Dual adversarial model for generating 3D point cloud

Anonymous authors
Paper under double-blind review

Abstract

Three-dimensional data, such as point clouds, are often composed of three co-ordinates with few featrues. In view of this, it is hard for common neural networks to learn and represent the characteristics directly. In this paper, we focus on latent space's representation of data characteristics, introduce a novel generative framework based on AutoEncoder(AE) and Generative Adversarial Network(GAN) with extra well-designed loss. We embed this framework directly into the raw 3D-GAN, and experiments demonstrate the potential of the framework in regard of improving the performance on the public dataset compared with other point cloud generation models proposed in recent years. It even achieves state-of-the-art performance. We also perform experiments on MNIST and exhibit an excellent result on 2D dataset.

1 Introduction

Three-dimensional data points begin to play a key role in many practical applications recently, including vision, robotics, 3D object detection and so on. While merely occupying a rather small space, three-dimensional data points are capable of providing expressive and compact representation of surface-based geometry, which means representing geometric details of the object. In the past few years point clouds are more and more popular and particularly suitable for simple geometric transformation. However, point cloud data have some disadvantages, for instance, the geometrical data points of objects in the distance are often more sparse than those at close range under the same scene, and more difficult to identify due to the imaging principle; raw and high-dimensional representations are typically not well suited for the design of generative models via classic statistical methods and so on Gumhold et al. (2001); Daniels et al. (2007); Achlioptas et al. (2018). Thus how to effectively and accurately use and generate point cloud data is an urgent problem to be solved at present.

A qualified point cloud data generation network should satisfy the following conditions: 1) It should be able to synthesize 3D objects that are both highly varied and vivid, which means that there need to be fine details in the generated examples Wu et al. (2016). 2) The distribution of the data generated should be similar to that of the original data. If the dimension of the generated sample is higher than that of the raw data, the additional generated portion must follow the potential distribution of raw data points. In the past few years, many researchers have developed a lot of meaningful work in this direction, such as 3D modeling and synthesis Carlson (1982); Kaick et al. (2011). However, many of these traditional methods synthesize new objects by borrowing components from existing CAD model libraries. Therefore, synthetic objects may look realistic, but they are not novel in concept in fact.

In recent years, the development of deep learning has provided a new way to solve this problem. Deep learning can well learn the characteristics of data in a domain by self-updating. Girdhar et al. (2016); Qi et al. (2016) propose voxelized-object based method in learning deep object representations, while Kalogerakis et al. (2017); Su et al. (2015)(add) from the perspective of view-based projections. Besides these, graph methods Mikael Henaff (2015); Defferrard et al. (2016); Yi et al. (2017); Bruna et al. (2014) are also considered for processing algorithms for 3D data. The method of generating point clouds at this stage no longer retrieves them from the object database, instead, most of them synthesize new objects based on learning objects. It is a more difficult task compared to operating on 2D data because of the higher data dimension and fewer data features.

Generative Adversarial Network(GAN) Goodfellow et al. (2014) is the most widely used model in the current generation domain and has achieved great success in 2D image generation field. Wu et al. (2016) first applied this network to the 3D point cloud generation field, and named it as 3D-GAN and achieved great results. It introduces an adversarial discriminator to differentiate whether an object is synthesized or real, which may own the potential to capture the structural difference of two 3D objects.

Although 3D-GAN has been proved to be successful, it still has some problems. Because of the fact that GAN training is difficult and unstable, its huge network parameters drag down the calculation speed Salimans et al. (2016). Achlioptas et al. (2018) adopts the idea of adversarial in latent space, but the specific approach is completely different. The former propose a workflow that first learns a representation by training an AE with a compact bottleneck layer, then trains a plain GAN in that fixed latent representation. The two parts are trained separately, and their sampling and coding representations are different thus they can't be translated or interpreted within both spaces; the latter has used the Adversarial Autoencoder(AAE) Makhzani et al. (2015) for 3D generation, having the distribution of latent space of Autoencoder trained on point cloud data close to the prior distribution, such as standard gaussian distribution.

In this paper we propose a *dual generation model*, which can fully make use of the characteristics of data in low-dimensional latent space to help the network better learn and generate point cloud data. Our contributions are mainly from the following aspects:

- Compared to the literature study on the latent space of raw data, our method, through reasonable theory explaining and experimental verification, proposes dual generate adversarial networks architecture, from which the two networks are divided into the primary and secondary part. The secondary network assists the primary network by learning that the data is compressed to features on different dimensions, and our results exceed the same type of articles that focus on hidden space learning Achlioptas et al. (2018).

- Our model consists of three parts and can be trained end-to-end, what's more, each part interacts and restricts each other, which can be trained efficiently and steadily without using the pre-trained weights related to point cloud data and only adopt the normal random initialization method. The loss function is designed in the light of the function of each part in the network, and we managed to find the best strategy of each part's iteration renewal through experiments.

- The framework we proposed on learning different low-dimension channel can be transferred to other vision tasks easily. We do experiment on the MNIST dataset to demonstrate its implementability on 2D image tasks.

2 RELATED WORK

In this section, we mainly introduce the necessary research backgrounds of this paper, including the characteristics of point cloud data and some related work.

point cloud   The Point cloud is a collection of massive points that form characteristic of the target's surface represents the shape of the object in 3D space. It consists of many x,y,z coordinates points, which have a matrix structrue of $N \times 3$, where N is the number of points in a set. Compared with image data, there are few information and extractable features as to point cloud data, so how to effectively learn and represent the featrues of point cloud data has become the focus of current research. Wu et al. (2015) took an input point cloud as the voxelized representation, other methods like Maturana & Scherer (2015) using 3D-CNN to calculate occupancy grid. Charles et al. (2017) proposed the PointNet architecture, it can solve the difficulty of dealing with unordered sets by introducing permutation-invariant function instead of convolutions.

deep learning for generating 3D data   Learning and generating 3D object is an essential task in the graphics and vision community. In 3D generating field, Wu et al. (2016) proposed the 3D-GAN, which can generate 3D object from a probabilistic space by leveraging recent advances in volumetric convolutional networks and generative adversarial nets. Achlioptas et al. (2018) designed a generative model called latent-GAN which is composed of two procedures: first using the point

cloud data as an input to train an Autoencoder(AE)Rumelhart et al. (1988), and then using the pre-trained encoder output as the label to train GAN, instead of training on the raw point cloud. Compared with 3D-GAN, its generative ability has improved, but the drawback of this model is that it is not an end-to-end one. Sampling and coding representations are different and can not be translated or interpreted within both spaces. Our model solves this shortcoming by proposing a dual GAN architecture and the result shows that it can largely improves the generation ability.

3 BACKGROUND

In this part we give the necessary introduction on some basic concepts which we will use in the rest of this paper, including point cloud metrics and some fundamental building blocks.

Autoencoder Autoencoder(AE) is a multi-layer neural network that can learn the efficient representation of input data through unsupervised learning. It is composed of two parts: Encoder and Decoder. The networks' width of the former part gradually becomes shallower as the network becomes deeper, while the latter part is reversed, as is shown on the right. The Encoder learns a low-dimensional latent representation z of the input data x and the Decoder restores z to original x.
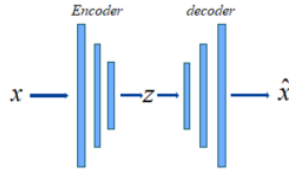
Generative Adversarial Network Generative Adversarial Network(GAN) is the most widely used generative model today. It consists of a model $generator(G)$ and a model $discriminator(D)$ and establishes a min-max adversarial game between these two models. The generator tries to get the function $G(z)$ that maps z sample from the prior $p(z)$ to the data space. The discriminator tries to distinguish the real data between the generated output. The loss function of this game can be expressed as:

$$\min_G \max_D E_{x \sim p_{data}}[\log D(x)] + E_{z \sim p(z)}[\log 1 - D(G(z))] \tag{1}$$

The model mainly includes two parts of training process:1) Train the discriminator to distinguish the true label between the fake data generated from sample noise by generator. 2) Train the generator to generate samples which aims to fool the discriminator.

Point Cloud Metrics We adopt two permutation-invariant metrics to compare unordered point sets, which was proposed by Fan et al. (2017).
The $Earth\ Mover's\ distance$(EMD): Introduced in Rubner et al. (2000), it is a metrics of a transpotation problem based on the minimal cost to transform one distribution into the other. For two equally sized subsets $s_1 \subseteq R^3$, $s_2 \subseteq R^3$, their EMD is difined by:

$$d_{EMD}(S_1, S_2) = \min_{\emptyset:S_1 \to S_2} \sum_{x \in S_1} \|x - \emptyset(x)\|_2 \tag{2}$$

where $\emptyset(x)$ is a bijection, EMD is differentialble almost everywhere as a loss.

$Chamfer\ pseudo-distance$(CD): It measures the squared distance between each point in one set to its nearest neighbor in another set, and the function is shown below:

$$CD(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2 \tag{3}$$

Compared to EMD , CD is more efficient to compute as a loss and is totally differentiable.

4 DUAL GENERATION MODEL FOR LEARNING REPRESENTATION

In this section we first describe the architecture of our model and the motivation why we design such a network. Then we give a brief introduction to some similar previous work on generating point cloud, and make a comparison of these models to our model. Finally we present a detail explanation of the loss function in our model and how to train it.
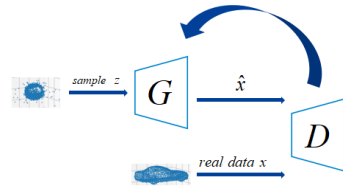
4.1 OUR ARCHITECTURE

The dataset we use to present our result is $ShapeNet$ Chang et al. (2015), the dimension of each sample is 2048×3. Inspired by the article Achlioptas et al. (2018) which proposes that using the latent codes instead of raw point data to train 3D-GAN can achieve more precise metrics and less network computing consumption, it is sensible to make a hypothesis that if learning the latent codes can achieve better result, the generator, in the process of generating images from noise, must have a certain dimension that the feature map generated in this dimension is the effective encoding of the raw data. In other words, the function of the latter multiple layers is simply to decode this feature.
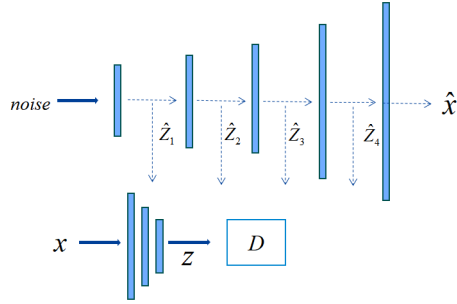

Figure 1: The structure of our assumption

As the figure shows above, the top of figure is the generator architecture, dimensional transformation occurs at the dotted line. x represents the raw data while D stands for $Discriminator$, $\hat{z}_i$(i=1, 2, 3 ,...) stands for the hidden output derived from each dimension of the generator. According to the above hypothesis, we may make an assumption that $\hat{z}_i$(i=1 or 2 or 3,...) is an effective compress of the raw data, which will be evaluated at the next section, then we can achieve high level on each indicator by adding an adversarial branch to the 3D-GAN in appropriate dimension and precisely designing loss function and updating strategy of each module.
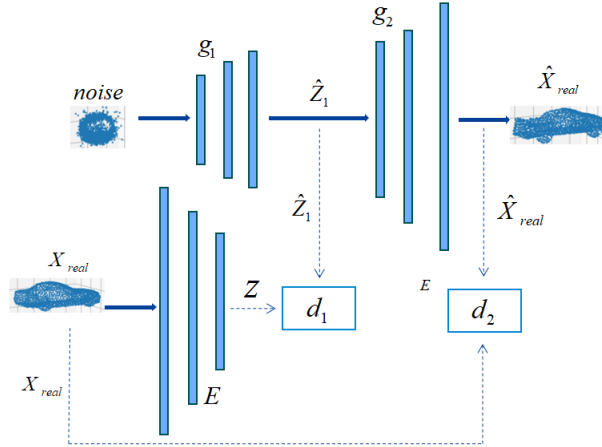

Figure 2: The structure of our approach

From what has been discussed above, we propose a dual generative model which is presented in Figure 2. To make it easier to illustrate, the modules in the structure are named separately as $g1, g2, d1, d2$ and E. $g1$ and $d1$ are combined as generative adversarial model $G1$ and they are against each other in latent space. The raw data are trained as the label of $G1$ after being encoded by module E while $g1, g2, d2$ are combined as the 3D-GAN model named $G2$ as figure shows above. To expressly state it, we name the lower part of the structure as E which is similar to the model $Encoder$ in order to be easily understood. In fact, there is no real "$decoder$" in this model to form

the autoencoder. During the training process, $G1$ will provide auxiliary training for $G2$, we can consider $G1$ as a regularization for $G2$, since it helps $G2$ more fully trained and better capture the distribution of the raw data.

Our structure combines the advantages of many classic structures in forward propagation, if we focus on module E and $g2$, this branch is like Autoencoder; if we pay attention to $g1,g2$ and $d2$, we will get the structure of raw-gan; if we only use $g1,d1$ and E, they make up latent-gan proposed by Achlioptas et al. (2018). In other words, our model can be transformed into the three frameworks mentioned above by some tricks, so we can achieve better performance on their basis.

4.2 LOSS DESIGN AND TRAINING PROCEDURE

In this section, we will show our design of the error function, as well as the update strategy of each part. We use the loss function $\mathcal{L}_{GAN}$ and $\mathcal{L}_{AE}$ referred to in Section 3 to compose our loss. The following statements are details of the loss function in each module. We use leaky-ReLU Andrew L. Maas (2013) as our active function and there is no batch-norm.

- E: Since the module E is to make an effective dimension reduction for the raw point cloud data, we use the $d_{EMD}(\hat{X}_{real}, X_{real})$ or $CD(\hat{X}_{real}, X_{real})$ to update its parameters. $\hat{X}_{real}$ is the output of G, so the update process of this module is in cooperation with the auxiliary generator training. The loss function of module E is named as $\mathcal{L}_E$.

- $G1$: The purpose of module G1 is to constrain the output of the hidden layer in the entire model, so that the output distribution of the hidden layer is close to the output distribution after the dimension reduction which is operated by module E. We use loss function $\mathcal{L}_{GAN}(noise{\to}z)$ to train this module.

- $G2$: G2 is a typical 3D-GAN structure, and its loss function must contain $\mathcal{L}_{GAN}(noise{\to}\hat{X}_{real})$. Inspired by Makhzani et al. (2015) & Larsen et al. (2016), the difference between the generated data and the raw data can be considered as a constraint while updating generation network. Therefore, the loss function of this module is $\mathcal{L}_G = \mathcal{L}_{GAN}(noise{\to}\hat{X}_{real}) + \lambda\mathcal{L}_E (0 < \lambda < 1)$.

The flowchart of the updating process is as follows:

Algorithm 1 Training the Dual Generation Model

---
initialize paramaters of each module $\theta_{G1}, \theta_G, \theta_E$
repeat
    $X_{real} \leftarrow$ random mini-batch from dataset
    $noise \leftarrow$ sampled from prior distribution $\mathcal{N}(0, I)$
    $z \leftarrow E(X_{real})$
    $\hat{z} \leftarrow g1(noise)$
    $\hat{X}_{real} \leftarrow G(noise)$
    $\mathcal{L}_E \leftarrow d_{EMD}(\hat{X}_{real}, X_{real})$ or $CD(\hat{X}_{real}, X_{real})$
    update parameter $\theta_E \leftarrow \nabla_{\theta_E}\mathcal{L}_E$
    $\mathcal{L}_{G1} \leftarrow \mathcal{L}_{GAN}(noise{\to}z)$
    update parameter $\theta_{G1} \leftarrow \nabla_{\theta_{G1}}\mathcal{L}_{GAN}(noise{\to}z)$
    update parameter $\theta_G \leftarrow \nabla_{\theta_G}(\mathcal{L}_{GAN} + \lambda\mathcal{L}_E)$
until algorithm convergence

---

5 EVALUATION

In this section we will present our results and the enhancement on 2D and 3D data. In 3D case, if we want to evaluate how well the fake data generated by our model matches the given set, we need a comparison to evaluate the faithfulness and diversity of a generative model, referring to Achlioptas et al. (2018), we will use the following metrics:

Coverage It is defined as the fraction of the point clouds in B, from which we first find its clostest neighbor in A. It has two forms – COV-CD and COV-EMD, depending on which metrics in Section 3 we use to compute point-set distance.The value of coverage will highly increase if most of the point clouds in B are more matched to point clouds in A.

Minimum Matching Distance(MMD) Unlike COV, which only focus on the closest point cloud and isn't the representative of $fidelity$, we consider MMD as a metric to report the average of distance in the matching. Just like COV, we can use either of the structural distance, so there are MMD-CD and MMD-EMD. The lower the value of MMD is, the stronger the model generation ability will be.

Jensen − Shannon Divergence(JSD) The Jensen-Shannon Divergence(JSD) is used to measure the difference between two empirical distributions. Assume that we have two distribution P and Q, so JSD is defined as:

$$JSD(P\|Q) = \frac{1}{2}KL\left(P(x)\left\|\frac{P(x)+Q(x)}{2}\right.\right) + \frac{1}{2}KL\left(Q(x)\left\|\frac{P(x)+Q(x)}{2}\right.\right) \quad (4)$$

where $KL(P(x)\|Q(x))$ means the KL-Divergence between P and Q Kullback & Leibler (1951)

5.1 OTHER WORK FOR CONTRAST

Referring to what was described in Achlioptas et al. (2018), we use the following algorithm framework as comparison:

- **Raw-GAN**: It use the basic architecture of GAN which directly generates the raw point cloud data using noise sampled from prior distribution $\mathcal{N}(0, I)$.
- **l-GAN**: Instead of operating on the raw point cloud input, it passes the data through a pretrained autoencoder, and each object class is trained separately with EMD or CD loss. The dimensional of bottleneck variable of AE is 128.

5.2 EVALUATING THE LATENT SPCACE REPRESENTATION

We use the linear model on latent space which is generated by our model E, and its performance can reflect the quality of the latent representation. We train our module E across all shape categories. To obtain features for an input of 3D shape, we feed the point-cloud into module E and then extract the 128-dimensional bottleneck layer vector. The features are processed by a linear classification SVM trained on the 3D classification benchmark of ModelNet Wu et al. (2015). Table 1 shows the comparative results.

Dataset	SPH	LFD	3D-GAN	l-GAN	3D-AAE	ours-CD
MN10	0.798	0.799	0.910	**0.954**	-	0.932
MN40	0.682	0.755	0.833	0.845	**0.847**	840

Table 1: The Result of Classification on $ModelNet40$ and $ModelNet10$, from left to right we refer to Kazhdan et al. (2003); Chen et al. (2003); Wu et al. (2016); Achlioptas et al. (2018)

We can clearly see from the table that our method's performance is between the performance of l-GAN and 3D-GAN. This result is very consistent with our previous inferences. The results of l-GAN's latent space will be directly used for the generation of point cloud images, but the representation of our latent space is only used as an assistance for the main framework. Our results, better than 3D-GAN, also prove that the generation framework, after adding the adversarial branch, has better representation in the latent space.

5.3 EVALUATING THE RECONSTRUCTION

In this section, we evaluate the reconstruction capabilities of the proposed autoencoders using unseen test examples. We confront the reconstruction results obtained by the E model and use G2 model as decoder with our approaches to examine the influence of a prior regularization on reconstruction

quality. In Table 2 we show the MMD-CD and MMD-EMD value between reconstruction and corresponding ground-truth in the dataset of chair object class.

Method	JSD(tr)	JSD(te)	MMD-CD	MMD-EMD
AE-CD	**0.0028**	0.0067	**0.0004**	**0.0527**
ours-E	0.0026	**0.0067**	0.0011	0.0562

Table 2: The Result of Latent Space Reconstruction Ability(tr:train-split, te:test-split)

The result in Table 2 shows that our end-to-end model can approximate result from l-GAN when trained by step. This encouraging result indicates that the generation ability of this network will probably exceed the result of l-GAN step-by-step generation. We will introduce the experiment of this part in the next section.

5.4 EVALUATING THE GENERATION

We train and compare a total of five generative models on the point clouds data of *car* category. The model E is established with 128-dimensional bottleneck, and trained with CD or EMD loss respectively. The training procedure will be stopped once our model's performance on the validation dataset is poor. According to the loss used by module E, we have two models of evaluating, which are DG-CD and DG-EMD. We also train 3D-GAN directly on the raw point cloud data as comparison.

Method	JSD	MMD-CD	MMD-EMD	COV-EMD	COV-CD
R-GAN	0.1764	0.0020	0.1230	19.0	52.3
l-GAN(CD)	0.0486	0.0020	0.0796	32.2	59.4
l-GAN(EMD)	0.0227	0.0019	0.0660	66.9	67.6
DG-CD	0.0135	0.0017	0.0632	56.3	61.7
DG-EMD	**0.0008**	**0.0013**	**0.0581**	**68.2**	**68.5**

Table 3: evaluating 5 generators on test-split of *car* dataset, for our model we did five experiments and reported its average value

In two different sets of experiments, we measure how well the distribution of the generated samples resembles both the train and test splits of the ground truth distribution, by using our models to generate a set of synthetic point clouds and employing the metrics which is introduced above to compare with the train or test set distributions respectively. All experimental results are shown in Table 3, and it can be seen that our method has achieved the best result on all five indicators, which means that in the algorithm based on generative adversarial network structure, our method has achieved the-state-of-art performance. Figure 3 is a comparison of the results of our method with the results of the R-GAN in generating point cloud data. We can clearly see when generating the same dimensional point cloud data, our method is more concentrated and its contour is more obvious. Figure 4 shows some other examples and results.
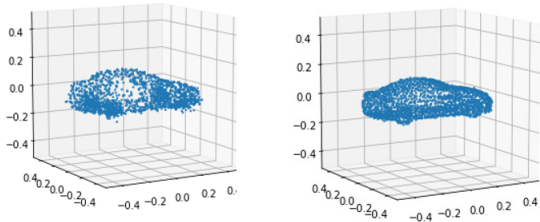

Figure 3: Left:Image generated by R-GAN   Right:Image generated by our method
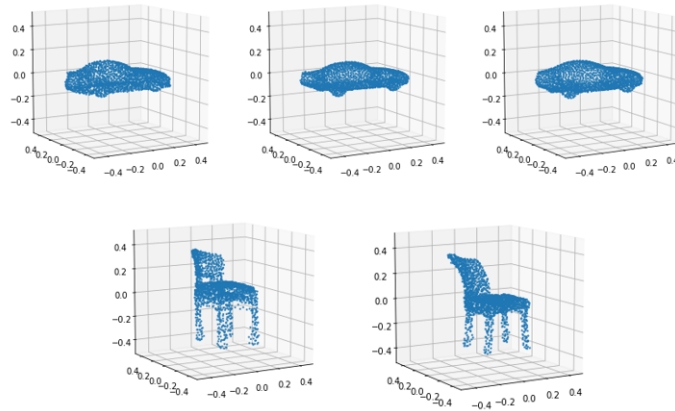
Figure 4: Some Results generated by our method

5.5 2D RESULT

To prove the mobility of this structure, we did some experiments on the 2D image dataset. We use our method to generate $MNIST$ dataset and compare it with the result of Goodfellow et al. (2014), we can see that the edges of the images we generate are sharper and the texture of the picture is more clear.
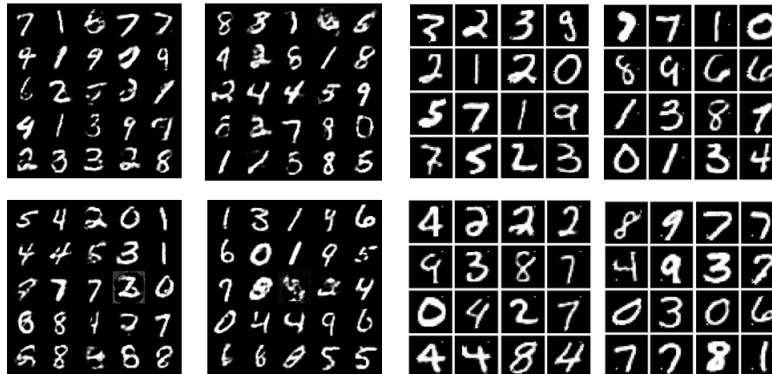

Figure 5: Left:Image generated by GAN   Right:Image generated by our method

6 CONCLUSION

In this paper we propose a dual generation architecture for learning and generating 3D point cloud data, and our experiments show that learning the characteristics of raw input data in different coding-and-low dimension can indeed help the GAN to generate object. Compared with other generative frameworks, our model is an end-to-end architecture which has better performance and higher efficiency. Although our experiment result may not be sufficient due to limited time, the insight of our work is worth for further study and discussion, what's more, the framework based on our learning and representing latent space feature will surely achieve a better result in the future.

REFERENCES

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International Conference on Machine Learning*, pp. 40–49, 2018.

Andrew Y. Ng Andrew L. Maas, Awni Y. Hannun. Rectifier nonlinearities improve neural network acoustic models. *International Conference on Machine Learning*, 2013.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, pp. http–openreview, 2014.

Wayne E. Carlson. An algorithm and data structure for 3d object synthesis using surface patch intersections. *ACM SIGGRAPH Computer Graphics*, 16(3):255–263, 1982.

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, and Su Hao. Shapenet: An information-rich 3d model repository. *Computer Science*, 2015.

R. Qi Charles, Su Hao, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2017.

Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, pp. 223–232. Wiley Online Library, 2003.

Joel II Daniels, Linh K Ha, Tilo Ochotta, and Claudio T Silva. Robust smooth feature extraction from point clouds. In *IEEE International Conference on Shape Modeling and Applications 2007 (SMI'07)*, pp. 123–136. IEEE, 2007.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pp. 3844–3852, 2016.

Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 605–613, 2017.

Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pp. 484–499. Springer, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Stefan Gumhold, Xinlong Wang, and Rob S MacLeod. Feature extraction from point clouds. In *IMR*. Citeseer, 2001.

Oliver Kaick, Hao Zhang, Ghassan Hamarneh, Daniel Cohen-Or, Oliver Kaick, Hao Zhang, Ghassan Hamarneh, Daniel Cohen-Or, Oliver Kaick, and Hao Zhang. A survey on shape correspondence. *Computer Graphics Forum*, 30(6):1681–1707, 2011.

Evangelos Kalogerakis, Melinos Averkiou, Subhransu Maji, and Siddhartha Chaudhuri. 3d shape segmentation with projective convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3779–3788, 2017.

Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3 d shape descriptors. In *Symposium on geometry processing*, volume 6, pp. 156–164, 2003.

S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *33rd International Conference on Machine Learning (ICML 2016)*, 2016.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *International Conference on Learning Representation*, 2015.

Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IEEE/RSJ International Conference on Intelligent Robots & Systems*, 2015.

Yann LeCun Mikael Henaff, Joan Bruna. Deep convolutional networks on graph-structured data. *CoRR*, 2015.

Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. 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, 2016.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000.

David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, and Chen Xi. Improved techniques for training gans. *Advances in Neural Information Processing Systems*, 2016.

Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pp. 945–953, 2015.

Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in neural information processing systems*, pp. 82–90, 2016.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, Jianxiong Xiao, Zhirong Wu, Shuran Song, and Aditya Khosla. 3d shapenets: A deep representation for volumetric shapes. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2015.

Li Yi, Hao Su, Xingwen Guo, and Leonidas J Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2282–2290, 2017.