Vox2Net: From 3D Shapes to Network Sculptures

Nima Dehmamy¹, Luca Stornaiuolo^{2,3} & Mauro Martino⁴

Abstract

Sculptures provide the highest degree of physicality in art, allowing the human viewer to make full use of their three dimensional understanding and connect with the piece of art. The advent of Generative Adversarial Networks (GAN) has afforded Artificial Intelligence with a means to create imagery, music and other products which rival human creations. Here we introduce a modified pix2pix GAN, which we call Vox2Vox, that is able to convert different 3D representations of a 3D object to one another. In particular, we teach Vox2Vox to construct a 3D network as an abstract way to represent a sculpture, a construction we call Vox2Net. The input of Vox2Net is a point cloud of the 3D sculpture and its output is spherical nodes and tubular links which together mimic the abstract shape of the original sculpture. Vox2Vox allows the user to convert 3D shapes to any abstract representation of the shape, as well as different styles using the appropriate training data.

Artificial Intelligence (AI) is seeing an enormous growth in popularity in the realm of creative art and design. It has found applications in style transfer [3, 10], video editing, and image reconstruction, to name a few. Generative Adversarial Networks (GAN) [4], in particular, have played an integral role in the recent surge, owing to their ability to learn representations of data and generating outputs in the form of the natural images and objects [7]. Algorithms such as pix2pix [5] are also able to recognize and reconstruct images from abstract drawings. While most of these efforts in using GAN have focused on images, some have also applied these methods to 3D shapes, reconstructing the 3D shape from images [9]. The goal of the current work is to take one further step and use AI to directly construct a different 3D representation of an existing 3D shape using a GAN which we call Vox2Vox.

1 Results and Methods

Our full pipeline, Vox2Net, consists of a Vox2Vox GAN (described below) combined with the FUEL 3D network layout algorithm [2, 1] used to improve the final outcome, similar to the WonderNet project [6]. We perform some post-processing to infer and extract a graph from the Vox2Vox output and then use FUEL on the inferred network. We want to convert 3D shapes into a 3D network. A network (graph) has two main components: 1) Nodes, being the entities to be connected and 2) Links, which are the wires connecting the nodes. Similar to a scaffold that mimics the shape of the 3D object, we want the nodes to be placed in suitable locations inside the shape and links that connect these nodes to form a 3D network.

Since we want the output of Vox2Vox to mimic the shape of the input, we choose pix2pix [5] as the backbone of the Vox2Vox architecture. The power of pix2pix, and hence Vox2Vox, is that we can teach it to convert shapes into any abstract representation of it, as long as there is a spatial correlation

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

^{*1} nimadehmamy.com, Department of Physics, Northeastern University, Boston, MA 02115, n.dehmamy@northeastern.edu

^{†2} Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, 20133 Milano, Italia, luca.stornaiuolo@polimi.it

^{‡3} Visual AI Lab, IBM Research, Cambridge, MA 02141, luca.stornaiuolo@ibm.com

 $^{^{\}S4}$ www.mamartino.com, Head of the Visual AI Lab, IBM Research, Cambridge, MA 02141, mmartino@us.ibm.com

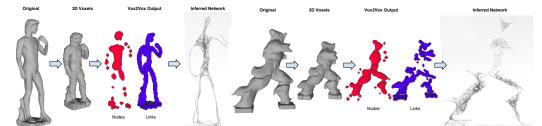


Figure 1: **Vox2Net Pipeline Illustration:** We begin from a 3D file (OBJ, STL, etc.). We first convert the data to filled 3D voxel data. The 3D voxel array is fed as input to the generator of Vox2Vox, which returns an array with two channels for nodes and links of the network. We then use K-means, random geometric graphs and the A* search algorithms to infer a network from the neural net output, keeping the paths of the links.

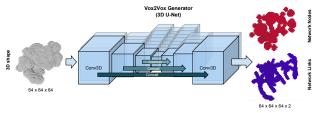


Figure 2: The Vox2Net generator has a U-Net architecture, taking a $64 \times 64 \times 64$ binary array of a 3D shape as input and generating a $64 \times 64 \times 64 \times 2$ array as output. The first layer of the output are the nodes and the second layer are the links of the network. The full Vox2Vox GAN mimics the pix2pix GAN, except that all convolutional layers are 3D instead of 2D.

between the two shapes. Our main modification to pix2pix is that all convolutional layers have been changed to 3D convolutions (Conv3D), instead of 2D (Fig. 2). During training, the input is a $64 \times 64 \times 64$ binary array (0 in empty spaces and 1 inside the 3D shape) and output of the generator is a $64 \times 64 \times 64 \times C$ array, where C is the number of channels. The generator has a U-Net [8] architecture, with 4 3D convolutional layers with 32, 64, 128 and 128 filters, encoding the input to $4 \times 4 \times 4 \times 128$ shape, and then 4 3D deconvolution layers with 128, 64, 32 and 2 filters decoding it to the output shape. The deconvolution layers also get the output of the encoding layers as input, as in U-Net (Fig. 2). The discriminator consists of 3 3D convolutional layers, yielding a $4 \times 4 \times 4$ output. The cost function for the discriminator decides whether on all the $4 \times 4 \times 4$ patches of the output the input 3D shape and the output network matched satisfactorily. Since we want a network as the output, we choose the number of output channels of Vox2Vox to be C = 2, encoding nodes in the first channel and links in the second (Fig. 2). A nice feature of pix2pix is that, since all layers are convolutional, we can produce larger outputs by simply modifying the input shape of the trained network. Therefore, after we train Vox2Vox on $64 \times 64 \times 64$ voxels, we change the input shape to $128 \times 128 \times 128$ and produce higher quality outputs. We used Vox2Vox on a variety of different 3D sculptures and used some post-processing to extract the generated 3D network to complete the Vox2Net pipeline (Fig. 1)

The training procedure of Vox2Vox is the same as pix2pix: We present the discriminator with a pair of inputs, which is the concatenation of the $64 \times 64 \times 64$ array of the 3D shape with the $64 \times 64 \times 64 \times C$ array of real or fake output along the channels. We create a variety of networks first and then lay them out in 3D using our custom force-directed layout algorithm [2] which makes sure nodes do not overlap⁵. We then create two arrays from the network. One is a $64 \times 64 \times 64 \times 64 \times 64 \times 2$ array, merging nodes and link segments by replacing them with overlapping large and small spheres, respectively. This 3D shape is the input of the generator. The second is a $64 \times 64 \times 64 \times 64 \times 2$ array, with the first channel being the nodes and the second the links, which are again replaced by spheres, this time with smaller where radii (Fig. 2). We generate about 700 such networks and heavily augment the dataset by rotating them in multiples of 20 degrees about the x, y and z axes to create a dataset of about 30,000 data points.

To extract a network from the Vox2Vox output, we first use K-means to choose centers for nodes and points along the links. We then form a network from points that are close to each other, and finally use the A* search algorithm to extract links between nodes.

⁵The full algorithm also bends links to avoid link crossings, but we used a simpler version for this work, keeping links straight

References

- Nima Dehmamy. FUEL and ELI 3D network layout algorithms. https://github.com/ nimadehmamy/3D-ELI-FUEL.
- [2] Nima Dehmamy, Soodabeh Milanlouei, and Albert-László Barabási. Structural transition in physical networks. *Nature (to appear)*, 2018.
- [3] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016.
- [4] 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*, pages 2672–2680, 2014.
- [5] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017.
- [6] Mauro Martino, Alice Grishchenko, Nima Dehmamy, Hendrik Strobelt, and Albert-László Barabási. Wondernet, 3d network layouts. http://netwonder.net/.
- [7] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.
- [8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
- [9] 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*, pages 82–90, 2016.
- [10] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *Computer Vision (ICCV)*, 2017 *IEEE International Conference on*, 2017.