Group-Free 3D Object Detection via Transformers

Ze Liu[1,2*] Zheng Zhang[2†] Yue Cao[2] Han Hu[2] Xin Tong[2]
[1]University of Science and Technology of China
liuze@mail.ustc.edu.cn
[2]Microsoft Research Asia
{zhez,yuecao,hanhu,xtong}@microsoft.com

Abstract

Recently, directly detecting 3D objects from 3D point clouds has received increasing attention. To extract object representation from an irregular point cloud, existing methods usually take a point grouping step to assign the points to an object candidate so that a PointNet-like network could be used to derive object features from the grouped points. However, the inaccurate point assignments caused by the hand-crafted grouping scheme decrease the performance of 3D object detection.

In this paper, we present a simple yet effective method for directly detecting 3D objects from the 3D point cloud. Instead of grouping local points to each object candidate, our method computes the feature of an object from all the points in the point cloud with the help of an attention mechanism in the Transformers [42], where the contribution of each point is automatically learned in the network training. With an improved attention stacking scheme, our method fuses object features in different stages and generates more accurate object detection results. With few bells and whistles, the proposed method achieves state-of-the-art 3D object detection performance on two widely used benchmarks, ScanNet V2 and SUN RGB-D. The code and models are publicly available at https://github.com/zeliu98/Group-Free-3D

1. Introduction

3D object detection on point cloud simultaneously localizes and recognizes 3D objects from a 3D point set. As a fundamental technique for 3D scene understanding, it plays an important role in many applications such as autonomous driving, robotics manipulation, and augmented reality.

Different from 2D object detection that works on 2D regular images, 3D object detection takes irregular and sparse point cloud as input, which makes it difficult to directly apply techniques used for 2D object detection techniques. Recent studies [27, 35, 26, 51] infer the object location and extract object features directly from the irregular input point cloud for object detection. In these methods, a point grouping step is required to assign a group of points to each object candidate, and then computes object features from assigned groups of points. For this purpose, different grouping strategies have been investigated. Frustum-PointNet [27] applies the Frustum envelop of a 2D proposal box for point grouping. Point R-CNN [35] groups points within the 3D box proposals to objects. VoteNet [26] determines the group as the points which vote to the same (or spatially-close) center point. Although these hand-crafted grouping schemes facilitate 3D object detection, the complexity and diversity of objects in real scene may lead to wrong point assignments (shown in Figure. 1) and degrade the 3D object detection performance.

In this paper, we propose a simple yet effective technique for detecting 3D objects from point clouds without the

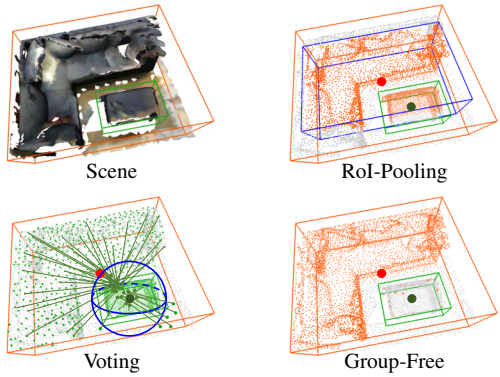

Figure 1. With the heuristic point grouping step, all points in blue box of RoI-Pooling or blue ball of Voting are assigned and aggregated to derive the object features, resulting in wrong assignments. Our group-free based approach automatically learn the contribution of all points to each object, which has ability to alleviate the drawbacks of the hand-crafted grouping.

*This work is done when Ze Liu is an intern at MSRA.
†Contact person

handcrafted grouping step. The key idea of our approach is to take all points in the point cloud for computing features for each object candidate, in which the contribution of each point is determined by an automatically learned attention module. Based on this idea, we adapt the Transformer to fit for 3D object detection, which could simultaneously model the object-object and object-pixel relationships, and extract the object features without handcrafted grouping.

To further release the power of the transformer architecture, we improve it in two aspects. First, we propose to iteratively refine the prediction of objects by updating the spatial encoding of objects in different stages, while the original application of Transformers adopt the fixed spatial encoding. Second, we use the ensemble of detection results predicted at all stages during inference, instead of only using the results in the last stage as the final results. These two modifications efficiently improve the performance of 3D object detection with few computational overheads.

We validate our method with both ScanNet V2 [6] and SUN RGB-D [52] benchmarks. Results show that our method is effective and robust to the quality of initial object candidates, where even a simple farthest point sampling approach has been able to produce strong results on ScanNet V2 and SUN RGB-D benchmarks. For the SUN RGB-D dataset, our method with the ensemble scheme results in significant performance improvement (+3.8 mAP@0.25). With few bells and whistles, the proposed approach achieved state-of-the-art performance on both benchmarks.

We believe that our method also advocates a strong potential by using the attention mechanism or Transformers for point cloud modeling, as it naturally addresses the intrinsic irregular and sparse distribution problems encountered by 3D point clouds. This is contrary to 2D image modeling, where such modeling tools mainly act as a challenger or a complementary component to the mature grid modeling tools such as ConvNets variants [16, 32, 46] and RoI Align [2, 5].

2. Related Work

Grid Projection/Voxelization based Detection Early 3D object detection approaches project point cloud to 2D grids or 3D voxels so that the advanced convolutional networks can be directly applied. A set of methods [18, 19, 50] project point cloud to the bird's view and then employ 2D ConvNets for learning features and generate 3D boxes. These methods are mainly applied for the outdoor scenes in autonomous driving where objects are distributed on a horizontal plane so that their projections on the bird-view are occlusion-free. Note these approaches also need to address the irregular and sparse distribution issues of the 2D point projections, usually by pixelization. Other methods [4, 48] project point clouds into frontal views and then apply 2D ConvNets for object detection. Voxel-based methods [37, 53] convert points into voxels and employ 3D ConvNets to generate features for 3D box generation. All these projection/voxelization based methods suffer from quantization errors. The voxel-based methods also suffer from the large memory and computational cost of 3D convolutions.

Point based Detection Recent methods directly process point clouds for 3D object detection. A core task of these methods is to compute object features from the irregularly and sparsely distributed points. All existing methods first assign a group of points to each object candidate and then compute object features from each point group. Frustum-PointNet [27] groups points by the 3D Frustum envelope of a 2D box detected using an RGB object detector, and applies a PointNet on the grouped points to extract object features for 3D box prediction. Point R-CNN [35] directly computes 3D box proposals, where the points within this 3D box are used for object feature extraction. PV-RCNN [34] leverages the voxel representation to complement the point-based representation in Point R-CNN [35] for 3D object detection and achieves better performance.

VoteNet [26] groups points according to their voted centers and extract object features from grouped points by the PointNet. Some follow-up works further improve the point group generation procedure [51] or the object box localization and recognition procedure [3].

Our method is also a point-based detection approach. Unlike existing point-based approaches, our method involves all the points for computing the features of each object candidate by an attention module. We also stack the attention modules to iteratively refine the detection results while maintaining the simplicity of our method.

Network architecture for Point Cloud A large set of network architectures [38, 12, 29, 9, 47, 23, 44, 39, 45, 28, 30, 33, 43, 40, 17, 1, 41, 49, 10, 22] have been proposed for various point cloud based learning tasks. [13] provides a good taxonomy and review of all these architectures, and discussing all of them is beyond the scope of this paper. Our method can take any point cloud architecture as the backbone network for computing the point features. We adopt PointNet++ [30] used in previous methods [26, 25, 51] in our implementation for a fair comparison.

Attention Mechanism/Transformer in NLP and 2D Image Recognition The attention-based Transformer is the dominant network architecture for the learning tasks in the field of NLP [42, 7, 21]. They have been also applied in the field of 2D image recognition [16, 32, 46] as a strong competitor to the dominant grid/dense modeling tools such as ConvNets and RoI-Align. The most related works in 2D

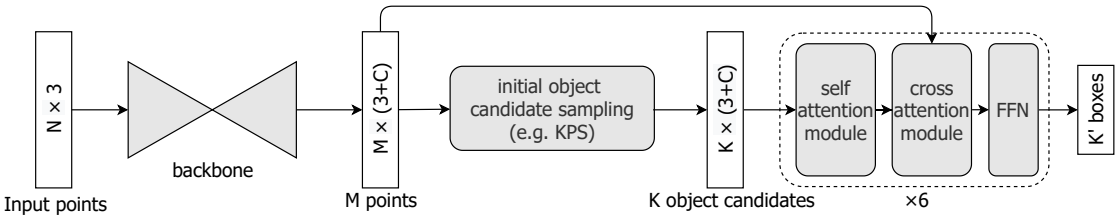

Figure 2. This figure illustrates the simple architecture of our approach, including three major components: a backbone network to extract feature representations for each point in the point cloud, a sampling method to generate initial object candidates, and stacked attention modules to extract and refine object representations from all points.

image recognition to this paper are those who apply the attention mechanism or Transformer architectures into 2D object detection [15, 11, 5, 2].

Among these approaches, our method is most similar to [2], which also applies a Transformer architecture for 2D object detection. However, we found that directly applying this method to point clouds leads to significantly lower performance than our approach in 3D object detection task. On the one hand, this is caused by the new technologies we proposed, and on the other hand, it probably because our method better integrated the advantage of traditional 3D detection framework. We discussed these factors in Sec. 4.6.

Our approach improves the Transformer models to better adapt the 3D object detection task, including the update of object query locations in the multi-stage iterative box prediction, and an ensemble of detection results of stages. Although the attention mechanisms still have a certain performance gap compared to the dominant convolution-based methods in other tasks, we found that this architecture may well address the point grouping issue for object detection on point clouds. As a result, we advocate a strong potential of this architecture for modeling irregular 3D point clouds.

3. Methodology

In 3D object detection on point clouds, we are given a set of N points $S \in \mathbb{R}^{N\times 3}$ and the goal is to produce a set of 3D (oriented) bounding boxes with categorization scores $\mathcal{O}_S$ to cover all ground-truth objects. Our overall architecture is illustrated in Figure 2, involving three major components: a *backbone network* to extract feature representations for each point in point clouds, a *sampling method* to generate initial object candidates, and *stacked attention modules* to extract and refine object representations from all points.

Backbone Architecture While our framework can leverage any point cloud network to extract point features, we adopt PointNet++ [30] as the backbone network for a fair comparison with the recent methods [26, 51].

The backbone network receives a point cloud of N points (i.e. 2048) as input. We follow the encoder-decoder architecture in [30] to first down-sample the point cloud input into $8\times$ resolution (i.e. 256 points) through four stages of set abstraction layers, and then up-sample it to the resolution of $2\times$ (i.e. 1024 points) by feature propagation layers. The network will produce a C-channel vector representation for each point on the $2\times$ resolution, denoted as $\{\mathbf{z}_i\}_{i=1}^{M}$, which are then used in the *initial object candidates sampling* module and the *stacked attention* modules. In the following parts, we will first describe these two modules in detail, and then present the loss function and head design for this framework.

3.1. Initial Object Candidate Sampling

While object detection on 2D images usually adopts data-independent anchor boxes as initial object candidates, it is generally intractable or impractical for 3D object detection to apply this simple top-down strategy, as the number of anchor boxes in 3D search space is too huge to handle. Instead, we follow recent practice [35, 26] to sample initial object candidates directly from the points on a point cloud, by a bottom-up way.

We consider three simple strategies to sample initial object candidates from a point cloud:

- *Farthest Point Sampling (FPS)*. The FPS approach has been widely adopted to generate a point cloud from a 3D shape or to down-sample the point clouds to a lower resolution. This method can be also employed to sample initial candidates from a point cloud. Firstly, a point is randomly sampled from the point cloud. Then the farthest point to the already-chosen point set is iteratively selected until the number of chosen points meets the candidate budget. Though it is simple, we show in experiments that this sampling approach along with our framework has been able to be comparable to the previous state-of-the-art 3D object detectors.

- *k-Closest Points Sampling (KPS)*. In this approach, we classify each point on a point cloud to be a real object candidate or not. The label assignment in training follows this rule: a point is assigned positive if it is inside a ground-truth object box and it is one of the k-closest points to the object center. In inference, the initial candidates are selected according to the classification score of the point.

- *KPS with non-maximal suppression (KPS-NMS)*. Built on the above *KPS* method, we introduce an additional non-maximal suppression (NMS) step, which iteratively removes spatially close object candidates, to improve the recall of sampled object candidates given a fixed number of objects, following the common practice in 2D object detection. In addition to the *objectness* scores, we predict also the object center that each point belongs to, where the NMS is conducted accordingly. Specifically, the candidates locating within a radius of the selected object center will be suppressed. The radius is set to 0.05 in our experiments.

In experiments, we will demonstrate that our framework has strong compatibility with the choice of these sampling approaches, mainly ascribed to the robust object feature extraction approach described in the next subsection (see Table 3). **We use the *KPS* approach by default**, due to its better performance than the *FPS* approach, and the same effectiveness as the more complex *KPS-NMS* approach.

3.2. Iterative Object Feature Extraction and Box Prediction by Transformer Decoder

With the initial object candidates generated by a sampling approach, we adopt the Transformer as the decoder to leverage all points on a point cloud to compute the object feature of each candidate. The multi-head attention network is the foundation of Transformer, it has three input sets: query set, key set and value set. Usually, the key set and value set are different projections of the same set of elements. Given a query set $\{\mathbf{q}_i\}$ and a common element set $\{\mathbf{p}_k\}$ of key set and value set, the output feature of the multi-head attention of each query element is the aggregation of the values that weighted by the attention weights, formulated as:

$$\text{Att}(\mathbf{q}_i, \{\mathbf{p}_k\}) = \sum_{h=1}^{H} W_h \Big(\sum_{k=1}^{K} A_{i,k}^h \cdot V_h \mathbf{p}_k\Big), \tag{1}$$

$$A_{i,k}^h = \frac{\exp[(Q_h \mathbf{q}_i)^T (U_h \mathbf{p}_k)]}{\sum_{k=1}^{K} \exp[(Q_h \mathbf{q}_i)^T (U_h \mathbf{p}_k)]} \tag{2}$$

where h indexes over attention heads, A_h is the attention weight, Q_h, V_h, U_h, W_h indicate the query projection weight, value projection weight, key projection weight, and output projection weight, respectively.

While the standard Transformer predicts the sentence of a target language sequentially in an auto-regressive way, our Transformer computes object features and predicts 3D object boxes in parallel. The Transformer consists of several stacked multi-head *self-attention* and multi-head *cross-attention* modules, as illustrated in Figure 3.

Denote the input point features at stage l as $\{\mathbf{z}_i^{(l)}\}_{i=1}^M$ and the object features at the same stage as $\{\mathbf{o}_i^{(l)}\}_{i=1}^K$. A self-attention module models interaction between object features, formulated as:

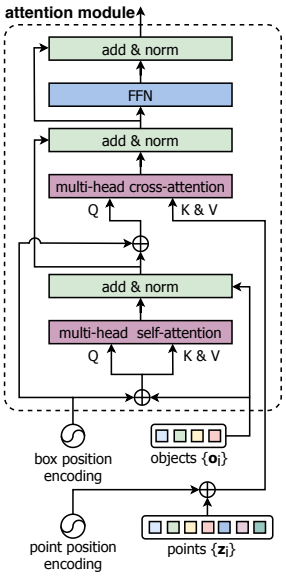

Figure 3. Architecture of the attention module.

$$\text{Self-Att}(\mathbf{o}_i^{(l)}, \{\mathbf{o}_j^{(l)}\}) = \text{Att}(\mathbf{o}_i^{(l)}, \{\mathbf{o}_j^{(l)}\}), \tag{3}$$

A cross-attention module leverages point features to compute object features, formulated as:

$$\text{Cross-Att}(\mathbf{o}_i^{(l)}, \{\mathbf{z}_j^{(l)}\}) = \text{Att}(\mathbf{o}_i^{(l)}, \{\mathbf{z}_j^{(l)}\}), \tag{4}$$

where the notations are similar to those in Eq. (3). After the object feature are updated through the self-attention module and cross attention module, a feed-forward network (FFN) is then applied to further transformed feature of each object.

There are a few differences compared to the original Transformer decoders, as described below.

Iterative Object Box Prediction and Spatial Encoding The original Transformer adopts a fixed spatial encoding for all of the stacked attention modules, indicating the indices of each word. The application of Transformers to 2D object detection [2] instantiate the spatial encoding (object prior) as a learnable weight. During inference, the spatial encoding is fixed and same for any images.

In this work, we propose to refine the spatial encodings of an object candidate stage by stage. Specifically, we predict the 3D box locations and categories at each decoder stage, and the predicted location of a box in one stage will be used to produce the refined spatial encoding of the same object, the refined spatial encoding vector is then added to the output feature of this decoder stage and fed into the next stage. The spatial encodings of an object and a point are computed by applying independent linear layers on the parameterization vector of a 3D box (x, y, z, l, h, w) and a point (x, y, z), respectively. In the experiments, we

will show this approach can improve the mAP@0.25 and mAP@0.5 by 1.6 and 5.0 on the ScanNet V2 benchmark, compared to the approach without iterative refinement.

Ensemble from Multi-Stage Predictions Another difference is that we ensemble the predictions of different stages to produce final detection results, while previous methods usually adopt the output of the last stage as the final results. Concretely, the detection results of different stages are combined and they together go through an NMS (IoU threshold of 0.25) procedure to generate the final object detection results. We find this approach can significantly improve the performance of some benchmarks, e.g. +3.8 mAP@0.25 on the SUN RGB-D dataset. Also note the overhead of this ensembling approach is marginal, mainly ascribed to the multi-stage nature of the Transformer decoder.

3.3. Heads and Loss Functions

Decoder Head We apply head networks on all decoder stages, with each mostly following the setting in [26]. There are 5 prediction tasks: objectness prediction with a binary focal loss [20] $\mathcal{L}_{\text{obj}}$, box classification with a cross entropy loss $\mathcal{L}_{\text{cls}}$, center offset prediction with a smooth-L1 loss $\mathcal{L}_{\text{center\_off}}$, size classification with a cross entropy loss $\mathcal{L}_{\text{sz\_cls}}$, and size offset prediction with a smooth-L1 loss $\mathcal{L}_{\text{sz\_off}}$. Also, all 5 prediction tasks are obtained by a shared 2-layer MLP and an independent linear layer.

The loss of l-th decoder stage is the combination of these 5 loss terms by weighted summation:

$$\mathcal{L}^{(l)}_{\text{decoder}} = \beta_1\mathcal{L}^{(l)}_{\text{obj}} + \beta_2\mathcal{L}^{(l)}_{\text{cls}} + \beta_3\mathcal{L}^{(l)}_{\text{center\_off}} + \beta_4\mathcal{L}^{(l)}_{\text{sz\_cls}} + \beta_5\mathcal{L}^{(l)}_{\text{sz\_off}}, \tag{5}$$

where the balancing factors are set default as $\beta_1 = 0.5$, $\beta_2 = 0.1$, $\beta_3 = 1.0$, $\beta_4 = 0.1$ and $\beta_5 = 0.1$. The losses on all decoder stages are averaged to form the final loss:

$$\mathcal{L}_{\text{decoder}} = \frac{1}{L}\sum_{l=1}^{L}\mathcal{L}^{(l)}_{\text{decoder}}. \tag{6}$$

Sampling Head The head designs and the loss functions of the sampling module are similar to those of the decoders. There are two differences: firstly, the box classification task is not involved; secondly, the objectness task follows the label assignment as described in Sec. 3.1. Our final loss is the sum of decoder and sampling heads:

$$\mathcal{L} = \mathcal{L}_{\text{decoder}} + \mathcal{L}_{\text{sampler}} \tag{7}$$

4. Experiments

4.1. Datasets and Evaluation Protocol

We validate our approach on two widely-used 3D object detection datasets: ScanNet V2 [6] and SUN RGB-D [36], and we follow the standard data splits [26] for them both.

ScanNet V2 [6] is constructed from an 3D reconstruction dataset of indoor scenes by enriched annotations. It consists of 1513 indoor scenes and 18 object categories. The annotations of per-point instance, semantic labels, and 3D bounding boxes are provided. We follow a standard evaluation protocol [26] by using mean Average Precision(mAP) under different IoU thresholds, without considering the orientation of bounding boxes.

SUN RGB-D [36] is a single-view RGB-D dataset for 3D scene understanding, consisting of ∼5K indoor RGB and depth images. The annotation consists of per-point semantic labels and oriented bounding object bounding boxes of 37 object categories. The standard mean Average Precision is used as evaluation metrics and the evaluation is reported on the 10 most common categories, following [26].

4.2. Implementation Details

ScanNet V2 We follow recent practice [26, 31] to use PointNet++ as default backbone network for a fair comparison. The backbone has 4 set abstraction layers and 2 feature propagation layers. For each set abstraction layer, the input point cloud is sub-sampled to 2048, 1024, 512, and 256 points with the increasing receptive radius of 0.2, 0.4, 0.8, and 1.2, respectively. Then, two feature propagation layers successively up-sample the points to 512 and 1024. More training details are given in Appendix.

SUN RGB-D The implementation mostly follow [26]. We use 20k points as input for each point cloud. The network architecture and the data augmentation are the same as that for ScanNet V2. As the orientation of the 3D box is required in evaluation, we include an additional orientation prediction branch for all decoder stages. More training details are given in Appendix.

4.3. System-level Comparison

In this section, we compare with previous state-of-the-arts on ScanNet V2 and SUN RGB-D. Since previous works [26, 24] usually report the best results of multiple times on training and testing in the system-level comparison, we report both best results and average results[1]

ScanNet V2 The results are shown in Table 1. With the same backbone network of a standard PointNet++, the proposed approach achieves 67.3 mAP@0.25 and 48.9 mAP@0.5 using 6 decoder stages and 256 object candidates, which is 2.8 and 5.5 better than previous best results using the same backbones. By more decoder stages as 12, the gap increases to 6.3 on mAP@0.5.

With stronger backbones and more sampled object candidates, i.e. 2× more channels and 512 candidates, the

[1]We train each setting 5 times and test each training trial 5 times. The average performance of these 25 trials is reported to account for algorithm randomness.

methods	backbone	mAP@0.25	mAP@0.5
HGNet [3]	GU-net	61.3	34.4
GSDN [14]	MinkNet	62.8	34.8
3D-MPA [8]	MinkNet	64.2	49.2
VoteNet [26][2]	PointNet++	62.9	39.9
MLCVNet [31]	PointNet++	64.5	41.4
H3DNet [51]	PointNet++	64.4	43.4
H3DNet [51]	4×PointNet++	67.2	48.1
Ours (L6, O256)	PointNet++	67.3 (66.3)	48.9 (48.5)
Ours (L12, O256)	PointNet++	67.2 (66.6)	49.7 (49.0)
Ours (L12, O256)	PointNet++w2×	68.8 (67.7)	52.1 (50.6)
Ours (L12, O512)	PointNet++w2×	**69.1** (68.6)	**52.8** (51.8)

Table 1. System level comparison on ScanNet V2 with state-of-the-arts. The main comparison is based on the best results of multiple experiments between different methods, and the number within the bracket is the average result.
Notations: 4×PointNet++ denotes 4 individual PointNet++; PointNet++w2× denotes the backbone width is expanded by 2 times; L denotes the decoder depth, and O denotes the number of object candidates, e.g. Ours (L6, O256) denotes a model with 6-layer decoder(i.e. 6 attention modules) and 256 object candidates.

methods	backbone	inputs	mAP@0.25	mAP@0.5
VoteNet [26][2]	PointNet++	point	59.1	35.8
MLCVNet [31]	PointNet++	point	59.8	-
HGNet [3]	GU-net	point	61.6	-
H3DNet [51]	4×PointNet++	point	60.1	39.0
imVoteNet [25]*	PointNet++	point+RGB	**63.4**	-
Ours (L6, O256)	PointNet++	point	63.0 (62.6)	**45.2** (44.4)

Table 2. System level comparison on SUN RGB-D with state-of-the-arts. The main comparison is based on the best results of multiple experiments between different methods, and the number within the bracket is the average result. *imVoteNet use RGB images as addition inputs.

sampling method	mAP@0.25	mAP@0.5
FPS	64.5	46.2
KPS-NMS	65.8	**48.7**
KPS	**66.3**	48.5

Table 3. Ablation study on applying different sampling strategies.

k	mAP@0.25	mAP@0.5
1	65.7	**48.7**
2	65.8	48.3
4	**66.3**	48.5
6	66.1	48.4

Table 4. Ablation study on different values of k in KPS strategy.

iterative	position encoding	mAP@0.25	mAP@0.5
	none	64.7	43.4
	center+size	64.6	43.5
✓	center	65.2	47.5
✓	center+size	**66.3**	**48.5**

Table 5. Ablation study on the effectiveness of iterative box prediction.

performance of the proposed approach is improved to 69.1 mAP@0.25 and 52.8 mAP@0.5, outperforming previous best method by a large margin.

SUN RGB-D We also compare the proposed approach with previous state-of-the-arts on the SUN RGB-D dataset, which is another widely used 3D object detection benchmark. In this dataset, the ensemble approach over multiple stages is used by default during inference. The results are shown in Table. 2. Our base model achieves 63.0 on mAP@0.25 and 45.2 on mAP@0.5, which outperforms all previous state-of-the-arts that only use the point cloud. In particular, it outperforms the H3DNet on mAP@0.5 by 6.2.

4.4. Ablation Study

In this section, we validate our key designs on ScanNet V2. If not specified, all models have 6 attention modules, 256 sampled candidates, and are equipped with the proposed iterative object prediction approach. In evaluation, we report the average performance of 25 trials by default.

Sampling Strategy We first ablate the effects of different sampling strategies in Table. 3. It shows that our approach performs well by using different sampling strategies. It also works well in a wide range of hyper-parameters, such as k in the *KPS* sampling approach (see Table. 4).

These results indicate the robustness of our framework for choosing different sampling approaches.

Iterative Box Prediction Table 5 ablates several design

[2]We report the results of MMDetection3D(https://github.com/open-mmlab/mmdetection3d) instead of the official paper, which reported 46.8 mAP@0.25 and 24.7 mAP@0.5 on ScanNet V2, and 57.7 mAP@0.25 and 32.0 mAP@0.5 on SUN RGB-D.

# of layers	mAP@0.25	mAP@0.5
0	63.3	40.7
1	64.8	43.9
2	66.0	45.6
3	66.4	46.6
4	66.2	47.9
5	66.3	48.3
6	**66.3**	**48.5**

Table 6. Ablation study on the performance of iterative box prediction with different decoder layers.

ensemble	ScanNet V2		SUN RGB-D	
	mAP@0.25	mAP@0.5	mAP@0.25	mAP@0.5
	66.3	48.5	59.2	43.3
✓	**66.4**	**48.7**	**63.0**	**45.2**

Table 7. Ablation study on the effectiveness of multi-stage ensemble.

choices for iterative box prediction. With a naive iterative method where no spatial encoding is involved in the decoder stages, the approach shows reasonably good performance of 64.7 mAP@0.25 and 43.4 mAP@0.25, likely because the location information may have been implicitly included in the input object features. Actually, an additional fixed position encoding does not improve detection performance (64.6 mAP@0.25 and 43.5 mAP@0.5).

By refining the encodings of the box location stage by stage, the localization ability of the approach is significantly improved of the 4.1 points gains on the mAP@0.5 metric over the naive implementation (47.5 vs. 43.4). Also, more detailed spatial encoding by both box center and size is beneficial, compared to that only encodes box centers (66.3 vs. 65.2 on mAP@0.25 and 48.5 vs. 47.5 on mAP@0.5).

Table. 6 shows the performance of iterative box prediction with different decoder stages. More stages can bring significant performance improvement, especially in the mAP@0.5. Compared with not applying any attention modules, our 6-stage model performs better on mAP@0.25 and mAP@0.5 by 3.0 and 7.8, respectively.

Ensemble Multi-stage Predictions Each decoder stage of our approach will predict a set of 3D boxes. It is natural to ensemble these results of different decoder stages in expecting better final detection results. Table 7 shows the results, where significantly performance improvements are observed on SUN RGB-D (+3.8 mAP@0.25 and +1.9 mAP@0.5) and maintained performance on ScanNet V2. We hypothesize that it is because the point clouds of SUN RGB-D have lower quality than those of ScanNet V2: SUN RGB-D adopts real RGB-D signals to generate point clouds that many objects have missing parts due to occlusion, while the ScanNet V2 generate point clouds from 3D shape meshes which are more complete. The ensemble method can boost the performance more on real 3D scenes.

Comparison with Group-based Approaches Aggregating point features through RoI-Pooing, or according to the voted centers are two typical handcrafted grouping strategies [35, 26] in 3D object detection. We refer these two grouping strategies as baselines and compare with them. For a fair comparison, we only switch the feature aggregation mechanism while all other settings (e.g. the 6-stage decoder) remain unchanged. More details are in Appendix. Table 8 show the results. Although RoI-Pooling outperforms than the voting approach, it is still worse than our group-free approach by 1.2 points on mAP@0.25 and 4.1 points on mAP@0.5.

method	mAP@0.25	mAP@0.5
RoI-Pooing	65.1	44.4
Voting	64.2	44.1
Ours	**66.3**	**48.5**

Table 8. Comparison with grouping-based approaches.

method	backbone	mAP		frames/s
		0.25	0.5	
MLCVNet [31]	PointNet++	64.5	41.4	5.44
H3DNet [51]	4×PointNet++	67.2	48.1	3.76
Ours (L6, O256)	PointNet++	67.3	48.9	**6.71**
Ours (L12, O256)	PointNet++	67.2	49.7	5.70
Ours (L12, O256)	PointNet++w2×	68.8	52.1	5.23
Ours (L12, O512)	PointNet++w2×	**69.1**	**52.8**	5.17

Table 9. Comparison on realistic inference speed on ScanNet V2.

4.5. Inference Speed

The computational complexity of the attention model is determined by the number of points in a point cloud and the number of sampled object candidates. In our approach, only a small number of object candidates are sampled, which makes the cost of the attention model insignificant. With our default setting (256 object candidates, 1024 output points), stacking one attention model brings 0.95 GFLOPs, which is quite light compared to the backbone.

In addition, the realistic inference speed of our method is also very competitive, compared to other state-of-the-art methods. For a fair comparison, all experiments are run on the same workstation (single Titan-XP GPU, 256G RAM, and Xeon E5-2650 v3) and environment (Ubuntu-16.04, Python 3.6, Cuda-10.1, and PyTorch-1.3.1). The official code of other methods is used for evaluation. The batch size of all experiments is set to 1 (i.e. single image). The results are shown in Table. 9. Our method achieves better performance and also higher inference speed.

4.6. Comparison with DETR

DETR [2] is a pioneer work that applies the Transformer to 2D object detection. Compared with DETR, our method involves more domain knowledge, such as the data-dependent initial object candidate generation, where DETR uses a data-independent object prior to representing each

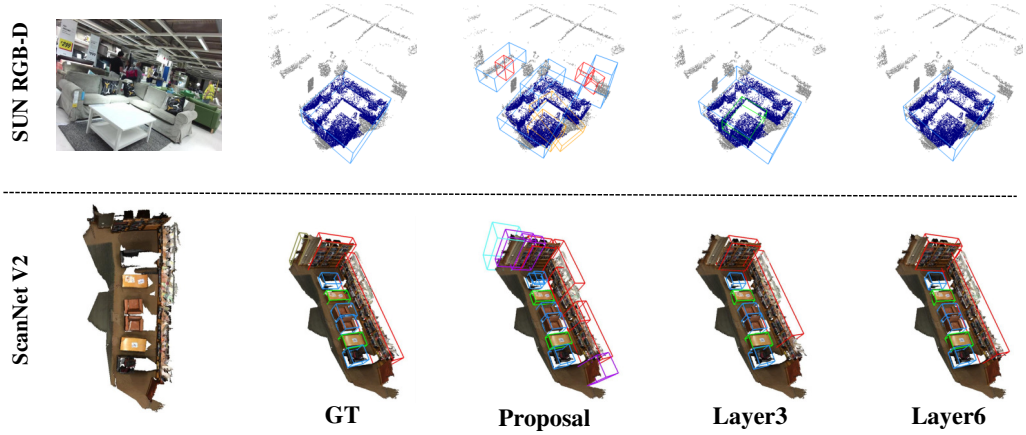

Figure 4. Qualitative results of different decoder stages. The first row is the results on SUN RGB-D, and the second row is the results on ScanNet V2. The color of bounding boxes represents their category.

method	epoch	mAP@0.25	mAP@0.5
DETR	400	39.6	21.4
DETR+KPS	400	59.6	41.0
DETR+KPS+iter pred	400	59.9	42.9
DETR+KPS+iter pred	1200	61.8	45.2
Ours	400	**66.3**	**48.5**

Table 10. The comparison between DETR and our method on ScanNet V2. *KPS* represent *k-Closest Points Sampling*, *iter pred* represents iterative prediction.

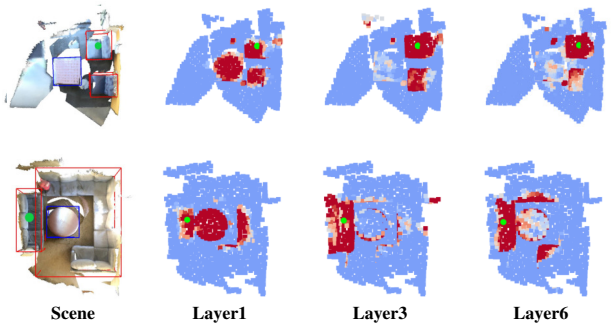

Figure 5. Visualizations on cross-attention weight in different decoder stages. The green point represents the reference object candidates. The redder color represent higher attention weight.

object candidate and is automatically learned without explicit supervision. Moreover, there is no iterative refinement on spatial encodings in DETR as in our approach. We evaluate these differences in 3D object detection. For a fair comparison, the backbone and decoder heads used in DETR are the same as in ours. We carefully tune the hyper-parameters for DETR and chose the best setting in comparison.

The results are shown in Table 10. With the same training length of 400 epochs, DETR achieves 39.6 mAP@0.25 and 21.4 mAP@0.5, significantly worse than our method. We guess it is mainly because of optimization difficulty by the data-independent object representation. The fixed spatial encoding also may contribute to inferior performance. In fact, the performance can be improved significantly by bridging these differences, reaching 59.9 mAP@0.25 and 42.9 mAP@0.5 using the same training epochs, and 61.8 mAP@0.25 and 45.2 mAP@0.5 by longer training.

The remaining performance gap is due to the difference in ground-truth assignments, where DETR adopts a set loss to automatically determine the assignments by detection losses and our approach manually assigns object candidates to ground-truths. This assignment may also be difficult for a network to learn.

4.7. Qualitative Results

Fig. 4 illustrates the qualitative results on both ScanNet V2 and SUN RGB-D. As the decoder networks go deeper, the more accurate detection results are observed.

Fig. 5 visualizes the learned cross-attention weights of different decoder stages. We could observe that the model of the lower stage always focuses on the surrounding points without considering the geometry. With the refinement, the model of the higher stage could focus more on the geometry and extract more high-quality object features.

5. Conclusion

In this paper, we present a simple yet effective 3D object detector based on the attention mechanism in Transformers. Unlike previous methods that require a grouping step for object feature computation, this detector is group-free which computes object features from all points in a point cloud, with the contribution of each point automatically determined by the attention modules. The proposed method achieves state-of-the-art performance on ScanNet V2 and SUN RGB-D benchmarks.

References

[1] Matan Atzmon, Haggai Maron, and Yaron Lipman. Point convolutional neural networks by extension operators. *arXiv preprint arXiv:1803.10091*, 2018. 2

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *arXiv preprint arXiv:2005.12872*, 2020. 2, 3, 4, 7

[3] Jintai Chen, Biwen Lei, Qingyu Song, Haochao Ying, Danny Z Chen, and Jian Wu. A hierarchical graph network for 3d object detection on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 392–401, 2020. 2, 6

[4] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017. 2

[5] Cheng Chi, Fangyun Wei, and Han Hu. Relationnet++: Bridging visual representations for object detection via transformer decoder. *arXiv preprint arXiv:2010.15831*, 2020. 2, 3

[6] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017. 2, 5

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[8] Francis Engelmann, Martin Bokeloh, Alireza Fathi, Bastian Leibe, and Matthias Nießner. 3d-mpa: Multi-proposal aggregation for 3d semantic instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9031–9040, 2020. 6

[9] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 264–272, 2018. 2

[10] Fabian Groh, Patrick Wieschollek, and Hendrik PA Lensch. Flex-convolution. In *Asian Conference on Computer Vision*, pages 105–122. Springer, 2018. 2

[11] Jiayuan Gu, Han Hu, Liwei Wang, Yichen Wei, and Jifeng Dai. Learning region features for object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 381–395, 2018. 3

[12] Haiyun Guo, Jinqiao Wang, Yue Gao, Jianqiang Li, and Hanqing Lu. Multi-view 3d object retrieval with deep embedding network. *IEEE Transactions on Image Processing*, 25(12):5526–5537, 2016. 2

[13] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 2

[14] JunYoung Gwak, Christopher Choy, and Silvio Savarese. Generative sparse detection networks for 3d single-shot object detection. *arXiv preprint arXiv:2006.12356*, 2020. 6

[15] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *CVPR*, pages 3588–3597, 2018. 3

[16] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3464–3473, 2019. 2

[17] Varun Jampani, Martin Kiefel, and Peter V Gehler. Learning sparse high dimensional filters: Image filtering, dense crfs and bilateral neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4452–4461, 2016. 2

[18] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018. 2

[19] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 641–656, 2018. 2

[20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5

[21] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 2

[22] Ze Liu, Han Hu, Yue Cao, Zheng Zhang, and Xin Tong. A closer look at local aggregation operators in point cloud analysis. *arXiv preprint arXiv:2007.01294*, 2020. 2

[23] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015. 2

[24] MMDetection3D. open-mmlab/mmdetection3d. 5

[25] Charles R Qi, Xinlei Chen, Or Litany, and Leonidas J Guibas. Imvotenet: Boosting 3d object detection in point clouds with image votes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4404–4413, 2020. 2, 6

[26] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9277–9286, 2019. 1, 2, 3, 5, 6, 7

[27] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 918–927, 2018. 1, 2

[28] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on*

Computer Vision and Pattern Recognition, pages 652–660, 2017. 2

[29] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016. 2

[30] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017. 2, 3

[31] Xie Qian, Lai Yu-kun, Wu Jing, Wang Zhoutao, Zhang Yiming, Xu Kai, and Wang Jun. Mlcvnet: Multi-level context votenet for 3d object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 5, 6, 7

[32] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. *arXiv preprint arXiv:1906.05909*, 2019. 2

[33] Yiru Shen, Chen Feng, Yaoqing Yang, and Dong Tian. Mining point cloud local structures by kernel correlation and graph pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4548–4557, 2018. 2

[34] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 2

[35] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019. 1, 2, 3, 7

[36] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 567–576, 2015. 5

[37] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 808–816, 2016. 2

[38] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015. 2

[39] 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. 2

[40] Gusi Te, Wei Hu, Amin Zheng, and Zongming Guo. Rgcnn: Regularized graph cnn for point cloud segmentation. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 746–754. ACM, 2018. 2

[41] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6411–6420, 2019. 2

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

[43] Chu Wang, Babak Samari, and Kaleem Siddiqi. Local spectral graph convolution for point set feature learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–66, 2018. 2

[44] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):72, 2017. 2

[45] Peng-Shuai Wang, Chun-Yu Sun, Yang Liu, and Xin Tong. Adaptive o-cnn: A patch-based deep representation of 3d shapes. *ACM Transactions on Graphics (TOG)*, 37(6):1–11, 2018. 2

[46] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. 2

[47] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 2

[48] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2345–2353, 2018. 2

[49] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 87–102, 2018. 2

[50] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018. 2

[51] Zaiwei Zhang, Bo Sun, Haitao Yang, and Qixing Huang. H3dnet: 3d object detection using hybrid geometric primitives. *arXiv preprint arXiv:2006.05682*, 2020. 1, 2, 3, 6, 7

[52] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014. 2

[53] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018. 2