Understanding the Data Behind the Pix2Vox Method

Cody Harris
Indiana University
Bloomington, IN
harrcody@iu.edu

Emma Cai
Indiana University
Bloomington, IN
caie@iu.edu

Neelan Scheumann
Indiana University
Bloomington, IN
nscheuma@iu.edu

Abstract

Dataset collection and curation can prove to be a time consuming and expensive endeavor, especially in the field of computer vision. When it comes to creating 3D volumes of an object, there are some great datasets out there, but they come with some caveats. The largest datasets (ShapeNet, Things3D) come from computer generated images, such as CAD models created for other purposes than computer vision research. These datasets include enough data to adequately train deep learning networks. On the other hand, there are datasets like Pix3D that include images of real objects in a real scene with binary masks that segment the item in the scene, along with the ground truth voxel model. A dataset like this could prove to be a great training set of real images, but it is generally too small for the task. This leads researchers to have to train on objects that exist in these large datasets made of computer generated images. We will show that it is better to train on the exact objects you would like to make a 3D reconstruction of, as this method does not always generalize well. Our main purpose of research is to understand what aspects of the dataset is important for a 3D reconstruction task, and if smaller datasets can be used to train the model with acceptable results.

1. Introduction

Generation of 3D volumes from 2D pictures is an area of computer vision that has an extremely wide array of applications for a variety of fields. There are obvious applications such as autonomous robots or vehicles being able to understand the contents of a 3D scene with only 2D images. Even if a robot were to have two eyes like a human, and thus able to get a sense of depth, occlusions could present a problem in understanding the 3D volume. For example, if a scene includes a view of a chair with arms, but only one arm is visible, the 3D reconstruction would require inferring that there should be a second arm.

Another application is CAD software. If one is to search the internet for solutions for 3D reconstruction from 2D images, much of the results and conversations are centered around using CAD software. This could be used in an application such as creating a bust of a person's head as a piece of art, all the way to helping with prototyping a manufactured part from a single photo. These applications make these models extremely important parts of current and future CAD software so that users who are unfamiliar with computer science can have access to this technology.

The main focus of the following experiments is to infer the 3D volume of objects from one or more 2D images, but there is somewhat related work that seeks to infer the 3D volume of an entire scene. In the past Markov Random Fields were used to reconstruct entire cities or large geographical areas in 3D from many 2D images [3]. These efforts were done shortly before the deep learning revolution. At this point state of the art models are exclusively using deep learning methods.

One of the best, if not the best, current methods comes from the Pix2Vox and Pix2Vox++ methods [11, 12]. These methods use a deep learning approach to build 3D volumes in voxel form from one or more images of an object. The images, or views, of the same object must only include a single object and have nothing else. Preliminary experiments show extremely promising results on the ShapeNet [1]. Due to being a synthetic dataset, this dataset has many (generally 20+) views of the same image. This allows a great deal of flexibility.

Something that is not explicitly discussed in the Pix2Vox [11] paper is how the training and testing data were selected. This leads to questions on if one were to want to implement this method for their own needs, what amount or type of data would they need? Is it more important to have multiple views of an image? If one were to use less data, what type of accuracy loss would be expected? Is it more important to use more a more diverse set of images? These are the type of questions we seek to answer over the course of our research.

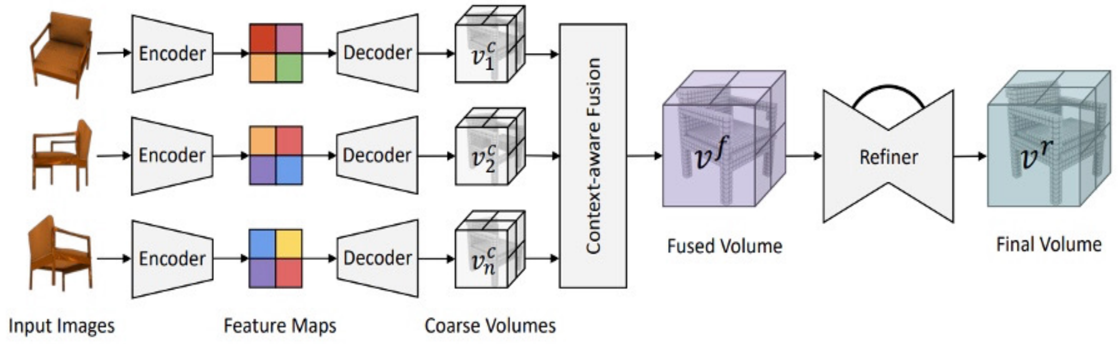

Figure 1. High-level workings of the Pix2Vox multi-view network. This is just an example of three views, but more or less can be used [11].

2. Background and Related Work

2.1. Attempts at 3D Reconstruction Before Pix2Vox

Not all methods work the same, or take the same output, but there are four methods used as benchmarks to compare results from Pix2Vox. 3D-r2n2 was the earliest of such methods that used a recurrent neural network for 3D reconstruction [2]. As mentioned in the original Pix2Vox paper, this method had caveats such as not being able to reproduce results regularly among other problems [11]. Octree Generating Networks or OGN [9] is a deep convolutional architecture that provided interesting, but ultimately not as good quantitative results on ShapeNet [1]. With that said OGN did tackle some things that Pix2Vox did not attempt, which is larger scale and higher resolution 3D scene reconstruction. Differentiable Ray Consistency (DRC) [10] and Point Set Generation Network (PSGN) [5] are both deep learning methods that were used to compare to Pix2Vox with the latter actually having better quantitative results on some image categories.

2.2. Pix2Vox Architecture

The Pix2Vox paper suggest two different architectures, for 3D reconstruction. There is Pix2Vox-A and Pix2Vox-F. In our research we only deal with Pix2Vox-A as Pix2Vox-F did not show much improvement, if any, over previous methods. The Pix2Vox architecture is made up of four main parts: encoder, decoder, context-aware fusion, and refiner. The context-aware fusion is only necessary for multi-view reconstructions. Figure 1 shows these four parts and how they interact.

The encoder is based on the VGG16 [8] model that has been trained on ImageNet [4] for the majority of the encoder. It is the job of the encoder to retrieve different features that make up an image so they may be fed to the decoder. The decoder then takes the features to create a 3D volume. It might seem as though many weights would be trained for the encoder and decoder when there is a multi-view training set, when looking at Figure 1, this is not the case, each encoder and decoder uses the same weights.

When using multiple views of the same object, each image is passed through the encoder-decoder and a 3D volume built for each view. These multiple volumes are "fused" according to the model and a new 3D volume is output via the content-aware fusion module. This output is used to calculate the loss (according to Equation (1)) to then back-propagate through the encoder-decoder. Although not specified, it is assumed that single view images are still processed through the content-aware fusion model, but due the equations provided, this would output the same as what was input.

The last part of the network is another encoder-decoder network known as the refiner. This refiner takes on a U-Net [7] architecture. The input is a 3D volume, and the output is a 3D volume. It is the job of the refiner to use context of the 3D world to then "smooth" the voxels in a way that makes sense. For example, if somehow the first encoder-decoder network found that with high probability there were some voxels that were floating and not connected to the main group of voxels, the refiner should eliminate or possibly connect these voxels because there should be no discontinuity in a single object. This is just an example of what the refiner could refine and is not limited to this sort of refinement.

The loss function used is simply the "voxel-wise binary cross entropies between the reconstructed object and the ground truth" [11], which is found in Equation (1). Figure 2 shows the exact architecture of the Pix2Vox-A model. For more details on this model, refer to the original publication.

$$l = \frac{1}{N}\sum_{i=1}^{N}[gt_i \log(p_i) + (1 - gt_i)\log(1 - p_i)] \quad (1)$$

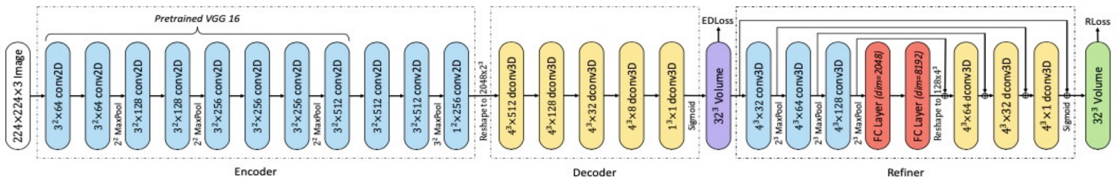

Figure 2. This is the network Architecture of Pix2Vox-A. Both Encoder-Decoder Loss (EDLoss) and Refiner Loss (RLoss) are calculated using Formula (1) [11]

2.3. Related Work

While there is not much published work related to evaluating the Pix2Vox model, there is a decent amount of collaboration on the Pix2Vox github issues section [1]. The only other related work that evaluates the Pix2Vox work and seeks to see improvements is the same team's subsequent publication titled Pix2Vox++ [12]. The main improvement made was replacing the VGG16 [8] in the encoder with a version of ResNet 18 [6]. Also the team create a way to do multi-scale fusion on the multiple views. While this is not directly related the questions asked by this team, it is interesting to see the improvements made over the course of a year.

3. Methods

We first began by implementing the Pix2Vox method that can be found at their Github repository. First their repository was cloned, and then the Python packages required were installed. Next the training and testing data must be downloaded and the paths to the files must be changed in the config file. From here the Pix2Vox method should be able to be reproduced. More information on this process can be found in the Readme file of their Github.

3.1. Additions and Changes to the Codebase

In order to get the program to work perfectly in our environment, some small tweaks to the code needed to be added. This was likely due to either differences in our environment, or possible changes to newer versions of packages used. The main changes were made so that a user could take pretrained weights and create a single visualization of the predicted voxels from the network. The current code did not have a way to do this, it did not have a way to get the IoU score for a single image as well. While for experiments sake, it is important to look at a wide range of images, sometimes it is helpful to see one off images as a litmus test of how well the model is doing to the human eye. Using suggestions and code from users within the issues section of the Pix2Vox Github, we add a way to run single images through the network. These code changes and everything needed for our experiments can be found on our fork of the repository[2]

3.2. Experiments

We sought out to investigate the training data's effect on the results in three different situations. The first situation was to investigate how the number of views used to predict the 3D volume would affect accuracy. Next, we investigated how the total size of the training data would affect results, it is known that creating datasets is extremely time consuming and expensive, knowing if you need a certain amount of data would be key in building a dataset for 3D reconstruction. Lastly, we built a dataset that contained roughly the same amount of training examples, but each of the three datasets had different number of categories. This led us to explore how diversity in training data would affect the results of the 3D reconstruction.

3.3. Method of Experimentation

We sought out to investigate the training data's effect on the results in three different situations. The first situation was to investigate how the number of views used to predict the 3D volume would affect accuracy. Next, we investigated how the total size of the training data would affect results, it is known that creating datasets is extremely time consuming and expensive, knowing if you need a certain amount of data would be key in building a dataset for 3D reconstruction. Lastly, we built a dataset that contained roughly the same amount of training examples, but each of the three datasets had different number of categories. This led us to explore how diversity in training data would affect the results of the 3D reconstruction.

3.4. Method of Measurement

We sought out to investigate the training data's effect on the results in three different situations. The first situation was to investigate how the number of views used to predict the 3D volume would affect accuracy. Next, we investigated how the total size of the training data would affect results, it is known that creating datasets is extremely time consuming and expensive, knowing if you need a certain amount of data

[1]https://github.com/hzxie/Pix2Vox/issues

[2]https://github.com/codyharris91/Pix2Vox-B657-Experiments

Experiment	Time (hrs)
1 View 250 Epoch Control	12
1 View 30 Epoch	1.5
4 View 30 Epoch	6
8 View 30 Epoch	9.75
12 View 30 Epoch	12
1 View Half-data 250 Epoch	7.5
1 view Quarter-data 250 Epoch	5
1 view 6 Categories 250 Epoch	12
1 view 13 Categories 250 Epoch	12

Table 1. Training times of all experiments. All training was done on a machine using a GTX 1080TI GPU, which is the same as the original Pix2Vox experiments.

Category	Our IoU Pix2Vox	Best Model IoU
Aeroplane	0.662	0.684
Table	0.586	0.601
Telephone	0.795	0.776
Average	0.681	0.687

Table 2. Comparison of our results from our model compared to Pix2Vox best model.

would be key in building a dataset for 3D reconstruction. Lastly, we built a dataset that contained roughly the same amount of training examples, but each of the three datasets had different number of categories. This led us to explore how diversity in training data would affect the results of the 3D reconstruction.

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{2}$$

4. Results

4.1. Baseline Results

In order to reduce the overall data size, the team randomly chose three categories and only used those for training instead of all thirteen. Within the chosen categories, all the same images will be used as the original Pix2Vox paper. The three chosen categories are: planes, tables, and telephones. A model was trained on these three categories and the results of the model compared to the best model from Pix2Vox is shown in Table 2. Surprisingly, the best model from Pix2Vox with much more training examples, ended up having relatively similar results. Our model ended up slightly worse on the airplanes and tables, while outperforming the telephone reconstruction. This could be due to the simplicity of the telephone models.

4.2. Multi-View

In order to gauge how important it is to have multiple views of the same object when reducing the size of the training data to three categories, we did an experiment that used either: one, four, eight, and twelve views. As shown in Table 1, the training times increased significantly when more views were trained. In order to accomplish these tests in a reasonable time, we had to come up with a solution. It is estimated that the twelve-view training would have taken in the realm of five to six days alone. The solution was to only train on 30 epochs. Without running this experiment fully, it was hard to know if reducing to 30 epochs would be too restrictive to provide relevant results.

	1View	4View	8View	12View
Airplane	0.575	0.586	0.539	0.527
Table	0.553	0.538	0.5032	0.491
Telephone	0.691	0.732	0.613	0.682
Average	0.606	0.619	0.552	0.567

Table 3. Results of the multiple view experiment.

The results of this experiment can be seen in Table 3. The values obtained show somewhat inconclusive results. It seems as though results improve marginally when moving to using four views over one view, but then drops off for eight to twelve views. While it is possible that more views is not important in this case, it goes against the results of the original Pix2Vox findings and what would be expected. This likely could be due to only using 30 epochs.

4.3. Reducing Training Data Size

The original data used by Pix2Vox used around 40,000 training images. Once we reduced down to three categories, we reduced the size of the data by about one quarter, or about 9500 objects. As show by our baseline results section, this reduction in training size showed only slight changes in accuracy. With a model such as this, there is a point where adding more training data will not increase the accuracy, and the accuracy will not approach perfect. This leads to the belief that a certain data size is optimal. While more data will never be a bad thing, spending more money on data collection and refinement could become a problem. Also, this dataset is large and unwieldy, so to be able to reduce the size of the training set would allow this method to be more accessible. From our already paired down dataset, we further test what happens if we randomly and proportionally remove half of the training set, and a quarter of the training set. This leaves with about 4700 and 2300 training examples. Table 4 shows the quantitative results. Obviously, less data is going to result in less accurate results, which is what we observed. With that said, it is hard to understand what this reduction in accuracy would mean to the human eye. Figure 5 shows one randomly picked example from each of the three categories and the resulting 3D reconstruction. While the table didn't come out looking good with only 25% of the data, the other two categories look

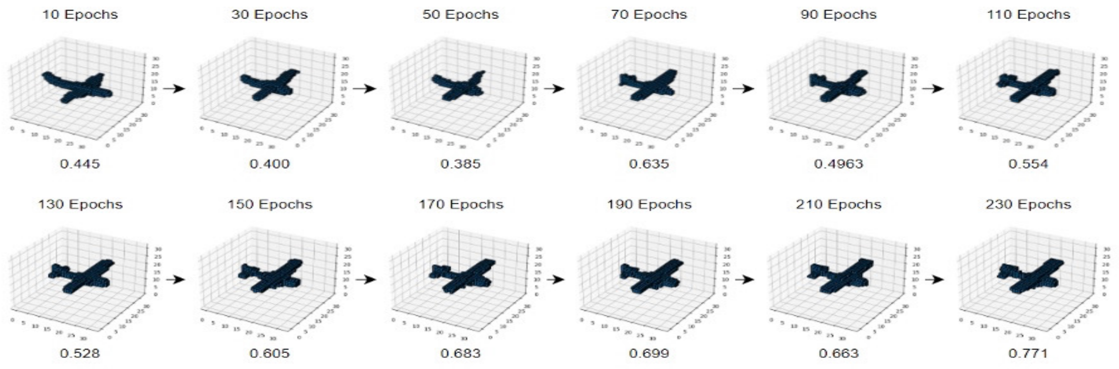

Figure 3. A progression of the predicted 3D volume based on the image in Figure 4 across 230 epochs along with the corresponding IoU value. [11].

	100%	50%	25%
Airplane	0.662	0.634	0.599
Table	0.586	0.553	0.516
Telephone	0.795	0.74	0.723
Average	0.691	0.647	0.62

Table 4. Results of reduced data experiment.

	3 Cat.	6 Cat.	13 Cat.
Airplane	0.662	0.65	0.6044
Table	0.586	0.19	0.498
Telephone	0.795	0.686	0.71
Average	0.691	0.509	0.604

Table 5. Results of trained models when different amounts of categories are used in training.

convincing with such little data. If someone were looking for a method that would give a rough understanding of the 3D object, even the 50% of the data would likely suffice for non-precise applications. For all of the experiments, the models were trained on 250 epochs.

4.4. More Categories Same Size

We sought to know if it was helpful to have more diversity in the training data in order to still reconstruct images from the same three categories we have been working with. This might seem like an odd endeavor, but within each category there is a great deal of variability. This lends itself to the thinking that maybe by training the model on other categories. Within each category there is a great deal of diversity as well, some categories more diverse than others. The telephone category does not have a great deal of diversity, but categories like planes and tables do.

In this experiment there are three trained models. One with three, six, and thirteen categories. For each of the models, approximately the same number of training examples are used in each test. Table 5 shows the results of this experiment. It seems as though more information will need to be collected and a further study on this specific problem. With a few data points there is no correlation, but that could just be due to the six categories chosen at random, or even the three categories that were originally chosen. Overall, it would seem that diversity in training is detrimental to the accuracy of the model.

What is interesting is that the table category was much more negatively impacted by a diverse training set then any other category. This leads to an interesting discussion as to what the model is actually learning.

5. Discussion

There is no question that our research and study had its limits. If we had more time or possibly better equipment for training, more models could be evaluated to find what is important in a dataset for 3D reconstruction from a 2D image. With that said, our study dove deeper into showing the results of the Pix2Vox method when certain changes are made. As there are more and more datasets emerging constantly to try to tackle this problem, there should also be a great deal of thought and research into what makes a good dataset.

When examining this work in a broader sense, sometimes we struggle to understand what our deep learning models are learning. Our last experiment about categories brings up a good question about what features are being learned by the model. How much importance is given to color and shading? Is the model learning a classification of sorts while doing the feature selection? These experiments can shed light onto how a particular model is being learned.

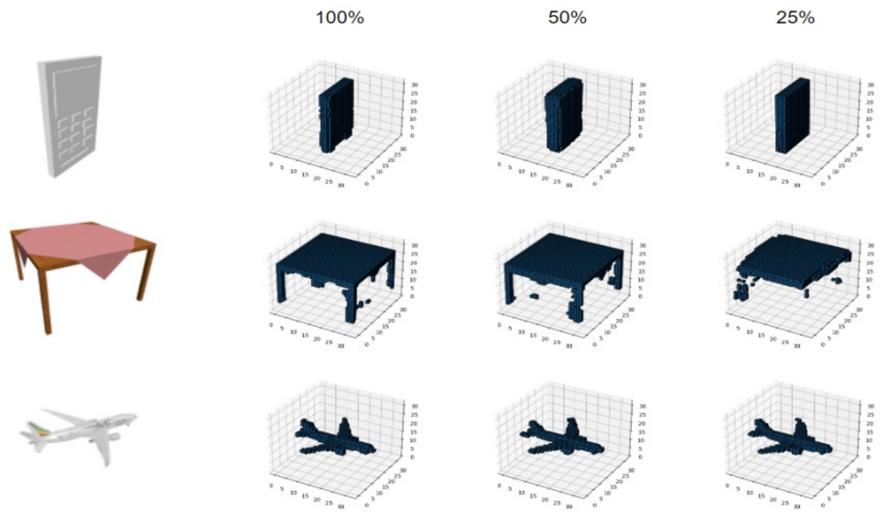

Figure 4. Three examples from each category at different levels of reduced training data. [11].

6. Conclusion

Our work creates a starting point for methods of evaluating a model for 3D reconstruction from a 2D image. In many cases these methods are presented in a conference paper or a publication in which the authors cannot explain their entire decision-making process and others are led to make certain assumptions that the authors did ample amounts of testing to make sure that their results are the best they can be. In the case of Pix2Vox, there is no explicit explanations on how their data was selected, split, or how that training had an effect on the end results.

Another interesting aspect that is not explored by Pix2Vox, that we are able to briefly show in our paper is that sometimes a lower IoU score would still allow for the accuracy needed for many tasks. While it is always good to try to go for the highest quantitative results, some other metrics such as time complexity or memory complexity might be of a higher interest than quantitative results alone. Our project could be extended to other datasets, or even trying more parameters within our current experiments. The hope is always to use the least amount of data as possible that still gives acceptable results.

7. Acknowledgements

The authors would like to thank Dr. Reza, Dr. Crandall and the associate instructors in the SP21-BL-CSCI-B657: Computer Vision course (offered in the Spring 2021 semester at Indiana University, Bloomington) for their continued assistance and suggestions with regard to exploring this idea.

References

[1] Angel X Chang et al. “Shapenet: An information-rich 3d model repository”. In: *arXiv preprint arXiv:1512.03012* (2015).

[2] Christopher B Choy et al. “3d-r2n2: A unified approach for single and multi-view 3d object reconstruction”. In: *European conference on computer vision*. Springer. 2016, pp. 628–644.

[3] David Crandall and Noah Snavely. “Modeling people and places with internet photo collections”. In: *Communications of the ACM* 55.6 (2012), pp. 52–60.

[4] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[5] 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.

[6] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Confer-*

ence on Medical image computing and computer-assisted intervention. Springer. 2015, pp. 234–241.

[8] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[9] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. "Octree Generating Networks: Efficient Convolutional Architectures for High-Resolution 3D Outputs". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.

[10] Shubham Tulsiani et al. "Multi-View Supervision for Single-View Reconstruction via Differentiable Ray Consistency". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[11] Haozhe Xie et al. "Pix2vox: Context-aware 3d reconstruction from single and multi-view images". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 2690–2698.

[12] Haozhe Xie et al. "Pix2Vox++: multi-scale context-aware 3D object reconstruction from single and multiple images". In: *International Journal of Computer Vision* 128.12 (2020), pp. 2919–2935.