TMVNet : Using Transformers for Multi-view Voxel-based 3D Reconstruction

Kebin Peng, Rifatul Islam, John Quarles, Kevin Desai
The University of Texas at San Antonio, TX 78249
{kebin.peng, rifatul.islam, john.quarles, kevin.desai}@utsa.edu

Abstract

Previous research in multi-view 3D reconstruction have used different convolution neural network (CNN) architectures to obtain a 3D voxel representation. Even though CNN works well, they have limitations in exploiting the long-range dependencies in sequence transduction tasks such as multi-view 3D reconstruction. In this paper, we propose TMVNet – a two-layer transformer encoder that can better use long-range dependencies information. In contrast to using a 2D CNN decoder by the previous approaches, our model uses a 3D CNN encoder to capture the relations between the voxels in the 3D space. Also, our proposed 3D feature fusion network aggregates 3D position feature from CNN and long-range dependencies feature from transformer together. The proposed TMVNet is trained and tested on the ShapeNet dataset. Comparison against ten state-of-the-art multi-view 3D reconstruction methods and the reported quantitative and qualitative results showcase the superiority of our method.

1. Introduction

3D reconstruction from single or multiple views is an ill-posed problem, making it a very challenging research problem. Structure from Motion (SfM) [18] is one of the classical approaches for 3D reconstruction, which requires capturing the subject from multiple views and then processing them with reconstruction algorithms [8]. However, extracting 2D-feature points is time-consuming and results in sparse reconstruction. In addition, establishing correspondence between feature points in multi-view reconstruction is more complicated when the view is separated by a large distance.

Inspired by the limitations of the prior approaches, researchers have proposed deep-learning based 3D reconstruction techniques [14, 19, 34], which can be classified into three categories: : 1) point-cloud based [19], 2) mesh based [14], and 3) voxel based [34]. The point-cloud approach outputs a series of points in 3D space, which describes the object with no connection between any of the points. In contrast to the point-cloud-based method, the mesh-based technique modeled the relationship between individual points in the point cloud. In voxel-based 3D reconstruction, a volume for the object is created and divided into small boxes. Each box can either be occupied or empty. If the box is occupied, it will be rendered as a pixel [34].

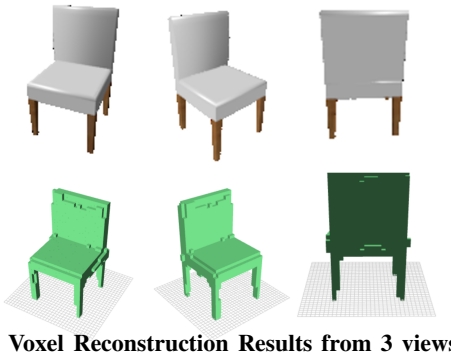

Figure 1. **Voxel Reconstruction Results from 3 views in 32^3 resolution.** An example from the ShapeNet dataset [2]. Our model estimates accurate voxel grids and shows more details.

Recently, researchers used a transformer-based encoder-decoder for 3D reconstruction, which outperformed prior approaches. Zhao et al. [39] introduced a method using a transformer architecture for 3D point cloud processing, which proposed a point transformer layer that applies self-attention in the local neighborhood of 3D points. Guo et al. [10] presented a transformer framework for point cloud learning which included a coordinate-based position encoder and an offset attention module that used neighbor embedding. In this paper, we use transformers for encoding the long-range dependencies between CNN features. Below, we provide the motivations behind the use of transformers.

1.1. Motivations for using Transformer

Although the previous works demonstrate satisfactory accuracy in 3D reconstruction, few of them discuss the long-range dependencies [29] in multi-view 3D reconstruction. The definition of long-range dependencies is: in a sequence signal $(x_1, ...x_i...x_j...x_n)$, x_j has relation not only with x_{j-1} but also with x_i, where $i < j$. Long-range dependencies also exist in multi-view 3D reconstruction. For

example, the first input image has overlapping parts with the last input image. A pixel in (i, j) in the m-th input image may move to (x, y) in the n-th input image. Such phenomena explain why RNN+CNN could perform better than pure CNN [4] in multi-view 3D reconstruction. However, the RNN layer has high computational complexity compared with self-attention in transformer [29]. This reason motivates us to apply transformer in 3D reconstruction.

Another benefit is that the transformer could also learn object's 3D position feature from image [21]. Such a 3D position feature could improve model accuracy. Meanwhile, CNN layers learn object surface detail features from image. Having two different features require them to be fused together to reconstruct the final 3D object. However, CNN features and transformer features may have different distributions. In our paper, we use 3D convolution decoder to upsample the features from the transformer, and later use 2D convolution to adapt the different distributions and aggregate them.

1.2. Contributions

The major contributions of this paper are as follows:

- A novel two-layer transformer neural network is proposed for voxel-based reconstruction from single or multiple views.
- We propose the 3D Feature Fusion Network to refine voxel reconstruction results.
- Our model uses fewer convolutional layers and only two transformer layers but preforms better than other voxel based reconstruction methods.

2. Related Work

In this section, we discuss relevant previous works. First, we discuss recent approaches that performing voxel 3D reconstruction. Then we discuss the works that use transformer and self-attention to perform object reconstruction.

Voxel-based 3D Reconstruction Early deep learning based 3D reconstruction techniques primarily used voxel representations of the subject to produce a 3D model. This approach allows represent a 3D shapes as voxel grids, which can be easily represented in binary form. A voxel is set to zero if it is not included in the object and vice versa. [5] presents an algorithm to reconstruct complex geometric models by turning the object into three voxel spaces where each voxel in the voxel space is encoded as a 2D texture. However, this algorithm required pre-processing before reconstruction. To solve this issue, [7] proposes a binary hierarchical voxel representation using a binary octree that does not rely on any pre-processing and produces a finished representation of the subject without holes.

Voxel grids are the most common form in deep learning based reconstruction methods. [4] proposes a recurrent neural network architecture that uses CNN layers as well as an LSTM. It takes one or more images of an object from different angles as the input and outputs a 3D occupancy grid as the reconstructed model. [13] proposes an image-to-semantic voxel model using a generator and discriminator to generate a voxel reconstruction, semantic segmentation, and object poses as the combined output.

Though these methods perform well, they are expensive in terms of memory requirements. To reduce the memory consumption, researchers aimed to lower the resolution of 3D volume reconstruction. [33] proposes a convolutional deep belief network that learns the distribution of complex 3D shapes and hierarchical representations between different object categories. To reconstruct an object from a single given image, [32] adds the MarrNet, which is an end-to-end deep learning model that can reconstruct 3D objects given an estimated 2.5D sketch with two-step disentangled formulation. The first step uses an encoder-decoder neural network to create 2.5D sketches, which are used as input for a second encoder-decoder model to create a 3D object. [24] used this two-step approach to create Pix3D, which is a multi-task learning approach to perform reconstruction and pose estimation from a single image.

Inspired by [33], [28] proposes the view consistency network by using differentiable ray consistency (DRC). They incorporate DRC into deep learning frameworks to regress the voxel grids. With DRC, deep learning frameworks can leverage different types of observations of a subject, such as foreground masks, depth, color, and semantics. [27] outlines a method of multi-view consistency to regress voxel grids. This method enforces the predicted voxels consistent with each other in the input image using their depth value. [36] proposes a unified framework that can use different types of data, like pose-annotated images and unlabeled images to perform regression. [16] introduces the Variational Shape Learner (VSL) with skip-connections which performs voxel regression by learning the underlying structure of 3D shapes in an unsupervised manner. This approach encodes 2D features into a latent variable, which is decoded into voxel grids.

Transformers for 3D Reconstruction: Transformer and self-attention are popular in natural language processing (NLP) and perform well in various NLP tasks, such as machine translation [29]. This performance attracted the interest of the computer vision community. Many researchers attempted to apply the transformer and self-attention to various computer vision tasks like image recognition, object detection, and 3D reconstruction. However, NLP tasks and the structure of NLP data are different from image and vision, which means transformer and self-attention is not always suitable for image-based workloads. As a result, many novel networks tried to make transformer and self-attention available to 3D reconstruction. [39] outlines self-attention

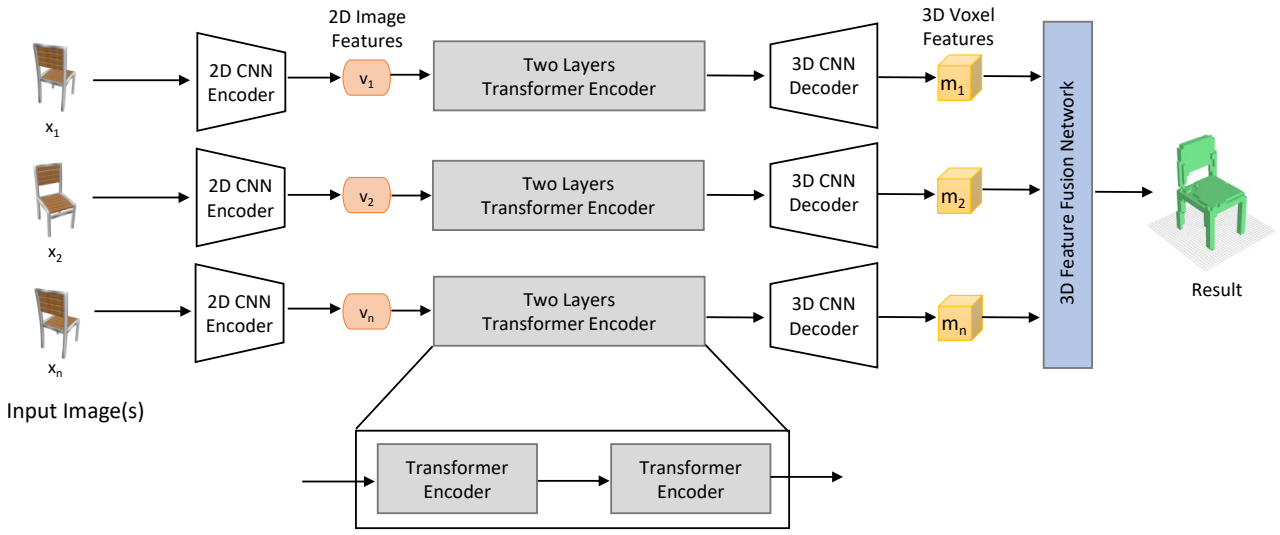

Figure 2. **Voxel Reconstruction Network Architecture Overview:** The model first extracts 2D features from a sequence of images using a residual CNN encoder. Those features are then passed through two transformer encoder layers with independent input dimensions. The obtained 3D feature vectors are then used by a 3D CNN decoder to obtain the 3D voxel features. Finally, a feature fusion layer is used to fuse the decoded 3D voxel features to obtain the final 3D reconstruction of the object.

layers to analyze point-clouds and uses self-attention layers to construct a point transformer. The point transformer models the relationship of the local neighbors of a point and encodes the 3D position information into a feature vector for the self-attention layers. However, this point cloud data is still highly irregular and lacks any ordering information. To solve this, [10] proposes a point cloud transformer that uses the order invariance of the transformer to define the order of points in the point cloud. The paper also proposes offset-attention with a Laplacian operator to refine the order of points. [38] introduces an iterative transformer network (IT-Net) that iteratively learns the 3D point shape and semantic segmentation of an object. [31] introduces a global voxel transformer network based on U-Net and built on Global Voxel Transformer Operators (GVTOs). GVTOs enable voxel transformers to aggregate the global information of an object while also retaining local information.

In our approach, we draw inspiration from [4,15,31] and propose a multi-layer transformer encoder, which takes image features as input from a CNN encoder to estimate 3D voxel representation from one or many images. The multi-layer transformer encoder can more accurately model the space position information for each voxel grid compared to the LSTM approach proposed in [4].

3. TMVNet Architecture

In this section, we discuss the network architecture. The overview of the model is illustrated in Figure 2. The proposed *TMVNet* consists of four separate layers – a 2D-CNN layer, a two-layered transformer Encoder layer, a 3D-CNN decoder, and a 3D feature fusion layer. It should be noted that there is only one each of the 2D CNN Encoder, the two-layer transformer encoder, and the 3D CNN Decoder. The figure shows them multiple times to denote that our proposed approach can work either take as input a single view or multiple views as a sequence. The following subsections detail the network architecture.

3.1. 2D-CNN Encoder Layer

The first part of our model utilizes the residual CNN in [4] as an encoder to extract 2D image features for each input view. We do not pre-train our CNN encoder on datasets, such as ImageNet [23], because the object shapes are not consistent between ImageNet and our training dataset, which could negatively impact model accuracy.

3.2. Transformer Encoder Layer

The extracted features from the 2D-CNN are forwarded to the two-layered transformer encoder layer. While existing transformer encoder architectures set constant dimensions for all transformer layers [1, 6], this is not suitable for voxel reconstruction. To overcome this limitation, we use linear projections in our transformer encoder to dynamically adjust the dimension of each output, performing gradual dimension reduction with several blocks, which was inspired by [11,15]. Because the output dimensions of a CNN encoder are quite small when given input images with a small size (i.e., 127×127 image), we set a small size square (3×3). The final output vectors of our transformer encoder are 3D feature vectors, which contain the 3D information required for a voxel as shown in Figure 2. To clarify, the

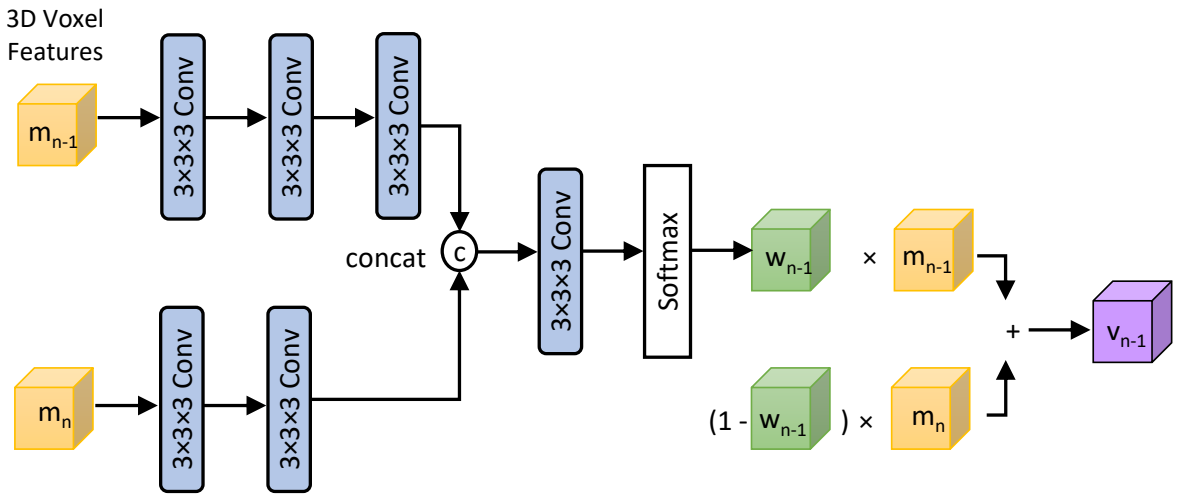

Figure 3. **3D Feature Fusion Network Overview:** The model first extracts 2D image features using the triple-layered CNN encoder. Those features are passed through two transformer encoder layers with independent input dimensions. Finally, the output of the second transformer is used to model the object.

extracted 2D features from each input view are processed by the two-layer transformer encoder one by one as a sequence. Hence, there is only a single two-layer transformer encoder, not one each for each view.

3.3. 3D-CNN Decoder Layer

The next layer of our model is the 3D CNN decoder. It takes as input the 3D feature vectors from the transformer decoder and performs 3D convolution to produce a $32\times32\times32$ vector. To save the computation time and memory usage, we propose a simple decoder network with four layers of $3 \times 3 \times 3$ convolutions (each layer follows a 3D relu layer). Different from [4, 34], our decoder is the 3D convolution and has no residual branches, which makes a better trade-off between accuracy and GPU memory usage.

3.4. Feature Fusion Layer

To fuse each 3D voxel feature from the 3D-CNN decoder, we propose the feature fusion layer. Figure 3 shows the overview of the fusion layer. The fusion layer takes as input the 3D voxel features from the 3D-CNN decoder. Assuming there are n 3D voxel features, namely, $m_1, m_2, \cdots m_n$. The 3D Feature Fusion Network uses m_{i-1} as input of the first branch that has three 3D convolution layers and uses m_i as input of the second branch that has two 3D convolution layers. After concatenation, the output from each branch in the channel direction, the concatenation output follows a 3D convolution layer and is normalized by a softmax layer, which gives a learned weight w_{i-1}. With w_{i-1}, m_i and m_{i-1} are fused as follows:

$$v_i = w_{i-1} \times m_{i-1} + (1 - w_{i-1}) \times m_i \tag{1}$$

Then v_i repeats the fusion steps with m_{i+1} until the last 3D voxel feature m_n. The final outputs are denoted as v_n. It passes through a softmax layer to normalize it to a range of $(0, 1)$, which is denoted as $p_{i,j,k}$. $p_{i,j,k}$ indicates the probability of the voxel being occupied at position (i, j, k).

3.5. Loss Function

The loss function is defined using the sum of voxel-wise cross-entropy (Eq. 2) [4]. The x denotes the input image (or images), $p_{i,j,k}$ represents the probability that the voxel at position (i, j, k) is occupied in the final reconstruction. The corresponding voxel in the ground truth is defined as $y(i, j, k) \in (0, 1)$ and the sum of our voxel-wise cross-entropy loss function is defined as follows:

$$L(x, y) = \sum_{i,j,k} y_{i,j,k} \log p_{i,j,k} + (1 - y_{i,j,k}) \log (1 - p_{i,j,k}) \tag{2}$$

4. Experiments

In this section, discuss the experiment setup, implementation details and the datasets.

4.1. Implementation Details

Our model is built on the PyTorch framework [20], and trained on the ShapeNet dataset, as explained below in Section 4.2. The training is performed on a NVIDIA RTX 3080 GPU for 60 epochs with a batch size of 4. We used the Adam optimizer [12] with the settings $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We utilize an adaptive learning rate, starting at 0.0001 and decreasing it by half every ten epochs after the 20^{th} epoch.

Category	3D-R2N2	OGN	AtlasNet	Pixel2Mesh	OccNet	IM-Net	AttSets	Pix2Vox++/F	Pix2Vox++/A	Mem3D	Ours
airplane	0.513	0.587	0.493	0.508	0.532	0.702	0.594	0.607	0.674	**0.767**	0.691
bench	0.421	0.481	0.431	0.379	0.597	0.564	0.552	0.544	0.608	0.651	**0.659**
cabinet	0.716	0.729	0.257	0.732	0.674	0.680	0.783	0.782	0.799	0.840	**0.853**
car	0.798	0.828	0.282	0.670	0.671	0.756	0.844	0.841	0.858	**0.877**	0.870
chair	0.466	0.483	0.328	0.484	0.583	0.644	0.559	0.548	0.581	0.712	**0.721**
display	0.468	0.502	0.457	0.582	0.651	0.585	0.565	0.529	0.548	**0.631**	0.595
lamp	0.381	0.398	0.261	0.399	0.474	0.433	0.445	0.448	0.457	**0.535**	0.534
speaker	0.662	0.637	0.296	0.672	0.655	0.683	0.721	0.721	0.721	**0.778**	0.712
rifle	0.544	0.593	0.573	0.468	0.656	0.723	0.601	0.594	0.617	0.746	**0.783**
sofa	0.628	0.646	0.354	0.622	0.669	0.694	0.703	0.696	0.725	**0.753**	0.701
table	0.513	0.536	0.301	0.536	0.659	0.621	0.590	0.609	0.620	**0.685**	0.660
telephone	0.661	0.702	0.543	0.762	0.794	0.762	0.743	0.782	0.809	**0.823**	0.801
watercraft	0.513	0.63	0.355	0.471	0.579	0.607	0.601	0.583	0.603	0.684	**0.685**
Overall	0.560	0.596	0.352	0.552	0.626	0.659	0.642	0.645	0.670	**0.729**	0.712

Table 1. **IoU Results on ShapeNet [2] for Single-View Reconstruction:** Thirteen test categories (first column) with one input view and the average IoU for each category. Best results for each metric are in **bold**; second best are underlined.

Category	3D-R2N2	OGN	AtlasNet	Pixel2Mesh	OccNet	IM-Net	AttSets	Pix2Vox++/F	Pix2Vox++/A	Mem3D	Ours
airplane	0.412	0.487	0.415	0.376	0.494	0.589	0.489	0.493	0.583	**0.671**	0.594
bench	0.345	0.364	0.439	0.313	0.318	0.361	0.406	0.399	0.478	0.525	**0.571**
cabinet	0.327	0.316	0.350	0.450	0.449	0.345	0.367	0.363	0.408	0.517	**0.453**
car	0.481	0.514	0.319	0.486	0.315	0.304	0.497	0.523	0.564	0.590	**0.602**
chair	0.238	0.226	0.406	0.386	0.365	0.442	0.334	0.262	0.309	0.503	**0.520**
display	0.227	0.215	0.451	0.319	0.468	0.466	0.310	0.253	0.296	**0.498**	0.475
lamp	0.267	0.249	0.217	0.219	0.361	0.371	0.315	0.287	0.315	**0.403**	0.368
speaker	0.231	0.225	0.199	0.190	0.249	0.200	0.211	0.256	0.152	**0.262**	0.242
rifle	0.521	0.541	0.405	0.340	0.219	0.407	0.524	0.553	0.574	0.626	**0.678**
sofa	0.274	0.290	0.337	0.343	0.324	0.354	0.334	0.320	0.377	0.434	**0.481**
table	0.340	0.352	0.373	0.502	0.549	0.461	0.419	0.385	0.406	0.569	**0.584**
telephone	0.504	0.528	0.545	0.485	0.273	0.423	0.469	0.588	0.633	0.674	**0.695**
watercraft	0.305	0.328	0.296	0.266	0.347	0.369	0.315	0.346	0.390	0.461	**0.470**
Overall	0.351	0.368	0.362	0.398	0.393	0.405	0.395	0.394	0.436	0.517	**0.518**

Table 2. **F-Score@1% Results on ShapeNet [2] for Single-View Reconstruction:** Thirteen test categories (first column) with one input view and the average F-Score@1% for each category. Best results for each metric are in **bold**; second best are underlined.

4.2. Datasets

We used the following two datsets for our study:

ShapeNet [2] contains 3D CAD models and is organized according to their WordNet classification. For convenience, we use a subset of the ShapeNet dataset that consists of $50,000$ models across 13 categories, such as plane and bench. We randomly split $2/3$ of the subset into the training set and the remaining $1/3$ into the testing set.

Pix3D [24] contains 395 3D models of nine object classes. It provides a set of real-world images for each CAD model. Following [24, 34], we use 2,894 untruncated and unoccluded images from the chair category for testing.

5. Results

We report two types of results for each experiment compared with other state-of-the-art voxel reconstruction methods: quantitative and qualitative. For *quantitative* analysis, Intersection-over-Union (IoU) and F-Score are used as the primary evaluation metrics, similarly to previous works [4, 25, 34]. IoU computes the intersecting areas between a 3D voxel reconstruction and its ground truth. Following [26], we take F-Score as a metric to evaluate the performance of 3D reconstruction results. We follow [34] for the evaluation on F-Score for all the reconstruction methods. For *qualitative* analysis, we visually compare the results of our approach against other methods.

5.1. TMVNet with ShapeNet Dataset [2]

Single-view reconstruction results: Tables 1 and 2 show the quantitative results, specifically IoU and F-score values respectively, for single view voxel reconstruction on the ShapeNet dataset. We compare our model with several state-of-the-art methods: 3D-R2N2 [4], OGN [25], AtlasNet [9], Pixel2Mesh [30], OccNet [17], IM-Net [3], AttSets [35], Pix2Vox++/F [34], and Pix2Vox++/A [34],Mem3D

Methods	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views
3D-R2N2	0.560	0.603	0.617	0.625	0.634	0.635	0.636	0.636
AttSets	0.642	0.662	0.670	0.675	0.677	0.685	0.688	0.692
Pix2Vox++/F	0.645	0.669	0.678	0.682	0.685	0.690	0.692	0.693
Pix2Vox++/A	0.670	0.695	0.704	0.708	0.711	0.715	0.717	0.718
Ours	**0.712**	**0.715**	**0.715**	**0.718**	**0.718**	**0.719**	**0.719**	**0.721**

Table 3. **IoU Results on ShapeNet [2] for Multi-View Reconstruction:** Comparison of multi-view 3D object reconstruction on ShapeNet at 32^3 resolution. Best results for each metric are in **bold**; second best are underlined.

Methods	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views
3D-R2N2	0.351	0.368	0.372	0.378	0.382	0.383	0.382	0.382
AttSets	0.395	0.418	0.426	0.430	0.432	0.444	0.445	0.447
Pix2Vox++/F	0.394	0.422	0.432	0.437	0.440	0.446	0.449	0.450
Pix2Vox++/A	0.436	0.452	0.455	0.457	0.458	0.459	0.460	0.461
Ours	**0.518**	**0.518**	**0.539**	**0.541**	**0.546**	**0.546**	**0.547**	**0.550**

Table 4. **F-Score@1% Results on ShapeNet [2] for Multi-View Reconstruction:** Comparison of multi-view 3D object reconstruction on ShapeNet at 32^3 resolution. Best results for each metric are in **bold**; second best are underlined.

[37]. As seen from the results, our model outperforms all other competitive methods on overall average IoU and average F-score results.

Multi-view reconstruction results: To evaluate the performance of 3D voxel reconstruction from multi-view images, we compare our model with 3D-R2N2 [4], AttSets [35], Pix2Vox++/F [34], and Pix2Vox++/A [34]. As shown in Tables 3 and 4, we conduct experiments on 8 different input view categories, namely 1-, 2-, 3-, 4-, 5-, 8-, 12-, and 16-views. Our model performs the best in 7 input view categories and second best in 1 input view category (4-views).

Figure 4 shows visual qualitative results of our multi-view voxel reconstruction approach on images from ShapeNet dataset on 3-views. From visual inspection, we can see that our model performs better in terms of the capture of details, as opposed to other methods.

High resolution reconstruction results: To further test the model performance at higher resolutions, namely, 64^3 and 128^3, we compare our model in single and multi-views situations against other state-of-the-art methods, including: OGN [25], Matryoshaka [22], Pix2Vox++/F [34], and Pix2Vox++/A [34]. We use the same experimental

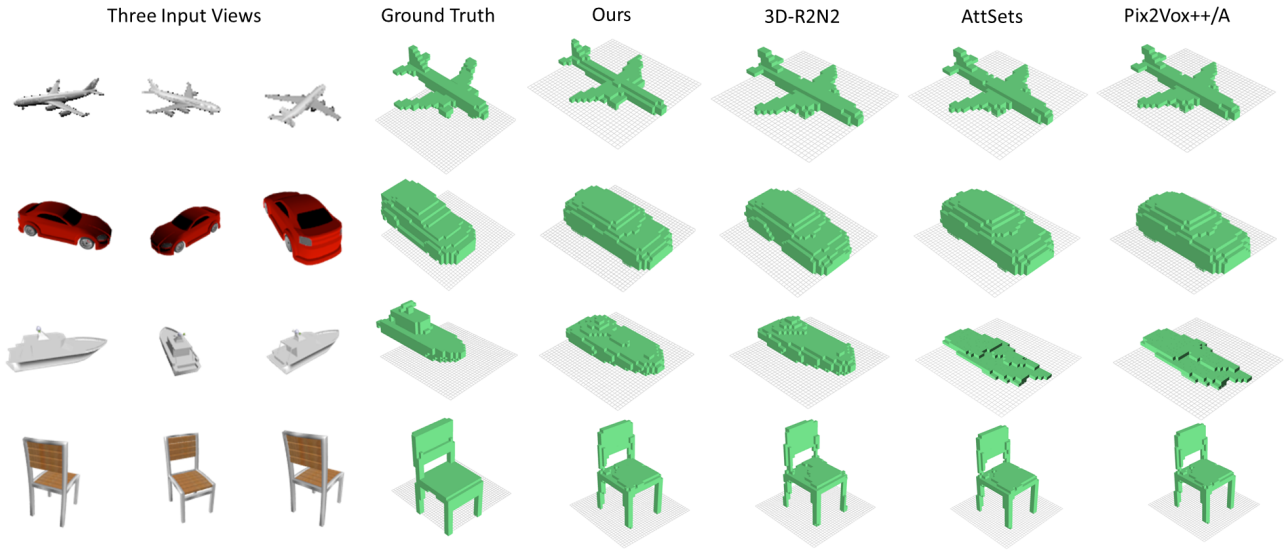

Figure 4. **Qualitative Results on ShapeNet [2] for Multi-View Reconstruction:** Comparison of our approach against against other approaches on four different test cases with ground truth data.

Methods	1 view	2 views	3 views
Resolution: 64^3			
OGN	0.771	N/A	N/A
Matryoshka	0.784	N/A	N/A
Pix2Vox++/F	0.793	0.807	0.809
Pix2Vox++/A	0.803	**0.813**	0.814
Ours	**0.805**	**0.813**	**0.818**
Resolution: 128^3			
OGN	0.782	N/A	N/A
Matryoshka	0.794	N/A	N/A
Pix2Vox++/F	0.817	0.832	0.838
Pix2Vox++/A	0.826	0.837	0.841
Ours	**0.831**	**0.843**	**0.849**

Table 5. **IoU Results on ShapeNet [2] Cars at Higher Resolutions:** Comparison of single and multi-view reconstruction at 64^3 resolution. Best results for each metric are in **bold**; second best are underlined.

Methods	1 view	2 views	3 views
Resolution: 64^3			
OGN	0.361	N/A	N/A
Matryoshka	0.380	N/A	N/A
Pix2Vox++/F	0.401	0.429	0.433
Pix2Vox++/A	0.418	0.448	0.450
Ours	**0.436**	**0.451**	**0.456**
Resolution: 128^3			
OGN	0.390	N/A	N/A
Matryoshka	0.426	N/A	N/A
Pix2Vox++/F	0.459	0.502	0.517
Pix2Vox++/A	0.475	0.509	0.521
Ours	**0.476**	**0.513**	**0.530**

Table 6. **F-Score@1% Results on ShapeNet [2] Cars at Higher Resolutions:** Comparison of single and multi-view reconstruction at 64^3 resolution. Best results for each metric are in **bold**; second best are underlined.

setup as [34], predicting 3D voxels of cars in the ShapeNet dataset. Tables 5 and 6 show the result of 64^3 and 128^3 resolutions respectively. As seen from the results, our model outperforms other state-of-the-art methods in both 64^3 and 128^3 resolution. The only exception is the result of 2 views at 64^3 resolution, our model obtains the same IoU as Pix2Vox++/A [34].

5.2. TMVNet with Pix3D Dataset [24]

We evaluated the performance of our network in single-view reconstruction on real-world images from the Pix3D Dataset [24]. We trained our network on ShapeNetChairs and tested it on the chair category of the Pix3D dataset. As shown in Table 7, our networks trained on ShapeNetChairs have better results than Pix2Vol++/A [34] and Pix2Vol++/F [34] trained on ShapeNet-Chairs.

Method	IoU	F-Score@1%
Pix2Vox++/F	0.179	0.012
Pix2Vox++/A	0.204	0.018
Ours	**0.210**	**0.021**

Table 7. **IoU and F-Score@1% Results on Pix3D [24] for Single-View Reconstruction:** Comparison F-Score@1% of single-view 3D object reconstruction on Pix3D at 32^3 resolution. Best results for each metric are in **bold**.

5.3. Ablation Study

We conduct an ablation study on the ShapeNet dataset to better understand the importance of the proposed modules: two-layers Transformer Encoder and 3D feature Fusion Network (3D FF). Table 8 shows the results of the different variants of our model on the ShapeNet dataset. As seen in the results, the baseline model with a 2D CNN Encoder, Res3D-GRU [4], and a 3D CNN Decoder neural, but without the proposed modules performs poorly compared to the others. On the other hand, the model with all our proposed modules - two-layer Transformer Encoder and 3D feature Fusion Network - performs the best with significant improvements compared to the baseline model and other partial configurations. This proposed two Transformer Encoder is the primary contributor towards successful voxel reconstruction. However, we also find that the baseline model can be improved highly even only using the 3D feature Fusion Network without the two Transformer Encoders. The variant with one Transformer Encoder works the second best. But the three Transformer Encoders perform worse than the one- and two-layer Transformer versions. The reason for this is that by adding too many transformer layers, the model easily overfits the dataset, resulting in poor performance.

		Avg IoU in Diffrernt View(s)				
	3D FF	1	2	3	4	5
Res3D-GRU		0.558	0.604	0.619	0.622	0.631
baseline	✓	0.583	0.606	0.626	0.638	0.635
One layer		0.482	0.436	0.512	0.511	0.541
Transformer	✓	0.628	0.632	0.637	0.645	0.651
Two layer		0.571	0.575	0.607	0.614	0.627
Transformer	✓	**0.712**	**0.715**	**0.715**	**0.718**	**0.718**
Three layer		0.543	0.504	0.529	0.537	0.563
Transformer	✓	0.615	0.622	0.640	0.641	0.643

Table 8. **ShapeNet [2] Ablation Study:** Ablation study of our model on all the categories from the ShapeNet [2]. Best results for each metric are in **bold**; second best are underlined.

5.4. Limitations & Future Work

Our approach for voxel reconstruction extracts image features from 2D image(s). There could be some edge areas

Figure 5. **Failure Case - ShapeNet Dataset [2]:** Our model fail to model the connection between chair legs.

in the image(s) that are hard to model. As Figure 5 shows, our model fails to model the connection between chair legs. This is because our CNN encoder may not be able to model object's detail when it is occluded. Another limitation is that the edge areas are not smooth, e.g., the edge area on the seat back. Adding gradient information into training may help us solve this issue. We will investigate such specific cases further as part of our future work and try to address the limitations of our proposed method.

6. Conclusion

In this paper, we proposed *TMVNet* – a novel and effective multi-layer transformer network to perform voxel-based multi-view 3D reconstruction using a transformer encoder and a convolutional decoder. A 2D CNN encoder was used to extract 2D image features for the proposed transformer encoder, which used a two-layer transformer and self-attention model to represent the 3D position of each voxel grid. To learn the mapping from 2D to 3D, a 3D CNN decoder was used to decode the output from the transformer encoder to generate voxel occupation probabilities. We also proposed a 3D feature fusion layer to fuse all the 3D voxel features. We showed experimental results on the widely-used ShapeNet and Pix3D datasets, which demonstrated the effectiveness of our proposed approach to voxel-based 3D reconstruction. The ablation study also showed that our proposed two-layer transformer encoder and 3D feature fusion network provided significant improvements.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 3

[2] 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, 5, 6, 7, 8

[3] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 5

[4] 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 *European conference on computer vision*, pages 628–644. Springer, 2016. 2, 3, 4, 5, 6, 7

[5] Zhao Dong, Wei Chen, Hujun Bao, Hongxin Zhang, and Qunsheng Peng. Real-time voxelization for complex polygonal models. In *12th Pacific Conference on Computer Graphics and Applications, 2004. PG 2004. Proceedings.*, pages 43–50, 2004. 2

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[7] Vincent Forest, Loic Barthe, and Mathias Paulin. Real-time hierarchical binary-scene voxelization. *journal of graphics, gpu, and game tools*, 14(3):21–34, 2009. 2

[8] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *Artificial intelligence review*, 43(1):55–81, 2015. 1

[9] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018. 5

[10] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *arXiv preprint arXiv:2012.09688*, 2020. 1, 3

[11] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006. 3

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[13] Vladimir V Kniaz, Vladimir A Knyaz, Fabio Remondino, Artem Bordodymov, and Petr Moshkantsev. Image-to-voxel model translation for 3d scene reconstruction and segmentation. In *European Conference on Computer Vision*, pages 105–124. Springer, 2020. 2

[14] Filippos Kokkinos and Iasonas Kokkinos. Learning monocular 3d reconstruction of articulated categories from motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1737–1746, 2021. 1

[15] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. *arXiv preprint arXiv:2012.09760*, 2020. 3

[16] Shikun Liu, II Ago, and C Lee Giles. Learning a hierarchical latent-variable model of voxelized 3d shapes. *arXiv preprint arXiv:1705.05994*, 2017. 2

[17] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 5

[18] Onur Özyeşil, Vladislav Voroninski, Ronen Basri, and Amit Singer. A survey of structure from motion*. *Acta Numerica*, 26:305–364, 2017. 1

[19] Jiahao Pang, Duanshun Li, and Dong Tian. Tearingnet: Point cloud autoencoder to learn topology-friendly representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7453–7462, 2021. 1

[20] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019. 4

[21] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *arXiv preprint arXiv:2103.13413*, 2021. 2

[22] Stephan R Richter and Stefan Roth. Matryoshka networks: Predicting 3d geometry via nested shape layers. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1936–1944, 2018. 6

[23] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 3

[24] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018. 2, 5, 7

[25] 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*, pages 2088–2096, 2017. 5, 6

[26] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019. 5

[27] Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Multi-view consistency as supervisory signal for learning shape and pose prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2897–2905, 2018. 2

[28] Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2626–2634, 2017. 2

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. 1, 2

[30] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 5

[31] Zhengyang Wang, Yaochen Xie, and Shuiwang Ji. Global voxel transformer networks for augmented microscopy. *Nature Machine Intelligence*, 3(2):161–171, 2021. 3

[32] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, William T Freeman, and Joshua B Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. *arXiv preprint arXiv:1711.03129*, 2017. 2

[33] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 2

[34] Haozhe Xie, Hongxun Yao, Shengping Zhang, Shangchen Zhou, and Wenxiu Sun. Pix2vox++: multi-scale context-aware 3d object reconstruction from single and multiple images. *International Journal of Computer Vision*, 128(12):2919–2935, 2020. 1, 4, 5, 6, 7

[35] Bo Yang, Sen Wang, Andrew Markham, and year=2018 Niki Trigoni, booktitle=IJCV 2019. Robust attentional aggregation of deep feature sets for multi-view 3d reconstruction. 5, 6

[36] Guandao Yang, Yin Cui, Serge Belongie, and Bharath Hariharan. Learning single-view 3d reconstruction with limited pose supervision. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 86–101, 2018. 2

[37] Shuo Yang, Min Xu, Haozhe Xie, Stuart Perry, and Jiahao Xia. Single-view 3d object reconstruction from shape priors in memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3152–3161, 2021. 6

[38] Wentao Yuan, David Held, Christoph Mertz, and Martial Hebert. Iterative transformer network for 3d point cloud. *arXiv preprint arXiv:1811.11209*, 2018. 3

[39] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. *arXiv preprint arXiv:2012.09164*, 2020. 1, 2