agnostic of the style of encoding-decoding mechanism used. To the best of our knowledge, we are the first to propose a deep learning based shape completion approach that works directly on point clouds and can handle arbitrary corruptions in the point cloud without requiring any special training. We also need very small networks with much fewer parameters as compared to voxel based approaches. Our approach is purely learning based and does not rely on exemplar-based retrieval from a database, and generalizes well to objects not seen during training.

3. Methods

3.1. Generative models for point clouds

We build upon a recent model for point cloud generation proposed by Achlioptas *et al.* [5], which extends [1] to learn an autoencoder for point clouds. The encoder consists of multiple layers of 1D convolutions followed by a symmetric pooling layer (max pool in our case) resulting in a single

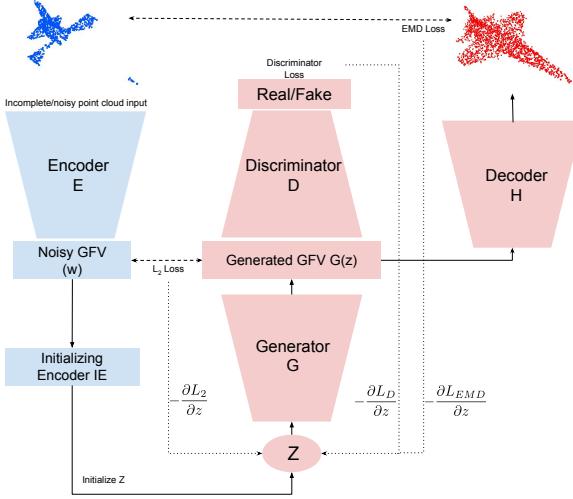


Figure 2. The proposed Point cloud completion algorithm. Loss terms of the LDO for an incomplete input point cloud are visualized with dotted lines. Blocks in blue are only used once for initialization. The losses are used to find the correct point in the latent space of the generator. No network weights are changed.

global feature vector (GFV) for the entire point cloud. The decoder consists of a set of stacked fully connected layers. The last layer of the decoder outputs an $N \times 3$ dimensional vector which corresponds to the N points of the point cloud. The Earth Mover’s distance (EMD), which is a permutation invariant metric, is used as the loss function. The EMD between two point clouds S_1, S_2 is given by:

$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \rightarrow S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2 \quad (1)$$

where $\phi : S_1 \rightarrow S_2$ is a bijection. The optimal bijection is unique and invariant under infinitesimal movement of the points. The autoencoder is trained on the ground truth complete point clouds in the training set. The trained encoder (E) is then used to extract the global feature vector encoding for each point cloud in the training set. As in [5], we then train a GAN on the extracted global feature vectors. New feature vectors generated from the generator (G) can be passed through the decoder (H) to generate point clouds. The GAN has the advantage of being a differentiable network - it is possible to take gradients through it from the output to the input distribution space. This is key for our latent-space optimization algorithm.

Generative Adversarial Network (GAN). GANs are a popular category of generative models which have been recently shown to produce state of the art results in image generation. GANs learn a mapping from an easy to sample distribution (say, a unit normal distribution) to the data

generating distribution using a function approximator like a neural network (generator). The generator(G) is trained in a game theoretic set up, where the objective of the generator is to generate samples that look indistinguishable (to another network, called the discriminator) from the data. The discriminator (D) is trained to distinguish between the real data and the samples generated by G. We introduce a third network, the Initializing Encoder (IE), that learns a mapping from the output of the generator to the latent space z . But the traditional GAN training scheme is known to be unstable. Recent advances in GANs [21, 22, 23] have tried to address this issue by modifying the loss or the training procedure itself. We use the loss modification proposed in [21] for more stable training. We train the GAN on the set of global feature vectors(GFVs) produced by the encoder. We use fully connected layers for both the generator and the discriminator. The training procedure for the AE and GAN is given in Algorithm 1. The losses optimized for training the GAN along side the IE have been described below :

$$J^{(D)} = \mathbb{E}_{z \sim p_z} D(G(z)) - \mathbb{E}_{x \sim p_{data}} D(x) \quad (2)$$

$$\begin{aligned} J^{(G)} = & \mathbb{E}_{z \sim p_z} [\|IE(G(z)) - z\|_2 - D(G(z))] \\ & + \mathbb{E}_{x \sim p_{data}} \|G(IE(x)) - x\|_2 \end{aligned} \quad (3)$$

$$\begin{aligned} J^{(IE)} = & \mathbb{E}_{z \sim p_z} \|IE(G(z)) - z\|_2 \\ & + \mathbb{E}_{x \sim p_{data}} \|G(IE(x)) - x\|_2 \end{aligned} \quad (4)$$

where, p_z is the unit normal distribution centered at the origin, p_{data} is the distribution of GFVs, z and x are samples from these distributions.

3.2. Point Cloud Completion using LDO

Consider an incomplete point cloud at test time, such as one generated via SfM. The point cloud may also be noisy and have uneven density. If this point cloud is passed through the encoder E, a “noisy” GFV is obtained, i.e. one that doesn’t lie on the manifold of representations learnt by the autoencoder. We model the task of completing the point cloud as obtaining a clean GFV corresponding to the noisy one, through an optimization procedure. The cleaned GFV can then be passed through the decoder (H) to obtain a completed point cloud.

Thus the task is reduced to projecting the noisy GFV onto the manifold of clean GFVs. This is not trivial, since we don’t have an analytical expression to represent the clean GFV manifold. Thus we use a GAN to represent the clean GFV manifold. As described in the previous section the GAN is trained on clean GFVs, extracted from the training set of complete point clouds. Projecting a noisy data point onto the manifold of clean GFVs can be reduced to finding the closest GFV to the noisy GFV, that is also classified as real by the discriminator. However, directly optimizing

over the space of GFVs would result in adversarial examples. Thus we choose to perform the optimization procedure in the latent space of the generator, represented by the latent vector z . First we produce an initialization for z by passing the noisy GFV through the Initializing Encoder, $z_{init} = IE(GFV)$. From this initial value, z is optimized so as to produce a clean GFV through the generator, $G(z)$. Specifically, the objective of our Latent Denoising Optimization (LDO) algorithm can be decomposed into three parts:

Discriminator Loss: This term ensures that the generated GFV is from the data manifold. We optimize to maximize the score given by the discriminator to the generated GFV:

$$L_D(z) = -D(G(z)) \quad (5)$$

Latent Least Squares Loss: This term ensures that the generated GFV, $G(z)$ is close to the noisy GFV, w_i during the optimization and thus remains semantically similar to the input point cloud. The noisy GFV was obtained using the encoder E, $w_i = E(S_i)$. We simply minimize the L2 distance between the generated GFV and the noisy GFV:

$$L_2(z; w) = \|G(z) - w_i\|_2^2 \quad (6)$$

Decoder EMD Loss: This term ensures that the generated GFV maps to a point cloud which is close to the input point cloud where it exists. Here, we minimize the Earth Mover's distance between the input point cloud and the point cloud decoded from the generated GFV:

$$L_{EMD}(z; S_i) = d_{EMD}(S_i, H(G(z))) \quad (7)$$

Thus our final loss becomes a weighted combination of these losses:

$$Loss(z) = L_{EMD}(z; S_i) + \lambda L_D(z) + \beta L_2(z; w_i) \quad (8)$$

We perform this optimization using the ADAM optimizer. Note that we use an exponential decay to reduce the value of λ and β , starting with an initial value of 0.001 each. This helps us ensure that in the initial stages of the optimization, the emphasis on obtaining a semantically consistent and real looking point cloud and in the latter stages, the emphasis is on reconstructing fine details of the point cloud. The optimization is stopped as soon as the loss $L_D(z)$ starts increasing. This ensures that the optimization does not lead to 'unreal' looking point clouds in order to get the details right. It is important to note here that we only minimize the loss with respect to z (which is the input to the Generator). We do not update the generator or discriminator parameters when performing this optimization. At the end of the optimization, we obtain the optimal latent vector, z^* . We simply pass this through the generator to get the clean GFV, $G(z)$, which is passed through the decoder to obtain the completed point

cloud, $H(G(z))$. The entire Latent Denoising Optimization (LDO) algorithm has been given in Algorithm 2 and visualized in fig 2. Note that the hyperparameter values stay the same for all our experiments. Thus no experiment specific tuning is required.

Algorithm 1 Training a generative model for use in LDO algorithm

Require: A training set of clean, complete point clouds S.

- 1: Train an autoencoder, with encoder E and decoder H, on the training set S, using EMD as the loss metric. For our experiments we use the implementation in [5], but our algorithm is completely transferable to other PointNet-style architectures such those presented in [2, 4].
 - 2: Using the trained Encoder, extract the global feature vectors (GFVs) for all examples in the training set.
 - 3: Train a GAN to fit on the distribution of extracted GFVs from training set.
-

Algorithm 2 Point cloud completion using LDO algorithm

- 1: Extract the GFV w_i for the partial cloud S_i by passing it through Encoder E. $w_i = E(S_i)$
 - 2: Initialize the latent vector z_i using the initializing encoder IE, $z_i = IE(w_i)$
 - 3: Set $prevLD = L_D(z_i)$, $\lambda = 0.001$, $\beta = 0.001$
 - 4: **for** k=1,2..N **do**
 - 5: Compute $\nabla_z Loss(z_i) = \nabla_z L_{EMD}(z_i; S_i) + \lambda \nabla_z L_D(z_i) + \beta \nabla_z L_2(z_i; w_i)$
 - 6: Update z_i using ADAM
 - 7: Update $\lambda = \lambda * 0.9998$
 - 8: Update $\beta = \beta * 0.9998$
 - 9: **if** $L_D(z_i) > prevLD$ **then**
 - 10: Exit Loop
 - 11: **end if**
 - 12: Update $prevLD = L_D(z_i)$
 - 13: **end for**
 - 14: Pass the cleaned GFV $G(z_i)$ through the previously trained autoencoder's decoder D to obtain the semantically completed point cloud $H(G(z_i))$
-

4. Experimental Results

In this section we demonstrate the salient features of our LDO algorithm by evaluating its quantitative and qualitative performance in multiple scenarios. We compare our model to 2 baseline models of point cloud completion, and show the improvement in reconstruction by applying LDO in conjunction with these baseline models. We show quantitative and qualitative results of the improvements gained by LDO on the tasks of point cloud completion and upsampling. Finally, we show experiments on a real world scenario (SfM)

where we demonstrate that our approach particularly shines in generalizing to real world data, while only having been trained on synthetic data. To summarize, we'll be comparing the following models:

1. **Autoencoder (AE):** An autoencoder trained only with complete point clouds. See Appendix for full implementation details. This baseline is used to demonstrate cases where no prior information is available about the deformities in the true data.
2. **Denoising Autoencoder (DAE):** Another intuitive baseline is an autoencoder with the same architecture as AE, trained with an augmented dataset of incomplete point clouds. While working on our experiments, we became aware that the authors of [5] incidentally updated their work to suggest a similar DAE based completion method. Our initial experiments found that DAEs have a tendency to overfit, and don't generalize well to different amounts and kinds (small holes, large missing region, low-resolution, SfM point clouds) of incompleteness (see appendix). Hence, we train different DAEs with different amounts of incompleteness and test them on the same amounts of incompleteness to get a competitive baseline for comparison. Although one would never have this luxury in the real world, this baseline is used to demonstrate the superiority of our method even when the exact kind and amount of deformity is known beforehand.
3. **Latent Denoising Optimization with AE (AE+LDO):** Our algorithm applied using AE and a GAN learnt on GFVs of the clean training data generated by the AE. We show that *despite never having been trained on noisy/incomplete point clouds*, AE+LDO is very effective at point cloud completion and achieves a huge boost over just the AE's performance.
4. **Latent Denoising Optimization with DAE (DAE+LDO):** To show the transferability of LDO, we also apply it on DAE, with a GAN trained on GFVs produced by the DAE on clean training data. We show that LDO is able to capitalize on the more robust representations learnt by DAE to improve performance even further than AE+LDO.

Dataset. We use ShapeNetCore, a subset of the full ShapeNet[10] dataset with manually verified category and alignment annotations. It covers 55 common object categories with about 51,300 unique 3D models. For the purposes of our experiments we use 4 classes with the most available data from the dataset, namely: airplane, car, chair and table. For each class, we split the models into 85/5/10 train-validation-test sets for our experiments and results. We use the models without any pose or scale augmentations.

We uniformly sample the point clouds (2048 points each) from these models, which serve as the ground truth for our training. In section 4.3, we experiment on a real-world data case we take sequences of images of faces and pass them through an SfM pipeline to get noisy point clouds of faces. We use the CMU Multi-PIE [36] dataset as the source of these face images and the Basel face model [37] to obtain a synthetic dataset of faces. More details on this are provided in section 4.3

4.1. Masking Experiments

For the first set of experiments we choose a synthetic masking scheme to demonstrate the benefits of using LDO in point cloud completion tasks. In order to perform masking on a point cloud, we first choose a random point from the point cloud, and remove its $2048^*(X/100)$ nearest neighbors of the point to obtain an $X\%$ masking. To ease batch processing of point clouds with unequal number of points, we replicate one of the non-masked points so that each point cloud is the same size. The PointNet architecture by its nature ignores replicated points. In latter sections we would look into more realistic scenarios where this would become important.

4.1.1 Varying levels of Incompletion

We train a vanilla autoencoder (AE) using the training set in the Airplane class. We then train a GAN on the GFVs of the AE. At test time, we test the AE and AE + LDO (Latent Denoising Optimization) with varying levels of masking (20%, 30%, 40% and 50%) in the input point cloud. The corresponding scores have been reported in Table 1. Note that we didn't have to retrain our model for the different levels of masking. We observe a clear improvement in performance using our method. We also note that the performance of AE decreases with increasing levels of masking whereas AE+LDO remains more or less robust. We also train multiple denoising autoencoders (DAE) along with the corresponding GANs with varying levels of masking in the input (20%, 30%, 40% and 50%). The DAEs are trained to reconstruct the ground truth given the masked input. In this case, we test DAE and DAE + LDO with masking amounts that the DAE and the corresponding GAN was trained on. So a DAE and GAN trained with 40% masking are tested on 40% masking. This is done to demonstrate the benefit of LDO even when the underlying model (DAE) already has prior knowledge about the incompleteness (since it was trained on the specific kind incompleteness). The corresponding scores have been reported in Table 1. We observe that AE + LDO perform on par with a DAE despite not having any prior knowledge about the kind or amount of incompleteness during training. Moreover, performing LDO on DAE provides further improvement as seen from the scores in

Table 1. This shows that our model can integrate with any AE architecture and capitalize on the robust representations learnt by the models. A visualization of how reconstruction quality varies with increasing percentage of missing data can be found in the appendix.

4.1.2 Different classes

To show the robustness of our methods, we test our model on other classes, namely Chair, Car and Table, both in single-class and multi-class setups. We train a separate AE and the corresponding GAN on the training set of each category. We also train separate DAEs and corresponding GANs with 30% and 50% masking for each category. We also train a multi-class AE, GAN pair on a training set with a combination of the four classes (Table, Chair, Car and Airplane). Correspondingly we train two DAE, GAN pairs with 30% and 50% masking respectively (referred to as Multi-Class in results tables). We test these models on 30% and 50% masking using the corresponding test sets and report the results in Table 1. As in the previous section, the DAEs trained with specific masking amounts are tested with the same masking amounts. We observe a similar pattern as was observed in Airplane in all classes except Cars. AE+LDO performs on par or better than DAE in most of these cases. Moreover, incorporating LDO with DAE further improves the results and provides better scores than all other models in most cases. Interestingly, DAE trained on masked Cars performs better or on par with our models. On further inspection we find that this is because there is very little variety in the dataset of cars. Thus DAE is able to easily transfer from the Car training set to the Car test set by simply producing the nearest neighbors from the training set. We visualize completion results with 50% masking using our best performing multi-class model (DAE + LDO) and compare it against its baseline (DAE) in Figure 3 (An enlarged version may be found in the appendix). It is seen that our multi-loss optimization ensures that both, a sharp, valid object is reconstructed, that also fits the available partial scan as best as possible. Figure 4 compares the point cloud completion results of AE and AE+LDO with 50% masking. We observe that AE produces meaningless point clouds when the inputs are very highly masked/distorted. Yet, just the addition of our algorithm drastically boosts the quality of results as shown in the figure, despite the models never having been trained on incomplete point clouds.

4.2. Upsampling Experiments

Upsampling is another important task that comes up in processing 3D data. This is especially important for SfM methods, that often rely on sparse feature points. We investigate the performance of LDO for upsampling point clouds that had been downsampled to 20% points of the original, using just a regular AE without any special training, and

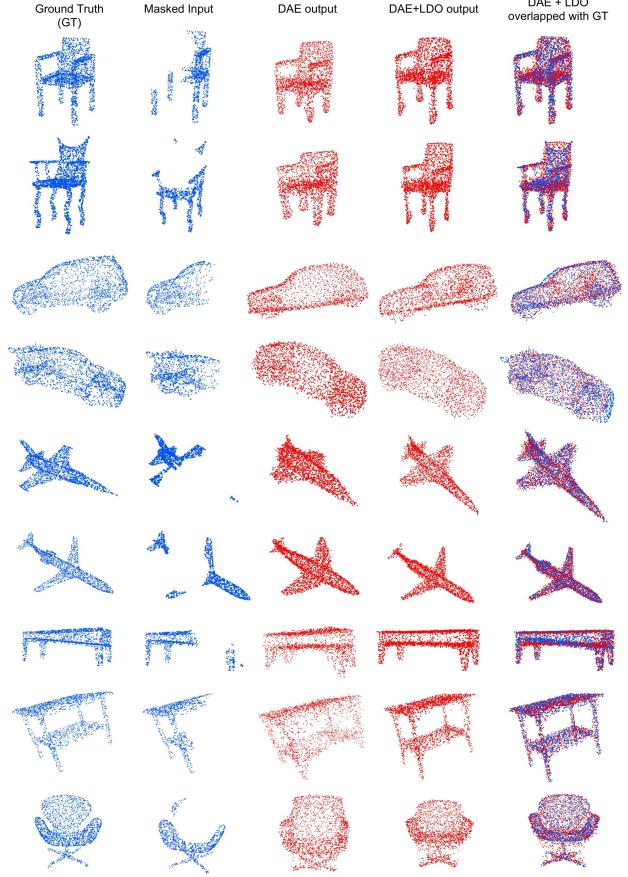


Figure 3. Visualizations of shape completions of LDO on a test set containing all 4 classes. The outputs under "DAE" are from single denoising autoencoder trained on objects of all 4 classes, with 50% missing data. Outputs of DAE+LDO are of the single DAE and a single GAN trained on global feature vectors of the DAE. DAE+LDO leads to much sharper outputs with more details of the partial shape captured. Last column shows ground truth and our results overlaid for ease of comparison.

see impressive results. We show the EMD loss for plain AE and our model in Table 2. The upsampled visualizations are given in Figure 5. We see that the AE struggles to reconstruct any meaningful point clouds. Yet, just by the addition of our algorithm (AE+LDO) we observe a tremendous improvement in the upsampling quality. This shows the versatility of our approach.

4.3. Real-world Experiments

To evaluate the real-world applicability of LDO, we test it on the task of completing input point clouds obtained from SfM. The aim is to see whether LDO can generalize to real-world point cloud data, *while having been trained only on synthetic data*. We use COLMAP [38], a general purpose SfM pipeline to generate point clouds using sequences of

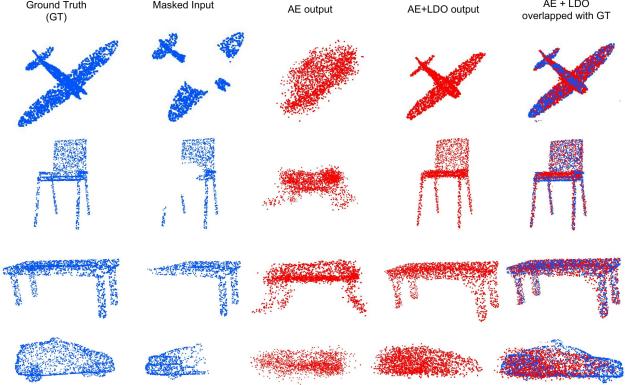


Figure 4. Visualizations of shape completion results of AE and AE+LDO on a test set containing all 4 classes. Random 50% chunks of the inputs are masked at test time. The outputs under "AE" are from a single autoencoder trained on objects of all 4 classes. The right most column shows the AE+LDO outputs overlapped with the ground truth for direct comparison. A massive improvement is seen in reconstruction quality with our method.

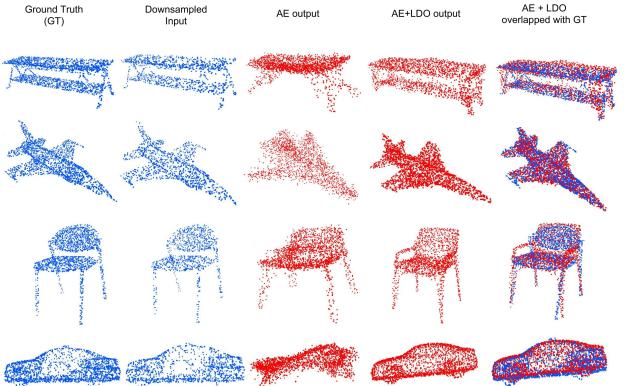


Figure 5. Visualizations of upsampling results of AE and AE+LDO on a test set containing all 4 classes. The inputs at test time are downsampled to 1/5th of the original points. The outputs under "AE" are from a single autoencoder trained on objects of all 4 classes. The right most column shows the AE+LDO outputs overlapped with the ground truth for direct comparison.

images. We experiment with the following classes:

1. Shapenet type models: We use some toy car and airplane models for testing. The models were placed on a rotating surface and multiple images were clicked from different poses around the object. These images were then processed through COLMAP. The output point cloud had incompletions due to severe lack of texture on these models. We test these incomplete point clouds on the multi-class DAE and DAE+LDO, trained with 50% masking on ShapeNet models, as described in section 4.1.2. We choose these, as they were the best

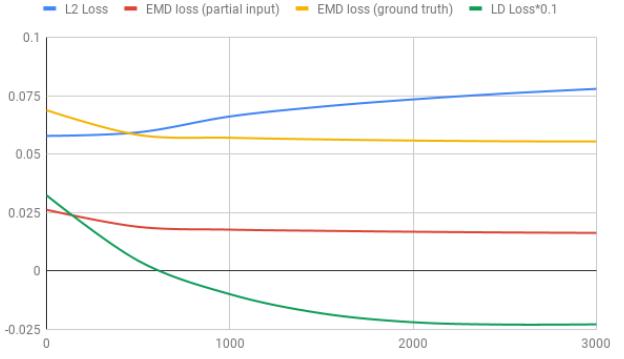


Figure 6. Plot of losses of a typical LDO optimization. EMD loss against ground truth is used for evaluation, not for optimization. LD loss is scaled by 0.1 for ease of visualization

Category	% Points Missing	AE	DAE	AE + LDO(ours)	DAE + LDO(ours)
Airplane	20%	0.061	0.033	0.030	0.028
Airplane	30%	0.079	0.036	0.037	0.033
Airplane	40%	0.083	0.039	0.041	0.034
Airplane	50%	0.097	0.039	0.038	0.037
Chair	30%	0.107	0.061	0.052	0.050
Chair	50%	0.120	0.064	0.069	0.055
Car	30%	0.096	0.0427	0.054	0.041
Car	50%	0.118	0.046	0.060	0.051
Table	30%	0.142	0.055	0.052	0.047
Table	50%	0.143	0.055	0.062	0.050
Multi-Class	30%	0.121	0.072	0.058	0.044
Multi-Class	50%	0.113	0.069	0.056	0.046

Table 1. EMD loss of completed point clouds against ground truth (lower is better). As baselines we compare against an autoencoder(AE) trained only with complete point clouds as well as a denoising AE (DAE) trained with partial point clouds. For fairness, the DAEs were trained with the same percentage of incompleteness as they were tested against. We report the performance of our LDO algorithm when used together with the AE (AE + LDO) and with the DAE(DAE+LDO). Multi-Class refers to training a single AE/DAE to reconstruct all 4 classes, as well as our own algorithm when used with these AE/DAEs

Category	Amount of down-sampling at input	AE	AE + LDO
Multi-Class	80%	0.073	0.058

Table 2. EMD loss of upsampled point clouds against ground truth(lower is better). As baselines we compare against a autoencoder(AE) trained only with complete point clouds. We report the performance of our LDO algorithm when used together with the AE (AE + LDO).

performing models in our previous experiments.

2. Faces : We first create a synthetic training set of 3000 face point clouds using the Basel 3D Morphable Model [37]. The model provides a PCA basis for faces, and different faces can be obtained by sampling the PCA coefficients from gaussians. We train a DAE with similar architecture as the ShapeNet DAEs, except with

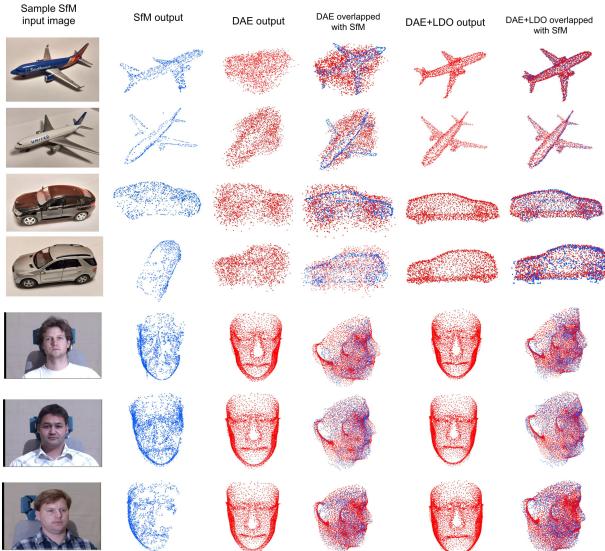


Figure 7. Visualizations of shape completion task on noisy and incomplete point clouds generated by a general-purpose SfM pipeline.

an input/output size of 8192x3. It is trained with 50% masking on the synthetic face dataset. We then also train a GAN on the GFVs obtained by this DAE, as in the regular procedure to setup LDO. Next, we use the CMU Multi-PIE [36] to obtain a sequence of images of human faces taken from different poses. These are processed through COLMAP to obtain point clouds. These are tested on the DAE and DAE+LDO models trained on the synthetic Basel dataset.

For both classes we align and do a "rough" cleaning of the obtained point cloud by aligning it against a template point cloud of the corresponding synthetic set, and removing points beyond a threshold distance from the template. Note that is only to remove background points - the intrinsic noisiness of the points characteristic of SfM is preserved. Where needed we also downsample them to fit our model resolution. The qualitative results can be seen in Fig 7. It is observed that the DAE by itself, having only been trained on synthetic data, fails completely on the ShapeNet-type models, and reconstructs badly fitting faces for the face models, since it fails to generalize beyond the PCA-basis constrained synthetic faces. However DAE+LDO gets high quality reconstructions that fit well with the partial input (A larger image visualizing the face reconstructions from profile views is given in the Appendix).

4.4. Analysis of loss functions

We show a plot of the 3 losses used in LDO optimization and the EMD loss against ground truth (used for evaluation)

during the optimization process of DAE + LDO in Fig 6. The x-axis shows the number of iterations and the y-axis shows the loss values as the optimization progresses. The plot shows that the initialization encoder provides a decent initialization for the optimization, as measured by the ground truth EMD Loss (EMD-GT). This shows that the initialization itself is decent enough to provide scores competitive to that of the DAE. The optimization that follows is responsible for the improvements over the baseline DAE model. We observe that throughout the optimization, the three losses, namely, Partial-EMD Loss, Discriminator loss(LD Loss) and EMD-GT Loss decay gradually until the end of optimization, whereas the L2 loss increases gradually. This indicates that the GFV is being cleaned as it moves away from the noisy GFV and moves closer to the clean GFV. The optimization highlighted here takes 324 sec to process 50 point clouds on a Titan X GPU. The batch size could be increased to achieve lower time per point cloud.

5. Discussion and Conclusion

In this work, we presented a novel scheme for point cloud completion using a purely learning based approach. We demonstrate the following salient features of our approach :

- We show the superiority of our algorithms on a variety of incompleteness types ranging from large missing regions (masked), low-density point clouds (upsampling) to real world SfM point clouds.
- Even when trained with only complete point clouds, our algorithm (AE+LDO) was able to obtain high quality reconstructions.
- Compared with a denoising autoencoder baseline, our approach was shown to generalize much better to unseen data (sec 4.3, 4.1). The reconstructions obtained by DAE+LDO were shown to have higher fidelity to match the partial input, whereas the DAEs overfit to the training data resulting in more generic reconstructions.
- LDO shows generalization even in scenarios when the underlying model is trained on synthetic data and then tested on real world data (sec 4.3). In fact, even in cases where the DAE fails completely, DAE+LDO is able to extract high quality reconstructions.

LDO is quite flexible to the actual autoencoder architecture and training mechanism used. We show that it is able to capitalize on the more robust representations learnt by a DAE. Recent works have proposed more robust formulations of encoder-decoder architectures for pointclouds, such as by incorporating local neighbourhood information [2, 4, 6]. In our future work we wish to explore the integration of LDO into these frameworks. We would also like to explore better optimization schemes than ADAM.

Acknowledgments

We would like to thank Panos Achlioptas for sharing his implementation of the autoencoder and Haoqiang Fan for the CUDA implementation of EMD loss function.

References

- [1] Qi, C.R., Su, H., Mo, K. and Guibas, L.J., 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. Proc. Computer Vision and Pattern Recognition (CVPR), IEEE, 1(2), p.4.
- [2] Qi, C.R., Yi, L., Su, H. and Guibas, L.J., 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Advances in Neural Information Processing Systems (pp. 5105-5114).
- [3] Fan, H., Su, H. and Guibas, L.J., 2017, July. A point set generation network for 3d object reconstruction from a single image. In Conference on Computer Vision and Pattern Recognition (CVPR) (Vol. 38).
- [4] Yang, Y., Feng, C., Shen, Y. and Tian, D., 2017. FoldingNet: Interpretable Unsupervised Learning on 3D Point Clouds. arXiv preprint arXiv:1712.07262.
- [5] Achlioptas, P., Diamanti, O., Mitliagkas, I. and Guibas, L., 2017. Representation learning and adversarial generation of 3D point clouds. arXiv preprint arXiv:1707.02392.
- [6] Shen, Y., Feng, C., Yang, Y. and Tian, D., 2017. Neighbors Do Help: Deeply Exploiting Local Structures of Point Clouds. arXiv preprint arXiv:1712.06760.
- [7] Yu, L., Li, X., Fu, C.W., Cohen-Or, D. and Heng, P.A., 2018. PU-Net: Point Cloud Upsampling Network. arXiv preprint arXiv:1801.06761.
- [8] Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M. and Solomon, J.M., 2018. Dynamic Graph CNN for Learning on Point Clouds. arXiv preprint arXiv:1801.07829.
- [9] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., & Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1912-1920).
- [10] Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., ... & Xiao, J. (2015). Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012.
- [11] Yeh, R. A., Chen, C., Lim, T. Y., Schwing, A. G., Hasegawa-Johnson, M., & Do, M. N. (2017, July). Semantic image inpainting with deep generative models. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5485-5493).
- [12] Iizuka, S., Simo-Serra, E., & Ishikawa, H. (2017). Globally and locally consistent image completion. ACM Transactions on Graphics (TOG), 36(4), 107.
- [13] Rubner, Y., Tomasi, C., & Guibas, L. J. (2000). The earth mover's distance as a metric for image retrieval. International journal of computer vision, 40(2), 99-121.
- [14] Maturana, D., & Scherer, S. (2015, September). Voxnet: A 3d convolutional neural network for real-time object recognition. In Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on (pp. 922-928). IEEE.
- [15] Qi, C. R., Su, H., Nießner, M., Dai, A., Yan, M., & Guibas, L. J. (2016). Volumetric and multi-view cnns for object classification on 3d data. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 5648-5656).
- [16] Boulch, A., Saux, B. L., & Audebert, N. (2017, April). Unstructured point cloud semantic labeling using deep segmentation networks. In Eurographics Workshop on 3D Object Retrieval (Vol. 2, p. 1).
- [17] Dohan, D., Matejek, B., & Funkhouser, T. (2015, October). Learning hierarchical semantic segmentations of lidar data. In 3D Vision (3DV), 2015 International Conference on (pp. 273-281). IEEE.
- [18] Hackel, T., Savinov, N., Ladicky, L., Wegner, J. D., Schindler, K., & Pollefeys, M. (2017). Semantic3D. net: A new large-scale point cloud classification benchmark. arXiv preprint arXiv:1704.03847.
- [19] Huang, J., & You, S. (2016, December). Point cloud labeling using 3d convolutional neural network. In Pattern Recognition (ICPR), 2016 23rd International Conference on (pp. 2670-2675). IEEE.
- [20] Wang, W., Huang, Q., You, S., Yang, C., & Neumann, U. (2017). Shape inpainting using 3d generative adversarial network and recurrent convolutional networks. arXiv preprint arXiv:1711.06375.
- [21] Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein gan. arXiv preprint arXiv:1701.07875.
- [22] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).

- [23] Mescheder, L., Nowozin, S., & Geiger, A. (2017). The numerics of gans. In Advances in Neural Information Processing Systems (pp. 1823-1833).
- [24] Li, J., Chen, B. M., & Lee, G. H. (2018, March). SO-Net: Self-Organizing Network for Point Cloud Analysis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 9397-9406).
- [25] Li, D., Shao, T., Wu, H., & Zhou, K. (2017). Shape completion from a single rgbd image. *IEEE transactions on visualization and computer graphics*, 23(7), 1809-1822.
- [26] Stutz, D., & Geiger, A. (2018). Learning 3D Shape Completion from Laser Scan Data with Weak Supervision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1955-1964).
- [27] Dai, A., Ritchie, D., Bokeloh, M., Reed, S., Sturm, J., & Nießner, M. (2018). ScanComplete: Large-Scale Scene Completion and Semantic Segmentation for 3D Scans. In CVPR (Vol. 1, p. 2).
- [28] Dai, A., Qi, C. R., & Nießner, M. (2017, July). Shape completion using 3d-encoder-predictor cnns and shape synthesis. In Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (Vol. 3).
- [29] Varley, J., DeChant, C., Richardson, A., Ruales, J., & Allen, P. (2017, September). Shape completion enabled robotic grasping. In Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on (pp. 2442-2447). IEEE.
- [30] Han, X., Li, Z., Huang, H., Kalogerakis, E., & Yu, Y. (2017, October). High-Resolution Shape Completion Using Deep Neural Networks for Global Structure and Local Geometry Inference. In Proceedings of IEEE International Conference on Computer Vision (ICCV).
- [31] Kazhdan, M., & Hoppe, H. (2013). Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3), 29.
- [32] Nealen, A., Igarashi, T., Sorkine, O., & Alexa, M. (2006, November). Laplacian mesh optimization. In Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia (pp. 381-389). ACM.
- [33] Sorkine, O., & Cohen-Or, D. (2004, June). Least-squares meshes. In Shape Modeling Applications, 2004. Proceedings (pp. 191-199). IEEE.
- [34] Li, Y., Dai, A., Guibas, L., & Nießner, M. (2015, May). Database-assisted object retrieval for real-time 3d reconstruction. In Computer Graphics Forum (Vol. 34, No. 2, pp. 435-446).
- [35] Shi, Y., Long, P., Xu, K., Huang, H., & Xiong, Y. (2016). Data-driven contextual modeling for 3d scene understanding. *Computers & Graphics*, 55, 55-67.
- [36] Gross, R., Matthews, I., Cohn, J., Kanade, T., & Baker, S. (2010). Multi-pie. *Image and Vision Computing*, 28(5), 807-813.
- [37] Paysan, P., Knothe, R., Amberg, B., Romdhani, S., & Vetter, T. (2009, September). A 3D face model for pose and illumination invariant face recognition. In Advanced video and signal based surveillance, 2009. AVSS'09. Sixth IEEE International Conference on (pp. 296-301). Ieee.
- [38] Schonberger, J. L., & Frahm, J. M. (2016). Structure-from-motion revisited. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4104-4113).
- [39] Shin, D., Fowlkes, C. C., & Hoiem, D. (2018). Pixels, voxels, and views: A study of shape representations for single view 3D object shape prediction. *arXiv preprint arXiv:1804.06032*.
- [40] Soltani, A. A., Huang, H., Wu, J., Kulkarni, T. D., & Tenenbaum, J. B. (2017, July). Synthesizing 3d shapes via modeling multi-view depth maps and silhouettes with deep generative networks. In The IEEE conference on computer vision and pattern recognition (CVPR) (Vol. 3, p. 4).
- [41] Tatarchenko, M., Dosovitskiy, A., & Brox, T. (2017, March). Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In Proc. of the IEEE International Conf. on Computer Vision (ICCV) (Vol. 2, p. 8).

5.1. Implementation Details

We implemented the Autoencoder as proposed by Achlioptas *et al.* in [5]. The encoder consists of 5 layers of 1x1 convolutions with 64, 128, 128, 256 and 128 filters respectively. Each layer is followed by a ReLU non-linearity. This is followed by a global max pooling operation leading to a bottleneck size of 128. The decoder consists of three fully-connected layers leading to hidden layers of size 256, 256 and 6144 respectively. The output of the last layer is reshaped to 2048×3 to get the output point cloud. We use the same architectures for the Autoencoder and Denoising Autoencoder. Figure 8 shows more details about the specific architecture. When feeding an incomplete point cloud we simply remove those points from the input. Since the operations performed are symmetric to all points and agnostic to the number of point clouds, the architecture works as is for incomplete point clouds as well. The AEs were trained using an optimized approximate implementation of the EMD loss, as used by Fan *et al.* in [3]. We also use EMD of the completed point clouds against the respective ground truth to evaluate our models. We use this AE to generate a set of global feature vectors for all the point clouds in the training set. We then train a W-GAN to be able to generate samples from this distribution of global feature vectors. The Generator samples from a unimodal gaussian noise distribution of 128 dimensions. It consists of two fully connected layers of size 128 each, and outputs a global feature vector of size 128. The discriminator takes a 128 dimension vector at its input. It has 2 fully connected layers of size 256 and 512 followed by a final layer which outputs one value and a sigmoid activation, resulting in a prediction of whether the input vector was real or fake. The initialization encoder is trained to map each GFV to a corresponding latent vector. It also consists of 2 fully connected layers of size 128 each. It takes the GFV as input and outputs the 128 dimensional latent vector which is taken as input by the Generator and mapped back to the corresponding GFV. The GAN is trained with a learning rate of 0.0001 and an ADAM optimizer, for 200 epochs. We use the same learning rate with ADAM optimizer for LDO as well. Initial values of λ , β and α are 0.1, 0.1, 0.001 respectively. λ and β are further decayed by a factor of 0.999 after every update.

6. DAE overfitting

We observe that the DAE overfit to the specific noise type they are trained on. If we train the DAE with a masking of 60% and test it on lower levels of masking, its performance decreases. This can be seen in Figure 10. Taking a closer look at the completion results of the DAE, we find that the DAE has simply learnt a fixed mapping in the training set from masked clouds to the completed point clouds. When exposed to unseen data such as the test set, it simply produces

a nearest neighbor from the training set. This can be seen in Figure 9.

6.1. Variation with masking

Figure 11 shows the performance of AE and AE+LDO with varying levels of masking on the Airplane category. The deterioration of performance with increasing levels of masking is clearly visible in the AE but the optimization procedure still manages to generate reasonable looking point clouds. This shows the robustness of our approach to different kinds of point cloud deformations.

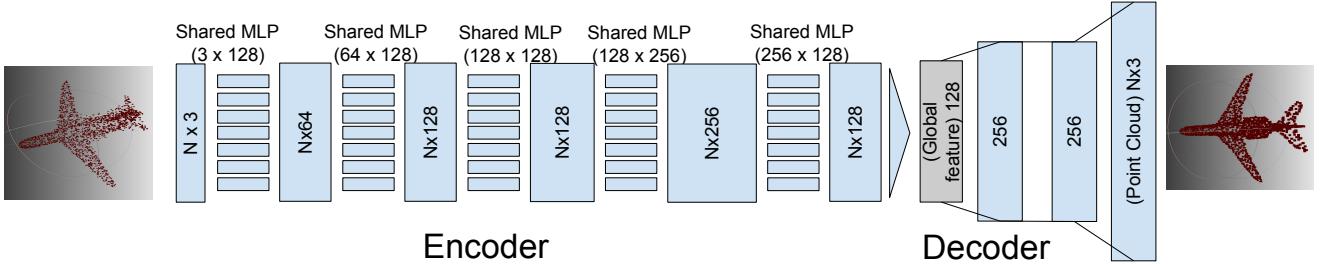


Figure 8. **Autoencoder Architecture.**

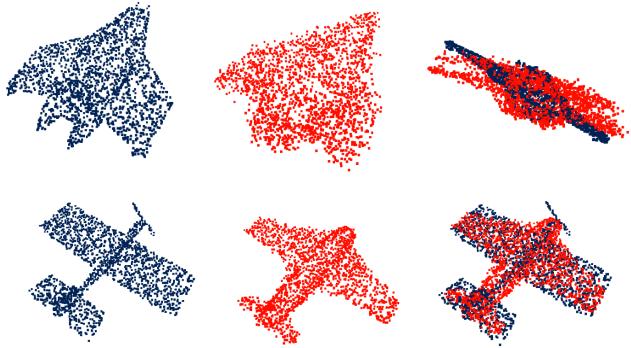


Figure 9. **DAE failure cases.** The Denoising Autoencoder is notoriously prone to overfitting on the training data. Overfitting increases with the percentage of the incompleteness with which they are trained. Column 1 has complete point clouds from the test set which were fed to the DAE without any masking. Even with a complete cloud as input, the DAE outputs a different point cloud (Column 2), since it essentially collapses to a Nearest Neighbour search against the training data. The difference between input and DAE output is visualized in column 3. Due to lack of large datasets for 3D, this is a huge disadvantage of the plain DAE. We also note in our experiments that a DAE trained with higher percentages of missing data performs poorly on lower percentages of corruption.

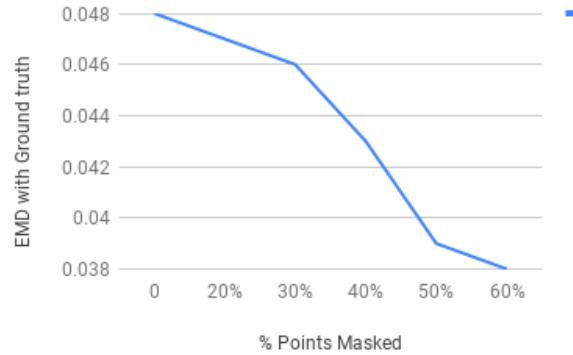


Figure 10. The plot shows the performance of Denoising Autoencoder(trained on point clouds with 60% masking) with different levels of masking. The plot shows that the DAE performs worse as the masking is reduced. Infact, it gives the worst performance when the ground truth (0% masking) is given as input.

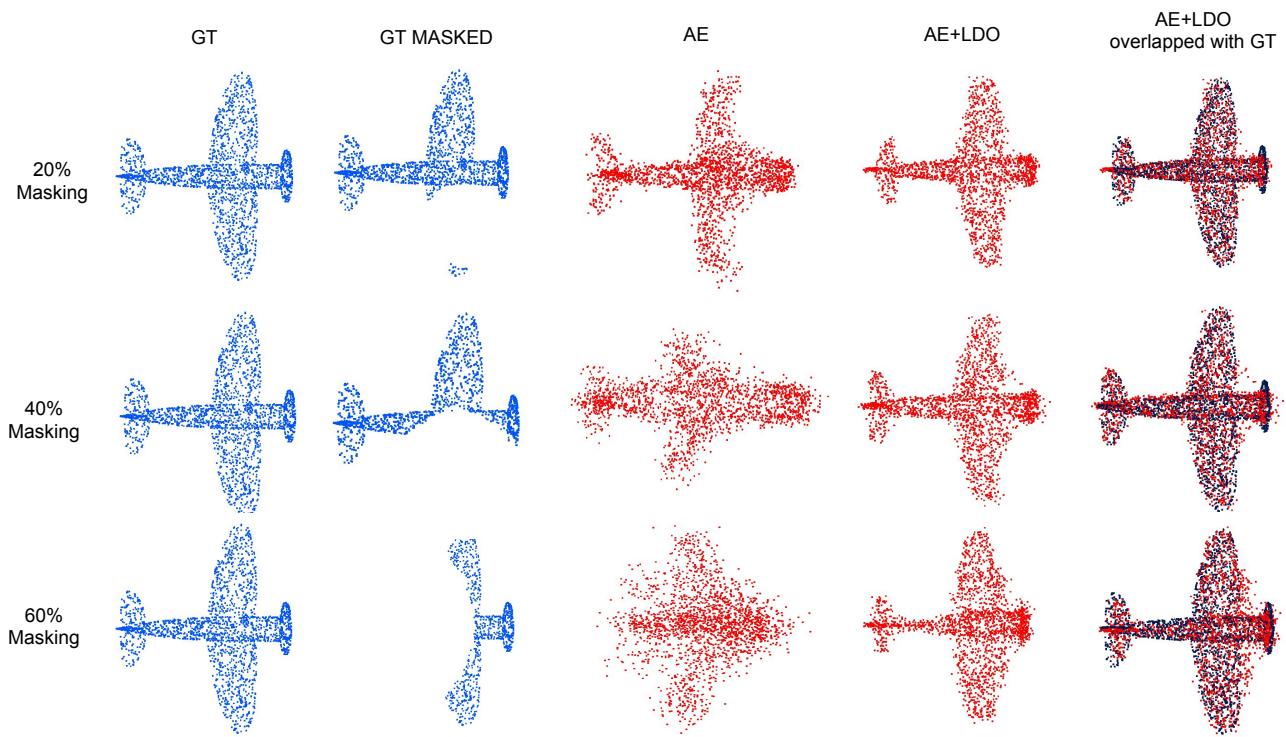


Figure 11. The figure shows the performance of AE and AE+LDO with varying levels of masking (20%, 40% and 60%). The AE in this case is trained specifically on the Airplane dataset.

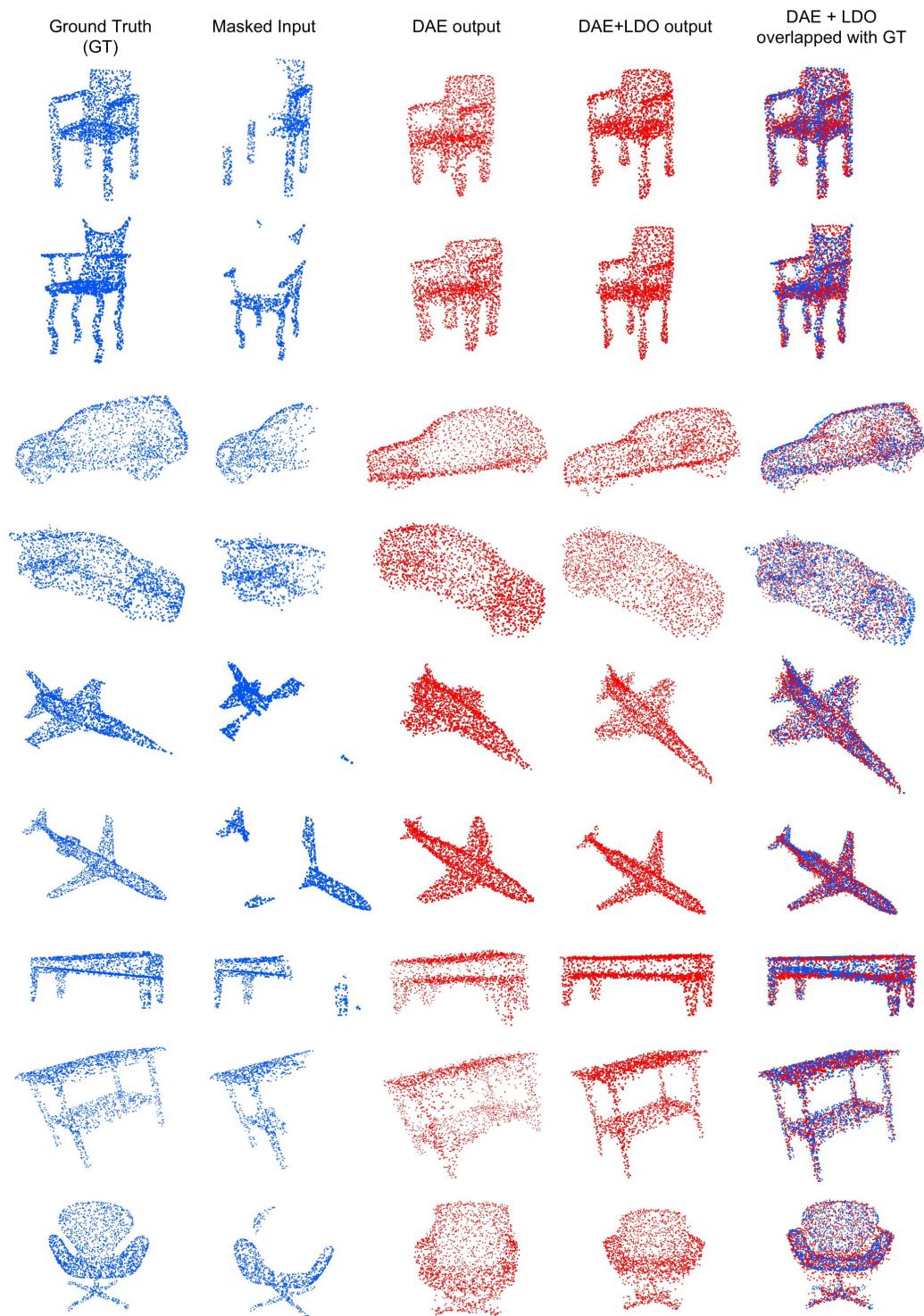


Figure 12. Enlarged version of Fig. 3 results in main paper, showing reconstructions of Multi-Class DAE and Multi-Class DAE+LDO

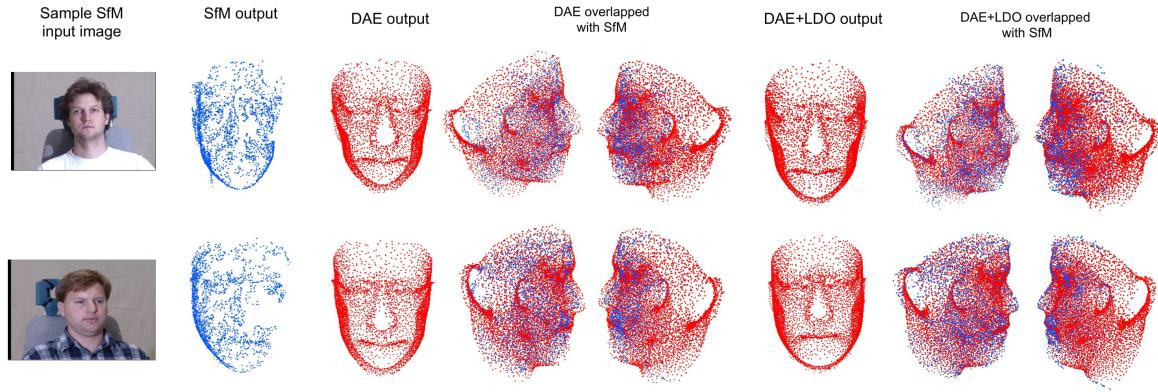


Figure 13. Profile views of reconstructions of the faces shown in Fig. 7 (SfM data) results in main paper, showing reconstructions of Face DAE and Face DAE+LDO (8k points)

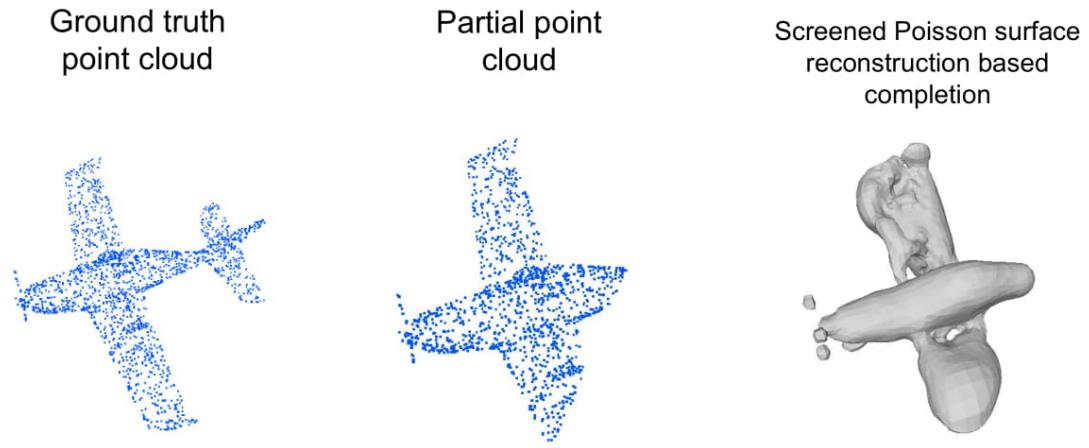


Figure 14. Geometric methods such as Screened poisson reconstruction, while effective at removing small holes in surfaces, fail on large missing regions.