The Fiat Lux Journal 2024

DYNAMIC CONVOLUTIONAL NETWORKS FOR 3-DIMENSIONAL RECONSTRUCTION

Dongchen Han

Phillips Academy Andover

January 22, 2024

Abstract

Modern convolutional neural networks (CNNs) require applying duplicate and redundant operations (such as alignment and matching) on different regions and pixels when processing three-dimensional (3D) reconstruction tasks. But different image areas and pixels are certainly not equally valuable for 3D reconstruction. In order to solve this problem, we propose a dynamic CNN structure for 3D reconstruction, which can dynamically modify the network based on the estimated impact of different regions and pixels on the 3D reconstruction task. The small gating branch learns which important areas or pixels need to be evaluated. The discrete gating decisions are trained with the Gumbel-Softmax trick, combined with a series of spatial and scale criteria. Our experiments on ShapeNet dataset shows that our method has higher accuracy than existing methods due to the better focus on important regions. Moreover, with an efficient CUDA implementation, our method achieves an improved inference speed on the most famous 3D reconstruction model Mesh R-CNN.

Keywords 3D Reconstruction, Dynamic Convolutional Neural Network

1 Introduction

With the development of virtual reality and deep learning, 3D reconstruction technology has been widely studied. It is worth noting that more and more preliminary research works began to use convolutional neural networks (CNNs) to reconstruct 3D shapes based on input RGB images. Although some progress has been made, these methods have a common significant shortcoming. In these methods, the same and redundant matching and alignment operations are performed for all regions and pixels. Obviously, these violent matching and alignment operations are unreasonable for 3D reconstruction tasks for three reasons. First, performing the same operation on different regions and pixels makes the existing 3D reconstruction system unable to effectively use different 3D reconstruction granularities in different image regions according to the characteristics of the object. Second, uniform matching and alignment operations also increase parameter redundancy and limit the performance of existing 3D reconstruction models on different scenes and objects in the natural world. Finally, redundant parameters can also significantly increase the training and test time. To solve these problems, we adopt an improved 3D reconstruction sub-network with a dynamic residual module, which can be trained end-to-end without additional space and scale constraints. Compared with the conventional CNN that performs the same convolution operation on all regions and pixels, our dynamic CNN can perform more convolution operations on image regions and pixels with more details. We borrowed design ideas of dynamic convolution [1] and residual convolution networks [2]. A dynamic residual module consists of several dynamic residual blocks. In every block, a small gating unit is used to select the regions that need to be focused on. The gating units are designed with the Gumbel-Softmax [3] trick to enable end-to-end training. Those decisions progress can extract features from important regions in the image, which enables the network to utilize higher-level information and process the regions of interest (ROIs) only. In addition, to improve the execution efficiency of the existing 3D reconstruction model, we implement the dynamic 3D reconstruction sub-network based on the CUDA platform of NVIDIA graphics cards. The core advantage of our approach is that we reduce as many CUDA modifications as possi-

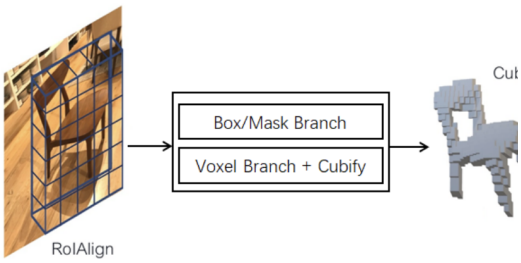

Figure 1: The overall structure of Mesh R-CNN. The voxel branch generates the coarse shape of each detected object. In the mesh refinement branch, several refinement stages can improve the details. We optimize the 3D reconstruction sub-network (i.e., mesh refinement branch) of Mesh R-CNN with the dynamic residual module.

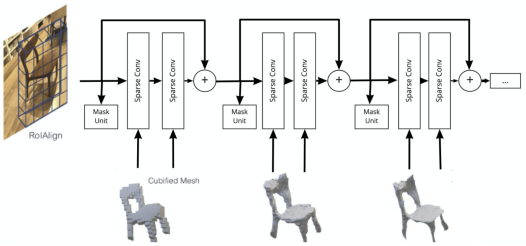

Figure 2: Our improved 3D reconstruction sub-network. Specifically, it adds multiple dynamic residual blocks to ROI Align to learn the target object's three-dimensional shape. Here we only list three dynamic residual blocks due to limited space. We use a mask unit and several spatial convolutions to learn the three-dimensional shape for each block.

ble. On the other hand, we optimized the Tensor data structure in memory. Compared to conventional CNNs, our method can perform convolution operations more accurately and sparsely. The main contributions of our paper are listed as follows: • We propose a dynamic 3D reconstruction sub-network to perform more precise and detailed convolution operations on image regions and pixels that need to be focused on in 3D reconstruction. We adopt the design idea of the Gumbel-Softmax trick to construct a dynamic residual module with gating masks. • The dynamic residual module can be integrated into existing 3D reconstruction networks based on convolutional layers. In the experiment, we combined our method into the most advanced Mesh R-CNN method. The results show that the proposed dynamic residual module can effectively improve the accuracy of Mesh R-CNN and accelerate the training and testing time of the entire network. • We implement the residual module on GPU with CUDA. It can not only theoretically reduce the number of floating-point operations but also practically accelerate the training and testing of Mesh R-CNN.

2 Methods

2.1 Tools

In this paper, we propose a dynamic 3D reconstruction sub-network based on dynamic residual modules, which can perform precise and detailed convolution in neural networks for 3D reconstruction. To show the effectiveness of the proposed method, we integrate it into the Mesh R-CNN system [26]. Figure 1 shows the system overview of Mesh R-CNN, where a 3D reconstruction sub-network is adopted in the mesh refinement branch. The original Mesh R-CNN has a common significant shortcoming, especially when facing redundant matching and alignment operations performed for all regions. The violent matching and alignment operations are inefficient in 3D reconstruction tasks. Performing redundant operations across different regions will be ineffective for applying different granularities on the final reconstructed object. This violent alignment operation will also increase parameter redundancy, causing difficulty processing other natural world scenes and increasing training time. To solve this problem, we optimize the 3D reconstruction sub-network of Mesh R-CNN with a dynamic residual module. With the dynamic processes in different regions, it can perform convolution operations with more details. In this section, we describe the design of the dynamic 3D reconstruction sub-network, which is based on dynamic residual modules.

2.2 Related Work

Our research aims to optimize the convolution operation in the existing 3D reconstruction model. Recently, more and more methods have begun to use CNNs for single-image 3D reconstruction. In general, these methods can be divided into three categories as follows. (1) Some approaches can predict the orientation of the shape [4, 5], and others can construct a 3D pose model based on the existing shape [6, 7, 8]; (2) Other methods predict 3D shapes based on point clouds [9, 10], patches [11, 12], or geometric primitives [13, 14, 15]; (3) Others use CNNs to learn distance functions [16]. Although these methods can perform 3D shape reconstruction well, they rely too much on data preprocessing and data post-processing steps, and these methods generally lack scalability. In addition, there

are some studies that mainly focus on multi-view reconstruction. Various methods are designed for it, including classical binocular stereo [17, 18], shape priors [19, 20, 21, 22], and modern learning techniques [23, 24, 25]. Our improved 3D reconstruction sub-network is shown in Figure 2. Specifically, our method adds multiple dynamic residual blocks to ROI Align to learn the target object's three-dimensional shape. Here we only list three dynamic residual blocks due to limited space. In each block, we use a mask unit and several spatial convolution layers to learn the three-dimensional shape. The spatial positions to be processed are indicated by the pixel-wise masks. For every dynamic residual block, we use the Gumbel-Softmax trick to enable the end-to-end training of discrete decisions, which can achieve better performance than REINFORCE [27] with less complexity. In the following, we will introduce in detail the dynamic residual block. Specifically, the structure of the dynamic residual block is shown in Figure 4. We denote the input of a block b (i.e., Cubified Mesh) as $X_b \in R^{c_b \times w_b \times h_b}$ and its output as $X_{b+1} \in R^{c_{b+1} \times w_{b+1} \times h_{b+1}}$. The dynamic residual block is formulated as:

$$X_{b+1} = r(F(X_b) + X_b)$$

where F is the dynamic function and r is the activation function. We make the values of F conditional on X_b by adopting a mask unit $M(X_b)$, which outputs soft gating decisions. Furthermore, we use the Gumbel-Softmax unit G to turn soft decisions $M_b \in R^{w_{b+1} \times h_{b+1}}$ into hard decisions $G_b \in R^{w_{b+1} \times h_{b+1}}$, indicating which position in the residual block should be evaluated. The Gumbel-Softmax unit can be described by:

$$G_b = G(M(X_b)).$$

Thus, the dynamic function F can be defined as:

$$F(X_b) = f(X_b) \circ G_b$$

where f is the spatial convolution function, which typically consists of two or three convolution layers with batch normalization (BN) [28]. The operation $\circ$ is the element-wise multiplication over the spatial dimensions broadcasted to all channels. Thus, a dynamic residual block can be represented by:

$$X_{b+1} = r(f(X_b) \circ G_b + X_b)$$

Figure 3 presents an example of the dynamic function F on a 6 × 5 × 1 feature map. The Gumbel-Softmax unit G selects the entries to be passed to the next dynamic residual block. The non-zero entries in $F(X_b)$ are significantly less than $F(X_b)$, which inspires us to accelerate the computation utilizing the sparsity of dynamic residual blocks.

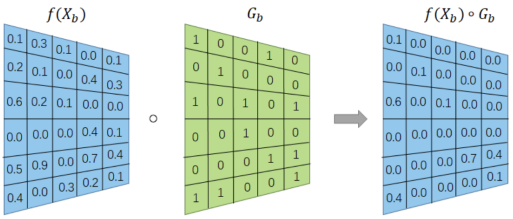

Figure 3: An example of the dynamic function $F(X_b) = f(X_b) \circ G_b$ The sparsity of $F(X_b$ is obviously higher than $F(X_b$, which can be utilized to reduce the amount of computation.

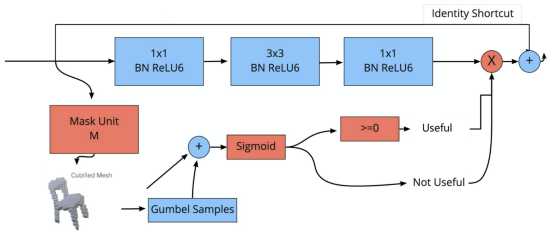

Figure 4: The execution flow of a mask unit with the Gumbel-Softmax trick. Firstly, a floating-point mask is generated. Then the soft decisions are converted into hard decisions by the Gumbel-Softmax trick, which enables the back-propagation for end-to-end learning.

2.3 Efficient inference implementation

Merely masking spatial locations in a convolutional block, as described above, does not lead to any speedup. During inference, our method needs evaluation on the active spatial positions only, indicated by Gb. Efficiently executing sparse operations is a challenging task. The overlap of convolutional kernels causes a single tensor element to be accessed several times, requiring advanced caching strategies. For instance, fast implementations of standard 3 × 3 convolutions use the Winograd algorithm [29], which processes image patches of typically 4 × 4 pixels. Conditionally executing individual pixels is not trivial in this case, and we leave the possibility to conditionally execute individual Winograd patches for future work. We propose a method to efficiently execute spatially sparse convolutions with minimal changes to existing operators. Our method executes individual operations conditionally by copying selected elements to an intermediate, dense tensor. For pointwise convolutions and activation functions, we can apply corresponding implementations on the intermediate tensor. Finally, the processed result is copied back to its original position.

3 Discussion

3.1 Dataset

ShapeNet [30] is a widely-used benchmark providing various 3D shapes, which are represented as textured CAD models and organized into semantic categories following WordNet [31]. We use a subset of ShapeNetCore.v1 to train and evaluate our method, like [26]. The meshes are rendered from at most 24 randomly chosen viewpoints. The RGB images are of size 137 × 137. The training and testing sets consist of 35,011 models (840,189 images) and 8,757 models (210,051 images), respectively. To evaluate the model, we input a single RGB image of a rendered ShapeNet model on a blank background and obtain the output, which is a 3D mesh of the object. The neural network is trained with pairs of images and meshes.

4 Discussion

4.1 Evaluation

We adopt evaluation metrics used in recent work [32, 33, 34, 26]. We compute Chamfer distance and $F1^{\tau}$ at various distance thresholds by sampling 10k points uniformly from the surface of predicted and ground-truth meshes. $F1^{\tau}$ is the harmonic mean of the precision and recall at *tau*. For Chamfer distance, lower is better. For $F1^{\tau}$, higher is better.

4.2 Analysis of Results

The results of our comparison with other state-of-the-art methods are shown in Table 1. For the sake of fairness, we comprehensively evaluated three evaluation measures. All the experimental results in the table are from the original paper. Because some methods do not share their source code, we cannot reproduce and make complete comparisons. In general, our method has achieved better results than other methods.

	Chamfer (↓)	$F1^{\tau}$ (↑)	$F1^{2\tau}$ (↑)
N3MR [35]	2.629	33.80	47.72
3D-R2N2 [36]	1.445	39.01	54.62
PSG [9]	0.593	48.58	69.78
Pixel2Mesh [34]	0.591	59.72	74.19
MVD [32]	-	66.39	-
GEOMetrics [33]	-	67.37	-
Mesh R-CNN [26]	**0.306**	**74.84**	**85.75**
Ours (Best)	**0.310**	**77.82**	**87.17**

Table 1: The comparison of single-image shape reconstruction on ShapeNet.

	Training time	Testing time
Mesh R-CNN	1.00	1.00
Ours (3 residuals)	0.82	0.93
Ours (4 residuals)	0.87	0.95
Ours (5 residuals)	0.92	0.97
Ours (6 residuals)	0.97	0.98

Table 2: The training and testing time of the original Mesh R-CNN and our method.

In careful comparison with the Mesh R-CNN method, our method has achieved better performance. In the experiment, the performance of our method will change with the number of dynamic residual blocks. Here we only list the best result of our method, which uses three dynamic residual blocks. In addition, we also compared the impact of our method on the training time and testing time of the original Mesh R-CNN method. The experimental results are shown in Table 2. Our method can optimize the testing and training time of Mesh R-CNN. Because of limited resources, we only test up to 6 residual modules. Our method achieves the best performance with three residual modules.

4.3 Examples of Reconstruction Results

We more intuitively compare our method and the original Mesh R-CNN on 3D reconstruction. We show part of the 3D reconstruction results in Figure 5. The last two columns of each result are the 3D triangle meshes generated by our method and Mesh R-CNN. From Figure 5, we can see that our method has achieved better results. Compared with the Mesh R-CNN method, our method can pay more attention to the detailed information in the 3D reconstruction details. Moreover, get a more accurate expression of the 3D triangle.

5 Conclusion

In our research project, we carefully survey the previous works on 3D construction. The state-of-the-art 3D construction models have similar problems: conventional CNNs can only operate the same convolution operations on all pixels of images and ignoring the detail of 3D shapes. Based on this observation, we propose a dynamic

Figure 5: We more intuitively compare our method and the original Mesh R-CNN on 3D reconstruction. The last two columns of each result are the 3D triangle meshes generated by our method and Mesh R-CNN.

3D reconstruction sub-network to improve the original frameworks of Mesh R-CNN. In the paper, we briefly illustrate our proposed method and analyze the experimental results. The proposed method outperforms Mesh R-CNN by a large margin with lower training and testing time.

References

[1] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In CVPR, pages 2317–2326, 2020.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In CVPR, 2016.

[3] Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparameterization with Gumbel-Softmax. 2016.

[4] David F. Fouhey, Abhinav Gupta, and Martial Hebert. Data-driven 3D primitives for single image understanding. In ICCV, 2013.

[5] Derek Hoiem, Alexei A. Efros, and Martial Hebert. Geometric context from a single image. In ICCV, 2005.

[6] Abhijit Kundu, Yin Li, and James M. Rehg. 3D-RCNN: Instance-level 3d object reconstruction via render-andcompare. In CVPR, 2018.

[7] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G. Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In ICRA, 2017.

[8] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In CVPR, 2015.

[9] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. In CVPR, 2017.

[10] Chen-Hsuan Lin, Chen Kong, and Simon Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. In AAAI, 2018.

[11] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In CVPR, 2018.

[12] Peng-Shuai Wang, Chun-Yu Sun, Yang Liu, and Xin Tong. Adaptive O-CNN: a patch-based deep representation of 3d shapes. In SIGGRAPH Asia, 2018.

[13] Sanja Fidler, Sven Dickinson, and Raquel Urtasun. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In NeurIPS, 2012.

[14] Yonglong Tian, Andrew Luo, Xingyuan Sun, Kevin Ellis, William T. Freeman, Joshua B. Tenenbaum, and Jiajun Wu. Learning to infer and execute 3d shape programs. In ICLR, 2019.

[15] Shubham Tulsiani, Hao Su, Leonidas J. Guibas, Alexei A. Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In CVPR, 2017.

[16] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In CVPR, 2019.

[17] Richard Hartley and Andrew Zisserman. Multiple view geometry in computer vision. Cambridge university press, 2003.

[18] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. IJCV, 2002.

[19] Sid Yingze Bao, Manmohan Chandraker, Yuanqing Lin, and Silvio Savarese. Dense object reconstruction with semantic priors. In CVPR, 2013.

[20] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In SIGGRAPH, 1999.

[21] Amaury Dame, Victor A. Prisacariu, Carl Y. Ren, and Ian Reid. Dense reconstruction using 3d object shape priors. In CVPR, 2013. [22] Christian Häne, Nikolay Savinov, and Marc Pollefeys. Class specific 3d object shape priors using surface normals. In CVPR, 2014.

[23] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In NeurIPS, 2017.

[24] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In ICCV, 2017.

[25] Tanner Schmidt, Richard Newcombe, and Dieter Fox. Self-supervised visual descriptor learning for dense correspondence. In IEEE Robotics and Automation Letters, 2017.

[26] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh R-CNN. In ICCV, 2019.

[27] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3-4):229–256, 1992.

[28] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift.

2015.
[29] Andrew Lavin and Scott Gray. Fast algorithms for convolutional neural networks. In CVPR, pages 4013–4021, 2016.
[30] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An informationrich 3d model repository. In CoRR 1512.03012, 2015.
[31] George A. Miller. WordNet: A Lexical Database for English. In Commun. ACM, 1995.
[32] Edward Smith, Scott Fujimoto, and David Meger. Multi-view silhouette and depth decomposition for high resolution 3d object representation. In NeurIPS, 2018.
[33] Edward J. Smith, Scott Fujimoto, Adriana Romero, and David Meger. GEOMetrics: Exploiting geometric structure for graph-encoded objects. In ICML, 2019.
[34] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2Mesh: Generating 3D mesh models from single RGB images. In ECCV, 2018.
[35] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3D mesh renderer. In CVPR, 2018.
[36] 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. In ECCV, 2016.