Reducing Complexity in 3D Shape Reconstruction

Chelsey Toribio

ct3113@columbia.edu

Abstract

This report introduces the methods used to construct a 3D Shape Reconstruction model using a 3D Recurrent Reconstruction Neural Network (3D-R2N2) architecture. This neural network takes in ShapeNet's [2] 2D images and outputs a voxelized 3D reconstruction shape. Building upon the 3D-R2N2 [1] architecture, the difference between the original network and my network is I simplified the architecture during decoding to reduce complexity. The architecture integrates an Encoder, a variation of the LSTM called Gated Recurrent Units (GRU), and a Decoder architecture. Evaluation on the ShapeNET test dataset shows the model producing enough voxels for the output to take shape. From visualization, the 3D shape reconstruction lacks the finer details. With a result of a low IoU score, this is apparent. Even though the IoU score is lower than the standard for consideration of a "good" score, the training and validation loss is indication of an average performing model.

1. Introduction

Performing a 3D shape reconstruction from a single image has been one of the main problems researchers have faced. It is a complex computational task as its goal is to capture three-dimensional objects from two-dimensional images by rebuilding. Many experiments were done to reconstruct a 2D image as a 3D shape using existing algorithms on small datasets, but it is still limited. With the technological evolution of 3D models built from Computer-Aided Design (CAD), there are now improved datasets to use and facilitate a new or existing algorithm. The ShapeNet [2] dataset accomplishes the task by offering a repository of 2D images and its 3D shapes to perform reconstruction. This problem, once solved, can revolutionize technology that relies on spatial recognition in various fields. Such technology includes autonomous driving, reconstructive surgery planning, constructing anatomy, and much more. Industries like gaming can take advantage of the advancement as 3D reconstruction can create an immersive gaming and virtual reality (VR) experience. The architectural sector can benefit from a new way of visualization where blueprints can come to life without manually drawing in a design application. Research in this area can pave the way for significant advancements in how 3D reconstruction interacts with the world.

2. Related Works

Historically, the approach to 3D shape reconstruction was done using a variety of algorithms, aimed at converting two-dimensional images into three-dimensional objects. One such experiment includes a neural network of point set prediction that has been developed that can generate 3D point clouds from a single 2D image.[3]. Its network uses an encoder and predictor architecture that discerns shape and depth. Another notable experiment introduces a TripoSR framework that can perform a fast feed-forward 3D reconstruction to process faster creation of 3D objects [4]. This significantly reduces the time it takes to generate 3D models and can be effectively used for real-time applications. More recently, another experiment used a NeuSDFusion network that can generate high-quality, smoothly contoured 3D shapes [5]. This result can bring us closer to producing 3D reconstructions with enhanced surface smoothness. Building upon these experiments, my project delves into the 3D Recurrent Reconstruction Neural Network (3D-R2N2) that integrates a Convolutional Neural Network (CNN), with either a Long Short-Term Memory Unit (LSTM) or Gated Recurrent Unit (GRU), and a Deconvolutional Neural Network (DCNN) [1].

My research aims to refine the 3D-R2N2 architecture to improve efficiency and computational resource

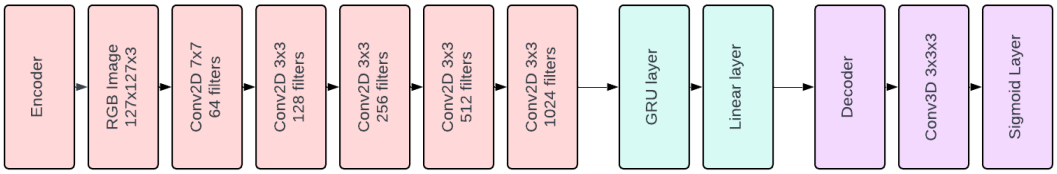

Figure 1. Neural Network Architecture. My model is built upon a 3D Recurrent Reconstruction Neural Network (3D-R2N2) [1]. The Encoder is the same architecture. The GRU layer is added instead of a 3D LSTM. The Decoder is shortened to only a singular 3D convolutional layer. And finally, instead of a softmax, a sigmoid function is used as the final output. Each convolutional layer from the encoder class is followed by a LeakyReLU and a 2x2 MaxPooling.

usage while preserving the quality of the generated shapes. The network's ability to not only recognize, but to accurately replicate complex shapes will hopefully advance the ongoing research of 3D reconstruction. The goal is to contribute to the field by creating a model that can be practical for real-world usage and still maintain a similar, or improved, 3D shapes.

3. Methodology

The hypothesis of this project centers on investigating whether a simplified architecture within a 3D Recurrent Reconstruction Neural Network (3D-R2N2) can effectively output a voxelized 3D shape reconstruction from 2D images. The aim is to retain the same results of a 3D shape reconstruction while reducing the architecture's complexity. To test this hypothesis, I used the ShapeNetCore dataset consisting of 48,600 3D models over 55 common categories with voxels used as the output 3D representations.

For preprocessing, to save computation time, I reduced the dataset and used 20,000 samples for the training set, 4,000 for validation set, and 1,000 for the test set. Each subset was processed through a PyTorch dataloader and were randomly shuffled to ensure the model's generalization. The image pixels are scaled from the range [0,255] to the range between [0,1] and resized to 127x127 pixels. I repeated the process for the train, validation, and test dataset to ensure all images are consistent in size. The resized images are converted from PIL format to PyTorch tensors. The images are then normalized using the mean and standard deviation values of [0.485, 0.456, 0.406] and [0.229, 0.224, 0.225], respectively, for each RGB channel. For voxel data, the MATLAB files are standardized to a size of 32x32x32 through adaptive max pooling 3D.

My model approach consists of a 3D Recurrent Reconstruction Neural Network (3D-R2N2) [1]. First, the encoder network is designed to down-sample spatial features from the input images. The network consists of multiple 2D convolutional layers followed by Leaky ReLU activation functions and max pooling 2D layers. Following the encoder, is the Gated Recurrent Unit (GRU) network which encodes the input features. It consists of a GRU layer function with an input size of the flattened encoder output, hidden size of 128, dropout of 0.5 to prevent overfitting, and a final linear layer. Finally, the decoder network is used for reconstructing the encoded features to a 3D volume. It consists of 3D convolutional layers followed by a sigmoid activation function. The decoder will utilize the output and transform it into a 3D voxel.

For training, I used an Adam optimizer with a learning rate of 0.001 and a weight decay of 1e-4. Weight decay is utilized to prevent weights from becoming too large. I used binary cross-entropy loss (BCELoss) for a binary classification task appropriate for voxel prediction. I implemented a multi-step learning rate scheduler to reduce the learning rate by a factor of 0.5 at a specific number of epochs. This will ensure learning rate is reduced during training leading to improved loss convergence and avoid overfitting. For regularization, gradient clipping is utilized to prevent potential exploding gradient. The model is evaluated using a valid loader and checkpoints are saved after each epoch. I used TensorBoard to verify visualization of 2D images while training the model as well as for logging training metrics. I am using PyTorch to build and train the model.

Once training is finished, I saved and loaded the model to generate predictions. For evaluation, each image in the dataset is transformed into a tensor and passed through the encoder, GRU, and decoder to generate voxel grid predictions. The voxel predictions are then used for visualization of 3D shapes. I added a threshold of 0.3 to set the values below 0.3 to be a

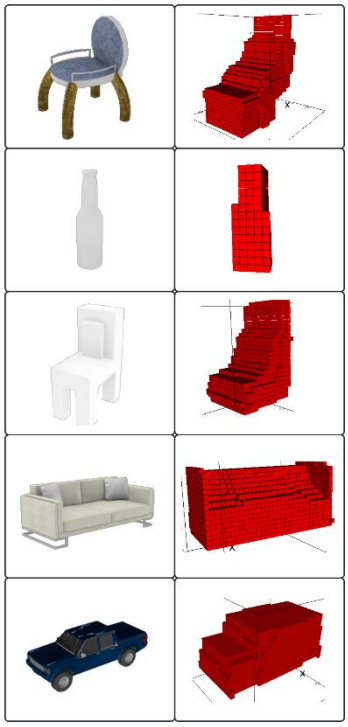

Figure 2. 3D Shape Reconstruction Actual Object vs Predicted.

value of 0 and above 0.3 to be a value of 1. This threshold will display the values as binary data and display a shape in a 3D scatter plot.

To evaluate the reconstructed voxels, I used the Intersection Over Union (IoU) evaluation code given by the ShapeNet challenge website. The IoU will compare the intersection of the predicted voxels and the ground truth voxels and result in a value. The higher the IoU value, the better the 3D reconstruction.

$$IoU = \frac{\Sigma_{i,j,k}\left[I\left(p_{(i,j,k)}>t\right)I\left(y_{(i,j,k)}\right)\right]}{\Sigma_{i,j,k}\left[I\left(I\left(p_{(i,j,k)}>t\right)+I\left(y_{(i,j,k)}\right)\right)\right]} \quad (1)$$

I(.) is an indicator function and t is the voxelization threshold. The numerator represents the intersection of the prediction and the ground truth. It takes the sum of pixels where the prediction and ground truth overlap are true. The denominator represents the union of prediction and the ground truth. It takes the sum of pixels where either the prediction or the ground truth are true, then subtracting the intersection. The IoU output will be between 0 and 1 where 1 indicates a perfect overlap and 0 indicates no overlap.

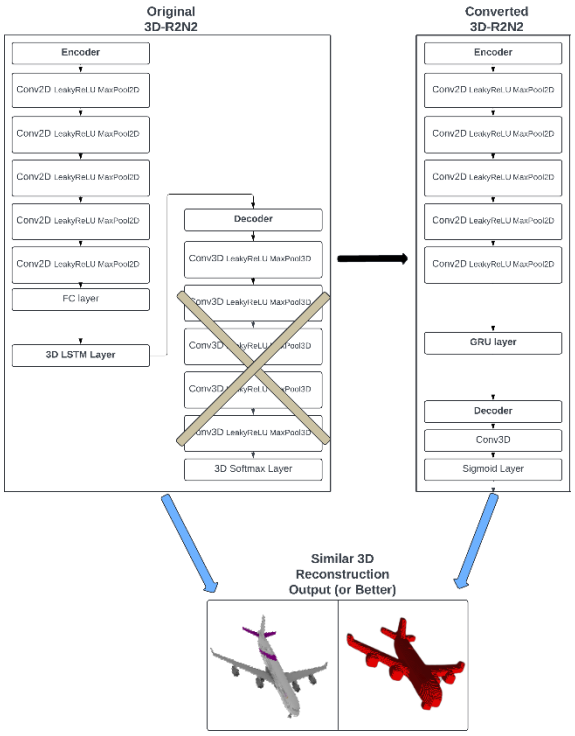

Figure 3. Illustration of the problem I want to solve. Using the 3D-R2N2 original network, I want to reduce its complexity while retaining the same 3D reconstruction prediction.

4. Results & Discussion

Training finished with a loss value of 0.178 and a validation loss of 0.204. It was run in 50 epoch and a batch size of 8. The training process was conducted using an NVIDIA GeForce RTX 4070 Ti graphics card. The model is generalizing well to unseen data based on the loss values. The Intersection over Union (IoU) score calculated to be 0.107. This is relatively low compared to a standard of 0.5 which is commonly considered a "good" score. This means that the overlap between predicted and ground truth is minimal. To consider the possibilities of a low IoU score, the model may have an issue with accuracy of the predictions. While the loss values show a reasonable model performance, the IoU score indicates that the model may have failed in predicting. I used a threshold of 0.3 to determine positive predictions. Figure 2 shows the actual image and the predicted 3D reconstruction. The model was able to predict some shape resembling the actual object, but it appears to be lacking detail. It seems to suggest the model was struggling predicting the finer details of the object.

Loss	Val Loss	IoU
0.178	0.204	0.107

Table 1. Model Reconstruction Performance

The purpose of this experiment was to decrease the complexity of the 3D Recurrent Reconstruction Neural Network (3D-R2N2) architecture while maintaining the same, or improved, level of predictions for 3D shape reconstruction. Despite having the latest GeForce RTX 40 series graphics card, one of the limitations I encountered was how extensive computational resources were required to train the model. It took 3.6 days for the model to finish training with 50 epoch and 2500 steps. The original architecture and using ShapeNet's total dataset would require a much more powerful GPU and computational memory. It would also require more days to run the training process. I implemented a shallow 3D-R2N2 network as opposed to a deep residual network [6], and for the purpose of this experiment, I decided to reduce the dataset to an acceptable level my resources can handle. Due to the size of the dataset, the performance of the model was affected as it generalized to unseen data subaverage. This proves that the performance of the model relies on a good quality of training data to be able to generalize well. Optimization has been seen to have slow convergence, to combat this, I included gradient clipping and a learning rate scheduler. These adjustments have improved the training process a small amount, but not significantly enough as I hoped in terms of my goal.

As shown in Figure 3, the problem I wanted to solve was reducing the 3D-R2N2 network complexity. Additionally, I also changed the activation function and loss function in order to improve optimization. My aim was to convert the 3D-R2N2 architecture and retain the same 3D shape reconstruction output as the original 3D-R2N2 or, better yet, improve it. Doing so will also reduce computation resources and improve efficiency. Due to time constraints, the model resulted in a lower performance than what I wanted to achieve. This is evident through the IoU score as it is lower than other experiments [1]. The results learned here will serve as future reference for 3D reconstruction.

5. Conclusion

The trained model displays good loss value in terms of generalizing to unseen data. However, the low IoU score indicates a problem with predictions from the model and will need more improvement. As shown from Figure 2 results, the model, although displays a 3D shape, lacks predicting finer details of the object. Possible ways to enhance the model would have to be addressed with further research and experiments.

To further improve the model, in the future, I would consider using the entire ShapeNet dataset to fully implement it into the training process. With this, I anticipate a better loss value, IoU score, and 3D shape reconstruction prediction result. I believe with a higher epoch duration more than 50 epoch; it may lower loss values. Reviewing the loss from the training process, it continued to converge with each epoch. Additionally, I want to further reduce the model's complexity by removing layers from the architecture for better computation. For analysis, I want to implement and compare both LSTM and GRU within the model. This will help to understand how each implementation will affect 3D reconstruction results and computation time. This comparison could give insight of the trade-offs between model complexity and 3D shape reconstruction. I would also like to consider adding a dataset with more synthetic images. This will diversify the training data and improve the model's robustness to unseen, new data.

These changes aim to improve the model's performance to 3D shape reconstruction while maintaining the same goal of reducing complexity and computation usage. I hope to advance the research so far in 3D reconstruction, so it does effectively well in real-time applications. It would be innovative to see 3D reconstruction implemented in many scenarios.

GitHub Repository

References

[1] Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. “3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction.” arXiv:1604.00449, 2016. 1,2,4

[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, Fisher Yu. “Shapenet: an information-rich 3d model repository,” arXiv:1512.03012, 2015. 1

[3] Haoqiang Fan, Hao Su, Leonidas Guibas. “APoint Set Generation Network for 3D Object Reconstruction from a Single Image.” arXiv:1612.00603, 2016. 1

[4] Dmitry Tochilkin, David Pankratz, Zexiang Lu, Zixuan Huang, Adam Letts, Yangguang Li, Ding Liang, Christian Laforte, Varun Jampani, Yan-Pei Cao. “TripoSR: Fast 3D Object Reconstruction from a Single Image.” arXiv:2403.02151, 2024. 1

[5] Ruikai Cui, Weizhe Liu, Weixuan Sun, Senbo Wang, Taizhang Shng, Yang Li, Han Yan, Zhennan Wu, Shenzhou Chen, Hongdong Li, Pan Ji. “NeuSDFusion: A Spatial-Aware Generative Model for 3D Shape Completion, Reconstruction, and Generation.” arXiv:2403.18241, 2024. 1

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. “Deep Residual Learning for Image Recognition.” arXiv:1512.03385, 2015. 4