Robust Attentional Aggregation of Deep Feature Sets for Multi-view 3D Reconstruction

Bo Yang · Sen Wang · Andrew Markham · Niki Trigoni

Abstract We study the problem of recovering an underlying 3D shape from a set of images. Existing learning based approaches usually resort to recurrent neural nets, *e.g.*, GRU, or intuitive pooling operations, *e.g.*, max/mean poolings, to fuse multiple deep features encoded from input images. However, GRU based approaches are unable to consistently estimate 3D shapes given different permutations of the same set of input images as the recurrent unit is permutation variant. It is also unlikely to refine the 3D shape given more images due to the long-term memory loss of GRU. Commonly used pooling approaches are limited to capturing partial information, *e.g.*, max/mean values, ignoring other valuable features. In this paper, we present a new feedforward neural module, named **AttSets**, together with a dedicated training algorithm, named **FASet**, to attentively aggregate an arbitrarily sized deep feature set for multi-view 3D reconstruction. The AttSets module is permutation invariant, computationally efficient and flexible to implement, while the FASet algorithm enables the AttSets based network to be remarkably robust and generalize to an arbitrary number of input images. We thoroughly evaluate FASet and the properties of AttSets on multiple large public datasets. Extensive experiments show that AttSets together with FASet algorithm significantly outperforms existing aggregation approaches.

Bo Yang, Andrew Markham, Niki Trigoni
Department of Computer Science,
University of Oxford
E-mail: firstname.lastname@cs.ox.ac.uk

Sen Wang
School of Engineering & Physical Sciences,
Heriot-Watt University
E-mail: s.wang@hw.ac.uk

1 Introduction

The problem of recovering a geometric representation of the 3D world given a set of images is classically defined as multi-view 3D reconstruction in computer vision. Traditional pipelines such as Structure from Motion (SfM) (Ozyesil et al., 2017) and visual Simultaneous Localization and Mapping (vSLAM) (Cadena et al., 2016) typically rely on hand-crafted feature extraction and matching across multiple views to reconstruct the underlying 3D model. However, if the multiple viewpoints are separated by large baselines, it can be extremely challenging for the feature matching approach due to significant changes of appearance or self occlusions (Lowe, 2004). Furthermore, the reconstructed 3D shape is usually a sparse point cloud without geometric details.

Recently, a number of deep learning approaches, such as 3D-R2N2 (Choy et al., 2016), LSM (Kar et al., 2017), DeepMVS (Huang et al., 2018) and RayNet (Paschalidou et al., 2018) have been proposed to estimate the 3D dense shape from multiple images and have shown encouraging results. Both 3D-R2N2 (Choy et al., 2016) and LSM (Kar et al., 2017) formulate multi-view reconstruction as a sequence learning problem, and leverage recurrent neural networks (RNNs), particularly GRU, to fuse the multiple deep features extracted by a shared encoder from input images. However, there are three limitations. First, the recurrent network is permutation variant, *i.e.*, different permutations of the input image sequence give different reconstruction results (Vinyals et al., 2015). Therefore, inconsistent 3D shapes are estimated from the same image set with different per-
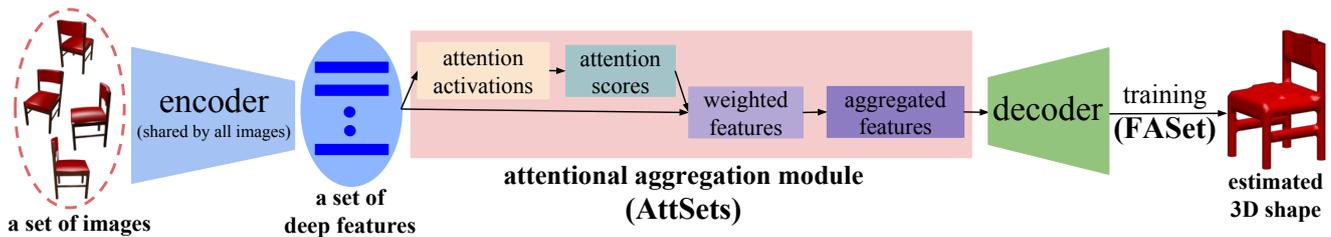
Fig. 1 Overview of our attentional aggregation module for multi-view 3D reconstruction. A set of N images is passed through a common encoder to be a set of deep features, one element for each image. The network is trained with our FASet algorithm.

mutations. Second, it is difficult to capture long-term dependencies in the sequence because of the gradient vanishing or exploding (Bengio et al., 1994; Kolen and Kremer, 2001), so the estimated 3D shapes are unlikely to be refined even if more images are given during training and testing. Third, the RNN unit is inefficient as each element of the input sequence must be sequentially processed without parallelization (Martin and Cundy, 2018), so is time-consuming to generate the final 3D shape given a sequence of images.

The recent DeepMVS (Huang et al., 2018) applies max pooling to aggregate deep features across a set of unordered images for multi-view stereo reconstruction, while RayNet (Paschalidou et al., 2018) adopts average pooling to aggregate the deep features corresponding to the same voxel from multiple images to recover a dense 3D model. The very recent GQN (Eslami et al., 2018) uses sum pooling to aggregate an arbitrary number of orderless images for 3D scene representation. Although max, average and summation poolings do not suffer from the above limitations of RNN, they tend to be 'hard attentive', since they only capture the max/mean values or the summation without learning to attentively preserve the useful information. In addition, the above pooling based neural nets are usually optimized with a specific number of input images during training, therefore being not robust and general to a dynamic number of input images during testing. This critical issue is also observed in GQN (Eslami et al., 2018).

In this paper, we introduce a simple yet efficient attentional aggregation module, named **AttSets** [1]. It can be easily included in an existing multi-view 3D reconstruction network to aggregate an arbitrary number of elements of a deep feature set. Inspired by the attention mechanism which shows great success in natural language processing (Bahdanau et al., 2015; Raffel and Ellis, 2016), image captioning (Xu et al., 2015), *etc.*, we design a feed-forward neural module that can automatically learn to aggregate each element of the input deep feature set. In particular, as shown in Figure 1, given a variable sized deep feature set, which

are usually learnt view-invariant visual representations from a shared encoder (Paschalidou et al., 2018), our AttSets module firstly learns an **attention activation** for each latent feature through a standard neural layer (*e.g.*, a fully connected layer, a 2D or 3D convolutional layer), after which an **attention score** is computed for the corresponding feature. Subsequently, the attention scores are simply multiplied by the original elements of the deep feature set, generating a set of **weighted features**. At last, the weighted features are summed across different elements of the deep feature set, producing a fixed size of **aggregated features** which are then fed into a decoder to estimate 3D shapes. Basically, this AttSets module can be seen as a natural extension of sum pooling into a "weighted" sum pooling with learnt feature-specific weights. AttSets shares similar concepts with the concurrent work (Ilse et al., 2018), but it does not require the additional gating mechanism in (Ilse et al., 2018). Notably, our simple feed-forward design allows the attention module to be separately trainable according to the property of its gradients.

In addition, we propose a new **Feature-Attention Separate training (FASet)** algorithm that elegantly decouples the base encoder-decoder (to learn deep features) from the AttSets module (to learn attention scores for features). This allows the AttSets module to learn desired attention scores for deep feature sets and guarantees the AttSets based neural networks to be robust and general to dynamic sized deep feature sets. Basically, in the proposed training algorithm, the base encoder-decoder neural layers are only optimized when the number of input images is **1**, while the AttSets module is only optimized where there are more than **1** input images. Eventually, the whole optimized AttSets based neural network achieves superior performance with a large number of input images, while simultaneously being extremely robust and able to generalize to a small number of input images, even to a single image in the extreme case. Comparing with the widely used feed-forward attention mechanisms for visual recognition (Jie Hu et al., 2018; Rodríguez et al., 2018; Liu et al., 2018; Sarafianos et al., 2018; Girdhar and Ramanan, 2017), our FASet algorithm is the first to investigate

---

[1] Code is available at *https://github.com/Yang7879/AttSets*

and improve the robustness of attention modules to dynamically sized input feature sets, whilst existing works are only applicable to fixed sized input data.

Overall, our novel AttSets module and FASet algorithm are distinguished from all existing aggregation approaches in three ways. 1) Compared with RNN approaches, AttSets is permutation invariant and computationally efficient. 2) Compared with the widely used pooling operations, AttSets learns to attentively select and weight important deep features, thereby being more effective to aggregate useful information for better 3D reconstruction. 3) Compared with existing visual attention mechanisms, our FASet algorithm enables the whole network to be general to variable sized sets, being more robust and suitable for realistic multi-view 3D reconstruction scenarios where the number of input images usually varies dramatically.

Our key contributions are:

- We propose an efficient feed-forward attention module, AttSets, to effectively aggregate deep feature sets. Our design allows the attention module to be separately optimizable according to the property of its gradients.
- We propose a new two-stage training algorithm, FASet, to decouple the base encoder/decoder and the attention module, guaranteeing the whole network to be robust and general to an arbitrary number of input images.
- We conduct extensive experiments on multiple public datasets, demonstrating consistent improvement over existing aggregation approaches for 3D object reconstruction from either single or multiple views.

2 Related Work

(1) Multi-view 3D Reconstruction. 3D shapes can be recovered from multiple color images or depth scans. To estimate the underlying 3D shape from **multiple color images**, classic SfM (Ozyesil et al., 2017) and vSLAM (Cadena et al., 2016) algorithms firstly extract and match hand-crafted geometric features (Hartley and Zisserman, 2004) and then apply bundle adjustment (Triggs et al., 1999) for both shape and camera motion estimation. Ji *et al.* (Ji et al., 2017b) use "maximizing rigidity" for reconstruction, but this requires 2D point correspondences across images. Recent deep neural net based approaches tend to recover dense 3D shapes through learnt features from multiple images and achieve compelling results. To fuse the deep features from multiple images, both 3D-R2N2 (Choy et al., 2016) and LSM (Kar et al., 2017) apply the recurrent unit GRU, resulting in the networks being permutation

variant and inefficient for aggregating long sequence of images. Recent SilNet (Wiles and Zisserman, 2017, 2018) and DeepMVS (Huang et al., 2018) simply use max pooling to preserve the first order information of multiple images, while RayNet (Paschalidou et al., 2018) applies average pooling to reserve the first moment information of multiple deep features. MVSNet (Yao et al., 2018) proposes a variance-based approach to capture the second moment information for multiple feature aggregation. These pooling techniques only capture partial information, ignoring the majority of the deep features. Recent SurfaceNet (Ji et al., 2017a) and SuperPixel Soup (Kumar et al., 2017) can reconstruct 3D shapes from two images, but they are unable to process an arbitrary number of images. As for **multiple depth image** reconstruction, the traditional volumetric fusion method (Curless and Levoy, 1996; Cao et al., 2018) integrates multiple viewpoint information by averaging truncated signed distance functions (TSDF). Recent learning based OctNetFusion (Riegler et al., 2017) also adopts a similar strategy to integrate multiple depth information. However, this integration might result in information loss since TSDF values are averaged (Riegler et al., 2017). PSDF (Dong et al., 2018) is recently proposed to learn a probabilistic distribution through Bayesian updating in order to fuse multiple depth images, but it is not straightforward to include the module into existing encoder-decoder networks.

(2) Deep Learning on Sets. In contrast to traditional approaches operating on fixed dimensional vectors or matrices, deep learning tasks defined on sets usually require learning functions to be permutation invariant and able to process an arbitrary number of elements in a set (Zaheer et al., 2017). Such problems are widespread. Zaheer *et al.* introduce general permutation invariant and equivariant models in (Zaheer et al., 2017), and they end up with a **sum pooling** for permutation invariant tasks such as population statistics estimation and point cloud classification. In the very recent GQN (Eslami et al., 2018), sum pooling is also used to aggregate an arbitrary number of orderless images for 3D scene representation. Gardner *et al.* (Gardner et al., 2017) use **average pooling** to integrate an unordered deep feature set for classification task. Su *et al.* (Su et al., 2015) use **max pooling** to fuse the deep feature set of multiple views for 3D shape recognition. Similarly, PointNet (Qi et al., 2017) also uses max pooling to aggregate the set of features learnt from point clouds for 3D classification and segmentation. In addition, the higher-order statistics based pooling approaches are widely used for 3D object recognition from multiple images. Vanilla **bilinear pooling** is applied for fine-grained recognition in (Lin et al., 2015) and is further improved in (Lin and
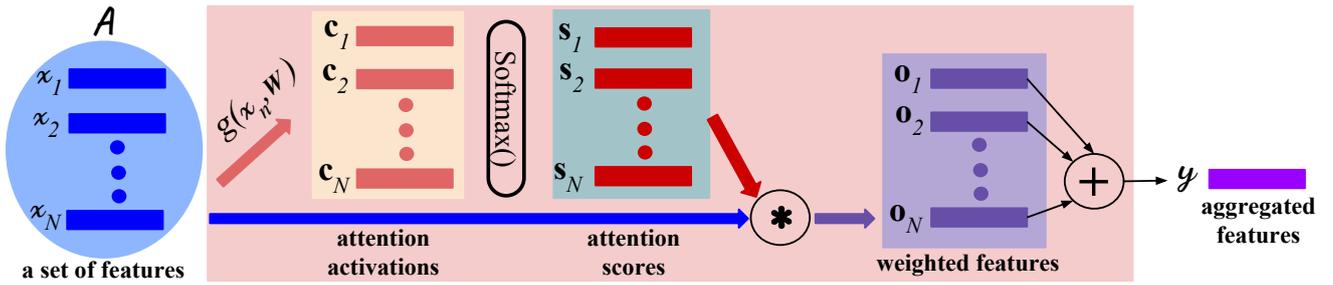
Fig. 2 Attentional aggregation module on sets. This module learns an attention score for each individual deep feature.

Maji, 2017). Concurrently, **log-covariance pooling** is proposed in (Ionescu et al., 2015), and is recently generalized by **harmonized bilinear pooling** in (Yu et al., 2018). Bilinear pooling techniques are further improved in the recent work (Yu and Salzmann, 2018; Lin et al., 2018). However, both first-order and higher-order pooling operations ignore a majority of the information of a set. In addition, the first-order poolings do not have trainable parameters, while the higher-order poolings have only few parameters available for the network to learn. These limitations lead to the pooling based neural networks to be optimized with regards to the specific statistics of data batches during training, and therefore unable to be robust and generalize well to variable sized deep feature sets during testing.

(3) Attention Mechanism. The attention mechanism was originally proposed for natural language processing (Bahdanau et al., 2015). Being coupled with RNNs, it achieves compelling results in neural machine translation (Bahdanau et al., 2015), image captioning (Xu et al., 2015), image question answering (Yang et al., 2016), *etc.* However, all these coupled attention approaches are permutation variant and computationally time-consuming. Dispensing with recurrence and convolutions entirely and solely relying on attention mechanism, Transformer (Vaswani et al., 2017) achieves superior performance in machine translation tasks. Similarly, being decoupled with RNNs, attention mechanisms are also applied for visual recognition (Jie Hu et al., 2018; Rodríguez et al., 2018; Liu et al., 2018; Sarafianos et al., 2018; Zhu et al., 2018; Nakka and Salzmann, 2018; Girdhar and Ramanan, 2017), semantic segmentation (Li et al., 2018), long sequence learning (Raffel and Ellis, 2016), and image generation (Zhang et al., 2018). Although the above decoupled attention modules can be used to aggregate variable sized deep feature sets, they are literally designed to operate on fixed sized features for tasks such as image recognition and generation. The robustness of attention modules regarding dynamic deep feature sets has not been investigated yet.

Compared with the original attention mechanism, our AttSets does not couple with RNNs. Instead, AttSets is a simplified feed-forward module which shares similar concepts with the concurrent work (Ilse et al., 2018). However, our AttSets is much simpler, without requiring the additional gating mechanism in (Ilse et al., 2018). Besides, we further propose a dedicated FASet algorithm, enabling the AttSets based network to be remarkably robust and general to arbitrarily sized deep sets. This algorithm is the first to investigate and improve the robustness of feed-forward attention mechanisms.

3 AttSets

3.1 Problem Definition

This paper considers the problem of aggregating an arbitrary number of elements of a set $\mathcal{A}$ into a fixed single output $\boldsymbol{y}$. Each element of set $\mathcal{A}$ is a feature vector extracted from a shared encoder, and the fixed dimension output $\boldsymbol{y}$ is fed into a subsequent decoder, such that the whole network can process an arbitrary number of input elements.

Given N elements in the input deep feature set $\mathcal{A} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^{1 \times D}$, where N is an arbitrary value, while D is fixed for a specific encoder, and the output $\boldsymbol{y} \in \mathbb{R}^{1 \times D}$, which is then fed into the subsequent decoder, our task is to design an aggregation function f with learnable weights $\boldsymbol{W}$: $\boldsymbol{y} = f(\mathcal{A}, \boldsymbol{W})$, which should be permutation invariant, *i.e.*, for any permutation π:

$$f(\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\}, \boldsymbol{W}) = f(\{\boldsymbol{x}_{\pi(1)}, \cdots, \boldsymbol{x}_{\pi(N)}\}, \boldsymbol{W}) \quad (1)$$

The common pooling operations, *e.g.*, max/mean/sum, are the simplest instantiations of function f where $\boldsymbol{W} \in \emptyset$. However, these pooling operations are predefined to capture partial information.

3.2 AttSets Module

The basic idea of our AttSets module is to learn an attention score for each latent feature of the whole deep feature set. In this paper, each latent feature refers to each entry of an individual element of the feature set,
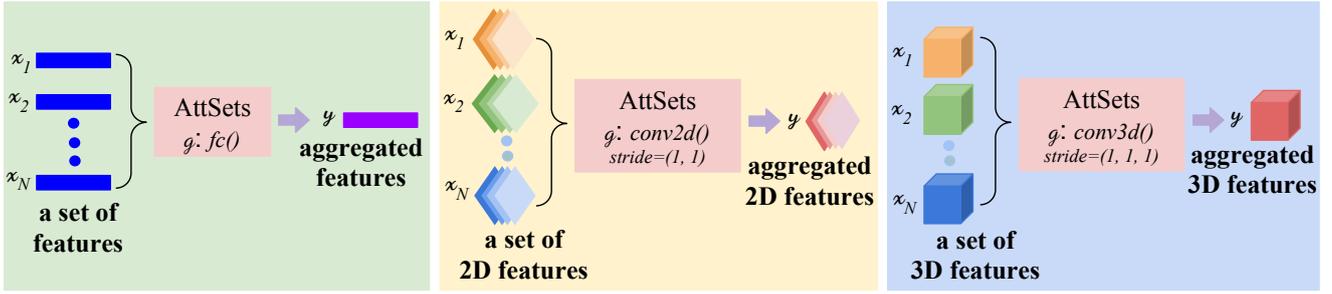
Fig. 3 Implementation of AttSets with fully connected layer, 2D ConvNet, and 3D ConvNet. These three variants of AttSets can be flexibly plugged into different locations of an existing encoder-decoder network.

with an individual element usually represented by a latent vector, i.e., $\boldsymbol{x}_n$. The learnt scores can be regarded as a mask that automatically selects useful latent features across the set. The selected features are then summed across multiple elements of the set.

As shown in Figure 2, given a set of features $\mathcal{A} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^{1 \times D}$, AttSets aims to fuse it into a fixed dimensional output $\boldsymbol{y}$, where $\boldsymbol{y} \in \mathbb{R}^{1 \times D}$.

To build the AttSets module, we first feed each element of the feature set $\mathcal{A}$ into a shared function g which can be a standard neural layer, i.e., a linear transformation layer without any non-linear activation functions. Here we use a fully connected layer as an example, the bias term is dropped for simplicity. The output of function g is a set of learnt attention activations $\mathcal{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \cdots, \boldsymbol{c}_N\}$, where

$$\boldsymbol{c}_n = g(\boldsymbol{x}_n, \boldsymbol{W}) = \boldsymbol{x}_n \boldsymbol{W},$$
$$(\boldsymbol{x}_n \in \mathbb{R}^{1 \times D}, \quad \boldsymbol{W} \in \mathbb{R}^{D \times D}, \quad \boldsymbol{c}_n \in \mathbb{R}^{1 \times D}) \quad (2)$$

Secondly, the learnt attention activations are normalized across the N elements of the set, computing a set of attention scores $\mathcal{S} = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_N\}$. We choose $softmax$ as the normalization operation, so the attention scores for the nth feature element are

$$\boldsymbol{s}_n = [s_n^1, s_n^2, \cdots, s_n^d, \cdots, s_n^D],$$
$$s_n^d = \frac{e^{c_n^d}}{\sum_{j=1}^N e^{c_j^d}}, c_n^d, c_j^d \text{ are the } d^{th} \text{ entry of } \boldsymbol{c}_n, \boldsymbol{c}_j. \quad (3)$$

Thirdly, the computed attention scores $\mathcal{S}$ are multiplied by their corresponding original feature set $\mathcal{A}$, generating a new set of deep features, denoted as weighted features $\mathcal{O} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \cdots, \boldsymbol{o}_N\}$, where

$$\boldsymbol{o}_n = \boldsymbol{x}_n * \boldsymbol{s}_n \quad (4)$$

Lastly, the set of weighted features $\mathcal{O}$ are summed up across the total N elements to get a fixed size feature vector, denoted as $\boldsymbol{y}$, where

$$\boldsymbol{y} = [y^1, y^2, \cdots, y^d, \cdots, y^D],$$
$$y^d = \sum_{n=1}^N o_n^d, \qquad o_n^d \text{ is the } d^{th} \text{ entry of } \boldsymbol{o}_n. \quad (5)$$

In the above formulation, we show how AttSets gradually aggregates a set of N feature vectors $\mathcal{A}$ into a single vector $\boldsymbol{y}$, where $\boldsymbol{y} \in \mathbb{R}^{1 \times D}$.

3.3 Permutation Invariance

The output of AttSets module $\boldsymbol{y}$ is permutation invariant with regard to the input deep feature set $\mathcal{A}$. Here is the simple proof.

$$[y^1, \cdots y^d \cdots, y^D] = f(\{\boldsymbol{x}_1, \cdots \boldsymbol{x}_n \cdots, \boldsymbol{x}_N\}, \boldsymbol{W}) \quad (6)$$

In Equation 6, the dth entry of the output $\boldsymbol{y}$ is computed as follows:

$$\begin{aligned} y^d &= \sum_{n=1}^N o_n^d = \sum_{n=1}^N (x_n^d * s_n^d) \\ &= \sum_{n=1}^N \left(x_n^d * \frac{e^{c_n^d}}{\sum_{j=1}^N e^{c_j^d}} \right) \\ &= \sum_{n=1}^N \left(x_n^d * \frac{e^{(\boldsymbol{x}_n \boldsymbol{w}^d)}}{\sum_{j=1}^N e^{(\boldsymbol{x}_j \boldsymbol{w}^d)}} \right) \\ &= \frac{\sum_{n=1}^N \left(x_n^d * e^{(\boldsymbol{x}_n \boldsymbol{w}^d)} \right)}{\sum_{j=1}^N e^{(\boldsymbol{x}_j \boldsymbol{w}^d)}}, \end{aligned} \quad (7)$$

where $\boldsymbol{w}^d$ is the dth column of the weights $\boldsymbol{W}$. In above Equation 7, both the denominator and numerator are a summation of a permutation equivariant term. Therefore the value y^d, and also the full vector $\boldsymbol{y}$, is invariant to different permutations of the deep feature set $\mathcal{A} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n, \cdots, \boldsymbol{x}_N\}$ (Zaheer et al., 2017).

3.4 Implementation

In Section 3.2, we described how our AttSets aggregates an arbitrary number of vector features into a single vector, where the attention activation learning function g embeds a fully connected (fc) layer. AttSets can also

be easily implemented with both 2D and 3D convolutional neural layers to aggregate both 2D and 3D deep feature sets, thus being flexible to be included into a 2D encoder/decoder or 3D encoder/decoder. Particularly, as shown in Figure 3, to aggregate a set of 2D features, *i.e.*, a tensor of ($width \times height \times channels$), the attention activation learning function g embeds a standard *conv2d* layer with a stride of (1×1). Similarly, to fuse a set of 3D features, *i.e.*, a tensor of ($width \times height \times depth \times channels$), the function g embeds a standard *conv3d* layer with a stride of ($1 \times 1 \times 1$). For the above *conv2d/conv3d* layer, the filter size can be 1, 3 or many. The larger the filter size, the learnt attention score is considered to be correlated with the larger local spatial area.

Instead of embedding a single neural layer, the function g is also flexible to include multiple layers, but the tensor shape of the output of function g is required to be consistent with the input element $\boldsymbol{x}_n$. This guarantees each individual feature of the input set $\mathcal{A}$ will be associated with a learnt and unique weight. For example, a standard 2-layer or 3-layer ResNet module (He et al., 2016) could be a candidate of the function g. The more layers that g embeds, the capability of AttSets module is expected to increase accordingly.

Compared with fc enabled AttSets, the *conv2d* or *conv3d* based AttSets variants tend to have fewer learnable parameters. Note that both the *conv2d* and *conv3d* based AttSets are still permutation invariant, as the function g is shared across all elements of the deep feature set and it does not depend on the order of the elements (Zaheer et al., 2017).

4 FASet

4.1 Motivation

Our AttSets module can be included in an existing encoder-decoder multi-view 3D reconstruction network, replacing the RNN units or pooling operations. Basically, in an AttSets enabled encoder-decoder net, the encoder-decoder serves as the base architecture to learn visual features for shape estimation, while the AttSets module learns to assign different attention scores to combine those features. As such, the base network tends to have robustness and generality with regard to different input image content, while the AttSets module tends to be general regarding an arbitrary number of input images.

However, to achieve this robustness is not straightforward. The standard end-to-end joint optimization approach is unable to guarantee that the base encoder-decoder and AttSets are able to learn visual features and the corresponding scores separately, because there are no explicit feature score labels available to directly supervise the AttSets module.

Let us revisit the previous Equation 7 as follows and draw insights from it.

$$y^d = \frac{\sum_{n=1}^{N} \left(x_n^d * e^{(\boldsymbol{x}_n \boldsymbol{w}^d)} \right)}{\sum_{j=1}^{N} e^{(\boldsymbol{x}_j \boldsymbol{w}^d)}} \qquad (8)$$

where N is the size of an arbitrary input set and $\boldsymbol{w}^d$ are the AttSets parameters to be optimized. If N is 1, then the equation can be simplified as

$$y^d = x_n^d \qquad (9)$$

$$\frac{\partial y^d}{\partial x_n^d} = 1, \qquad \frac{\partial y^d}{\partial \boldsymbol{w}^d} = \boldsymbol{0}, \qquad N = 1 \qquad (10)$$

This shows that all parameters, *i.e.*, $\boldsymbol{w}^d$, of the AttSets module are not going to be optimized when the size of the input feature set is 1.

However, if $N > 1$, Equation 8 is unable to be simplified to Equation 9. Therefore,

$$\frac{\partial y^d}{\partial x_n^d} \neq 1, \qquad \frac{\partial y^d}{\partial \boldsymbol{w}^d} \neq \boldsymbol{0}, \qquad N > 1 \qquad (11)$$

This shows that both the parameters of AttSets and the base encoder-decoder layers will be optimized simultaneously, if the whole network is trained in the standard end-to-end fashion.

Here arises the critical issue. When $N > 1$, all derivatives of the parameters in the **encoder** are different from the derivatives when $N = 1$ due to the chain rule of differentiation applied backwards from $\frac{\partial y^d}{\partial x_n^d}$. Put simply, the derivatives of encoder are N-*dependent*. As a consequence, the encoded visual features and the learnt attention scores would be N-*biased* if the whole network is jointly trained. This biased network is unable to generalize to an arbitrary value of N during testing.

To illustrate the above issue, assuming the base encoder-decoder and the AttSets module are jointly trained given 5 images to reconstruct every object, the base encoder will be eventually optimized towards 5-view object reconstruction during training. The trained network can indeed perform well given 5 views during testing, but it is unable to predict a satisfactory object shape given only 1 image.

To alleviate the above problem, a naive approach is to enumerate various values of N during the jointly training, such that the final optimized network can be somehow robust and general to arbitrary N during testing. However, this approach would inevitably optimize the encoder to learn the *mean* features of input data for varying N. The overall performance will hence not be optimal. In addition, it is impractical and also time-consuming to enumerate all values of N during training.

Algorithm 1 Feature-Attention Separate training of an AttSets enabled network. M is batch size, N is image number.

Stage 1:

for number of training iterations **do**

  • Sample M sets of images $\{\mathcal{I}_1, \cdots, \mathcal{I}_m, \cdots, \mathcal{I}_M\}$ and sample N images for each set, *i.e.*, $\mathcal{I}_m = \{i_m^1, \cdots, i_m^n, \cdots, i_m^N\}$. Sample M 3D shape labels $\{v_1, \cdots, v_m, \cdots, v_M\}$.

  • Update the base network by ascending its stochastic gradient:

$$\nabla_{\Theta_{base}} \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} [\ell(\hat{v}_m^n, v_m)], \ where \ \hat{v}_m^n \ \text{is the estimated 3D shape of single image} \ \{i_m^n\}.$$

Stage 2:

for number of training iterations **do**

  • Sample M sets of images $\{\mathcal{I}_1, \cdots, \mathcal{I}_m, \cdots, \mathcal{I}_M\}$ and sample N images for each set, *i.e.*, $\mathcal{I}_m = \{i_m^1, \cdots, i_m^n, \cdots, i_m^N\}$. Sample M 3D shape labels $\{v_1, \cdots, v_m, \cdots, v_M\}$.

  • Update the AttSets module by ascending its stochastic gradient:

$$\nabla_{\Theta_{att}} \frac{1}{M} \sum_{m=1}^{M} [\ell(\hat{v}_m, v_m)], where \ \hat{v}_m \ \text{is the estimated 3D shape of the image set} \ \mathcal{I}_m.$$

The gradient-based updates can use any gradient optimization algorithm.

4.2 Algorithm

To resolve the critical issue discussed in Section 4.1, we propose a **Feature-Attention Separate training (FASet)** algorithm that decouples the base encoder-decoder and the AttSets module, enabling the base encoder-decoder to learn robust deep features and the AttSets module to learn the desired attention scores for the feature sets.

In particular, the base encoder-decoder neural layers are only optimized when the number of input images is 1, while the AttSets module is only optimized where there are more than 1 input images. In this regard, the parameters of the base encoding layers would have consistent derivatives in the whole training stage, thus being optimized steadily. In the meantime, the AttSets module would be optimized solely based on multiple elements of learnt visual features from the shared encoder.

The trainable parameters of the base encoder-decoder are denoted as Θ_{base}, and the trainable parameters of AttSets module denoted as Θ_{att}, and the loss function of the whole network is represented by ℓ which is determined by the specific supervision signal of the base network. Our FASet is shown in Algorithm 1. It can be seen that Θ_{base} and Θ_{att} are completely decoupled from one another, thus being separately optimized in two stages. In stage 1, the Θ_{base} is firstly well optimized until convergence, which guarantees the base encoder-decoder is able to learn robust and general visual features. In stage 2, the Θ_{att} is optimized to learn attention scores for individual visual features. Basically, this module learns to select and weight important deep features automatically.

Fig. 4 The architecture of Base$_{r2n2}$-AttSets for multi-view 3D reconstruction network. The base encoder-decoder is the same as 3D-R2N2.

Fig. 5 The architecture of Base$_{silnet}$-AttSets for multi-view 3D shape learning. The base encoder-decoder is the same as SilNet.

In FASet algorithm, once the Θ_{base} is well optimized in stage 1, it is not necessary to train it again, since the two-stage algorithm guarantees that optimizing Θ_{base} is agnostic to the attention module. The FASet algorithm is a crucial component to maintain the superior robustness of the AttSets module, as shown in Section 5.9. Without it, the feed-forward attention mechanism is ineffective with respect to dynamic input sets.

5 Evaluation

Base Networks. To evaluate the performance and various properties of AttSets, we choose the encoder-decoders of 3D-R2N2 (Choy et al., 2016) and SilNet (Wiles and Zisserman, 2017) as two base networks.

– Encoder-decoder of 3D-R2N2. The original 3D-R2N2 consists of (1) a shared ResNet-based 2D encoder which encodes a size of $127 \times 127 \times 3$ images into

1024 dimensional latent vectors, (2) a GRU module which fuses N 1024 dimensional latent vectors into a single $4 \times 4 \times 4 \times 128$ tensor, and (3) a ResNet-based 3D decoder which decodes the single tensor into a $32 \times 32 \times 32$ voxel grid representing the 3D shape. Figure 4 shows the architecture of AttSets based multi-view 3D reconstruction network where the only difference is that the original GRU module is replaced by AttSets in the middle. This network is called $\text{Base}_{\text{r2n2}}$-AttSets.

– Encoder-decoder of SilNet. The original SilNet consists of (1) a shared 2D encoder which encodes a size of $127 \times 127 \times 3$ images together with image viewing angles into 160 dimensional latent vectors, (2) a max pooling module which aggregates N latent vectors into a single vector, and (3) a 2D decoder which estimates an object silhouette (57×57) from the single latent vector and a new viewing angle. Instead of being explicitly supervised by 3D shape labels, SilNet aims to implicitly learn a 3D shape representation from multiple images via the supervision of 2D silhouettes. Figure 5 shows the architecture of AttSets based SilNet where the only difference is that the original max pooling is replaced by AttSets in the middle. This network is called $\text{Base}_{\text{silnet}}$-AttSets.

Competing Approaches. We compare our AttSets and FASet with three groups of competing approaches. Note that all the following competing approaches are connected at the same location of the base encoder-decoder shown in the pink block of Figure 4 and 5, with the same network configurations and training/testing settings.

– RNNs. The original 3D-R2N2 makes use of the **GRU** (Choy et al., 2016; Kar et al., 2017) unit for feature aggregation and serves as a solid baseline.

– First-order poolings. The widely used **max/mean/ sum** pooling operations (Huang et al., 2018; Paschalidou et al., 2018; Eslami et al., 2018) are all implemented for comparison.

– Higher-order poolings. We also compare with the state-of-the-art higher-order pooling approaches, including bilinear pooling (**BP**) (Lin et al., 2015), and the very recent **MHBN** (Yu et al., 2018) and **SMSO** poolings (Yu and Salzmann, 2018).

Datasets. All approaches are evaluated on four large open datasets.

– $\text{ShapeNet}_{\text{r2n2}}$ Dataset (Choy et al., 2016). The released 3D-R2N2 dataset consists of 13 categories of $43,783$ common objects with synthesized RGB images from the large scale ShapeNet 3D repository (Chang et al., 2015). For each 3D object, 24 images are rendered from different viewing angles circling around each object. The train/test dataset split is $0.8 : 0.2$.

– $\text{ShapeNet}_{\text{lsm}}$ Dataset (Kar et al., 2017). Both LSM and 3D-R2N2 datasets are generated from the same 3D ShapeNet repository (Chang et al., 2015), *i.e.*, they have the same ground truth labels regarding the same object. However, the $\text{ShapeNet}_{\text{lsm}}$ dataset has totally different camera viewing angles and lighting sources for the rendered RGB images. Therefore, we use the $\text{ShapeNet}_{\text{lsm}}$ dataset to evaluate the robustness and generality of all approaches. All images of $\text{ShapeNet}_{\text{lsm}}$ dataset are resized from 224×224 to 127×127 through linear interpolation.

– ModelNet40 Dataset. ModelNet40 (Wu et al., 2015) consists of $12,311$ objects belonging to 40 categories. The 3D models are split into $9,843$ training samples and $2,468$ testing samples. For each 3D model, it is voxelized into a $30 \times 30 \times 30$ shape in (Qi et al., 2016), and 12 RGB images are rendered from different viewing angles. All 3D shapes are zero-padded to be $32 \times 32 \times 32$, and the images are linearly resized from 224×224 to 127×127 for training and testing.

– Blobby Dataset (Wiles and Zisserman, 2017). It contains $11,706$ blobby objects. Each object has 5 RGB images paired with viewing angles and the corresponding silhouettes, which are generated from Cycles in Blender under different lighting sources and texture models.

Metrics. The explicit 3D voxel reconstruction performance of $\text{Base}_{\text{r2n2}}$-AttSets and the competing approaches is evaluated on three datasets: $\text{ShapeNet}_{\text{r2n2}}$, $\text{ShapeNet}_{\text{lsm}}$ and ModelNet40. We use the mean Intersection-over-Union (IoU) (Choy et al., 2016) between predicted 3D voxel grids and their ground truth as the metric. The IoU for an individual voxel grid is formally defined as follows:

$$IoU = \frac{\sum_{i=1}^{L} \left[I(h_i > p) * I(\bar{h}_i) \right]}{\sum_{i=1}^{L} \left[I \left(I(h_i > p) + I(\bar{h}_i) \right) \right]}$$

where $I(\cdot)$ is an indicator function, h_i is the predicted value for the ith voxel, $\bar{h}_i$ is the corresponding ground truth, p is the threshold for voxelization, L is the total number of voxels in a whole voxel grid. As there is no validation split in the above three datasets, to calculate the IoU scores, we independently search the optimal binarization threshold value from $0.2 \sim 0.8$ with a step 0.05 for all approaches for fair comparison. In our experiments, we found that all optimal thresholds of different approaches end up with 0.3 or 0.35.

The implicit 3D shape learning performance of $\text{Base}_{\text{silnet}}$-AttSets and the competing approaches is evaluated on the Blobby dataset. The mean IoU between predicted 2D silhouettes and the ground truth is used as the metric (Wiles and Zisserman, 2017).

Table 1 Group 1: mean IoU for multi-view reconstruction of all 13 categories in ShapeNet$_{r2n2}$ testing split. All networks are firstly trained given only 1 image for each object in Stage 1. The AttSets module is further trained given **2 images** per object in Stage 2, while other competing approaches are fine-tuned given **2 images** per object in Stage 2.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views	20 views	24 views
Base$_{r2n2}$-GRU	0.574	0.608	0.622	0.629	0.633	0.639	0.642	0.642	0.641	0.640
Base$_{r2n2}$-max pooling	0.620	0.651	0.660	0.665	0.666	0.671	0.672	0.674	0.673	0.673
Base$_{r2n2}$-mean pooling	0.632	0.654	0.661	0.666	0.667	0.674	0.676	0.680	0.680	0.681
Base$_{r2n2}$-sum pooling	0.633	0.657	0.665	0.669	0.669	0.670	0.666	0.667	0.666	0.665
Base$_{r2n2}$-BP pooling	0.588	0.608	0.615	0.620	0.621	0.627	0.628	0.632	0.633	0.633
Base$_{r2n2}$-MHBN pooling	0.578	0.599	0.606	0.611	0.612	0.618	0.620	0.623	0.624	0.624
Base$_{r2n2}$-SMSO pooling	0.623	0.654	0.664	0.670	0.672	0.679	0.679	0.682	0.680	0.678
Base$_{r2n2}$-AttSets(Ours)	**0.642**	**0.665**	**0.672**	**0.677**	**0.678**	**0.684**	**0.686**	**0.690**	**0.690**	**0.690**

Table 2 Group 2: mean IoU for multi-view reconstruction of all 13 categories in ShapeNet$_{r2n2}$ testing split. All networks are firstly trained given only 1 image for each object in Stage 1. The AttSets module is further trained given **8 images** per object in Stage 2, while other competing approaches are fine-tuned given **8 images** per object in Stage 2.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views	20 views	24 views
Base$_{r2n2}$-GRU	0.580	0.616	0.629	0.637	0.641	0.649	0.652	0.652	0.652	0.652
Base$_{r2n2}$-max pooling	0.524	0.615	0.641	0.655	0.661	0.674	0.678	0.683	0.684	0.684
Base$_{r2n2}$-mean pooling	0.632	0.657	0.665	0.670	0.672	0.679	0.681	0.685	0.686	0.686
Base$_{r2n2}$-sum pooling	0.580	0.628	0.644	0.656	0.660	0.672	0.677	0.682	0.684	0.685
Base$_{r2n2}$-BP pooling	0.544	0.599	0.618	0.628	0.632	0.644	0.648	0.654	0.655	0.656
Base$_{r2n2}$-MHBN pooling	0.570	0.596	0.606	0.612	0.614	0.621	0.624	0.628	0.629	0.629
Base$_{r2n2}$-SMSO pooling	0.570	0.621	0.641	0.652	0.656	0.668	0.673	0.679	0.680	0.681
Base$_{r2n2}$-AttSets(Ours)	**0.642**	**0.662**	**0.671**	**0.676**	**0.678**	**0.686**	**0.688**	**0.693**	**0.694**	**0.694**

Table 3 Group 3: mean IoU for multi-view reconstruction of all 13 categories in ShapeNet$_{r2n2}$ testing split. All networks are firstly trained given only 1 image for each object in Stage 1. The AttSets module is further trained given **16 images** per object in Stage 2, while other competing approaches are fine-tuned given **16 images** per object in Stage 2.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views	20 views	24 views
Base$_{r2n2}$-GRU	0.579	0.614	0.628	0.636	0.640	0.647	0.651	0.652	0.653	0.653
Base$_{r2n2}$-max pooling	0.511	0.604	0.633	0.649	0.656	0.671	0.678	0.684	0.686	0.686
Base$_{r2n2}$-mean pooling	0.594	0.637	0.652	0.662	0.667	0.677	0.682	0.687	0.688	0.689
Base$_{r2n2}$-sum pooling	0.570	0.629	0.647	0.657	0.664	0.678	0.684	0.690	0.692	0.692
Base$_{r2n2}$-BP pooling	0.545	0.593	0.611	0.621	0.627	0.637	0.642	0.647	0.648	0.649
Base$_{r2n2}$-MHBN pooling	0.570	0.596	0.606	0.612	0.614	0.621	0.624	0.627	0.628	0.629
Base$_{r2n2}$-SMSO pooling	0.580	0.627	0.643	0.652	0.656	0.668	0.673	0.679	0.680	0.681
Base$_{r2n2}$-AttSets(Ours)	**0.642**	**0.660**	**0.668**	**0.673**	**0.676**	**0.683**	**0.687**	**0.691**	**0.692**	**0.693**

5.1 Evaluation on ShapeNet$_{r2n2}$ Dataset

To fully evaluate the aggregation performance and robustness, we train the Base$_{r2n2}$-AttSets and its competing approaches on ShapeNet$_{r2n2}$ dataset. For fair comparison, all networks (the pooling/GRU/AttSets based approaches) are trained according to the proposed two-stage training algorithm.

Training Stage 1. All networks are trained given only 1 image for each object, *i.e.*, $N = 1$ in all training iterations, until convergence. Basically, this is to guarantee all networks are well optimized for the extreme case where there is only one input image.

Training Stage 2. To enable these networks to be more robust for multiple input images, all networks are further trained given more images per object. Particularly, we conduct the following five parallel groups of training experiments.

- Group 1. All networks are further trained given only 2 images for each object, *i.e.*, $N = 2$ in all iterations. As to our Base$_{r2n2}$-AttSets, the well-trained encoder-decoder in previous stage 1 is frozen, and we only optimize the AttSets module according to our FASet algorithm 1. As to the competing approaches, *e.g.*, GRU and all poolings, we turn to fine-tune the whole networks because they do not have separate parameters suitable for special training. To be specific, we use smaller learning rate (1e-5) to carefully train these networks to achieve better performance where $N = 2$ until convergence.
- Group 2/3/4. Similarly, in these three groups of second-stage training experiments, N is set to be 8, 16, 24 separately.
- Group 5. All networks are further trained until convergence, but N is uniformly and randomly sampled from $[1, 24]$ for each object during training. In the above

Table 4 Group 4: mean IoU for multi-view reconstruction of all 13 categories in ShapeNet$_{r2n2}$ testing split. All networks are firstly trained given only 1 image for each object in Stage 1. The AttSets module is further trained given **24 images** per object in Stage 2, while other competing approaches are fine-tuned given **24 images** per object in Stage 2.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views	20 views	24 views
Base$_{r2n2}$-GRU	0.578	0.613	0.627	0.635	0.639	0.647	0.651	0.653	0.653	0.654
Base$_{r2n2}$-max pooling	0.504	0.600	0.631	0.648	0.655	0.671	0.679	0.685	0.688	0.689
Base$_{r2n2}$-mean pooling	0.593	0.634	0.649	0.659	0.663	0.673	0.677	0.683	0.684	0.685
Base$_{r2n2}$-sum pooling	0.580	0.634	0.650	0.658	0.660	0.678	0.682	0.689	0.690	0.691
Base$_{r2n2}$-BP pooling	0.524	0.585	0.609	0.623	0.630	0.644	0.650	0.656	0.659	0.660
Base$_{r2n2}$-MHBN pooling	0.566	0.595	0.606	0.613	0.616	0.624	0.627	0.631	0.632	0.632
Base$_{r2n2}$-SMSO pooling	0.556	0.613	0.635	0.647	0.653	0.668	0.674	0.681	0.682	0.684
Base$_{r2n2}$-AttSets(Ours)	**0.642**	**0.660**	**0.668**	**0.674**	**0.676**	**0.684**	**0.688**	**0.693**	**0.694**	**0.695**

Table 5 Group 5: mean IoU for multi-view reconstruction of all 13 categories in ShapeNet$_{r2n2}$ testing split. All networks are firstly trained given only 1 image for each object in Stage 1. The AttSets module is further trained given random number of images per object in Stage 2, *i.e.*, N is uniformly sampled from [**1, 24**], while other competing approaches are fine-tuned given random number of views per object in Stage 2.

|	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views	20 views	24 views
Base$_{r2n2}$-GRU	0.580	0.615	0.629	0.637	0.641	0.648	0.651	0.651	0.651	0.651
Base$_{r2n2}$-max pooling	0.601	0.638	0.652	0.660	0.663	0.673	0.677	0.682	0.683	0.684
Base$_{r2n2}$-mean pooling	0.598	0.637	0.652	0.660	0.664	0.675	0.679	0.684	0.685	0.686
Base$_{r2n2}$-sum pooling	0.587	0.631	0.646	0.656	0.660	0.672	0.678	0.683	0.684	0.685
Base$_{r2n2}$-BP pooling	0.582	0.610	0.620	0.626	0.628	0.635	0.638	0.641	0.642	0.643
Base$_{r2n2}$-MHBN pooling	0.575	0.599	0.608	0.613	0.615	0.622	0.624	0.628	0.629	0.629
Base$_{r2n2}$-SMSO pooling	0.580	0.624	0.641	0.652	0.657	0.669	0.674	0.679	0.681	0.682
Base$_{r2n2}$-AttSets(Ours)	**0.642**	**0.662**	**0.670**	**0.675**	**0.677**	**0.685**	**0.688**	**0.692**	**0.693**	**0.694**
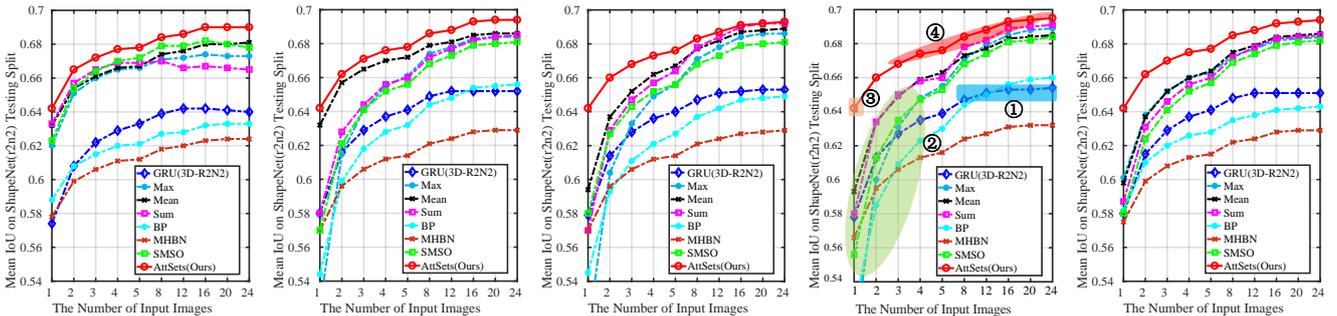
Fig. 6 IoUs of Group 1. **Fig. 7** IoUs of Group 2. **Fig. 8** IoUs of Group 3. **Fig. 9** IoUs of Group 4. **Fig. 10** IoUs of Group 5.

Group 1/2/3/4, N is fixed for each object, while N is dynamic for each object in this Group 5.

The above experiment Groups 1/2/3/4 are designed to investigate how all competing approaches would be further optimized towards the statistics of a fixed N during training, thus resulting in different level of robustness given an arbitrary number of N during testing. By contrast, the paradigm in Group 5 aims at enumerating all possible N values during training. Therefore the overall performance might be more robust regarding an arbitrary number of input images during testing, compared with the above Group 1/2/3/4 experiments.

Testing Stage. All networks trained in above five groups of experiments are separately tested given $N = \{1, 2, 3, 4, 5, 8, 12, 16, 20, 24\}$. The permutations of input images are the same for all different approaches for fair comparison. Note that, we do not test the networks which are only trained in Stage 1, because the AttSets

module is not optimized and the corresponding Base$_{r2n2}$-AttSets is unable to generalize to multiple input images during testing. Therefore, it is meaningless to compare the performance when the network is solely trained on a single image.

Results. Tables 1 ∼ 5 show the mean IoU scores of all 13 categories for experiments of Group 1 ∼ 5, while Figures 6 ∼ 10 show the trends of mean IoU changes in different Groups. Figure 11 shows the estimated 3D shapes in experiment Group 5, with an increasing number of images from 1 to 5 for different approaches.

We notice that the reported IoU scores of ShapeNet data repository in original LSM (Kar et al., 2017) are higher than our scores. However, the experimental settings in LSM (Kar et al., 2017) are quite different from ours in the following two aspects. 1) The original LSM requires both RGB images and the corresponding viewing angles as input, while all our experiments do not. 2) The original LSM dataset has different styles of rendered
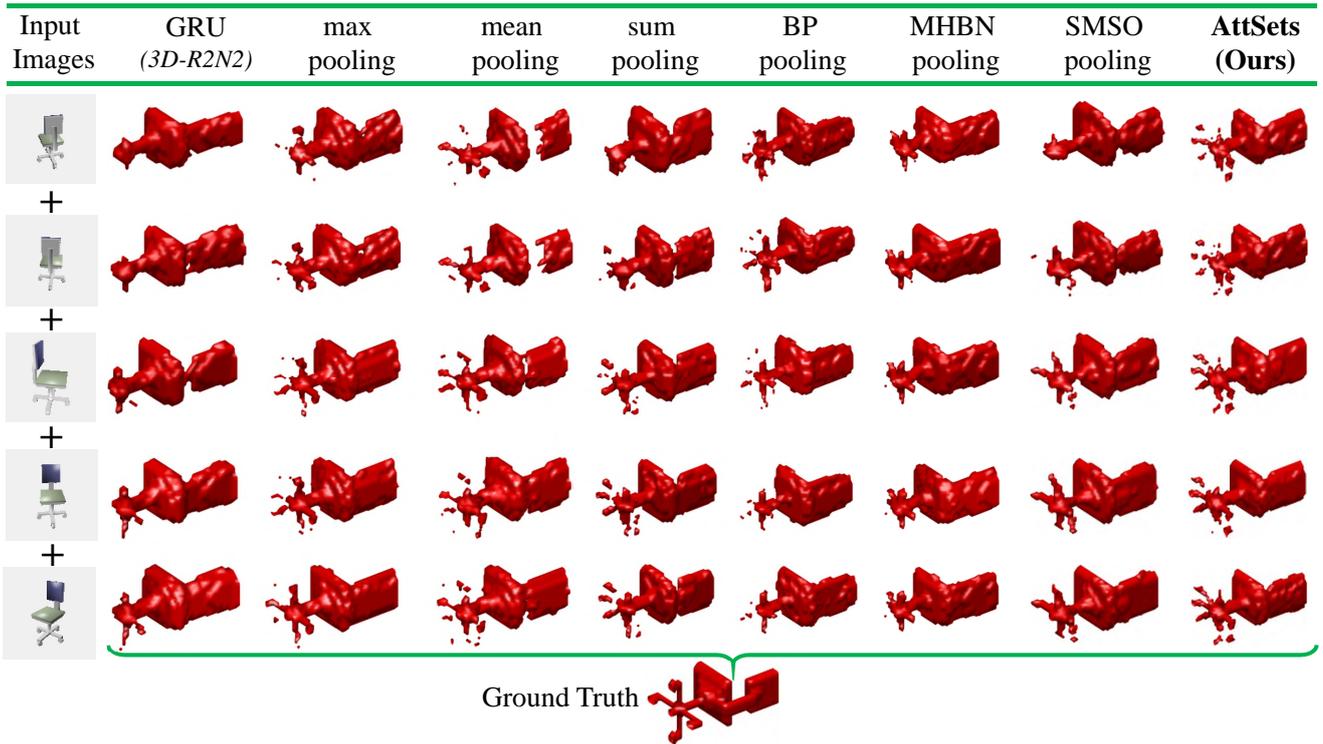
Fig. 11 Qualitative results of multi-view reconstruction from different approaches in experiment Group 5.

Table 6 Per-category mean IoU for single view reconstruction on ShapeNet$_{r2n2}$ testing split.

	plane	bench	cabinet	car	chair	monitor	lamp	speaker	firearm	couch	table	phone	watercraft	mean
Base$_{r2n2}$-GRU	0.530	0.449	0.730	0.807	0.487	0.497	0.391	0.671	0.553	0.631	0.515	0.683	0.535	0.580
Base$_{r2n2}$-max pooling	0.573	0.521	0.755	0.835	0.533	0.544	0.423	0.695	0.587	0.678	0.562	0.710	0.582	0.620
Base$_{r2n2}$-mean pooling	0.582	0.540	0.773	0.837	0.547	0.550	0.440	0.713	0.595	0.695	0.576	0.718	0.593	0.632
Base$_{r2n2}$-sum pooling	0.588	0.536	0.771	0.838	0.554	0.547	0.442	0.710	0.598	0.690	0.575	0.728	0.598	0.633
Base$_{r2n2}$-BP pooling	0.536	0.469	0.747	0.816	0.484	0.499	0.398	0.678	0.556	0.646	0.528	0.681	0.550	0.588
Base$_{r2n2}$-MHBN pooling	0.528	0.451	0.742	0.812	0.471	0.487	0.386	0.677	0.548	0.637	0.515	0.674	0.546	0.578
Base$_{r2n2}$-SMSO pooling	0.572	0.521	0.763	0.833	0.541	0.548	0.433	0.704	0.581	0.682	0.566	0.721	0.581	0.623
OGN	0.587	0.481	0.729	0.816	0.483	0.502	0.398	0.637	0.593	0.646	0.536	0.702	**0.632**	0.596
AORM	**0.605**	0.498	0.715	0.757	0.532	0.524	0.415	0.623	**0.618**	0.679	0.547	0.738	0.552	0.600
PointSet	0.601	0.550	0.771	0.831	0.544	0.552	**0.462**	**0.737**	0.604	**0.708**	**0.606**	**0.749**	0.611	0.640
Base$_{r2n2}$-AttSets(Ours)	0.594	**0.552**	**0.783**	**0.844**	**0.559**	**0.565**	0.445	0.721	0.601	0.703	0.590	0.743	0.601	**0.642**

color images and different train/test splits compared with our experimental settings. Therefore the reported IoU scores in LSM are not directly comparable with ours and we do not include the results in this paper to avoid confusion. Note that, the aggregation module of LSM (Kar et al., 2017), *i.e.*, GRU, is the same as used in 3D-R2N2 (Choy et al., 2016), and is indeed fully evaluated throughout our experiments.

To highlight the performance of single view 3D reconstruction, Table 6 shows the optimal per-category IoU scores for different competing approaches from experiments Group 1 ∼ 5. In addition, we also compare with the state-of-the-art dedicated single view reconstruction approaches including OGN (Tatarchenko et al., 2017), AORM (Yang et al., 2018) and PointSet (Fan et al., 2017) in Table 6. Overall, our AttSets based approach outperforms all others by a large margin for either single

view or multi view reconstruction, and generates much more compelling 3D shapes.

Analysis. We investigate the results as follows:

– The GRU based approach can generate reasonable 3D shapes in all experiments Group 1 ∼ 5 given either few or multiple images during testing, but the performance saturates quickly after being given more images, *e.g.*, 8 views, because the recurrent unit is hardly able to capture features from longer image sequences, as illustrated in Figure 9 ①.

– In Group 1 ∼ 4, all pooling based approaches are able to estimate satisfactory 3D shapes when given a similar number of images as given in training, but they are unlikely to predict reasonable shapes given an arbitrary number of images. For example, in experiment Group 4, all pooling based approaches have inferior IoU scores given only few images as shown in
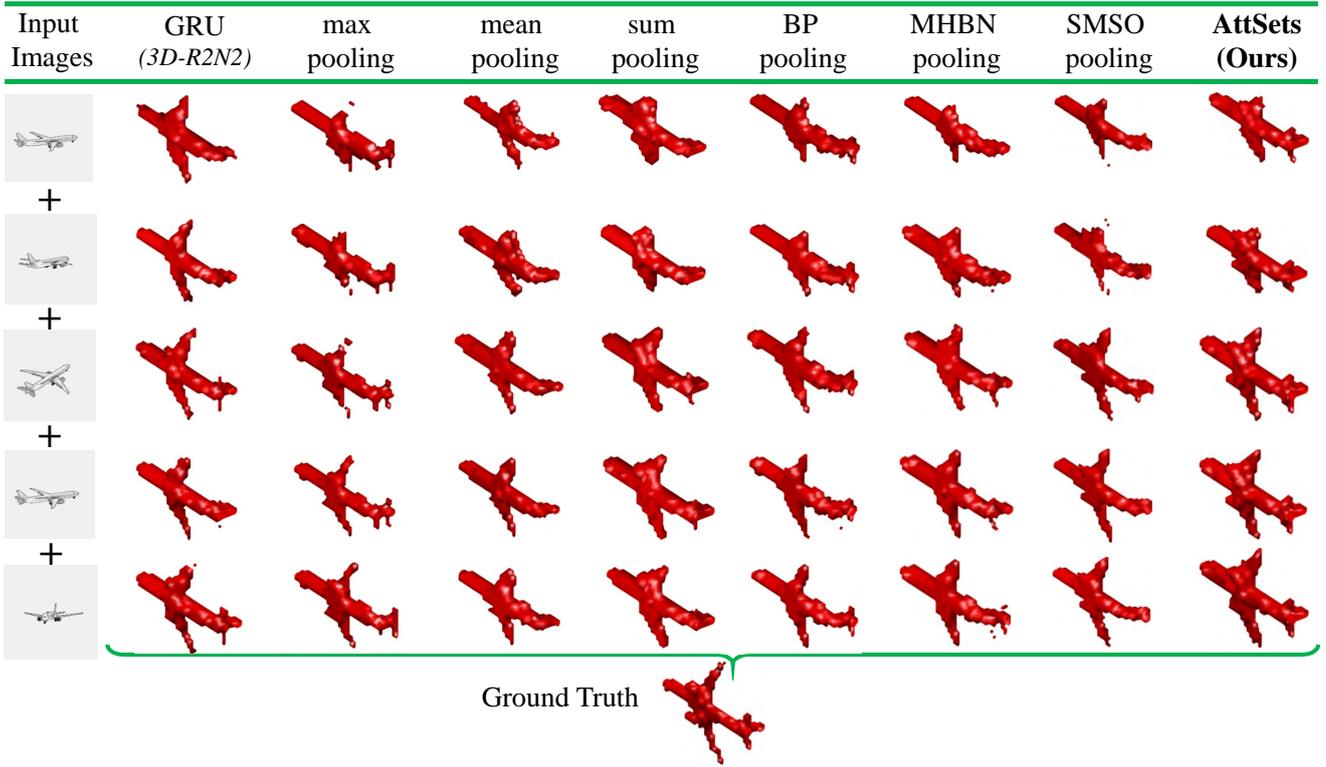
Input Images	GRU (3D-R2N2)	max pooling	mean pooling	sum pooling	BP pooling	MHBN pooling	SMSO pooling	**AttSets (Ours)**

Ground Truth

Fig. 12 Qualitative results of multi-view reconstruction from different approaches in ShapeNet$_{lsm}$ testing split.

Table 7 Mean IoU for multi-view reconstruction of all 13 categories from ShapeNet$_{lsm}$ dataset. All networks are well trained in previous experiment Group 5 of Section 5.1.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views	20 views
Base$_{r2n2}$-GRU	0.390	0.428	0.444	0.454	0.459	0.467	0.470	0.471	0.472
Base$_{r2n2}$-max pooling	0.276	0.388	0.433	0.459	0.473	0.497	0.510	0.515	0.518
Base$_{r2n2}$-mean pooling	0.365	0.426	0.452	0.468	0.477	0.493	0.503	0.508	0.511
Base$_{r2n2}$-sum pooling	0.363	0.421	0.445	0.459	0.466	0.481	0.492	0.499	0.503
Base$_{r2n2}$-BP pooling	0.359	0.407	0.426	0.436	0.442	0.453	0.459	0.462	0.463
Base$_{r2n2}$-MHBN pooling	0.365	0.403	0.418	0.427	0.431	0.441	0.446	0.449	0.450
Base$_{r2n2}$-SMSO pooling	0.364	0.419	0.445	0.460	0.469	0.488	0.500	0.506	0.510
Base$_{r2n2}$-AttSets(Ours)	**0.404**	**0.452**	**0.475**	**0.490**	**0.498**	**0.514**	**0.522**	**0.528**	**0.531**

Table 4 and Figure 9 ②, because the pooled features from fewer images during testing are unlikely to be as general and representative as pooled features from more images during training. Therefore, those models trained on 24 images fail to generalize well to only one image during testing.

– In Group 5, as shown in Table 5 and Figure 10, all pooling based approaches are much more robust compared with Group 1~4, because the networks are generally optimized according to an arbitrary number of images during training. However, these networks tend to have the performance *in the middle*. Compared with Group 4, all approaches in Group 5 tend to have better performance when $N = 1$, while being worse when $N = 24$. Compared with Group 1, all approaches in Group 5 are likely to be better when $N = 24$, while being worse when $N = 1$. Basically, these networks tend to be optimized to learn the *mean* features overall.

– In all experiments Group $1 \sim 5$, all approaches tend to have better performance when given enough input images, *i.e.*, $N = 24$, because more images are able to provide enough information for reconstruction.

– In all experiments Group $1 \sim 5$, our AttSets based approach clearly outperforms all others in either single or multiple view 3D reconstruction and it is more robust to a variable number of input images. Our FASet algorithm completely decouples the base network to learn visual features for accurate single view reconstruction as illustrated in Figure 9 ③, while the trainable parameters of AttSets module are separately responsible for learning attention scores for better multi-view reconstruction as shown in Figure 9 ④. Therefore, the whole network does not suffer from limitations of GRU or pooling approaches, and can achieve better performance for either fewer or more image reconstruction.

Table 8 Group 1: mean IoU for multi-view reconstruction of all 40 categories in ModelNet40 testing split. All networks are firstly trained given only 1 image for each object in Stage 1. The AttSets module is further trained given **12 images** per object in Stage 2, while other competing approaches are fine-tuned given **12 images** per object in Stage 2.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views
$Base_{r2n2}$-GRU	0.344	0.390	0.414	0.430	0.440	0.454	0.464
$Base_{r2n2}$-max pooling	0.393	0.468	0.490	0.504	0.511	0.523	0.525
$Base_{r2n2}$-mean pooling	0.415	0.464	0.481	0.495	0.502	0.515	0.520
$Base_{r2n2}$-sum pooling	0.332	0.441	0.473	0.492	0.500	0.514	0.520
$Base_{r2n2}$-BP pooling	0.431	0.466	0.479	0.492	0.497	0.509	0.515
$Base_{r2n2}$-MHBN pooling	0.423	0.462	0.478	0.491	0.497	0.509	0.515
$Base_{r2n2}$-SMSO pooling	0.441	0.476	0.490	0.500	0.506	0.517	0.520
$Base_{r2n2}$-AttSets(Ours)	**0.487**	**0.505**	**0.511**	**0.517**	**0.521**	**0.527**	**0.529**

Table 9 Group 2: mean IoU for multi-view reconstruction of all 40 categories in ModelNet40 testing split. All networks are firstly trained given only 1 image for each object in Stage 1. The AttSets module is further trained given random number of images per object in Stage 2, *i.e.*, N is uniformly sampled from [**1, 12**], while other competing approaches are fine-tuned given random number of views per object in Stage 2.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views
$Base_{r2n2}$-GRU	0.388	0.421	0.434	0.440	0.444	0.449	0.452
$Base_{r2n2}$-max pooling	0.461	0.489	0.498	0.506	0.509	0.515	0.517
$Base_{r2n2}$-mean pooling	0.455	0.487	0.498	0.507	0.512	0.520	0.523
$Base_{r2n2}$-sum pooling	0.453	0.484	0.494	0.503	0.506	0.514	0.517
$Base_{r2n2}$-BP pooling	0.454	0.479	0.487	0.496	0.499	0.507	0.510
$Base_{r2n2}$-MHBN pooling	0.453	0.480	0.488	0.497	0.500	0.507	0.509
$Base_{r2n2}$-SMSO pooling	0.462	0.488	0.497	0.505	0.509	0.516	0.519
$Base_{r2n2}$-AttSets(Ours)	**0.487**	**0.505**	**0.511**	**0.518**	**0.520**	**0.525**	**0.527**

5.2 Evaluation on ShapeNet$_{lsm}$ Dataset

To further investigate how well the learnt visual features and attention scores generalize across different style of images, we use the well trained networks of previous Group 5 of Section 5.1 to test on the large ShapeNet$_{lsm}$ dataset. Note that, we only borrow the synthesized images from ShapeNet$_{lsm}$ dataset corresponding to the objects in ShapeNet$_{r2n2}$ testing split. This guarantees that all the trained models have never seen either the style of LSM rendered images or the 3D object labels before. The image viewing angles from the original ShapeNet$_{lsm}$ dataset are not used in our experiments, since the Base$_{r2n2}$ network does not require image viewing angles as input. Table 7 shows the mean IoU scores of all approaches, while Figure 12 shows the qualitative results.

Our AttSets based approach outperforms all others given either few or multiple input images. This demonstrates that our Base$_{r2n2}$-AttSets approach does not overfit the training data, but has better generality and robustness over new styles of rendered color images compared with other approaches.

5.3 Evaluation on ModelNet40 Dataset

We train the Base$_{r2n2}$-AttSets and its competing approaches on ModelNet40 dataset from scratch. For fair comparison, all networks (the pooling/GRU/AttSets based approaches) are trained according to the pro-

posed FASet algorithm, which is similar to the two-stage training strategy of Section 5.1.

Training Stage 1. All networks are trained given only 1 image for each object, *i.e.*, $N = 1$ in all training iterations, until convergence. This guarantees all networks are well optimized for single view 3D reconstruction.

Training Stage 2. We further conduct the following two parallel groups of training experiments to optimize the networks for multi-view reconstruction.

– Group 1. All networks are further trained given all 12 images for each object, *i.e.*, $N = 12$ in all iterations, until convergence. As to our Base$_{r2n2}$-AttSets, the well-trained encoder-decoder in previous Stage 1 is frozen, and only the AttSets module is trained. All other competing approaches are fine-tuned using smaller learning rate (1e-5) in this stage.
– Group 2. All networks are further trained until convergence, but N is uniformly and randomly sampled from $[1, 12]$ for each object during training. Only the AttSets module is trained, while all other competing approaches are fine-tuned in this Stage 2.

Testing Stage. All networks trained in above two groups are separately tested given $N = [1, 2, 3, 4, 5, 8, 12]$. The permutations of input images are the same for all different approaches for fair comparison.

Results. Tables 8 and 9 show the mean IoU scores of Groups 1 and 2 respectively, and Figure 13 shows qualitative results of Group 2. The Base$_{r2n2}$-AttSets surpasses all competing approaches by a large margin
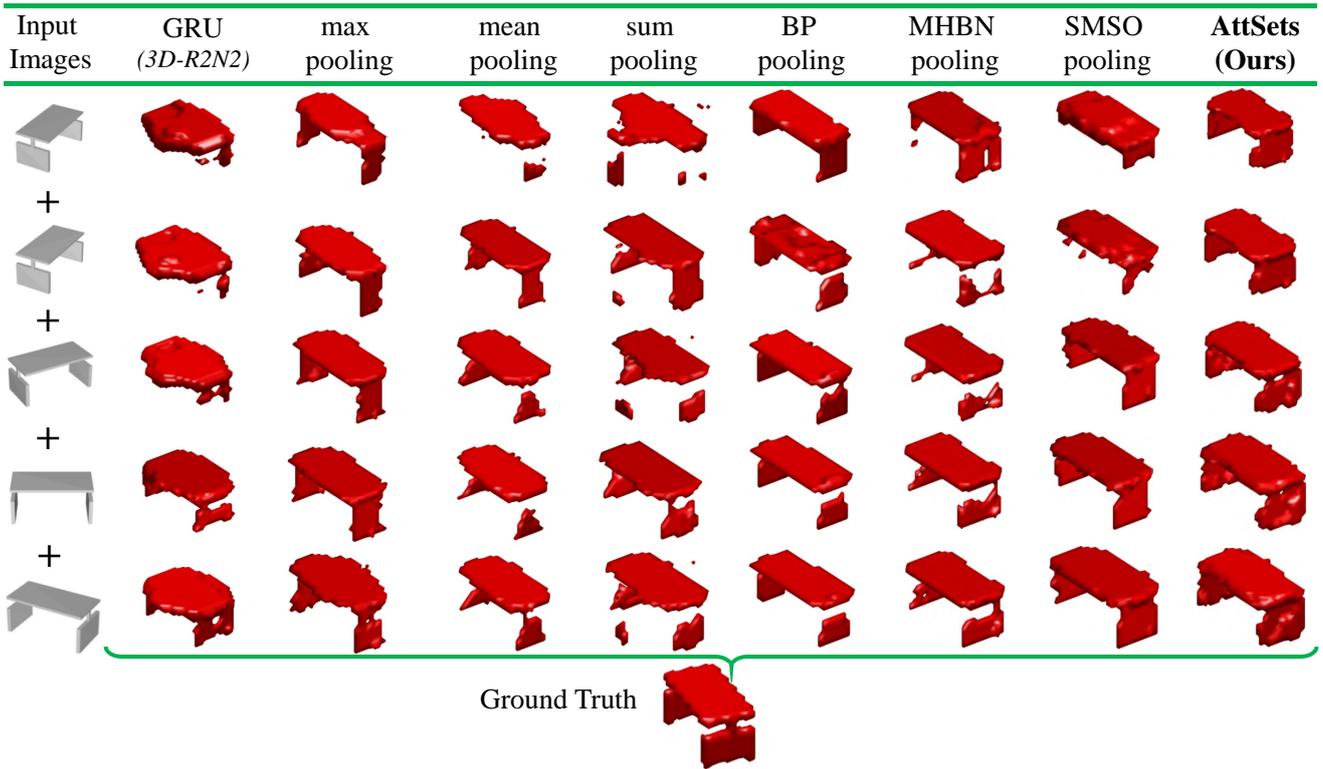
Fig. 13 Qualitative results of multi-view reconstruction from different approaches in ModelNet40 testing split.

Table 10 Group 1: mean IoU for silhouettes prediction on the Blobby dataset. All networks are firstly trained given only 1 image for each object in Stage 1. The AttSets module is further trained given **2 images** per object, *i.e.*, $N=2$, while other competing approaches are fine-tuned given 2 views per object in Stage 2.

	1 view	2 views	3 views	4 views
Base$_{silnet}$-GRU	0.857	0.860	0.860	0.860
Base$_{silnet}$-max pooling	0.922	0.923	0.924	0.924
Base$_{silnet}$-mean pooling	0.920	0.922	0.923	0.924
Base$_{silnet}$-sum pooling	0.913	0.918	0.917	0.916
Base$_{silnet}$-BP pooling	0.908	0.912	0.914	0.914
Base$_{silnet}$-MHBN pooling	0.901	0.904	0.906	0.906
Base$_{silnet}$-SMSO pooling	0.860	0.865	0.865	0.865
Base$_{silnet}$-AttSets(Ours)	**0.924**	**0.931**	**0.933**	**0.935**

Table 11 Group 2: mean IoU for silhouettes prediction on the Blobby dataset. All networks are firstly trained given only 1 image for each object in Stage 1. The AttSets module is further trained given **4 images** per object, *i.e.*, $N=4$, while other competing approaches are fine-tuned given 4 views per object in Stage 2.

	1 view	2 views	3 views	4 views
Base$_{silnet}$-GRU	0.863	0.865	0.865	0.865
Base$_{silnet}$-max pooling	0.923	0.927	0.929	0.929
Base$_{silnet}$-mean pooling	0.923	0.925	0.927	0.927
Base$_{silnet}$-sum pooling	0.902	0.917	0.921	0.924
Base$_{silnet}$-BP pooling	0.911	0.916	0.919	0.920
Base$_{silnet}$-MHBN pooling	0.904	0.908	0.911	0.911
Base$_{silnet}$-SMSO pooling	0.863	0.865	0.865	0.865
Base$_{silnet}$-AttSets(Ours)	**0.924**	**0.932**	**0.936**	**0.937**

for both single and multiple view 3D reconstruction, and all the results are consistent with previous experimental results on both ShapeNet$_{r2n2}$ and ShapeNet$_{lsm}$ datasets.

5.4 Evaluation on Blobby Dataset

In this section, we evaluate the Base$_{silnet}$-AttSets and the competing approaches on the Blobby dataset. For fair comparison, the GRU module is implemented with a single fully connected layer of 160 hidden units, which has similar network capacity with our AttSets based network. All networks (the pooling/GRU/AttSets based approaches) are trained with the proposed two-stage FASet algorithm as follows:

Training Stage 1. All networks are trained given only 1 image together with the viewing angle for each object, *i.e.*, $N=1$ in all training iterations, until convergence. This guarantees the performance of single view shape learning.

Training Stage 2. Another two parallel groups of training experiments are conducted to further optimize the networks for multi-view shape learning.

– Group 1. All networks are further trained given only 2 images for each object, *i.e.*, $N=2$ in all iterations. As to Base$_{silnet}$-AttSets, only the AttSets module is optimized with the well-trained base encoder-decoder being frozen. For fair comparison, all competing ap-
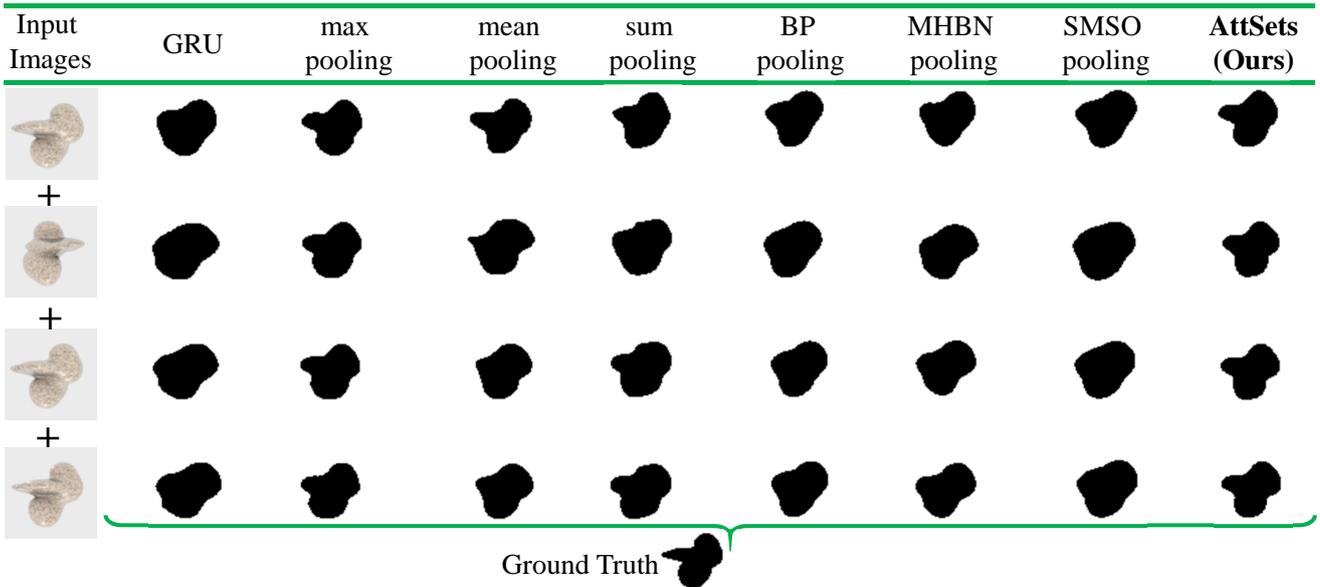
Fig. 14 Qualitative results of silhouettes prediction from different approaches on the Blobby dataset.

proaches are fine-tuned given 2 images per object for better performance where $N = 2$ until convergence.

– Group 2. Similar to the above Group 1, all networks are further trained given all 4 images for each object, *i.e.*, $N=4$, until convergence.

Testing Stage. All networks trained in above two groups are separately tested given $N = [1,2,3,4]$. The permutations of input images are the same for all different networks for fair comparison.

Results. Table 10 and 11 show the mean IoUs of above two groups of experiments and Figure 14 shows the qualitative results of Group 2. Note that, the IoUs are calculated on predicted 2D silhouettes instead of 3D voxels, so they are not numerically comparable with previous experiments on ShapeNet$_{\text{r2n2}}$, ShapeNet$_{\text{lsm}}$, and ModelNet40 datasets. We do not include the IoU scores of the original SilNet (Wiles and Zisserman, 2017), because the original IoU scores are obtained from an end-to-end training strategy. In this paper, we uniformly apply the proposed two-stage FASet training paradigm on all approaches for fair comparison. Our Base$_{\text{silnet}}$-AttSets consistently outperforms all competing approaches for shape learning from either single or multiple views.

5.5 Qualitative Results on Real-world Images

To the best of our knowledge, there is no public real-world dataset for multi-view 3D object reconstruction. Therefore, we manually collect real world images from Amazon online shops to qualitatively demonstrate the generality of all networks which are trained on the synthetic ShapeNet$_{\text{r2n2}}$ dataset in experiment Group 4 of Section 5.1, as shown in Figure 15.

In the meantime, we use these real-world images to qualitatively show the permutation invariance of different approaches. In particular, for each object, we use 6 different permutations in total for testing. As shown in Figure 16, the GRU based approach generates inconsistent 3D shapes given different image permutations. For example, the arm of a chair and the leg of a table can be reconstructed in permutation 1, but fail to be recovered in another permutation. By comparison, all other approaches are permutation invariant, as the results shown in Figure 15.

5.6 Computational Efficiency

To evaluate the computation and memory cost of AttSets, we implement Base$_{\text{r2n2}}$-AttSets and the competing approaches in Python 2.7 and Tensorflow 1.2 with CUDA 9.0 and cuDNN 7.1 as the back-end driver and library. All approaches share the same Base$_{\text{r2n2}}$ network and run in the same Titan X and software environments. Table 12 shows the average time consumption to reconstruct a single 3D object given different number of images. Our AttSets based approach is as efficient as the pooling methods, while Base$_{\text{r2n2}}$-GRU (*i.e.*, 3D-R2N2) takes more time when processing an increasing number of images due to the sequential computation mechanism of its GRU module. In terms of the total trainable weights, the max/mean/sum pooling based approaches have 16.66 million, while AttSets based net has 17.71 million. By contrast, the original 3D-R2N2 has 34.78 million, the BP/MHBN/SMSO have $141.57, 60.78$ and 17.71 million respectively. Overall, our AttSets outperforms the recurrent unit and pooling operations without incurring notable computation and memory cost.
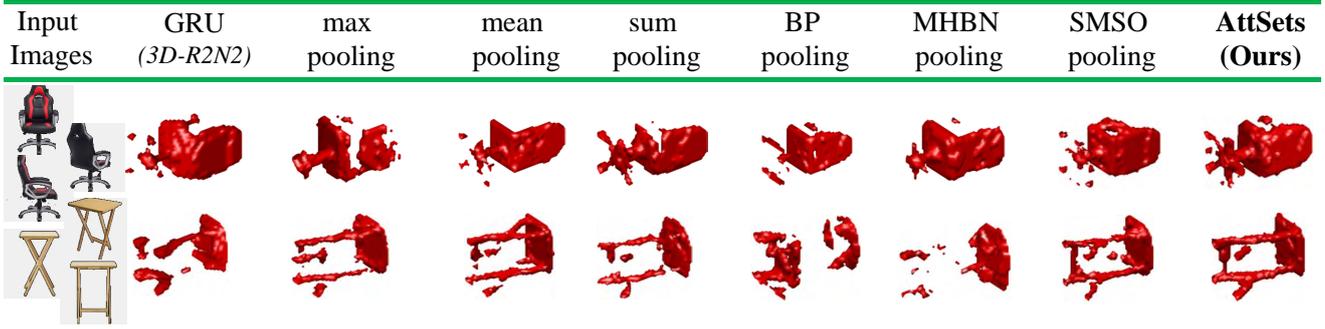
Input Images	GRU *(3D-R2N2)*	max pooling	mean pooling	sum pooling	BP pooling	MHBN pooling	SMSO pooling	**AttSets (Ours)**

Fig. 15 Qualitative results of multi-view 3D reconstruction from real-world images.

Input Images	GRU *(P1)*	GRU *(P2)*	GRU *(P3)*	GRU *(P4)*	GRU *(P5)*	GRU *(P6)*
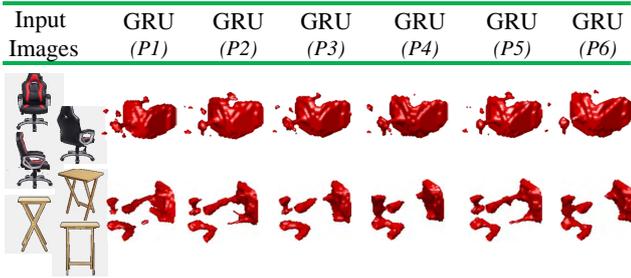
Fig. 16 Qualitative results of inconsistent 3D reconstruction from the GRU based approach.

Table 12 Mean time consumption for a single object (32^3 voxel grid) estimation from different number of images (milliseconds).

number of input images	1	4	8	12	16	20	24
Base_{r2n2}-GRU	6.9	11.2	17.0	22.8	28.8	34.7	40.7
Base_{r2n2}-max pooling	6.4	**10.0**	**15.1**	20.2	**25.3**	**30.2**	**35.4**
Base_{r2n2}-mean pooling	**6.3**	10.1	**15.1**	20.1	**25.3**	30.3	35.5
Base_{r2n2}-sum pooling	6.4	10.1	**15.1**	20.1	**25.3**	30.3	35.5
Base_{r2n2}-BP pooling	6.5	10.5	15.6	20.5	25.7	30.6	35.8
Base_{r2n2}-MHBN pooling	6.5	10.3	15.3	20.3	25.5	30.7	35.7
Base_{r2n2}-SMSO pooling	6.5	10.2	15.3	20.3	25.4	30.5	35.6
Base_{r2n2}-AttSets(Ours)	7.7	11.0	16.3	21.2	26.3	31.4	36.4

Input Images
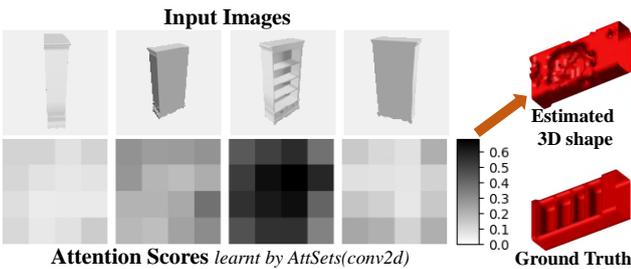

Attention Scores *learnt by AttSets(conv2d)*

Fig. 17 Learnt attention scores for deep feature sets via *conv*2*d* based AttSets.

5.7 Comparison between Variants of AttSets

We further compare the aggregation performance of *fc*, *conv*2*d* and *conv*3*d* based AttSets variants which are shown in Figure 3 in Section 3.4. The *fc* based AttSets net is the same as in Section 5.1. The *conv*2*d* based AttSets is plugged into the middle of the 2D encoder, fusing a $(N, 4, 4, 256)$ tensor into $(1, 4, 4, 256)$, where N is an arbitrary image number. The *conv*3*d* based AttSets

is plugged into the middle of the 3D decoder, integrating a $(N, 8, 8, 8, 128)$ tensor into $(1, 8, 8, 8, 128)$. All other layers of these variants are the same. Both the *conv*2*d* and *conv*3*d* based AttSets networks are trained using the paradigm of experiment Group 4 in Section 5.1. Table 13 shows the mean IoU scores of three variants on ShapeNet$_{r2n2}$ testing split. *fc* and *conv*3*d* based variants achieve similar IoU scores for either single or multi view 3D reconstruction, demonstrating the superior aggregation capability of AttSets. In the meantime, we observe that the overall performance of *conv*2*d* based AttSets net is slightly decreased compared with the other two. One possible reason is that the 2D feature set has been aggregated at the early layer of the network, resulting in features being lost early. Figure 17 visualizes the learnt attention scores for a 2D feature set, *i.e.*, $(N, 4, 4, 256)$ features, via the *conv*2*d* based AttSets net. To visualize 2D feature scores, we average the scores along the channel axis and then roughly trace back the spatial locations of those scores corresponding to the original input. The more visual information the input image has, the higher attention scores are learnt by AttSets for the corresponding latent features. For example, the third image has richer visual information than the first image, so its attention scores are higher. Note that, for a specific base network, there are many potential locations to plug in AttSets and it is also possible to include multiple AttSets modules into the same net. To fully evaluate these factors is out of the scope of this paper.

5.8 Feature-wise Attention *vs*. Element-wise Attention

Our AttSets module is initially designed to learn unique feature-wise attention scores for the whole input deep feature set, and we demonstrate that it significantly improves the aggregation performance over dynamic feature sets in previous Section 5.1, 5.2, 5.3 and 5.4. In this section, we further investigate the advantage of this feature-wise attentive pooling over element-wise attentional aggregation.

For element-wise attentional aggregation, the AttSets module turns to learn a single attention score for each el-

**Table 13** Mean IoU of AttSets variants on all 13 categories in ShapeNet$_{r2n2}$ testing split.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views	20 views	24 views
Base$_{r2n2}$-AttSets (*conv2d*)	**0.642**	0.648	0.651	0.655	0.657	0.664	0.668	0.674	0.675	0.676
Base$_{r2n2}$-AttSets (*conv3d*)	**0.642**	**0.663**	**0.671**	**0.676**	**0.677**	0.683	0.685	0.689	0.690	0.690
Base$_{r2n2}$-AttSets (*fc*)	**0.642**	0.660	0.668	0.674	0.676	**0.684**	**0.688**	**0.693**	**0.694**	**0.695**

Table 14 Mean IoU of all 13 categories in ShapeNet$_{r2n2}$ testing split for feature-wise and element-wise attentional aggregation.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views	20 views	24 views
Base$_{r2n2}$-AttSets (element-wise)	0.642	0.653	0.657	0.660	0.661	0.665	0.667	0.670	0.671	0.672
Base$_{r2n2}$-AttSets (feature-wise)	**0.642**	**0.660**	**0.668**	**0.674**	**0.676**	**0.684**	**0.688**	**0.693**	**0.694**	**0.695**

Table 15 Mean IoU of different training algorithms on all 13 categories in ShapeNet$_{r2n2}$ testing split.

	1 view	2 views	3 views	4 views	5 views	8 views	12 views	16 views	20 views	24 views
Base$_{r2n2}$-AttSets (JoinT)	0.307	0.437	0.516	0.563	0.595	0.639	0.659	0.673	0.677	0.680
Base$_{r2n2}$-AttSets (FASet)	**0.642**	**0.660**	**0.668**	**0.674**	**0.676**	**0.684**	**0.688**	**0.693**	**0.694**	**0.695**
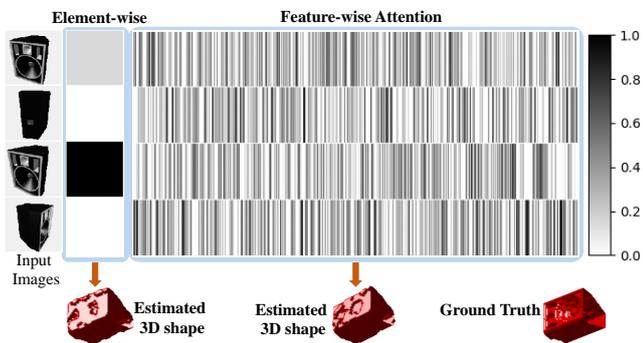
Fig. 18 Learnt attention scores for deep feature sets via element-wise attention and feature-wise attention AttSets.

ement of the feature set $\mathcal{A} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$, followed by the *softmax* normalization and weighted summation pooling. In particular, as shown in previous Figure 2, the shared function $g(\boldsymbol{x}_n, \boldsymbol{W})$ now learns a scalar, instead of a vector, as the attention activation for each input element. Eventually, all features within the same element are weighted by a learnt common attention score. Intuitively, the original feature-wise AttSets tends to be fine-grained aggregation, while the element-wise AttSets learns to coarsely aggregate features.

Following the same training settings of experiment Group 4 in Section 5.1, we conduct another group of experiment on ShapeNet$_{r2n2}$ dataset for element-wise attentional aggregation. Table 14 compares the mean IoU for 3D object reconstruction through feature-wise and element-wise attentional aggregation. Figure 18 shows an example of the learnt attention scores and the predicted 3D shapes. As expected, the feature-wise attention mechanism clearly achieves better aggregation performance compared with the coarsely element-wise approach. As shown in Figure 18, the element-wise attention mechanism tends to focus on few images, while completely ignoring others. By comparison, the feature-wise AttSets learns to fuse information from all images, thus achieving better aggregation performance.

5.9 Significance of FASet Algorithm

In this section, we investigate the impact of FASet algorithm by comparing it with the standard end-to-end joint training (JoinT). Particularly, in JoinT, all parameters Θ_{base} and Θ_{att} are jointly optimized with a single loss. Following the same training settings of experiment Group 4 in Section 5.1, we conduct another group of experiment on ShapeNet$_{r2n2}$ dataset under the JoinT training strategy. As its IoU scores shown in Table 15, the JoinT training approach tends to optimize the whole net regarding the training multi-view batches, thus being unable to generalize well for fewer images during testing. Basically, the network itself is unable to dedicate the base layers to learning visual features, while the AttSets module to learning attention scores, if it is not trained with the proposed FASet algorithm. The theoretical reason is discussed previously in Section 4.1. The FASet algorithm may also be applicable to other learning based aggregation approaches, as long as the aggregation module can be decoupled from the base encoder/decoder.

6 Conclusion

In this paper, we present AttSets module and FASet training algorithm to aggregate elements of deep feature sets. AttSets together with FASet has powerful permutation invariance, computation efficiency, robustness and flexible implementation properties, along with the theory and extensive experiments to support its performance for multi-view 3D reconstruction. Both quantitative and qualitative results explicitly show that AttSets significantly outperforms other widely used aggregation approaches. Nevertheless, all of our experiments are dedicated to multi-view 3D reconstruction. It would be interesting to explore the generality of AttSets and FASet over other set-based tasks, especially the tasks which constantly take multiple elements as input.

References

Bahdanau D, Cho K, Bengio Y (2015) Neural Machine Translation by Jointly Learning to Align and Translate. International Conference on Learning Representations

Bengio Y, Simard P, Frasconi P (1994) Learning Long-term Dependencies with Gradient Descent is Difficult. IEEE Transactions on Neural Networks 5(2):157–166

Cadena C, Carlone L, Carrillo H, Latif Y, Scaramuzza D, Neira J, Reid ID, Leonard JJ (2016) Past, Present, and Future of Simultaneous Localization and Mapping: Towards the Robust-Perception Age. IEEE Transactions on Robotics 32(6):1309–1332

Cao YP, Liu ZN, Kuang ZF, Kobbelt L, Hu SM (2018) Learning to Reconstruct High-quality 3D Shapes with Cascaded Fully Convolutional Networks. European Conference on Computer Vision pp 616–633

Chang AX, Funkhouser T, Guibas L, Hanrahan P, Huang Q, Li Z, Savarese S, Savva M, Song S, Su H, Xiao J, Yi L, Yu F (2015) ShapeNet: An Information-Rich 3D Model Repository. arXiv 1512.03012

Choy CB, Xu D, Gwak J, Chen K, Savarese S (2016) 3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction. European Conference on Computer Vision

Curless B, Levoy M (1996) A Volumetric Method for Building Complex Models from Range Images. Conference on Computer Graphics and Interactive Techniques pp 303–312

Dong W, Wang Q, Wang X, Zha H (2018) PSDF Fusion: Probabilistic Signed Distance Function for On-the-fly 3D Data Fusion and Scene Reconstruction. European Conference on Computer Vision pp 714–730

Eslami SA, Rezende DJ, Besse F, Viola F, Morcos AS, Garnelo M, Ruderman A, Rusu AA, Danihelka I, Gregor K, Reichert DP, Buesing L, Weber T, Vinyals O, Rosenbaum D, Rabinowitz N, King H, Hillier C, Botvinick M, Wierstra D, Kavukcuoglu K, Hassabis D (2018) Neural scene representation and rendering. Science 360(6394):1204–1210

Fan H, Su H, Guibas L (2017) A Point Set Generation Network for 3D Object Reconstruction from a Single Image. IEEE Conference on Computer Vision and Pattern Recognition pp 605–613

Gardner A, Kanno J, Duncan CA, Selmic RR (2017) Classifying Unordered Feature Sets with Convolutional Deep Averaging Networks. arXiv 1709.03019

Girdhar R, Ramanan D (2017) Attentional Pooling for Action Recognition. International Conference on Neural Information Processing Systems pp 33–44

Hartley R, Zisserman A (2004) Multiple View Geometry in Computer Vision. Cambridge University Press

He K, Zhang X, Ren S, Sun J (2016) Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition pp 770–778

Huang PH, Matzen K, Kopf J, Ahuja N, Huang JB (2018) DeepMVS: Learning Multi-view Stereopsis. IEEE Conference on Computer Vision and Pattern Recognition pp 2821–2830

Ilse M, Tomczak JM, Welling M (2018) Attention-based Deep Multiple Instance Learning. International Conference on Machine Learning pp 2127–2136

Ionescu C, Vantzos O, Sminchisescu C (2015) Matrix backpropagation for deep networks with structured layers. IEEE International Conference on Computer Vision pp 2965–2973

Ji M, Gall J, Zheng H, Liu Y, Fang L (2017a) SurfaceNet: An End-to-end 3D Neural Network for Multiview Stereopsis. IEEE International Conference on Computer Vision pp 2326–2334

Ji P, Li H, Dai Y, Reid I (2017b) "Maximizing rigidity" Revisited: a Convex Programming Approach for Generic 3D Shape Reconstruction from Multiple Perspective Views. IEEE International Conference on Computer Vision pp 929–937

Jie Hu, Li Shen, Albanie S, Sun G, Wu E (2018) Squeeze-and-Excitation Networks. IEEE Conference on Computer Vision and Pattern Recognition pp 7132–7141

Kar A, Häne C, Malik J (2017) Learning a Multi-View Stereo Machine. International Conference on Neural Information Processing Systems pp 364–375

Kolen JF, Kremer SC (2001) Gradient Flow in Recurrent Nets: The Difficulty of Learning Long Term Dependencies. A Field Guide to Dynamical Recurrent Networks

Kumar S, Dai Y, Li H (2017) Monocular Dense 3D Reconstruction of a Complex Dynamic Scene from Two Perspective Frames. IEEE International Conference on Computer Vision pp 4649–4657

Li H, Xiong P, An J, Wang L (2018) Pyramid Attention Network for Semantic Segmentation. arXiv 1805.10180

Lin TY, Maji S (2017) Improved Bilinear Pooling with CNNs. British Machine Vision Conference

Lin TY, Roychowdhury A, Maji S (2015) Bilinear CNN models for fine-grained visual recognition. IEEE International Conference on Computer Vision pp 1449–1457

Lin TY, Maji S, Koniusz P (2018) Second-order Democratic Aggregation. European Conference on Computer Vision pp 620–636

Liu X, Kumar BV, Yang C, Tang Q, You J (2018) Dependency-aware Attention Control for Unconstrained Face Recognition with Image Sets. European Conference on Computer Vision pp 548–565

Lowe DG (2004) Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60(2):91–110

Martin E, Cundy C (2018) Parallelizing Linear Recurrent Neural Nets over Sequence Length. International Conference on Learning Representations

Nakka KK, Salzmann M (2018) Deep Attentional Structured Representation Learning for Visual Recognition. British Machine Vision Conference

Ozyesil O, Voroninski V, Basri R, Singer A (2017) A Survey of Structure from Motion. Acta Numerica 26:305–364

Paschalidou D, Ulusoy AO, Schmitt C, Van Gool L, Geiger A (2018) RayNet: Learning Volumetric 3D Reconstruction with Ray Potentials. IEEE Conference on Computer Vision and Pattern Recognition pp 3897–3906

Qi CR, Su H, Nießner M, Dai A, Yan M, Guibas LJ (2016) Volumetric and Multi-View CNNs for Object Classification on 3D Data. IEEE Conference on Computer Vision and Pattern Recognition pp 5648–5656

Qi CR, Su H, Mo K, Guibas LJ (2017) PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. IEEE Conference on Computer Vision and Pattern Recognition pp 652–660

Raffel C, Ellis DPW (2016) Feed-Forward Networks with Attention Can Solve Some Long-Term Memory Problems. International Conference on Learning Representations Workshops

Riegler G, Ulusoy AO, Bischof H, Geiger A (2017) OctNetFusion: Learning Depth Fusion from Data. International Conference on 3D Vision pp 57–66

Rodríguez P, Gonfaus JM, Cucurull G, Roca FX, Gonzàlez J (2018) Attend and Rectify: a Gated Attention Mechanism for Fine-Grained Recovery. European Conference on Computer Vision pp 349–364

Sarafianos N, Xu X, Kakadiaris IA (2018) Deep Imbalanced Attribute Classification using Visual Attention Aggregation. European Conference on Computer Vision pp 680–697

Su H, Maji S, Kalogerakis E, Learned-Miller E (2015) Multi-view Convolutional Neural Networks for 3D Shape Recognition. IEEE International Conference on Computer Vision pp 945–953

Tatarchenko M, Dosovitskiy A, Brox T (2017) Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs. IEEE International Conference on Computer Vision pp 2088–2096

Triggs B, McLauchlan PF, Hartley RI, Fitzgibbon AW (1999) Bundle Adjustment - A Modern Synthesis. International Workshop on Vision Algorithms

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention Is All You Need. International Conference on Neural Information Processing Systems

Vinyals O, Bengio S, Kudlur M (2015) Order Matters: Sequence to Sequence for Sets. International Conference on Learning Representations

Wiles O, Zisserman A (2017) SilNet : Single- and Multi-View Reconstruction by Learning from Silhouettes. British Machine Vision Conference

Wiles O, Zisserman A (2018) Learning to Predict 3D Surfaces of Sculptures from Single and Multiple Views. International Journal of Computer Vision pp 1–21

Wu Z, Song S, Khosla A, Yu F, Zhang L, Tang X, Xiao J (2015) 3D ShapeNets: A Deep Representation for Volumetric Shapes. IEEE Conference on Computer Vision and Pattern Recognition pp 1912–1920

Xu K, Ba JL, Kiros R, Cho K, Courville A, Salakhutdinov R, Zemel RS, Bengio Y (2015) Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. International Conference on Machine Learning pp 2048–2057

Yang X, Wang Y, Wang Y, Yin B, Zhang Q, Wei X, Fu H (2018) Active Object Reconstruction Using a Guided View Planner. International Joint Conference on Artificial Intelligence pp 4965–4971

Yang Z, He X, Gao J, Deng L, Smola A (2016) Stacked Attention Networks for Image Question Answering. IEEE Conference on Computer Vision and Pattern Recognition pp 21–29

Yao Y, Luo Z, Li S, Fang T, Quan L (2018) MVSNet: Depth Inference for Unstructured Multi-view Stereo. European Conference on Computer Vision pp 767–783

Yu K, Salzmann M (2018) Statistically Motivated Second Order Pooling. European Conference on Computer Vision pp 600–616

Yu T, Meng J, Yuan J (2018) Multi-view Harmonized Bilinear Network for 3D Object Recognition. IEEE Conference on Computer Vision and Pattern Recognition pp 186–194

Zaheer M, Kottur S, Ravanbakhsh S, Poczos B, Salakhutdinov R, Smola A (2017) Deep Sets. International Conference on Neural Information Processing Systems

Zhang H, Goodfellow I, Metaxas D, Odena A (2018) Self-Attention Generative Adversarial Networks. arXiv 1805.08318

Zhu Y, Wang J, Xie L, Zheng L (2018) Attention-based Pyramid Aggregation Network for Visual Place Recognition. ACM International Conference on Multimedia