Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

Article

Contrasitive Learning for 3D Point Clouds Classification and Shape Completion

Danish Nazir [1,2,†], Muhammad Zeshan Afzal [1,2,3,*,†], Alain Pagani [3], Marcus Liwicki [4], and Didier Stricker [1,3]

1 Department of Computer Science, Technical University of Kaiserslautern, 67663 Kaiserslautern, Germany; danish.nazir@dfki.de (M.A.); didier.stricker@dfki.de (D.S.)
2 Mindgrage, Technical University of Kaiserslautern, 67663 Kaiserslautern, Germany
3 German Research Institute for Artificial Intelligence (DFKI), 67663 Kaiserslautern, Germany; alain.pagani@dfki.de
4 Department of Computer Science, Luleå University of Technology, 971 87 Luleå, Sweden; marcus.liwicki@ltu.se
* Correspondence: muhammad_zeshan.afzal@dfki.de (M.Z.A.)
† These authors contributed equally to this work.

Abstract: In this paper, we present the idea of Self Supervised learning on the Shape Completion and Classification of point clouds. Most 3D shape completion pipelines utilize autoencoders to extract features from point clouds used in downstream tasks such as Classification, Segmentation, Detection, and other related applications. Our idea is to add Contrastive Learning into Auto-Encoders to learn both global and local feature representations of point clouds. We use a combination of Triplet Loss and Chamfer distance to learn global and local feature representations. To evaluate the performance of embeddings for Classification, we utilize the PointNet classifier. We also extend the number of classes to evaluate our model from 4 to 10 to show the generalization ability of learned features. Based on our results, embedding generated from the Contrastive autoencoder enhances Shape Completion and Classification performance from 84.2% to 84.9% of point clouds achieving the state-of-the-art results with 10 classes.

Keywords: 3D point Cloud Classification, 3D point Cloud Shape Completion,Auto-Encoders, Contrastive Learning, Self-Supervised Learning

Citation: Nazir, D; Afzal, M.Z., Pagani, A.; Liwicki, M.; Stricker, D. Title. *Journal Not Specified* **2021**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Submitted to *Journal Not Specified* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Performing tasks on point clouds are considered more challenging than 2D images. It is because 2D images live on a nice 2D grid with regular spacing in an Image Plane, and in contrast, point clouds are represented by a list of sparse 3D Cartesian points. Point clouds are spread in 3D Coordinate System and lack the required topological information. It is also challenging to apply typical deep learning methods like Convolutional Neural Networks directly on point clouds to learn feature representations because Convolution operation requires its neighbouring samples to appear at some fixed spatial orientations and distances to facilitate Convolution which is not the case in point clouds.

Recently, Graph-based methods have been very successful in learning point cloud representations task, and Dynamic Graph Convolutional Neural Networks (DGCNN) [1] are introduced, which aims to recover the topology of the point clouds it can extract rich representations. The major drawback of GCNNs based methods is that they have millions of learnable parameters, making them vulnerable to over-fitting. We need large-scale annotated datasets for training GCNN's in order to have a generalizable solution to our shape completion problem. However, unlike Images, there are very limited datasets available for point clouds. The collection and the annotation of the new point cloud dataset are time-consuming and expensive since pixel-level annotations are needed. With their powerful ability to learn useful representations from unlabeled data,

© 2021 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

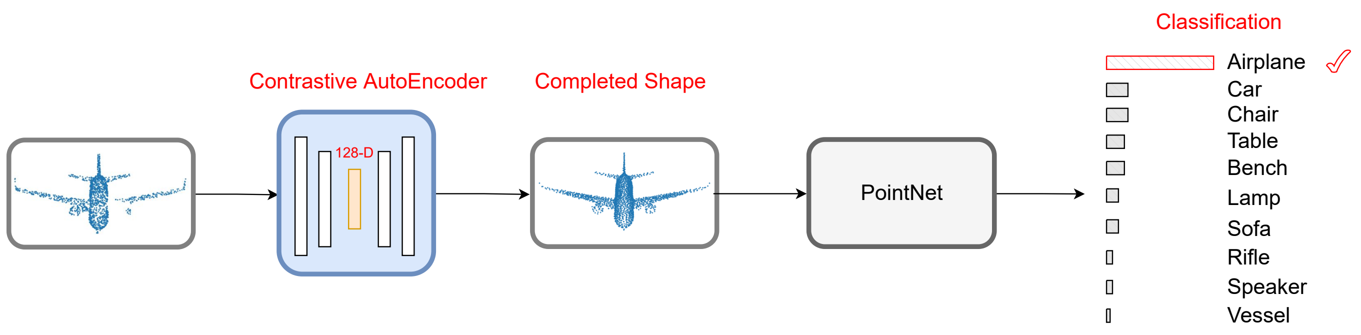


Figure 1. The incomplete shapes are passed through Contrasitive AutoEncoder which completes the shape. The completed shapes are sent to pre-trained PointNet for classification.

unsupervised learning methods, sometimes known as self-supervised learning methods, have drawn significant attention.

In this work, we focus on extracting representations of point clouds which are helpful in downstream tasks. We choose Classification as a downstream task and use PointNet [2] to measure the Classification accuracy. The classical approach to this problem is to voxelize point clouds to extract features, but this is computationally expensive, and representation accuracy is not enough, and we lose information. We present Contrastive Auto Encoder which is optimized over Contrasitive loss [3][4][5][6][7][8] and a Distance function. Contrastive Learning is a variant of Self Supervised Learning. The intuition behind Contrastive Learning is that objects belonging to the same class should be closer in feature space than objects of a different class. It will also work if we have limited labelled 3D data. Our network focuses on learning both global and local representations of the point clouds. Representations learned from the autoencoder are utilized to classify incomplete shapes.

Our key contributions are as follows:

- We train an end-to-end deep Contrasitive auto-encoder that consumes point clouds directly.
- To the best of our knowledge, in Self Supervised realms, the combination of local and global features has not been investigated much in the context of point cloud shape completion and Classification. It is used only to extract global features from point clouds and apply them directly in downstream tasks. We are the first to explore this combination.
- We extend our idea to shape a completion task to complete corrupt and noisy point clouds. The results show that our method can help in that task as well.
- We increase the number of classes for evaluation from 4 to 10.

The forward pass of the proposed network is given in Figure 2.

2. Related Works

Point clouds have obtained increased attention over the past few years. We will review related point cloud analysis and shape completion methods.

2.1. Point Cloud Analysis

There exist two subcategories for Point Cloud analysis methods.

2.1.1. Supervised Learning Methods

Supervised learning methods require labeled data of 3D point clouds. The types of methods which are used in this domain are given below.

Traditional Methods: They include hand-crafted-based techniques.They aim to capture local geometric structure information of point clouds such as intrinsic [9][10] or

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

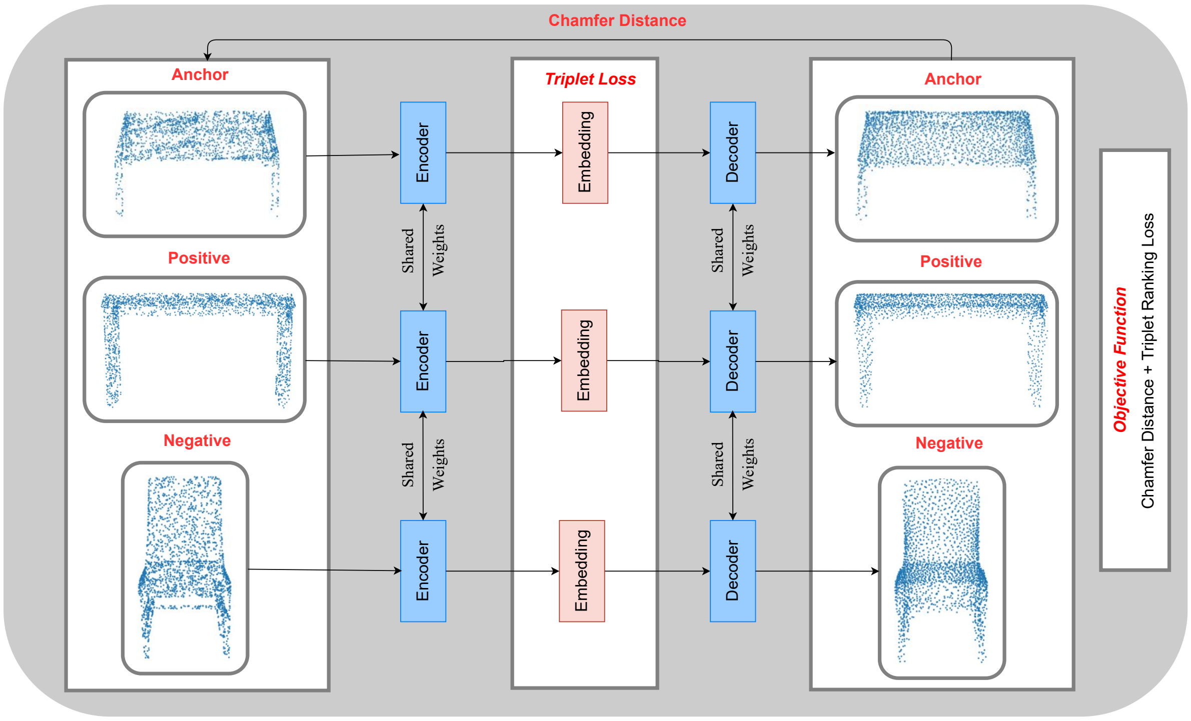


Figure 2. Forward Pass of the proposed Contrasitive Auto Encoder. The triplets are passed through the Encoder to transform high-dimensional point clouds to lower-dimensional feature representations of the point clouds. Triplet loss between Anchor, Positive and Negative feature representations is calculated, and they are sent to the decoder, which reconstructs the point clouds. Finally, the reconstruction loss is calculated between decoded and input point clouds. The objective function is formulated as the sum of these two losses. The network is trained in an end-to-end manner.

extrinsic feature descriptors [11][12]. These methods are limited in terms of performance and include manual work.

Convolutional Methods: We can also extend 2D Convolutions ideas to 3D and utilize them for feature extraction. Convolution-based methods have better performance on downstream tasks than traditional handcrafted features. There are various ideas to use CNN with point clouds. One way is to perform voxelization of 3D point clouds which means that we create a voxel structure of $[X \times Y \times Z]$ and then we can perform convolution operation using filters of size $[x \times y \times z]$ with $[x, y, z] \leq [X, Y, Z]$ [13] [14],[15],[16]. However, these methods have a quantization effect and have high computational costs. MultiView methods [17] [18] [19] is another way of applying CNN's to 3D data. The idea behind MultiView methods is to convert 3D data to 2D and then apply existing CNN techniques. The accuracy of MultiView methods is greater than voxel-based methods because they make use of existing state-of-the-art 2D techniques and extend them to 3D data.

Graph based Methods: The idea behind Graph-based Methods is to capture the local structure of the point clouds. They represent each point in the point cloud by a node, allowing us to model correlations within the point clouds. Deep Graph-based Convolutional Networks (DGCNN) [1] proposes a dynamic graph edge convolution on point clouds. The neighbours of a point are selected based on the distances of different point features. It applies EdgeConv to nearest k neighbours calculated by KNN in the metric space.

Learning from raw 3D point clouds: In other methods, e.g., in Convolutional based, we first had to calculate intermediate representation such as voxels or 2D grids and then learn features, but instead of computing intermediate representation, we can also learn features directly from unstructured raw point clouds.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

PointNet [2] is the first method to process unstructured point clouds directly. It individually processes each point in 3D data, and therefore disarrangement in the point clouds will not affect the model. However, as a consequence, PointNet will not be able to utilize the local structure of the 3D point cloud. To counter this, PointNet++ [20] was introduced later, which applied PointNet recursively on a nested partitioning of the input point set.

2.1.2. Unsupervised Learning Methods

Labelled 3D point cloud data is often hard to obtain for new applications. This fact allowed the boom of many Unsupervised learning methods which learn features from unlabeled data to solve this problem. One of the solutions to this problem is to introduce Pretext task learning. In Pretext task learning, our goal is to learn global features by performing a mainstream task, e.g. Orientation prediction [21] and Reconstruction [22] of point clouds. In Orientation estimation, we rotate point clouds and train the classifier to predict the rotation angle, whereas, in Reconstruction, we learn by applying reconstructions of generated deformed point cloud shapes.

Developing further the idea of pretext task, Contrast-Net [23] was introduced, which uses the concept of contrastive learning. The idea is to give Positive and Negative pairs to DGCNN and apply Pairwise ranking loss. Positive pairs are generated by selecting samples randomly and making sure they are from the same class, whereas Negative pairs are generated using the same technique but ensuring that Positive and Negative pairs represent different classes. We utilize learned representations in the Classification downstream task. LatentGANs [24] introduced a new deep auto-encoder network with state-of-the-art reconstruction quality and generalization ability for point cloud data. FoldingNet [25] is an end-to-end autoencoder that is state-of-the-art for unsupervised feature learning on point clouds. They introduced a Graph-based enhancement applied to the encoder to enforce local structures on top of PointNet, and a folding-based decoder deforms a canonical 2D grid onto the underlying 3D object surface of a point cloud.

2.2. Shape Completion

3D point cloud shape completion [26][27][28] is very challenging problem in Computer Vision. The applications of shape completion range from robotics to autonomous vehicles, and there has been significant development of methods in the field. Earlier approaches [26], [27], [29] to shape completion were inspired by the performance of 2D CNN operations on images, and they extended the idea and introduced 3D Convolutions on voxels. They have produced promising reconstruction results, However voxels data representation have memory constraints and cannot produce high-resolution outputs. Recent works have gradually discarded the voxel format, and now they are moving towards meshes, i.e., they use mesh format to represent the 3D point cloud shapes. Mesh representation allows the consumption of point clouds directly into the network without dealing with any intermediate representation. It provides a significant advantage over voxel representation. Therefore, we also use the meshes format to represent our point cloud shapes. We compare our shape completion results with RLGAN [30] framework, which uses Reinforcement learning-based agents and autoencoder to complete shapes. It uses the same backbone as ours in autoencoder, but it works with only four classes. Our method extends the number of classes and shows that it is more robust in completing different shapes.

3. Methods

We used autoencoder [31] as our backbone network to process unstructured 3D point clouds. We added more Convolutional layers into the backbone network and changed the objective function to adapt to our problem. It helped us achieve much stable and efficient training, and it also helped us get better reconstruction quality.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

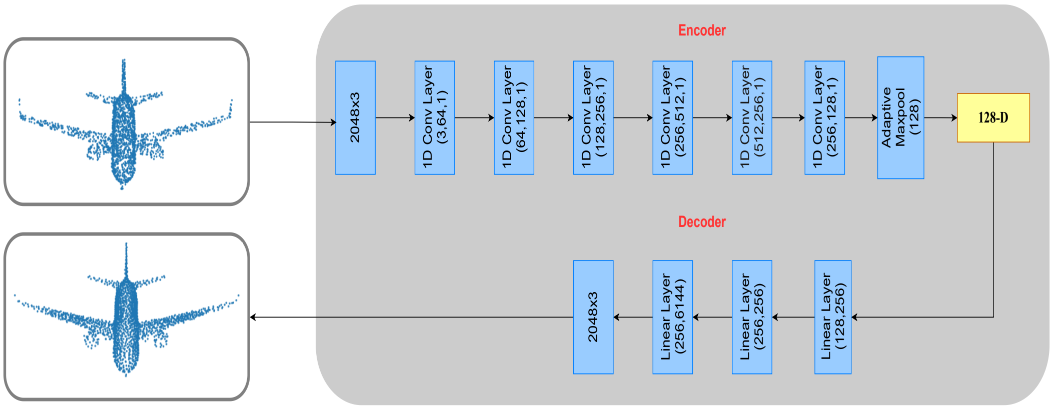


Figure 3. Auto Encoder architecture. The Encoder consists of 1D Convolutional layers which are applied until we get $128 - D$ embedding. Batch Normalization of input channel size is applied to each layer. The Decoder consists of only Fully Connected layers and maps the embedding to full point cloud of shape (2048×3).

3.1. AutoEncoder

AutoEncoder (AE) is an unsupervised method in which we learn feature representation of the data through two networks, i.e., Encoder and Decoder. The Encoder converts the complex input into encoded representation, whereas the Decoder reverts the encoded version to the original dimension. The loss is calculated between the output of the Decoder and input. However, to extract global representations, we also introduced Contrasitive loss at the encoded representations of the data. The following sections give more details about the objective function of our AutoEncoder.

3.1.1. Objective Function

We use a combination of a Distance function and Contrastive loss to train our network. From the Distance function, we want to extract local features of the point clouds, whereas by using Contrastive loss, we want to extract global features.

Distance Function: Distance functions that are suitable to our cause are Earth Mover's distance (EMD)[32], and Chamfer's distance [33]. We chose Chamfer distance because it gave us much better reconstruction quality. We use symmetric Chamfer distance to measure the quality between Input and Encoder's generated feature representations. Chamfer distance can be defined as follows.

$$\begin{aligned} \mathcal{L}(\mathbf{P_1}, \mathbf{P_2}) = &\sum_{a \in \mathbf{P_1}} min_{b \in \mathbf{P_2}} ||a - b||_2^2 \\ &+ \sum_{b \in \mathbf{P_2}} min_{a \in \mathbf{P_1}} ||a - b||_2^2 \end{aligned} \tag{1}$$

Where $\mathbf{P_1} \in \mathbb{R}^{2048 \times 3}$ denotes Input point cloud from training set and $\mathbf{P_2} \in \mathbb{R}^{2048 \times 3}$ denotes Encoder's generated point cloud representation which is decoded using Decoder. Our loss function consists of a Distance function and Constrasitive loss.

Contrasitive Loss: There are two main ideas for loss functions in Contrasitive learning space on which we can train our Network.

The first idea is Pairwise Ranking loss, which is also used in related work [23]. It uses use pairs of positive and negative training data points. Positive pairs are formulated by choosing an anchor sample and a positive sample similar to an anchor. Negative pairs consist of an anchor sample and a negative sample dissimilar to an anchor. The objective is to learn representations with smaller distance between them for positive pairs and greater distance than some margin values for negative pairs.

The second idea is Triplet Ranking loss [3][4][5][6][7][8] which uses the concept of Triplets. A Triplet is formed by an anchor sample, a positive sample, and a negative sample. A positive sample means that it should be similar to an anchor whereas a

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

negative sample means the opposite. This setup outperforms Pairwise Ranking loss, and also it is easier to optimize on. Hence we are using Triplet Ranking loss in our setup. Triplet Ranking loss can be defined as follows.

$$\begin{aligned}\mathcal{L}(\mathbf{r_a}, \mathbf{r_p}, \mathbf{r_n}) = max \Bigg(0\,,\; m + \sum_{i \in \mathbf{r_a}, j \in \mathbf{r_p}} ||r_a^i - r_p^i||_2^2 \\ - \sum_{i \in \mathbf{r_a}, j \in \mathbf{r_n}} ||r_a^i - r_n^i||_2^2 \Bigg)\end{aligned} \tag{2}$$

Where $\mathbf{r_a} \in \mathbb{R}^{2048\times3}$ denote anchor sample, $\mathbf{r_p} \in \mathbb{R}^{2048\times3}$ denote positive sample, $\mathbf{r_n} \in \mathbb{R}^{2048\times3}$ denote negative sample and $m \in \mathbb{R}^d$ denotes minimum margin between positive and negative samples. We use l_2 distance metric and $d(\mathbf{r_a}, \mathbf{r_p})$ represents distance between representations of anchor and positive sample and $d(\mathbf{r_a}, \mathbf{r_n})$ represents distance between representations of anchor and negative sample.

Let's analyze the three different situations of Equation 2.

- $d(\mathbf{r_a}, \mathbf{r_n}) > d(\mathbf{r_a}, \mathbf{r_p}) + m$: This means that negative samples are already sufficiently distant to anchor samples in the embedding space which will in turn cause loss to 0 and the network will not learn anything. This kind of Triplets are known as Easy Triplets.
- $d(\mathbf{r_a}, \mathbf{r_p}) > d(\mathbf{r_a}, \mathbf{r_n})$: Negative samples are closer to anchor than positive samples and this will cause loss to be positive and greater than margin m. The network will then learn something and such Triplets are known as Hard Triplets.
- $d(\mathbf{r_a}, \mathbf{r_p}) < d(\mathbf{r_a}, \mathbf{r_n}) < d(\mathbf{r_a}, \mathbf{r_p}) + m$: Negative samples are more distinct to the anchor but the distance is not greater than margin m. Hence the loss is still positive, encouraging the network to learn something but less than m. They are known as Semi-hard Triplets

3.1.2. Proposed Loss

The objective of our method is to encourage learning global and local feature representations. Using Equation 1 and 2, we define a unique loss function which aligns with our objective.

$$\begin{aligned}\mathcal{L}(\mathbf{r_a}, \mathbf{r_p}, \mathbf{r_n}) = \mathcal{L}_{triplet}(\mathbf{r_a}, \mathbf{r_p}, \mathbf{r_n}) + \mathcal{L}_{ch}(\mathbf{r_a}, dec(\mathbf{e_a})) \\ + \mathcal{L}_{ch}(\mathbf{r_p}, dec(\mathbf{e_p})) + \mathcal{L}_{ch}(\mathbf{r_n}, dec(\mathbf{e_n}))\end{aligned} \tag{3}$$

where embeddings $\mathbf{e_a}$,$\mathbf{e_p}$ and $\mathbf{e_n} \in \mathbb{R}^{128}$ are generated using encoder for anchor, positive and negative samples. The representations are sent to decoder to get full point cloud.

3.2. Implementation Details

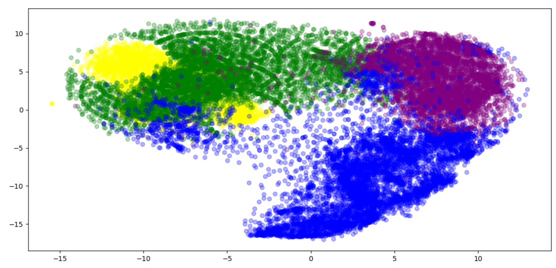
In the forward pass of the network, It measures Triplet Ranking loss and Chamfer distance, and then back-propagation is performed on the sum of these two losses as shown in Equation 3.

Computing Chamfer distance is straightforward. However, Triplet Ranking loss requires some careful design decisions. Triplet mining for Triplet Ranking loss is an essential factor, and chosen strategy for mining will have a high impact on training efficiency and final performance.

We should avoid training with Easy Triplets as much as possible since the resulting loss would be 0, which means no learning. There are 2 strategies for choosing triplets.

- **Offline Triplet Mining:** The traditional way of defining triplets is defining them either at the beginning of the training or after every epoch. .

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1


(a) Four Classes.

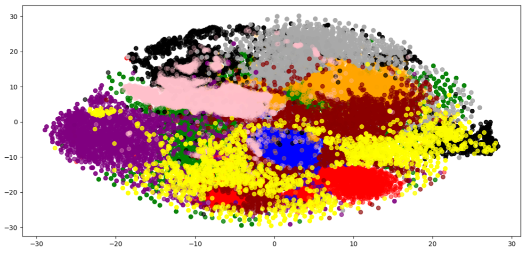

(b) Ten Classes.

Figure 4. The visualizations are made by projecting the dataset into two dimensional space. Part **a)** shows the dataset visualization of 4 classes whereas **b)** shows the visualization of 10 classes.

- **Online Triplet Mining:** The latest approach to triplet mining is that we define triplets for every batch during the training. It is better than Offline triplet mining and results in better training efficiency and higher performance.

In our training pipeline, we choose Online Triplet mining for the computation of Contrastive loss as it has better performance than the rest. This strategy also discourages Easy Triplets, and it forces the network to learn Contrasitive representations.

4. Experiments and Results

We used PyTorch [34] for the implementation and training of our network. We wanted to observe the effect of Contrastive learning on shape completion pipelines, and therefore we compare our results with autoencoders trained without Contrasitive learning. We also evaluate whether Contrasitive learning helps in the Classification of incomplete point clouds after completing shapes from our proposed autoencoder.

We refer to Naive autoencoder to an autoencoder with the same architecture as a Contrastive autoencoder, but it is trained only with the Chamfer distance function. In the first part of our experiments, we choose four classes that are the same as used by RLGAN framework [30]. We trained naive autoencoder along with the RLGAN framework and our proposed autoencoder. After training, we calculate Classification accuracy using PointNet on shapes completed by the methods mentioned earlier.

4.1. Dataset

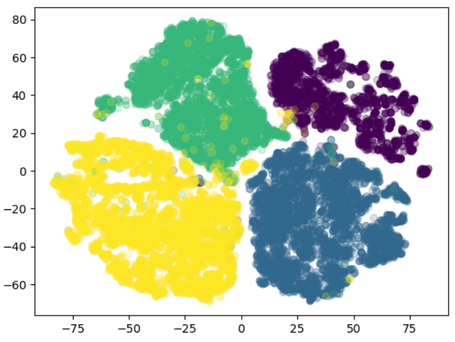
All of the experiments are done on two samples of ShapeNetCore dataset [35], that have four and ten classes based on having the most number of shapes. We chose the four classes: Airplane, Chair, Table, and Car, we then extend this set and add six more classes, Bench, Lamp, Speaker, Rifle, Sofa, and Vessel, to make ten classes dataset.

Due to a high imbalance in the dataset, we replicate samples so that all of the classes have approximately the same number of samples. The total number of shapes for the four classes dataset sums up to 22803, and for ten classes, it sums up to 70000 samples in our dataset. Each shape contains 2048 points, hence making our data-set shape of $(22803, 2048, 3)$ and $(70, 000, 2048, 3)$ respectively.

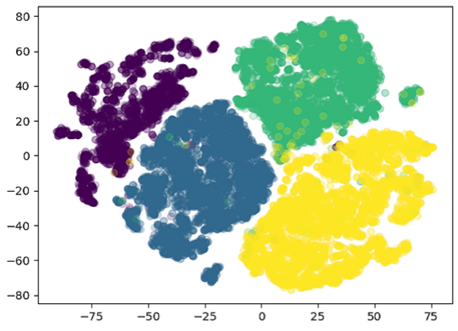
Before applying Contrasitive learning, we wanted to visualize the TSNE [36][37][38] plot of our dataset. TSNE applies dimensionality reduction to the data such that we can visualize High dimensional data. . In the case where we have four classes, we transformed $(22803, 6144)$ dimensional data to $(22803, 2)$ whereas, in the case of ten classes, we transformed $(70000, 6144)$ dimensional data to $(70000, 2)$. We wanted to ensure that the data is not linearly separable because if it is, then it does not make any sense to apply Contrasitive learning to such a dataset.

Figure 4 shows the TSNE plots. Figure 4a shows that there are somewhat clear separation between the classes. From the ten classes plot given in Figure 4b, we can infer

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

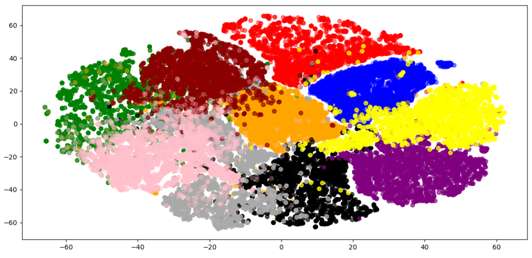


(a) Contrasitive AutoEncoder.

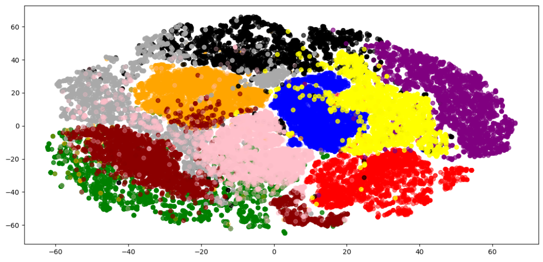


(b) Naive AutoEncoder.

Figure 5. Embedding Visualization of dataset with 4 classes. Part **a)** shows the learned Contrasitive AutoEncoder i.e. with Contrasitive loss embeddings visualization whereas **b)** shows the visualization of the Naive AutoEncoder i.e. without contrasitive loss. Constrasitive AutoEncoder embeddings are similar to naive autoencoder.



(a) Contrasitive AutoEncoder.



(b) Naive AutoEncoder.

Figure 6. Embedding Visualization of dataset with 10 classes. Part **a)** shows the learned Contrasitive AutoEncoder i.e. with Contrasitive loss embeddings visualization whereas **b)** shows the visualization of the AutoEncoder without contrasitive loss.

that Contrasitive learning can be applied to this dataset as the boundaries are not clearly defined for each class.

4.2. Embeddings

Embeddings produced by all of the methods which we are using in this work are 128 dimensional for each shape. In the case where we have four classes, we transformed the embeddings from $(22803, 128)$ to $(22803, 2)$. On the other hand, for ten classes we transformed $(70000, 6144)$ dimensional data to $(70000, 2)$. We use TSNE to apply the transformations mentioned above. It also allows us to compare the embeddings plot with the original dataset.

Figure 5a and 6a shows the embeddings generated by Contrasitive autoencoder and the boundaries for each class are very well defined except for two classes. We investigated this behavior and found out that the shapes for these classes are similar and challenging to distinguish even for human observers. On the other hand, Figures 5b and 6b represents the embeddings learned by autoencoder trained only on Chamfer distance.

4.3. Hyper-parameters and Optimization:

We use Adam optimizer along with a learning rate of 0.001 and weight decay of 0.001 which helps us to stabilize Constrasitive loss. We also use a learning rate scheduler to further reduce learning rate after $[100, 175, 250, 400$
$, 800]$ epochs. The optimal value for the margin m of Triplet Ranking loss for our data-set

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

is 0.5 and we use l_2 norm as our distance measure between anchor, positive and negative representations.

4.4. Shape Completion

We generated test data shapes with 20% missing points to test our autoencoder shape completion capabilities. It is done by randomly removing 20% 3D points from all test shapes. After that, we passed these shapes from the shape completion pipeline, as shown in Figure 7, and save completed shapes. The quantitative results i.e. mean Chamfer distance per point for completed shapes is given in Table 1 whereas the qualitative results is given in Table 3.

Table 1. Quantitative results computed by average Chamfer distance (10^{-4}) between Ground truth and completed shapes with respective methods. Lower the Chamfer distance from the ground truth, better is the completed shape. Naive AutoEncoder performs better than all of the available methods.

RL-GAN	Naive AE	Contrasitive AE
10.67	2.34	4.67

4.5. Classification

After completing shapes using RLGAN, naive autoencoder, and Contrastive autoencoder, we also measure classification accuracy using PointNet. The completed shapes are sent to a pre-trained PointNet which outputs the classes for each completed shape. The results by are shown in Table 2.

When the number of classes is four, naive autoencoder and Contrastive autoencoder almost produces the same classification accuracy. The shapes used in the four classes are semantically very different, making them easier to classify, and using Contrastive learning does not make any significant difference.

Table 2. Classification results on completed shapes with 20% missing data.

No. of Classes	RL-GAN	Naive AE	Contrasitive AE
4	96.12%	**97.34** %	97.31 %
10	58%	84.2%	**84.9** %

On the other hand, for ten classes, the dataset plot in Figure 3 shows that there is little to no separation between the shapes of different classes, and they are also much more semantically similar, e.g., the shape of Car is similar to a certain kind of Vessel, and this makes classification a much more challenging task. In this case, classification accuracy for the Contrastive autoencoder is better than other methods, which shows that Contrasitive learning helps separate the embeddings of similar classes, which then helps in classification, as shown in Figure 5 a.

Last but not least, from Table 1, it is evident that the mean Chamfer distance for Naive autoencoder is slightly lower than other methods, but the overall Classification accuracy for the Contrastive autoencoder is higher than others. It also shows how crucial global feature learning is for downstream tasks.

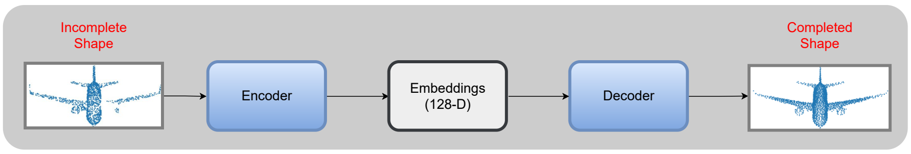


Figure 7. Shape completion pipeline. The incomplete shapes with 1638 × 3 are passed through the Contrasitive autoencoder to receive complete 2048 × 3 shape.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

5. Conclusions

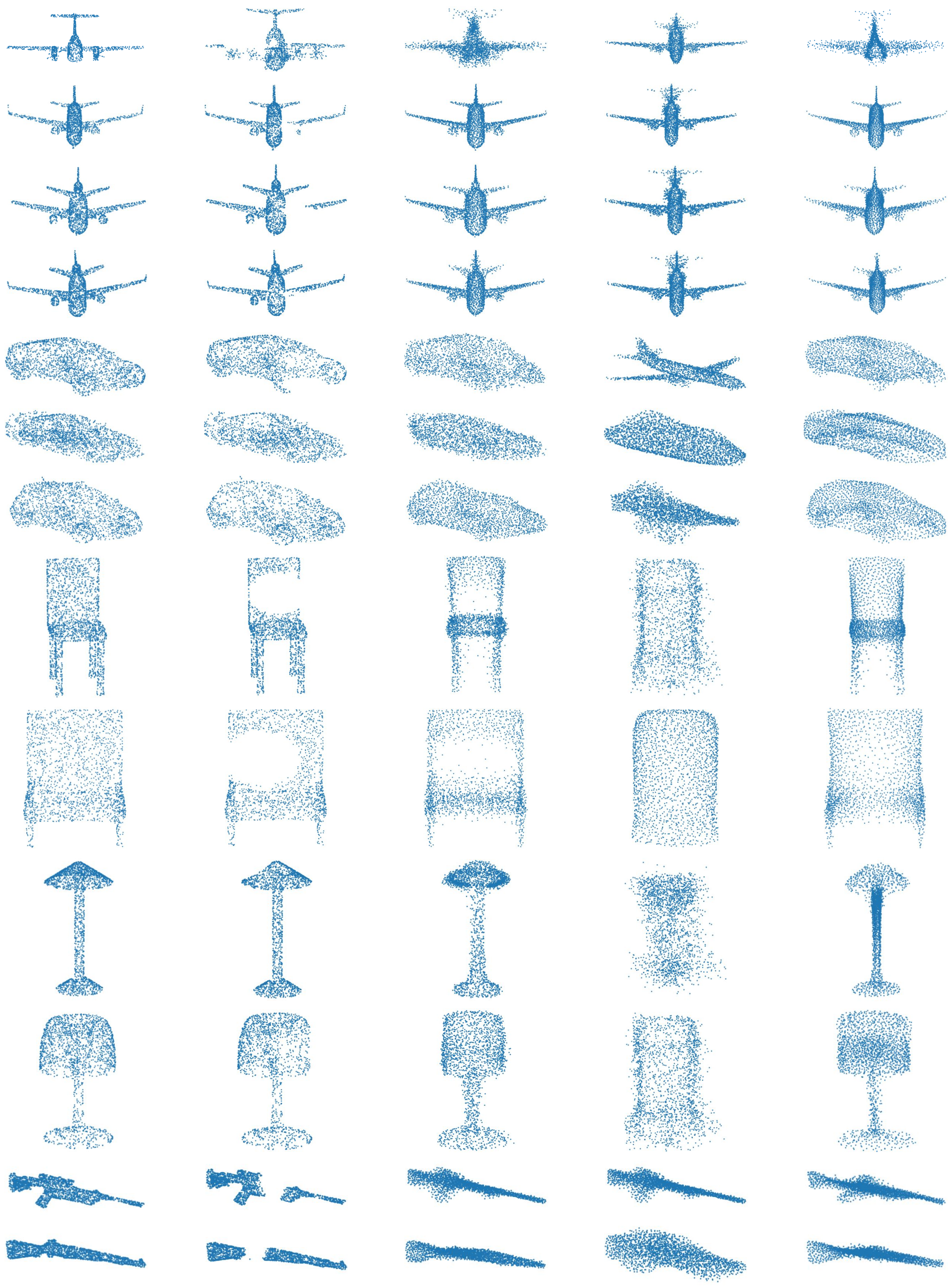
We proposed Contrastive Learning for the 3D point cloud shape completion and classification task. Contrastive learning provides us with the global features of point clouds, and we use Chamfer distance to extract local features. We combined both feature extractors and trained our network to learn both the global and local feature sets. We provide benchmarks on ShapeNetCore [28] dataset for 4 and 10 classes. Our results for both shape completion and classification are very promising, and as a possible extension, we would like to further look into other pretext tasks which help extract more useful global features.

References

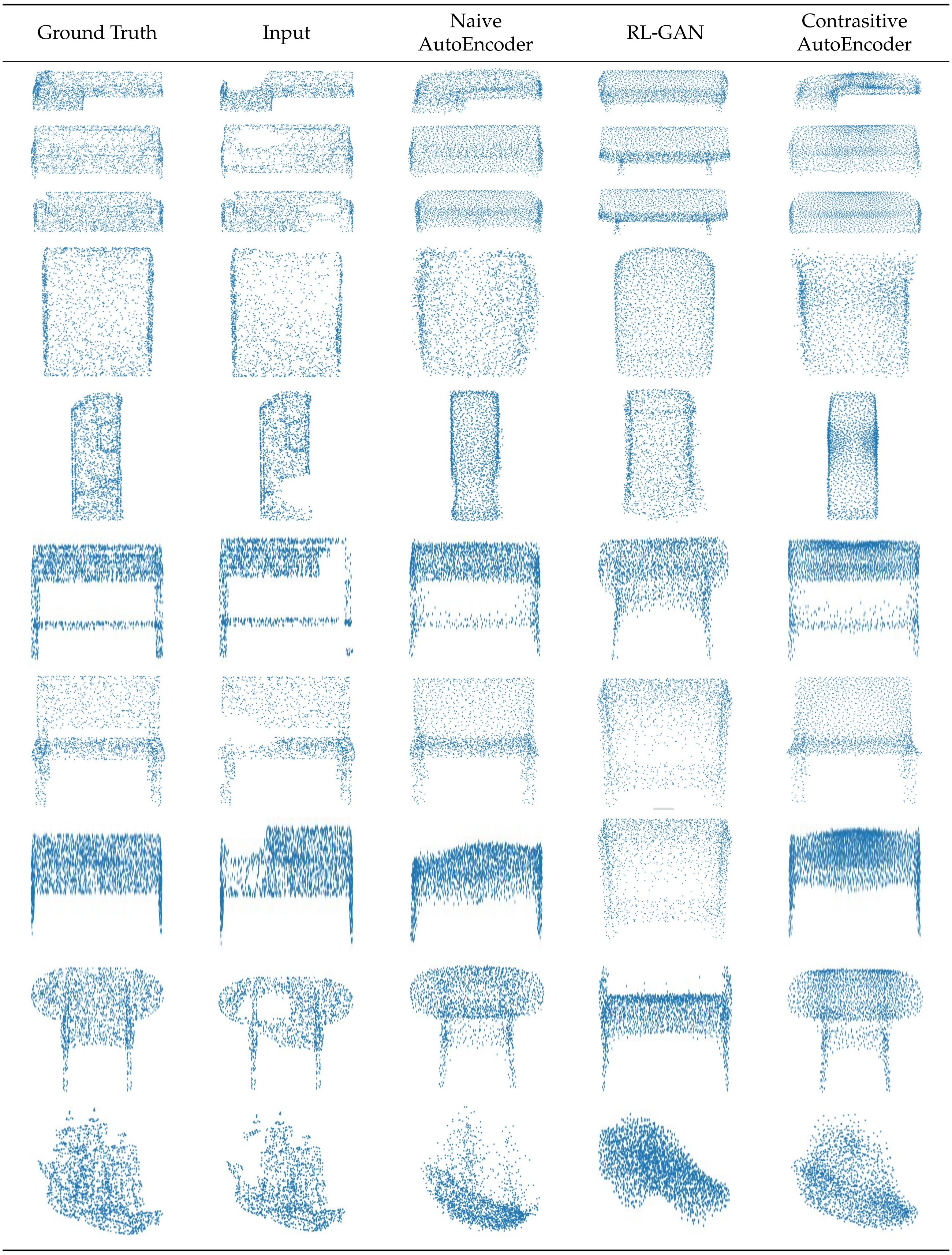
1. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic Graph CNN for Learning on Point Clouds. *ACM Trans. Graph.* **2019**, *38*. doi:10.1145/3326362.
2. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
3. Chechik, G.; Sharma, V.; Shalit, U.; Bengio, S. Large Scale Online Learning of Image Similarity Through Ranking. *Journal of Machine Learning Research, JMLR* **2010**, pp. 1109–1135.
4. Schroff, F.; Kalenichenko, D.; Philbin, J. FaceNet: A Unified Embedding for Face Recognition and Clustering. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.
5. Law, M.T.; Thome, N.; Cord, M. Learning a distance metric from relative comparisons between quadruplets of images. *International Journal of Computer Vision* **2017**, *121*, 65–94.
6. Hoffer, E.; Ailon, N. Deep Metric Learning Using Triplet Network. Similarity-Based Pattern Recognition; Feragen, A.; Pelillo, M.; Loog, M., Eds.; Springer International Publishing: Cham, 2015; pp. 84–92.
7. Hermans, A.; Beyer, L.; Leibe, B. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737* **2017**.
8. Zhou, M.; Niu, Z.; Wang, L.; Gao, Z.; Zhang, Q.; Hua, G. Ladder Loss for Coherent Visual-Semantic Embedding. Proceedings of the AAAI Conference on Artificial Intelligence, 2020, Vol. 34, pp. 13050–13057.
9. Aubry, M.; Schlickewei, U.; Cremers, D. The wave kernel signature: A quantum mechanical approach to shape analysis. 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), 2011, pp. 1626–1633. doi:10.1109/ICCVW.2011.6130444.
10. Bronstein, M.M.; Kokkinos, I. Scale-invariant heat kernel signatures for non-rigid shape recognition. 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010, pp. 1704–1711. doi:10.1109/CVPR.2010.5539838.
11. Rusu, R.B.; Blodow, N.; Beetz, M. Fast Point Feature Histograms (FPFH) for 3D registration. 2009 IEEE International Conference on Robotics and Automation, 2009, pp. 3212–3217. doi:10.1109/ROBOT.2009.5152473.
12. Rusu, R.B.; Blodow, N.; Marton, Z.C.; Beetz, M. Aligning point cloud views using persistent feature histograms. 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008, pp. 3384–3391. doi:10.1109/IROS.2008.4650967.
13. Qi, C.R.; Su, H.; Niessner, M.; Dai, A.; Yan, M.; Guibas, L.J. Volumetric and Multi-View CNNs for Object Classification on 3D Data. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
14. Wang, C.; Cheng, M.; Sohel, F.; Bennamoun, M.; Li, J. NormalNet: A voxel-based CNN for 3D object classification and retrieval. *Neurocomputing* **2019**, *323*, 139–147. doi:https://doi.org/10.1016/j.neucom.2018.09.075.
15. Maturana, D.; Scherer, S. 3d convolutional neural networks for landing zone detection from lidar. 2015 IEEE international conference on robotics and automation (ICRA). IEEE, 2015, pp. 3471–3478.
16. Maturana, D.; Scherer, S. Voxnet: A 3d convolutional neural network for real-time object recognition. 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2015, pp. 922–928.
17. Kalogerakis, E.; Averkiou, M.; Maji, S.; Chaudhuri, S. 3D Shape Segmentation With Projective Convolutional Networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
18. Cao, Z.; Huang, Q.; Karthik, R. 3D Object Classification via Spherical Projections. 2017 International Conference on 3D Vision (3DV), 2017, pp. 566–574. doi:10.1109/3DV.2017.00070.
19. Zhang, L.; Sun, J.; Zheng, Q. 3D Point Cloud Recognition Based on a Multi-View Convolutional Neural Network. *Sensors* **2018**, *18*. doi:10.3390/s18113681.
20. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space, 2017, [arXiv:cs.CV/1706.02413].
21. Poursaeed, O.; Jiang, T.; Qiao, Q.; Xu, N.; Kim, V.G. Self-Supervised Learning of Point Clouds via Orientation Estimation. *arXiv preprint arXiv:2008.00305* **2020**.
22. Sauder, J.; Sievers, B. Self-Supervised Deep Learning on Point Clouds by Reconstructing Space, 2019, [arXiv:cs.LG/1901.08396].
23. Zhang, L.; Zhu, Z. Unsupervised Feature Learning for Point Cloud Understanding by Contrasting and Clustering Using Graph Convolutional Neural Networks. 2019 International Conference on 3D Vision (3DV), 2019, pp. 395–404. doi:10.1109/3DV.2019.00051.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1

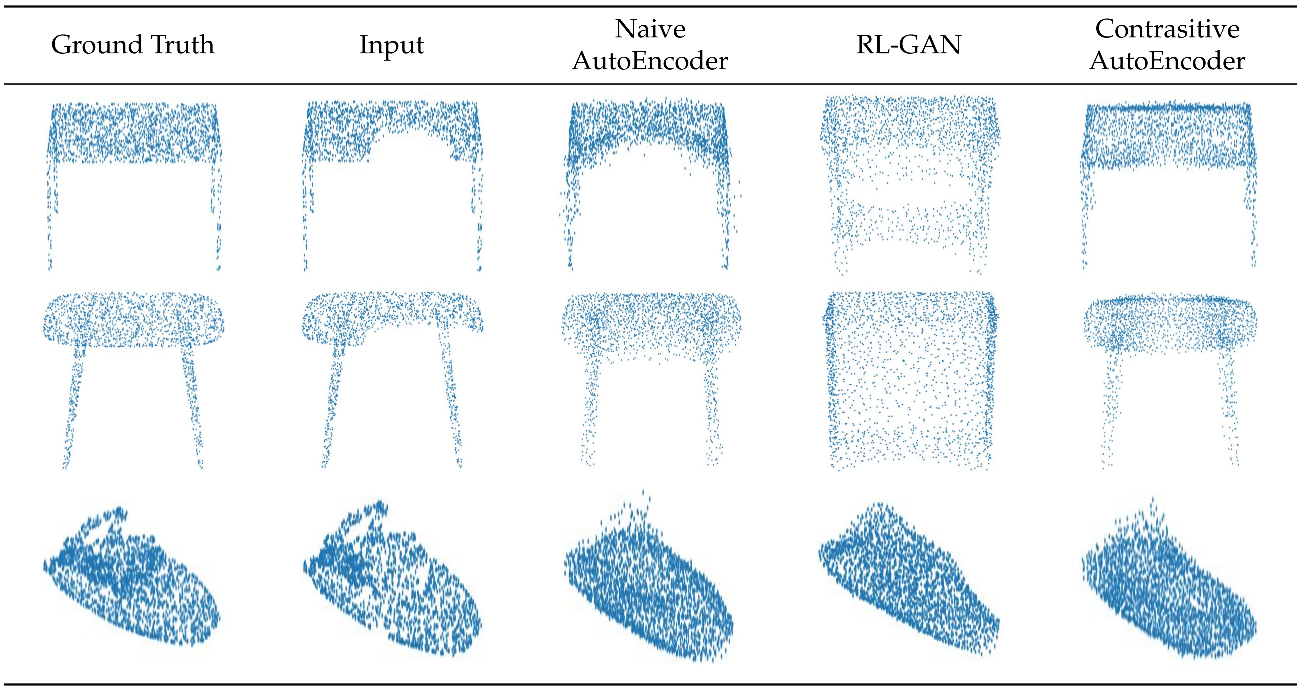
Table 3. Comparison of qualitative results of our Shape Completion pipeline with other methods on 20% missing object shapes.

Ground Truth	Input	Naive AutoEncoder	RL-GAN	Contrasitive AutoEncoder
				

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1



Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 September 2021 doi:10.20944/preprints202109.0112.v1



24. Achlioptas, P.; Diamanti, O.; Mitliagkas, I.; Guibas, L. Representation learning and adversarial generation of 3d point clouds. *arXiv preprint arXiv:1707.02392* **2017**, *2*, 4.
25. Yang, Y.; Feng, C.; Shen, Y.; Tian, D. FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
26. Dai, A.; Ruizhongtai Qi, C.; Nießner, M. Shape completion using 3d-encoder-predictor cnns and shape synthesis. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5868–5877.
27. Han, X.; Li, Z.; Huang, H.; Kalogerakis, E.; Yu, Y. High-resolution shape completion using deep neural networks for global structure and local geometry inference. Proceedings of the IEEE international conference on computer vision, 2017, pp. 85–93.
28. Stutz, D.; Geiger, A. Learning 3d shape completion from laser scan data with weak supervision. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1955–1964.
29. Le, T.; Duan, Y. Pointgrid: A deep network for 3d shape understanding. Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 9204–9214.
30. Sarmad, M.; Lee, H.J.; Kim, Y.M. RL-GAN-Net: A Reinforcement Learning Agent Controlled GAN Network for Real-Time Point Cloud Shape Completion. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
31. Achlioptas, P.; Diamanti, O.; Mitliagkas, I.; Guibas, L. Learning Representations and Generative Models for 3D Point Clouds. Proceedings of the 35th International Conference on Machine Learning; Dy, J.; Krause, A., Eds. PMLR, 2018, Vol. 80, *Proceedings of Machine Learning Research*, pp. 40–49.
32. Rubner, Y.; Tomasi, C.; Guibas, L.J. The Earth Mover's Distance as a Metric for Image Retrieval. *Int. J. Comput. Vision* **2000**, *40*, 99–121. doi:10.1023/A:1026543900054.
33. Fan, H.; Su, H.; Guibas, L.J. A Point Set Generation Network for 3D Object Reconstruction From a Single Image. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
34. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*; Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; Garnett, R., Eds.; Curran Associates, Inc., 2019; pp. 8024–8035.
35. Chang, A.X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; Xiao, J.; Yi, L.; Yu, F. ShapeNet: An Information-Rich 3D Model Repository, 2015, [arXiv:cs.GR/1512.03012].
36. van der Maaten, L.; Hinton, G.E. Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research* **2008**, *9*, 2579–2605.
37. Bunte, K.; Haase, S.; Biehl, M.; Villmann, T. Stochastic Neighbor Embedding (SNE) for Dimension Reduction and Visualization Using Arbitrary Divergences. *Neurocomput.* **2012**, *90*, 23–45. doi:10.1016/j.neucom.2012.02.034.
38. Van Der Maaten, L. Accelerating T-SNE Using Tree-Based Algorithms. *J. Mach. Learn. Res.* **2014**, *15*, 3221–3245.