WEAKLY SUPERVISED POINT CLOUD UPSAMPLING VIA OPTIMAL TRANSPORT

Zezeng Li[1] *Weimin Wang*[1] *Na Lei*[1⋆] *Rui Wang*[1]

[1] School of Software, Dalian University of Technology, Dalian, 116024, People's Republic of China

ABSTRACT

Existing learning-based methods usually train a point cloud upsampling model with synthesized, paired sparse-dense point clouds. However, the distribution gap between synthesized and real data limits the performance and generalization. To solve this problem, we innovatively regard the upsamplig task as an optimal transport (OT) problem from sparse to dense point cloud. Further we propose **PU-CycGAN**, a cycle network that consists of a Densifier, Sparsifier and two discriminators. It can be directly trained for upsampling with unpaired real sparse point clouds, so that the distribution gap can be filled via the learning. Especially, quadratic Wasserstein distance is introduced for the stable training. Extensive experiments on both synthetic and real-scanned datasets validate the effectiveness and advantages in terms of distribution uniformity, underlying surface representation and applicability to real data. The source code is available at https://github.com/cognaclee/PU-CycGAN.

***Index Terms*—** Point Cloud Upsampling, Weakly Supervised, Optimal Transport

1. INTRODUCTION

In real world, the raw point clouds produced from depth cameras and LiDAR sensors are often sparse, noisy, and non-uniform. To solve this problem, the point cloud upsampling technology that aims at generating dense, uniform and complete point clouds has been widely studied. Especially in recent years, with the increase of point cloud datasets, deep neural network has exerted its power on point cloud upsampling. Since the seminal work of employing Multi-layer Perceptron for point cloud upsampling [1], various learning-based methods with different network architectures and training strategies have been proposed to continuously improve the upsampling performance [2, 3, 4, 5, 6]. These methods can learn to generate high-quality upsampled point clouds on synthetic datasets. However, they require paired sparse-dense data in the network training. The paired point clouds are constructed by sampling on synthetic mesh models, which are time-consuming and cumbersome. Consequently, these supervised methods cannot be trained with real-scanned datasets such as ScanNet [7] and KITTI [8] where paired dense point clouds are unavailable. Furthermore, the distributions gap between synthetic point cloud data and real scans usually degrades the performance of the model trained on synthetic data when applied to real scans. Therefore, it is promising to propose a weakly supervised method that can be trained with unpaired sparse-dense data.

Recently, a few weakly supervised and unsupervised point cloud upsampling methods have been proposed. L2G-AE [9] concentrated on capturing global shape information via local-to-global reconstruction, which limits the network in capturing inherent upsampling patterns. SPU-Net [10] constructs the paired sparse point cloud by the farthest point sampling and then trains the model with the sparse and original dense point cloud pairs. The final upsampling result is obtained by repeating and aggregating the first two stages r times. However, the way of repetition and aggregation can not reflect the real pattern of dense target, and the farthest point sampling makes it not suitable for the real sparse point clouds.

To resolve the above problems, we cast point cloud sampling as the OT problem and propose **PU-CycGAN**. Through the design of Densifier, Sparsifier and consistency loss, self-restraint loss, our model can be trained with unpaired point sets. In addition, based on OT quadratic transport cost, our upsampling model can converge to a local equilibrium point. In summary, our main contributions are: *(i)* We propose a weakly supervised point cloud upsampling framework that trains the model with unpaired point clouds. *(ii)* We notably regard point cloud upsampling as an OT problem, and design a quadratic Wasserstein distance to stabilize GAN's training. *(iii)* We introduce consistency loss and self-restraint loss to improve the performance of the model in underlying surface representation. *(iv)* Extensive experiments demonstrate that our method achieves comparable results to the SOTA supervised methods, especially on real data.

2. PROPOSED METHOD

2.1. Preliminary

Given a real-scanned sparse point clouds dataset, how can it participate in the training of the upsampling model? Inspired by prior work [11], to relax the requirement of paired data, we propose **PU-CycGAN** which is essentially two symmetric GANs, forming a cycle network. **PU-CycGAN** shares two generators, and each GAN has a discriminator. Through mu-

tual generation, unpaired point clouds with different densities can be transferred to each other.

As mentioned in [12], GANs accomplish two major tasks: manifold learning and probability distribution transformation. Specifically, the generator computes the OT map, while the discriminator computes the Wasserstein distance between the generated and the real distribution. GANs need to solve a min-max saddle point optimization problem [13], which is essentially a Monge-Kantorovich duality problem of OT. In practice, given empirical distributions and two disjoint sets X and Y, the discrete Monge-Kantorovich dual problem is:

$$\begin{aligned} &\max_{\phi,\psi} \frac{1}{m}\sum_{y_i\in Y}\phi\left(y_i\right)-\frac{1}{n}\sum_{x_j\in X}\psi\left(x_j\right) \\ &s.t.\ \phi\left(y_i\right)-\psi\left(x_j\right)\leq c\left(x_j,y_i\right) \quad \forall y_i\in Y, \forall x_j\in X\ . \end{aligned} \tag{1}$$

Where n and m are sample sizes of X and Y respectively. $\phi : Y \to \mathbb{R}$ and $\psi : X \to \mathbb{R}$ are two potential functions. $c(x_j, y_i)$ is transport cost between x_j and y_i.

While widely used, GANs are known to be hard to train. To improve stability, Liu [14] et al. proposed WGAN-QC which is based on the quadratic transport cost in Eq. (2).

$$c\left(x_j,y_i\right)=\frac{1}{2}||x_j-y_i||_2^2\ . \tag{2}$$

When Equation (2) is applied, the optimal objective in Equation (1) equals to the quadratic Wasserstein distance [15, 14].

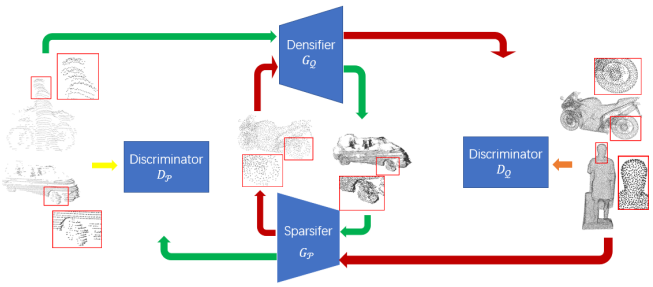

Fig. 1: The diagram of the proposed **PU-CycGAN**.

2.2. Proposed Framework

Given M sparse point sets $\mathcal{P} = \left\{\left\{\boldsymbol{p}_i^j\right\}_{j=1}^{N}\right\}_{i=1}^{M}$ and unpaired dense point sets $\mathcal{Q} = \left\{\left\{\boldsymbol{q}_i^k\right\}_{k=1}^{rN}\right\}_{i=1}^{M}$, we aim to learn a map which transports the sparse point sets to dense and uniformly distributed point set. Where N denotes the number of points in each sparse point set, r is the upsampling rate.

Fig. 1 shows the overall network architecture of **PU-CycGAN**. Herein, Densifier $G_{\mathcal{Q}}$ and Sparsifer $G_{\mathcal{P}}$ are generators which are used to fit the map $\mathcal{P} \to \mathcal{Q}$ and $\mathcal{Q} \to \mathcal{P}$. In addition, we introduce Sparse discriminator $D_{\mathcal{P}}$ and Dense discriminator $D_{\mathcal{Q}}$, where $D_{\mathcal{P}}$ aims to distinguish between $\mathcal{P}$ and generated sparse point sets $G_{\mathcal{P}}(\mathcal{Q})$, $D_{\mathcal{Q}}$ aims to discriminate between $\mathcal{Q}$ and $G_{\mathcal{Q}}(\mathcal{P})$. Through the sparse-dense-sparse (shown in green arrows) and dense-sparse-dense data (shown in red arrows) cycles, our model is expected to capture the inherent upsampling patterns and generate dense patches that are uniformly distributed on the target surface.

2.3. Densifier, Sparsifier and Discriminator

Densifier and Sparsifier. Densifier adopts the same architecture with PU-GAN's generator [4]. It contains three components to successively process: feature extraction, up-down-up feature expansion, and point set generation. The network structure of Sparsifier is similar to that of Densifier, but the up-down-up feature expansion is replaced with a down-feature aggregation module. To downsample the point's feature, the down-feature aggregation module reshapes the features and then uses MLP to regress the original features. Note that the Sparsifier learns to downsample to the distribution of sparse data rather than traditional downsampling operation, as illustrated in Fig. 1.

Discriminator. The purpose of discriminator is to distinguish whether its input is produced by the generator. For both $D_{\mathcal{P}}$ and $D_{\mathcal{Q}}$, we use a similar network structure with PU-GAN [4]. Considering that the number of points to be processed by $D_{\mathcal{P}}$ is less than that in $D_{\mathcal{Q}}$, the number of convolution kernels in $D_{\mathcal{P}}$ is reduced to half of that in $D_{\mathcal{Q}}$.

2.4. Loss Functions

Quadratic Wasserstein Loss The proposed **PU-CycGAN** is based on the Monge-Kantorovich duality problem whose solution can be obtained by GAN's min-max adversarial learning. Although WGAN-QC [14] improves the stability of GANs and achieves excellent performance in image generation, it cannot be directly applied to non-Euclidean point cloud data. To improve stability and convergence speed of **PU-CycGAN**, we introduce the Earth Mover's distance (EMD) [16] which is a suitable measure for point clouds into WGAN-QC's objective function. In specific, the objective function of discriminator $D_{\mathcal{P}}$ and $D_{\mathcal{Q}}$ is:

$$\begin{aligned} \min_{w}\ & \frac{1}{2}\left(\frac{1}{B}\sum_{i=1}^{B}D_w\left(y_i\right)-\frac{1}{B}\sum_{i=1}^{B}\phi\left(y_i\right)\right)^2 \\ &+\frac{1}{2}\left(\frac{1}{B}\sum_{i=1}^{B}\left(D_w\left(x_i\right)-\psi\left(x_i\right)\right)^2\right)+ \\ &\frac{\gamma}{B}\sum_{i=1}^{B}\left(||\nabla_x D_w\left(x_i\right)||-d_{EM}(x_i,y_i)\right)^2\ . \end{aligned} \tag{3}$$

Where y_i and x_i are patches with N points from real and generated points set respectively. B denotes batch size. $\phi\left(y_i\right)$ and $\psi\left(x_i\right)$ are obtained by linear programming of Eq. (1).

Here D_w can represent $D_{\mathcal{P}}$ or $D_{\mathcal{Q}}$. ∇_x is the partial derivative of D with respect to x. $d_{EM}(\cdot)$ denotes EMD. According to Brenier's theorem [12, 17], if x_i is transformed to y_i,the optimal discriminator exists at the minimization of Eq. (3).

Further, to stabilize the optimization of the generator, we set the adversarial loss of the generators $G_{\mathcal{Q}}$ and $G_{\mathcal{P}}$ as a quadratic function, which is

$$\mathcal{L}_{adv} = \left(\frac{1}{B} \sum_{i=1}^{B} D_w \left(y_i \right) - \frac{1}{B} \sum_{j=1}^{B} D_w \left(G_\theta \left(z_j \right) \right) \right)^2 . \quad (4)$$

Here, when $x_j = G_{\mathcal{Q}}(G_{\mathcal{P}}(q_j))$ and $y_i = q_i$, D_w is $D_{\mathcal{Q}}$. Similarly, when $x_j = G_{\mathcal{P}}(G_{\mathcal{Q}}(p_j))$ and $y_i = p_i$, D_w is $D_{\mathcal{P}}$.

Cycle Consistency Loss. To eliminates the need of paired data, inspired by CycleGAN [11], we proposed an point cloud upsampling consistency loss which is defined as follows:

$$\mathcal{L}_{cyc} = d_{EM}(p_i, G_{\mathcal{P}}(G_{\mathcal{Q}}(p_i))) + d_{EM}(q_j, G_{\mathcal{Q}}(G_{\mathcal{P}}(q_j))). \quad (5)$$

Where p_i is a real sparse point set, q_i is a real dense point set.

Self Restraint Loss. Without paired point sets as the supervision, we define a self-restraint loss to ensure that the generated points are distributed on the underlying surface. Herein, self-restraint loss uses the Chamfer distance [16] to measure the loss between sparsified or densified point set and the original one, which is defined as follows:

$$\begin{aligned} \mathcal{L}_{sel}(x_i, z_i) = & \frac{1}{N} \sum_{j=1}^{N} \min_{\boldsymbol{x}_{ik} \in x_i} ||\boldsymbol{z}_{ij} - \boldsymbol{x}_{ik}||_2^2 \\ & + \frac{1}{rN} \sum_{k=1}^{rN} \min_{\boldsymbol{z}_{ij} \in z_i} ||\boldsymbol{z}_{ij} - \boldsymbol{x}_{ik}||_2^2 . \end{aligned} \quad (6)$$

In the upsampling direction, x_i and z_i are upsampled results and input sparse point set respectively. In the dual task, x_i and z_i are downsampled results and dense input respectively.

The final total loss of **PU-CycGAN's** generator Densifier and Sparsifier is the weighted sum of consistency loss, self-restraint loss, uniform loss and adversarial loss:

$$\begin{aligned} \mathcal{L}_G = & \lambda_{cyc}\mathcal{L}_{cyc} + \lambda_{sel}(\mathcal{L}_{sel}(G_{\mathcal{Q}}(p_i), p_i) + \mathcal{L}_{sel}(G_{\mathcal{P}}(q_i), q_i)) \\ & + \lambda_{uni}(\mathcal{L}_{uni}(G_{\mathcal{Q}}(p_i)) + \mathcal{L}_{uni}(G_{\mathcal{Q}}(G_{\mathcal{P}}(q_i))))+ \\ & \lambda_{adv}(\mathcal{L}_{adv}(q_i, G_{\mathcal{Q}}(G_{\mathcal{P}}(q_i))) + \mathcal{L}_{adv}(p_i, G_{\mathcal{P}}(G_{\mathcal{Q}}(p_i)))) . \end{aligned} \quad (7)$$

$\mathcal{L}_{uni}$ in (7) is uniformity loss from PU-GAN [4]. λ_{cyc}, λ_{sel}, λ_{uni}, λ_{adv} are set as 100, 120, 40, and 5.8 empirically. As aforementioned, objective functions of discriminator $D_{\mathcal{P}}$ and $D_{\mathcal{Q}}$ are defined in Eq. (3). γ in Eq. (3) is set as 0.5.

3. EXPERIMENTAL RESULTS

3.1. Datasets and Network Configurations

To compare with baseline methods, we first train models on PU1K [5] and PU-GAN's datasets which are cropped into dense and downsampled sparse point clouds patches respectively. Then we train a model with upaired real-scanned sparse KITTI [8] and dense SEMANTIC3D [18] data to demonstrate the capability and advantages of our method in the real applications with unpaired data.

Each model is trained for 100 epochs using the Adam algorithm [19]. By default, we set $N = 256$, $r = 4$. Quantitative and qualitative comparisons with SOTA pair-data-based methods on the test set of PU1K and PU-GAN are performed. Since there is no dense GT for KITTI dataset, only qualitative comparison is conducted. In the quantitative comparisons, we employ the same four evaluation metrics as PU-GAN, including uniformity (UNI), point-to-surface (P2F), Chamfer distance (CD), and Hausdorff distance (HD).

Table 1: Comparisons on PU-GAN's dataset. ↓ mean lower is desired. The best result is shown in bold and the suboptimal result is shown in underline.

Methods	Uniformity for different p ↓ (10^{-3})					P2F ↓	CD ↓	HD ↓
	0.4%	0.6%	0.8%	1.0%	1.2%	(10^{-3})	(10^{-3})	(10^{-3})
PU-Net [1]	29.74	31.33	33.86	36.94	40.43	6.84	0.72	8.94
MPU [3]	7.51	7.41	8.35	9.62	11.13	3.96	0.49	6.11
PU-GAN [4]	**3.38**	**3.49**	**3.44**	**3.91**	**4.64**	**2.33**	**0.28**	4.64
L2G-AE [9]	24.61	34.61	44.86	55.31	64.94	39.37	6.31	63.23
SPU-Net [10]	4.53	4.82	5.68	6.69	7.95	5.97	0.38	**2.24**
Ours(unpaired data)	<u>4.17</u>	<u>4.11</u>	<u>4.71</u>	<u>5.53</u>	<u>6.51</u>	<u>3.35</u>	<u>0.32</u>	<u>2.92</u>

3.2. Comparisons with Baselines on Synthetic Datasets

We qualitatively and quantitatively compare our method with several baselines: PU-Net [1], MPU [3], PU-GAN [4], PU-GCN [5], Dis-PU [6], L2G-AE [9] and SPU-Net [10]. Among these methods, L2G-AE is unsupervised, SPU-Net is self-supervised, and all other methods are supervised. In the comparison experiment, if a baseline method provides a pretraining model, we directly use it for comparison. If not, we retrain a model on the corresponding dataset for fairness.

PU-GAN's dataset. In Table 1, **PU-CycGAN** outperforms PU-Net, MPU, L2G-AE and SPU-Net. Although it works with unpaired data in a weakly supervised manner, it also achieved comparable performance compared with the supervised SOTA methods. All evaluation metrics are the same as the ones employed in [4]. Fig. 2 shows the visual comparisons of upsampled results on two sparse point clouds. Besides, we also show the associated error maps, where the colors reveal the nearest distance for each point in the target point set to the predicted point set. We can see that the errors of our upsampled results are comparable to these supervised methods, which are also verified by both CD and HD values.

PU1K dataset. Due to the inaccessibility of code, we abandoned L2G-AE and SPU-Net in this experiment. For a fair comparison, we retrained PU-GAN and Dis-PU's networks with PU1K. As shown in Table 2, although **PU-CycGAN** is weakly supervised, it clearly outperforms PU-Net, MPU, PU-GCN and Dis-PU in all three metrics. On account of limited

space, visual comparisons on PU1K are not shown here.

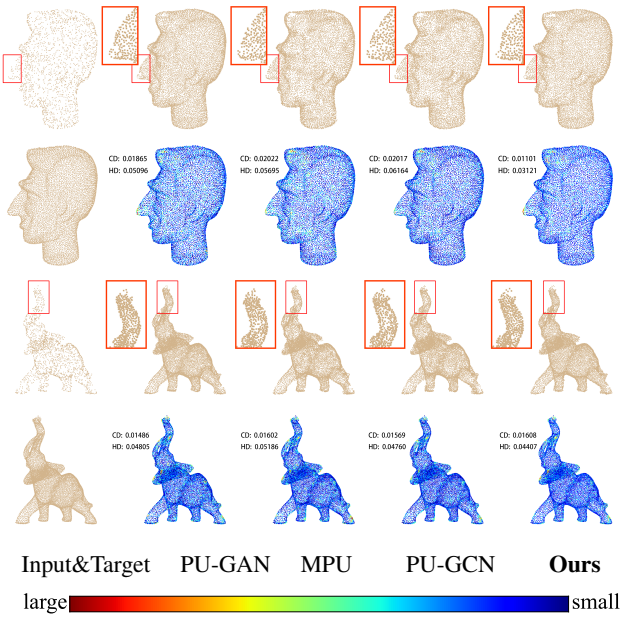

Fig. 2: Upsampling results and the associated error maps.

Table 2: Comparisons on PU1K against supervised methods.

Method	P2F(10^{-3})	CD(10^{-3})	HD(10^{-3})
PU-Net [1]	4.834	1.155	15.170
MPU [3]	3.551	0.935	13.327
PU-GAN [4]	**1.590**	**0.420**	5.390
PU-GCN [5]	2.499	0.585	7.577
Dis-PU [6]	3.143	1.151	14.680
Ours	2.080	0.551	**2.919**

3.3. Upsampling Real-scanned Data:KITTI

To illustrate the applicability of our method in real scenes, we construct our own real scene training data. We crop sparse point cloud patches from KITTI through polar equidistant sampling, and dense patches from SEMANTIC3D according to farthest point sampling and KNN sampling. After the training, the Densifier of **PU-CycGAN** is used to upsample sparse point clouds. More details and settings can be found in our open-source code.

As shown in the top row of Fig. 3, real-scanned inputs are large-scale, sparse, noisy, non-uniform, which are greatly different from the synthetic sparse data. Thus, the model trained on synthetic data has limited generalization to real scanned data. Fig. 3 compares **PU-CycGAN** against the most competitive upsampling methods PU-GAN on KITTI. We observe that PU-GAN tends to produce more outliers and overfill holes (e.g. the first close-up on bicycle wheels and the second close-up), while **PU-CycGAN** preserves better underlying surface and repair fine-grained details.

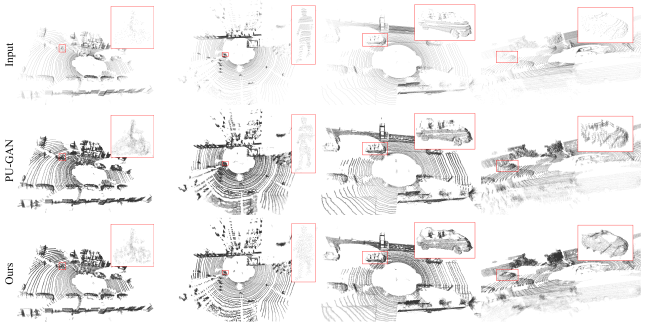

Fig. 3: Comparisons on KITTI.

3.4. Ablation Study and Baseline Comparison

To evaluate the effectiveness of the major components designed in our framework, we conduct the ablation study on in three cases: (A) removing $\mathcal{L}_{sel}$; (B) replace quadratic (l_2) Wasserstein loss in equations(3) and (4) with least square loss which is used in PU-GAN; (C) A&B. Tab. 3 summarizes the results of each case in terms of CD value. By comparing with our full pipeline (bottom row), the necessity of both components can be validated. Besides, we also present a comparison of convergence speed associated with the ablation study (B) in Fig. 4. Therein, we can see that l_2 loss converges to a stable small value after 8000 iterations of training, while least square loss still fluctuates in a large range. This shows that l_2 is more effective and suitable for training **PU-CycGAN** model.

Model	l_2	$\mathcal{L}_{sel}$	CD(10^{-3})
A	✓		24.197
B		✓	0.774
C			27.577
Full	✓	✓	**0.551**

Table 3: Ablation on PU1K

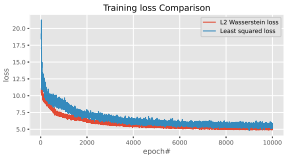

Fig. 4: Loss comparison

4. CONCLUSIONS

In this paper, we propose a weakly supervised point cloud upsampling method to generate dense and uniform point set from sparse inputs without the supervision of paired ground-truth dense point clouds. Through the design of Densifier, Sparsifier and consistency loss, self-restraint loss, our **PU-CycGAN** can be trained on unpaired point clouds. Moreover, by using quadratic transport cost, our method greatly improved in stability and convergence speed. Extensive experiments show that **PU-CycGAN** rivals the state-of-the-art supervised methods. Especially, the proposed method produces higher upsampling quality on real-scanned point clouds compared to other methods.

5. REFERENCES

[1] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng, "Pu-net: Point cloud upsampling network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018, pp. 2790–2799.

[2] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng, "Ec-net: an edge-aware point set consolidation network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 386–402.

[3] Wang Yifan, Shihao Wu, Hui Huang, Daniel Cohen-Or, and Olga Sorkine-Hornung, "Patch-based progressive 3d point set upsampling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5958–5967.

[4] Ruihui Li, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng, "Pu-gan: a point cloud upsampling adversarial network," in *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[5] Guocheng Qian, Abdulellah Abualshour, Guohao Li, Ali Thabet, and Bernard Ghanem, "Pu-gcn: Point cloud upsampling using graph convolutional networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11683–11692.

[6] Ruihui Li, Xianzhi Li, Pheng-Ann Heng, and Chi-Wing Fu, "Point cloud upsampling via disentangled refinement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 344–353.

[7] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.

[8] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[9] Xinhai Liu, Zhizhong Han, Xin Wen, Yu-Shen Liu, and Matthias Zwicker, "L2g auto-encoder: Understanding point clouds by local-to-global reconstruction with hierarchical self-attention," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 989–997.

[10] Xinhai Liu, Xinchen Liu, Zhizhong Han, and Yu-Shen Liu, "Spu-net: Self-supervised point cloud upsampling by coarse-to-fine reconstruction with self-projection optimization," *arXiv preprint arXiv:2012.04439*, 2020.

[11] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[12] Na Lei, Dongsheng An, Yang Guo, Kehua Su, Shixia Liu, Zhongxuan Luo, Shing-Tung Yau, and Xianfeng Gu, "A geometric understanding of deep learning," *Engineering*, vol. 6, no. 3, pp. 361–374, 2020.

[13] Leonard Adolphs, Hadi Daneshmand, Aurelien Lucchi, and Thomas Hofmann, "Local saddle point optimization: A curvature exploitation approach," in *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, Kamalika Chaudhuri and Masashi Sugiyama, Eds. April 2019, vol. 89 of *Proceedings of Machine Learning Research*, pp. 486–495, PMLR.

[14] Huidong Liu, Xianfeng Gu, and Dimitris Samaras, "Wasserstein gan with quadratic transport cost," in *Proceedings of the International Conference on Computer Vision*, October 2019.

[15] I. Deshpande, Z. Zhang, and A. Schwing, "Generative modeling using the sliced wasserstein distance," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 3483–3491.

[16] 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*, 2017, pp. 605–613.

[17] Na Lei, Kehua Su, Li Cui, Shing-Tung Yau, and Xianfeng David Gu, "A geometric view of optimal transportation and generative model," *Computer Aided Geometric Design*, vol. 68, pp. 1–21, 2019.

[18] Timo Hackel, N. Savinov, L. Ladicky, Jan D. Wegner, K. Schindler, and M. Pollefeys, "SEMANTIC3D.NET: A new large-scale point cloud classification benchmark," in *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2017, vol. IV-1-W1, pp. 91–98.

[19] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun, Eds., 2015.