Pacific Graphics (2023)
R. Chaine, Z. Deng, and M. H. Kim (Editors)

TreeGCN-ED: A Tree-Structured Graph-Based Autoencoder Framework For Point Cloud Processing (Supplementary)

Prajwal Singh, Ashish Tiwari, Kaustubh Sadekar & Shanmuganathan Raman

CVIG Lab, Indian Institute of Technology Gandhinagar

The following is included as a part of an extension to the main paper.

1. Detailed description of tree-GAN and TreeGCN-ED pipeline
2. Implementation Details
3. Additional qualitative results

1. Detailed description of tree-GAN and TreeGCN-ED pipeline

Generating an image from noise vector [GPAM*14] is a well-known problem in the computer vision community. Recently, in [SPK19], the authors have proposed a deep generative model for the 3D point cloud generation. The proposed method is unique because the authors have used graph convolution in point clouds, which do not have any edge connection between points. They use a branching method to gather the information from neighbor points.

1.1. tree-GAN

tree-GAN [SPK19] proposes a deep generative model for 3D point cloud generation. It uses a branching method to gather information from neighboring points. The accumulated information is then distributed to other points using graph convolution. The point cloud thus generated through this method is implicitly segmented.

In tree-GAN [SPK19], a noise vector $z \in \mathbb{R}^{96}$ is sampled from $\mathcal{N}(0, I)$ and is given as input to the generator network. Each generator layer consists of a branching network and a graph convolution layer. For the first layer, there is no ancestor for the input vector; therefore, it is passed on to the graph convolution network, which then upsamples the input and passes the current and upsampled input to the next layer. The branching network accumulates the feature vectors from the previous layers, which is again upsampled by the graph convolution layer to generate a new feature vector for that layer. This is repeated until the point cloud of the desired dimension $\mathbb{R}^{n\times 3}$ is obtained at the output. Note that the feature vector for the first layer is the noise vector z itself. The generator and discriminator are trained under WGAN [ACB17].

1.2. TreeGCN Based Point Cloud Encoder-Decoder

This section discusses the proposed method for 3D point cloud processing. The key idea of this approach is inspired by the tree-GAN [SPK19], a deep learning-based model for generating a 3D point cloud from a noise vector. We use the idea of tree-based graph convolution from [SPK19] to develop an encoder that extracts rich embeddings to perform well on the unseen point cloud data. Our model takes a 3D point cloud of size $\mathbb{R}^{n\times 3}$ as input, then passes it through sequences of graph-based operation to generate encoding for the point cloud. The generated encoding is then passed through the decoder network where a sequence of graph-based operations upsample the encoding to obtain a $\mathbb{R}^{n\times 3}$ point cloud as the output. The complete network, called TreeGCN-ED, is trained end-to-end by minimizing chamfer loss [FSG17].

A $\mathbb{R}^{n\times 3}$ point cloud is given as input to the model, which then passes through a down-branching network for gathering features from the ancestors of each node. We first each ancestor to a sequence of fully connected layers and then apply max pooling to extract dominant features, passing this further to the graph convolution module. The point cloud continuously passes through the down branching and graph convolution module sequence until the desired encoding ψ is obtained. The generated encoding is given as input to the decoder network. The decoder architecture is similar to tree-GAN [SPK19] except it takes encoded embedding as input. The overall model is trained end-to-end using the Chamfer loss function [FSG17].

The branching network is an essential part of the TreeGCN-ED network. It helps in accumulating information from ancestors for each node. Every ancestor feature is passed through a fully connected layer at each stage to help the network learn the relation between a node and its neighbor. This is also useful because the point cloud does not have edge connections between points. We use max pooling for selecting essential features from the encoded point cloud. Max pooling has been proved to be a permutation invariant function [QSMG17]. We experimented with other pooling functions, such as averaging and adding feature vectors, but max-pooling works better than other methods. Tree graph convolution network learns the semantic segmentation of point cloud implicitly [SPK19].

2. Implementation Details

2.1. Loss Function

To train the TreeGCN-ED, we have used the Chamfer loss [FSG17] function as it shows promising results for point cloud-based reconstruction [SPK19].

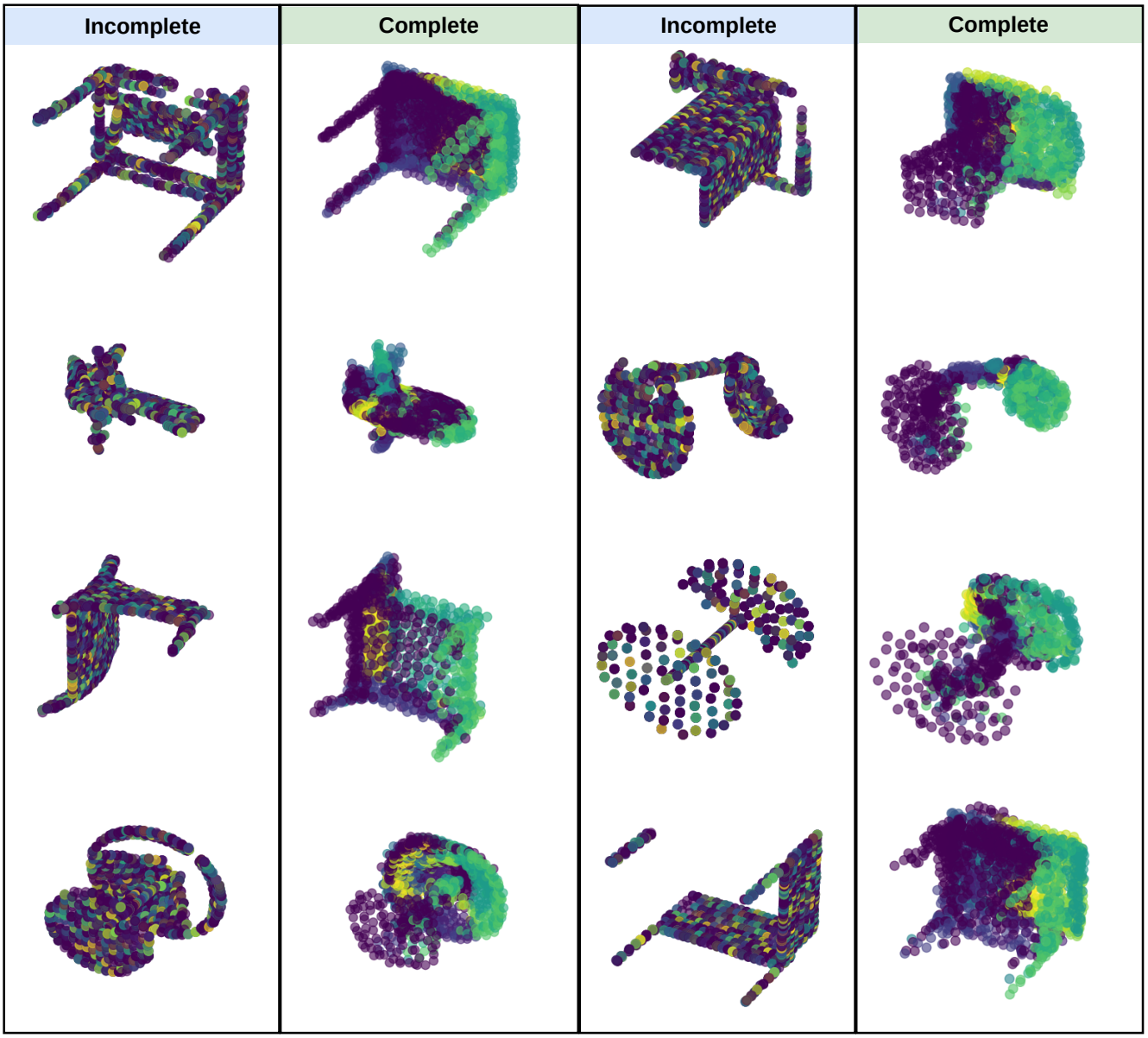

Figure 1: *Figure illustrates the qualitative result of point cloud shape completion.*

$$\mathcal{L}_{chamfer}(S_1,S_2) = \sum_{x\in S_1} min_{y\in S_2}||x-y||_2^2 + \sum_{y\in S_2} min_{x\in S_1}||x-y||_2^2 \tag{1}$$

In Equation 1, $S_1 \in \mathbb{R}^{n\times 3}$ and $S_2 \in \mathbb{R}^{n\times 3}$ represents two different point clouds. There are two specific reasons for using this loss function. First, it is permutation invariant [FSG17]. Second, it penalizes the loss function if a point from one set is not matched with its corresponding nearest neighbor in another set and vice-versa. This forces the model to learn information-preserving embedding for the point cloud.

2.2. Data Pre-processing

To train our model, we have used ShapeNetBenchmarkV0 dataset [CFG*15] consisting of 16 object classes.WE follow that same train-val-test split officially available along with the dataset. We uniformly sample 2048 points from the meshes of the ShapeNet dataset [CFG*15]. We use barycentric coordinates for the surface sampling to ensure uniform sampling of points.

2.3. Point Cloud Clustering

Table 1 shows the Chamfer Distance (CD) [FSG17] and Fréchet Point Cloud Distance (FPD) [SPK19] for each of the 16 different object classes in the ShapeNetBenchmarkV0 dataset.

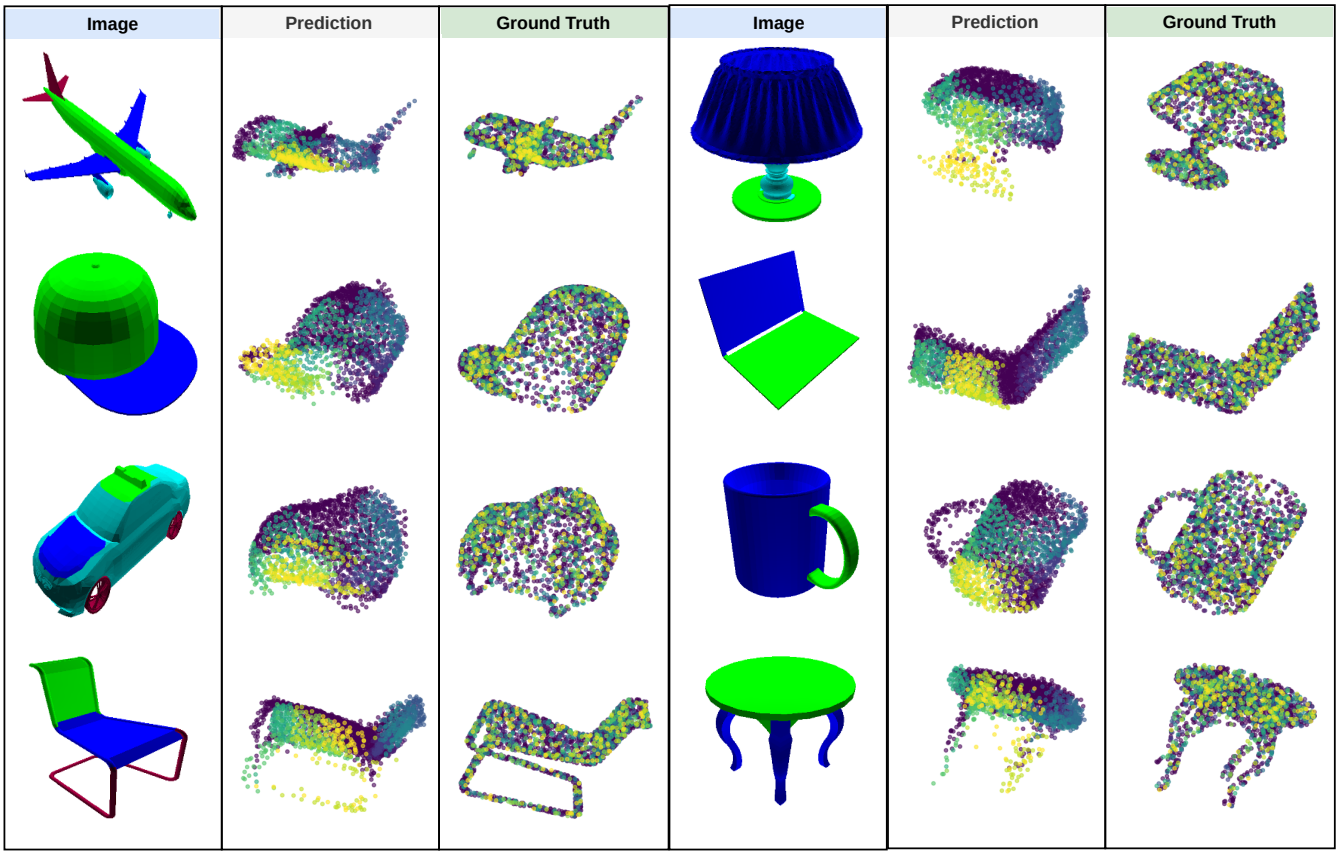

Figure 2: *Figure illustrates the qualitative result of 3D point cloud reconstruction from 2D images.*

2.4. Ablation Studies details.

We performed ablation studies to determine how feature embedding ψ dimension and data augmentation affect the ability of TreeGCN-ED to learn a meaningful feature representation. Four different training regimes are compared on the ShapeNetCore.v2 test-set [SYS*17], which consists of 55 classes. In *Regime* 1 *and* 2, the dimension of feature embedding is fixed to 256 and 512, respectively, without augmentation. Similarly, in *Regime* 3 *and* 4, the dimension of feature embedding is fixed to 256 and 512, respectively, but with augmentation. We use ShapeNetCore.v2 dataset [SYS*17] to train TreeGCN-ED for all the four regimes. Furthermore, we also evaluate the efficiency of feature representation learning of TreeGCN-ED on ModelNet10 and ModelNet40 datasets [WSK*15] for all four regimes. We follow the same procedure as mentioned in [YFST18] to train a linear SVM classifier on features extracted from trained TreeGCN-ED for the ModelNet datasets [WSK*15]. Regime 4 gives the best model performance.

It can be easily argued that the tree-GAN [SPK19] decoder itself is enough for point cloud processing at hand. However, to establish the need and examine the strength of the proposed encoder, we perform an additional experiment by replacing it with the PointNet [QSMG17] encoder to train the complete network on ShapeNetBenchmarkV0 dataset [CFG*15]. We observed that the average CD is 8.65 with the PointNet encoder and 1.21 with the proposed encoder. This establishes the efficacy of the proposed encoder.

2.5. Single Image-Based Point Cloud Reconstruction

We first train the TreeGCN-ED model on 16 different classes of the ShapeNetBenchmarkV0 dataset [CFG*15] till convergence. Later, we replace the encoder of TreeGCN-ED with a CNN-based architecture to extract image features. We freeze the trained weights of the decoder and train the image encoder network end-to-end for 3D reconstruction again using Chamfer Distance (CD) [FSG17] as the loss function. We use the synthesized images in the ShapeNetBenchmarkV0 dataset [CFG*15] to train the single image to a 3D shape reconstruction model.

3. Additional qualitative results

Figure 1, 2, and 3 show the additional qualitative results on point cloud completion, image-based reconstruction, and interpolation, respectively.

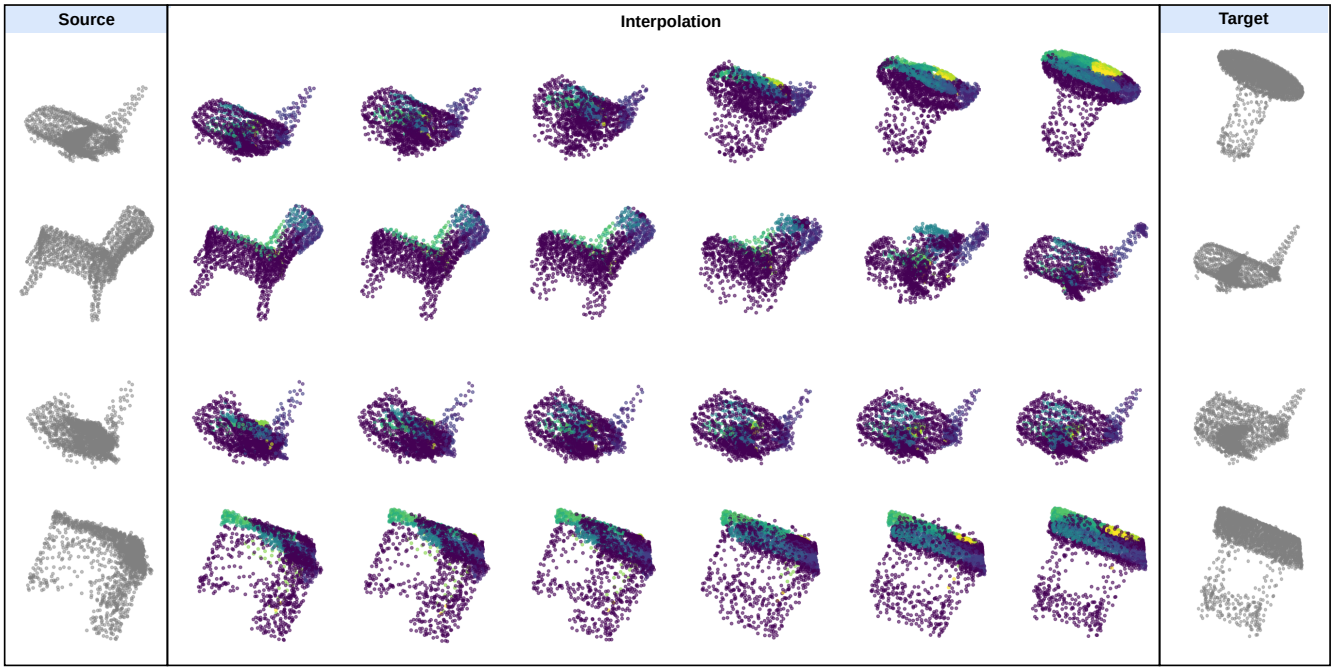

Figure 3: *Figure illustrates the qualitative result of smooth interpolation between point clouds for both inter and intra classes.*

Models	Metrics	Object Class																Average
		Airplane	Bag	Cap	Car	Chair	Earphone	Guitar	Knife	Lamp	Laptop	Motorbike	Mug	Pistol	Rocket	Skateboard	Table	
FoldingNet [YFST18]	CD	0.67	3.12	2.82	1.76	1.47	3.34	0.44	0.55	2.60	1.01	1.48	2.28	1.16	0.88	1.35	1.70	1.48
	FPD	11.10	87.45	117.36	28.47	12.00	152.04	19.55	19.56	45.19	11.19	33.91	40.17	30.14	32.53	47.17	24.62	44.52
TreeGCN-ED	CD	0.50	1.88	1.62	1.45	1.32	1.91	0.40	0.41	1.97	0.88	1.14	1.72	0.79	0.61	0.78	1.41	**1.21**
	FPD	5.79	21.02	16.14	9.47	7.85	51.79	13.90	14.80	21.82	2.56	14.67	12.70	9.62	23.91	13.90	13.90	**11.54**

Table 1: *Comparison of the efficiency for 3D point cloud encoding-decoding between our proposed architecture and the FoldingNet [YFST18] model on ShapeNetBenchmarkV0 dataset [CFG*15].*

References

[ACB17] ARJOVSKY M., CHINTALA S., BOTTOU L.: Wasserstein generative adversarial networks. In *34th ICML - Volume 70* (2017), ICML'17, JMLR.org, p. 214–223. 1

[CFG*15] 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:1512.03012. 2, 3, 4

[FSG17] FAN H., SU H., GUIBAS L. J.: A point set generation network for 3d object reconstruction from a single image. *2017 IEEE (CVPR)* (2017), 2463–2471. 1, 2, 3

[GPAM*14] GOODFELLOW I. J., POUGET-ABADIE J., MIRZA M., XU B., WARDE-FARLEY D., OZAIR S., COURVILLE A., BENGIO Y.: Generative adversarial networks, 2014. arXiv:1406.2661. 1

[QSMG17] QI C., SU H., MO K., GUIBAS L. J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. *2017 IEEE (CVPR)* (2017), 77–85. 1, 3

[SPK19] SHU D. W., PARK S. W., KWON J.: 3d point cloud generative adversarial network based on tree structured graph convolutions. *2019 IEEE/CVF (ICCV)* (2019), 3858–3867. 1, 2, 3

[SYS*17] SAVVA M., YU F., SU H., KANEZAKI A., FURUYA T., OHBUCHI R., ZHOU Z., YU R., BAI S., BAI X., AONO M., TATSUMA A., THERMOS S., AXENOPOULOS A., PAPADOPOULOS G. T., DARAS P., DENG X., LIAN Z., LI B., JOHAN H., LU Y., MK S.: Large-scale 3d shape retrieval from shapenet core55. In *3DOR@Eurographics* (2017). 3

[WSK*15] WU Z., SONG S., KHOSLA A., YU F., ZHANG L., TANG X., XIAO J.: 3d shapenets: A deep representation for volumetric shapes. *2015 IEEE (CVPR)* (2015), 1912–1920. 3

[YFST18] YANG Y., FENG C., SHEN Y., TIAN D.: Foldingnet: Point cloud auto-encoder via deep grid deformation. *2018 IEEE/CVF (CVPR)* (2018), 206–215. 3, 4