Abracadabra: Style Transfer in 3D

Ifrah Idrees, Abaho Katabarwa, Ben Abbatematteo, Usama Naseer
Brown University
Providence, RI

1. Introduction

The use of neural networks for style transfer has resulted in numerous practical applications ranging from digital art (sold for around half a million dollars) to generating fake videos (in form DeepFakes, used for fake news dissemination). Although the use of GANs for 2D images has been explored widely in the past, we further test the potential of GANs in the realm of 3D objects in this project. We leverage recent advancements in training GANs, in form of cycle consistency loss, to optimize our network to apply style transfer to 3D objects.

2. Related Work

3D model generation has received significant attention in the past. Early attempts at 3D model generation mostly focused on template-based solution that used union or modification of existing models [3, 1] or the use of RGB images and depth mapping to generate 3D models from images [5, 4]. However with the advent and success of deep learning, the use of GANs for 3D object generation provided a new platform and many recent stusides have explored this [8, 10, 9, 7]. The basic goal is to use creative power of generative adversarial networks to understand the details of 3D objects and generate new 3D objects from the learned distributions. GAN consists of a generator model G and a discriminator model D. Generator model generates new samples, whereas Discriminator model takes real and generated samples and tries to distinguish real ones from generated ones. Generator and discriminator are trained simultaneously so that while the generator learns to generate better samples, the discriminator becomes better at distinguishing samples, resulting in an improved sample generation performance at the end of the training.

Style transfer has emerged as a useful application for neural networks where texture of a set of images new images of (usually artworks) is applied to images from other domain to produce high perceptual quality images that blend them together such that the input image is transformed to look like the set of content images.

A recent study [11] explored the use of GANs for style transfer of 2D images with great success, especially when paired training data is not available. Their approach introduces cycle-consistency to GANs. Cycle consistency dictates that if we have a translator G : X -> Y and another translator F : Y -> X, then G and F should be inverses of each other, and both mappings should be bijections.

3. Our Solution

3.1. Dataset

We turn to the Shapenet dataset [2] for our source of 3D models. The full dataset covers 55 common object categories with about 51,300 unique 3D models. We represent shapes in voxel space, represented as 3D occupancy grids of size $64 \times 64 \times 64$. Some samples from the dataset are shown below in Figure 1.

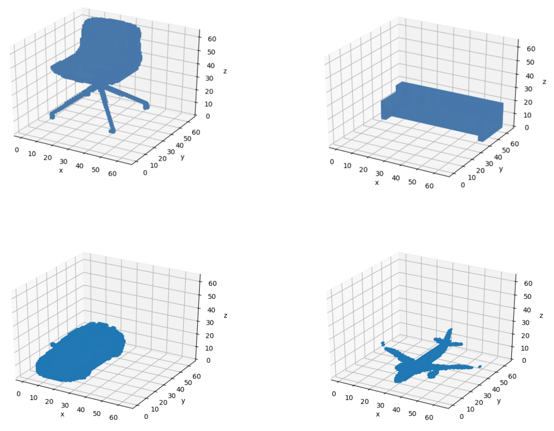


Figure 1. Example objects from the chair, sofa, car, and airplane object classes. We attempt to learn mappings between chairs and sofas, and between cars and airplanes.

3.2. Objective

3.3. Adversarial Losses

Our objective is to learn mappings between two domains X and Y given training examples $\{x_i \in X\}_i^N$, $\{y_i \in Y\}_i^M$.

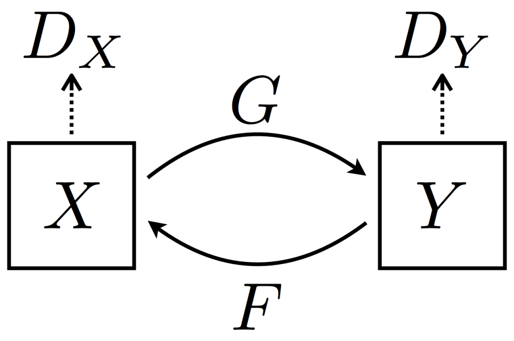


Figure 2. CycleGAN setup. Our objective is to learn mappings $G : X \mapsto Y$ and $F : Y \mapsto X$ in adversarial fashion.

We denote the two mappings $G : X \mapsto Y$ and $F : Y \mapsto X$, as shown in Figure 2. Each mapping is trained in adversarial fashion with a corresponding discriminator for each domain, D_X and D_Y. The adversarial loss for the mapping G is given as:

$$\begin{aligned}\mathcal{L}_{GAN}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}(y)}[\log D_Y(y)] \\ + \mathbb{E}_{x \sim p_{data}(x)}[\log(1 - D_Y(G(x))],\end{aligned} \tag{1}$$

and the loss $\mathcal{L}_{GAN}(F, D_X, Y, X)$ is defined analogously.

3.3.1 Cycle Consistency Loss

The adversarial objective given above leaves the mappings between the two domains largely unconstrained: the mappings are free to map each source shape to a random target shape, or even all source shapes to the same target shape. In order to regularize this objective, the authors of [10] introduce *cycle consistency loss*, the notion that the mappings should reproduce the input shape when composed, i.e. $F(G(x)) \approx x$ and $G(F(y)) \approx y$. The forces G to map each unique element in X to a unique element in Y such that the original shape may be reconstructed, and vice versa for F. Formally, cycle consistency loss is given as:

$$\begin{aligned}\mathcal{L}_{cyc}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}[||F(G(x)) - x||_1] \\ + \mathbb{E}_{y \sim p_{data}(y)}[||G(F(y)) - y||_1].\end{aligned} \tag{2}$$

Note that this loss is the L1 norm rather than an additional adversarial loss on the reconstruction; the authors reported improved performance with this metric in preliminary experiments.

3.4. Full Objective

The full objective is given as:

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{GAN}(G, D_Y, X, Y) \\ + \mathcal{L}_{GAN}(F, D_X, Y, X) \\ + \lambda \mathcal{L}_{cyc}(G, F).\end{aligned} \tag{3}$$

3.5. Model

The architecture we build is similar to that of cycleGAN [10] but substitutes the generator model from 3DGAN [9]. Below, we briefly discuss the constituent models of these architectures.
The structure of 3DGAN is similar to that of the standard GAN; however, 3DGAN makes use of 3 dimensional convolution and deconvolution in place of the 2 dimensional convolution and deconvolution present in the generator network [6] (see Figure 3).
The cycleGAN archicture makes use of two generator discriminator pairs. The first generator, G, takes voxel grids x from distribution X and outputs y' voxel grids of the same dimension as x. Similarly, the second generator, F, takes voxel grids y from distribution Y and outputs x'. The discriminator model is the same as the one used in the 3DGAN architecture (See Figure 2).

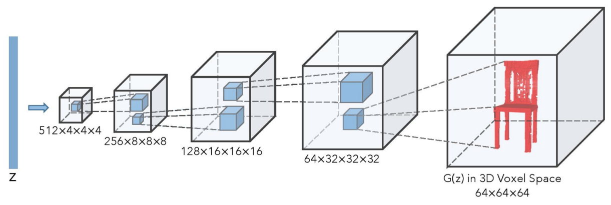


Figure 3. 3DGAN generator architecture

3.6. Training details

See Table 1 for an enumeration of the training experiments. Training experiments were ran on Tesla K80 or Tesla T4 GPU's. See Table 1 for the training parameters for each experiment. Two sets of mappings were attempted: chairs to/from couches and planes to/from cars. These taxonomy classes were chosen primarily because of the large number of available examples ($\geq$ 3000 examples per class) and the relative visual similarity between source and destination classes.

The authors of [10] replace the the standard adversarial negative log-likelihood loss with least squares loss, citing improved training stability. To investigate the effect of different loss functions, we attempted training our models with the standard objective (BCE) as well as least absolute deviation (L1) loss (see Table 1).

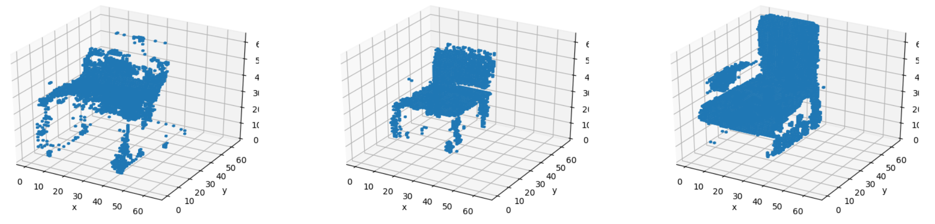


Figure 5. Fake chairs

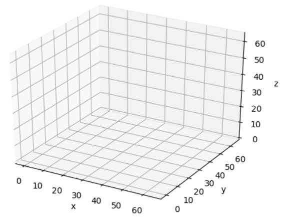


Figure 4. Reconstructed chair

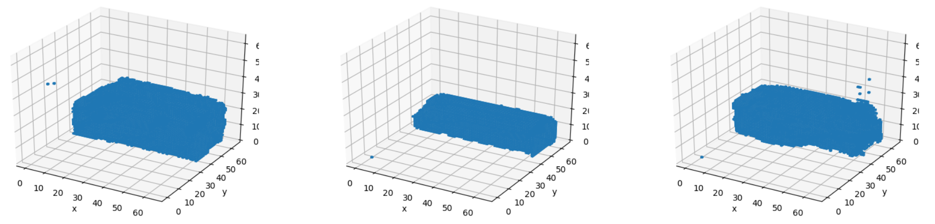


Figure 7. Fake couches

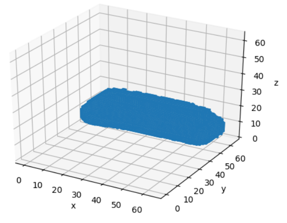


Figure 6. Reconstructed couch

Experiments						
No.	bs	d_lr	g_lr	ntrain	loss	type
1	16	1e-4	5e-4	6000	L1	A
2	8	1e-4	2.5e-3	4045	BCE	B
3	8	1e-5	2.5e-3	6000	L1	A
4	6	1e-4	5e-4	500	L1	B

Table 1. Experiment Parameters. *See the following legend for the symbols in the table.* bs: batch size, d_lr: discriminator learning rate, g_lr: generator learning rate, ntrain: number of training examples, loss[L1]: least absolute loss, loss[BCE]: binary cross entropy loss,type[A]: chairs to couches, type[B]: planes to cars

4. Results

4.1. Chairs to Couches

Figures 4-7 plots selective results for one of our Chair to Couches model attempt (Experiment 2). We observed the Generator and Discriminator accuracy to be climbing close to 1 fairly quickly for the default learning rates and this prompted us to use lower learning rates (listed in the table). Through trial and error, we found our batch size to be limited by GPU memory and 16 being the highest batch size the 12GB memory of GPU was able to support. We used Least absolute deviations (L1) as the loss function in this experiment. As it can be seen in Figures 4-7, our network is able to produce fake chairs and couches with some success. We note the reconstruction of couches appears to yield the mean couch object: the occupied space in the reconstructed samples is filled in nearly all couch datapoints, since couches have consistent geometry. Surprisingly, reconstruction for chair produced empty results. Our intuition is that chairs have less consistent geometry: a given voxel is more likely to be empty than full across the dataset, and therefore the mean chair sample is an empty voxel grid.

4.2. Planes to Cars Model

In addition to training the 3D chairs and couches model, we also attempted to train a planes and cars model. For this particular run, we set the discriminator loss to $1 - e4$, used a batch size of 6, used least absolute deviation loss as the objective function. The results were as follows. Generator A, which was expected to produce a car-like output, converged to structures that look car-like (See Appendix Table 3 Experiment 4). Generator B, expected to output a plane-like output, produced a nearly empty volume. Similarly to the couch reconstruction result above, the reconstructed cars resemble the mean car sample in the dataset. This may additionally be due to the empty grids (fake planes) from which the reconstructed cars where generated. While these results were unexpected, they demonstrate that the mapping in question can be learned. We hypothesize that the source of these results is the high loss penalization of the reconstruction generators ($\lambda_A = 10$, $\lambda_B = 10$) which attempt to minimize loss by converging to degenerate mean representations of the their intended outputs. As such, we attempted lower weights on the reconstructions. The results of this experiment is shown in Table 3.

5. Discussion

A few interesting insights that we were able to make gather from the runs that we did on training our model with different hyperparameters are as follows:

- Comparison of Experiment 2 and 3 showed that lowering the learning rate of the discriminator by 10 times greatly improved the construction of fake images of the domain Y. In case of the aforementioned experiments, domain Y was couches. Lowering the learning rate helped since we believed that it travelled slowed toward the downward gradient and did not miss any local minimas.
- The configuration of parameters show in table 1 was run the earliest by us. The results of experiment 2 and 1 can be visualised in the appendix. Experiment 2 generated nearly empty voxel grid representations for domain Y(couches) which led us to question if the batch sizes were an impedance is producing better results therefore we setup up experiment 1 with batch size of 16 since our Tesla K80 GPU supported 24GB of memory and as it turns out this helped in smoothing out back propagation, leading to generation of better 3-D representation of couches.
- While running the experiments we noticed better results with least absolute loss function as compared to the Binary Cross entropy. This outcome was in accordance with the statement of the authors of cycle GAN [10].

6. Conclusion

In this paper we explored style transfer in 3D using 3D generative adversarial networks. We combined CycleGAN [10] and 3DGAN [9]. We observed that our model was fairly successful in learning forward mappings between domains, but the reconstruction of the original domain converged to the mean distribution of the representative categories.

References

[1] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1

[2] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 1

[3] Siddhartha Chaudhuri, Evangelos Kalogerakis, Leonidas Guibas, and Vladlen Koltun. Probabilistic reasoning for assembly-based 3d modeling. In *ACM Transactions on Graphics (TOG)*, volume 30, page 35. ACM, 2011. 1

[4] Qixing Huang, Hai Wang, and Vladlen Koltun. Single-view reconstruction via joint analysis of image and shape collections. *ACM Transactions on Graphics (TOG)*, 34(4):87, 2015. 1

[5] Abhishek Kar, Shubham Tulsiani, Joao Carreira, and Jitendra Malik. Category-specific object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1966–1974, 2015. 1

[6] Thom Lane. 1d 3d convolutions explained with... ms excel!, 2018. 2

[7] Cihan Ongun and Alptekin Temizel. Paired 3d model generation with conditional generative adversarial networks. *CoRR*, abs/1808.03082, 2018. 1

[8] Edward J. Smith and David Meger. Improved adversarial systems for 3d object generation and reconstruction. *CoRR*, abs/1707.09557, 2017. 1

[9] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016. 1, 2, 4

[10] Jing Zhu, Jin Xie, and Yi Fang. Learning adversarial 3d model generation with 2d image enhancer. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 1, 2, 4

[11] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017. 1

A. Appendix

A.1. Visualization of Results

The figures below show additional runs of our system with various hyperparameters. See Table 1 for hyperparameters.

A.2. Contributions of Team Members

Contributions from our team members were remarkably even. Together we explored datasets, setup GCP infrastructure, and wrote code for building and training our 3DcycleGAN. Independently, we each ran several experiments and shared results with the group as a whole. Finally, each team member contributed various sections of the written report, drafting as well as editing.

Table 2. Visualisations for Experiment 1

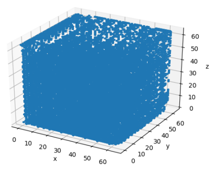
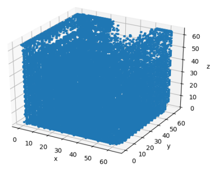
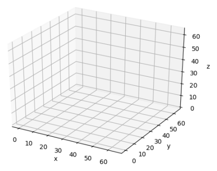
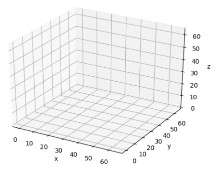
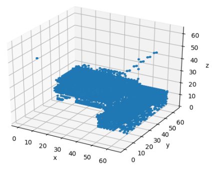
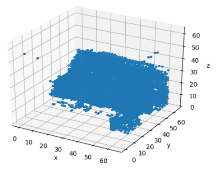
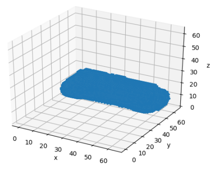
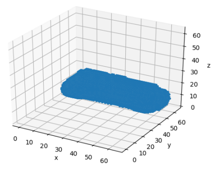
fake chair		reconstructed chairs	
			
fake couches		reconstructed couches	
			

Table 3. Visualisations for Experiment 3

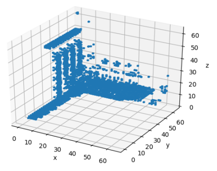
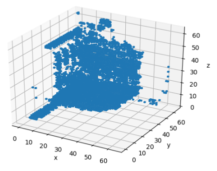
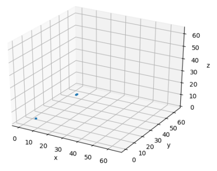
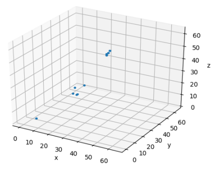
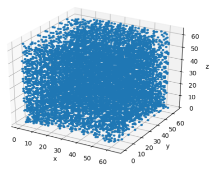
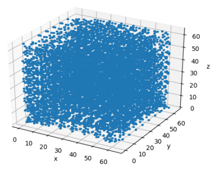
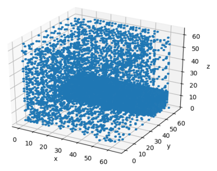
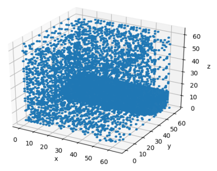
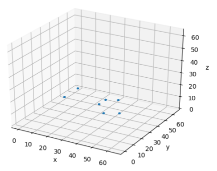
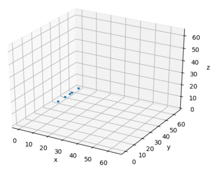
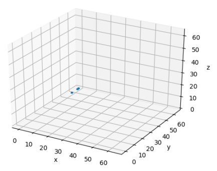
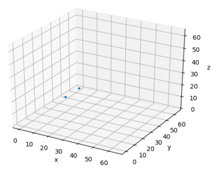
fake chair		reconstructed chairs	
			
fake couches		reconstructed couches	
			

Table 4. Visualisations for Experiment 4

fake planes		reconstructed planes	
			
fake cars		reconstructed cars