A Hierarchical Graph Network for 3D Object Detection on Point Clouds

Jintai Chen[1]*, Biwen Lei[1]*, Qingyu Song[1]*, Haochao Ying[1], Danny Z. Chen[2], Jian Wu[1]✉

[1]Zhejiang University, Hangzhou, 310027, China

[2]Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556, USA

{JTigerChen,biwen1996,qingyusong,haochaoying}@zju.edu.com, dchen@nd.edu, wujian2000@zju.edu.cn

Abstract

3D object detection on point clouds finds many applications. However, most known point cloud object detection methods did not adequately accommodate the characteristics (e.g., sparsity) of point clouds, and thus some key semantic information (e.g., shape information) is not well captured. In this paper, we propose a new graph convolution (GConv) based hierarchical graph network (HGNet) for 3D object detection, which processes raw point clouds directly to predict 3D bounding boxes. HGNet effectively captures the relationship of the points and utilizes the multi-level semantics for object detection. Specially, we propose a novel shape-attentive GConv (SA-GConv) to capture the local shape features, by modelling the relative geometric positions of points to describe object shapes. An SA-GConv based U-shape network captures the multi-level features, which are mapped into an identical feature space by an improved voting module and then further utilized to generate proposals. Next, a new GConv based Proposal Reasoning Module reasons on the proposals considering the global scene semantics, and the bounding boxes are then predicted. Consequently, our new framework outperforms state-of-the-art methods on two large-scale point cloud datasets, by ∼4% mean average precision (mAP) on SUN RGB-D and by ∼3% mAP on ScanNet-V2.

1. Introduction

3D object detection on point clouds has many applications, such as autonomous driving, fault detection for parts, housekeeping robots, and augmented reality. Since point clouds lie in irregular space and can be sparse, known methods (e.g., convolutional neural networks) designed for grid-structured data did not perform well on point clouds (e.g., see discussion in [2]). Many methods have been proposed for 3D object detection on point clouds, such as projection based methods [35, 4], volumetric convolution based methods [19, 8], and PointNet based methods [29, 30]. The former two types tried to stiffly transform point cloud data into grid-structured data, and the latter aggregated features without explicitly considering the geometric positions of points.

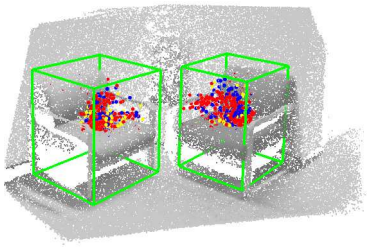

Figure 1. The predicted object centers and bounding boxes. Different colors of points indicate the center predictions based on the semantics of different levels. The semantics of different levels are then centralized and aggregated to predict the bounding boxes.

Compared to other known methods, PointNet++ [32] aimed to preserve the spatial structure of points, and thus was widely used as backbone for feature learning in state-of-the-art frameworks [29, 46, 30]. Recently, Charles *et al.* proposed VoteNet [29], voting for points to be at the object centers based on learned features from PointNet++ [32]. This method yielded excellent results. But, there are still some challenging drawbacks. First, using PointNet++ as backbone neglected some local shape information, since the relative geometric positions of points were not accounted for. Second, the multi-level semantics were not adequately utilized by the structures of the frameworks, which might neglect some helpful information for object detection.

In this paper, we propose a novel *Hierarchical Graph Network (HGNet)* for 3D object detection on point clouds, based on graph convolutions (GConvs). HGNet contains three main components: a GConv based U-shape network (GU-net), a Proposal Generator, and a Proposal Reasoning Module (ProRe Module). Specially, we develop a new

*These authors contributed equally to this work.

Shape-attentive GConv (SA-GConv), which captures the object shape information by modelling the relative geometric positions of points. In our pipeline, the SA-GConv based GU-net takes a point cloud as input and captures the semantics of multi-levels (see Fig. 2), which are further aggregated to generate proposals by the Proposal Generator that contains an improved voting module (see Sec. 3.4). Incorporating the global scene semantics, the novel Proposal Reasoning Module (ProRe Module) leverages a fully-connected graph to reason on the proposals, and the bounding boxes are predicted. The detection results are finally obtained after performing 3D non-maximum suppression (NMS). An example of our object detection results is shown in Fig. 1.

The entire HGNet is trained in end-to-end manner. In our framework, the local shape information, semantics of multi-levels, and global scene information (features of proposals) of point clouds are sufficiently captured, aggregated, and incorporated by the hierarchical graph model, giving full consideration of the characteristics of point cloud data.

Our main contributions in this work are as follows:
(A) We develop a novel Hierarchical Graph Network (HGNet) for 3D object detection on point clouds, which outperforms the state-of-the-art methods by a clear margin.
(B) We propose a novel SA-(De)GConv, which is effective at aggregating features and capturing shape information of objects in point clouds.
(C) We build a new GU-net for generating multi-level features, which are vital for 3D object detection.
(D) Leveraging global information, we propose the ProRe Module to promote performance by reasoning on proposals.

2. Related Work

2.1. 3D Object Detection on Point Clouds

Point clouds have some special characteristics (e.g., sparse and irregular), which are often not suitable for convolutional neural networks to process. Many methods [2, 38, 20, 44, 9, 23] have been proposed for 3D object detection on point clouds, such as projection methods (e.g., Complex-YOLO [35], BirdNet [4]), volumetric convolution based methods (e.g., 3DFCN [19], Vote3Deep [8]), and PointNet based methods (e.g., F-PointNet [30], STD [46]). PointNet [31] pioneered a method using raw points as input and obtained good performances, followed by many frameworks [31, 32, 14, 29, 42]. Lang *et al.* [17] introduced the Pillar Feature Network, encoding point clouds into pseudo images and being processed by 2D CNN. Although novel and fast, the localization information of the framework [17] was not well preserved. PointNet based methods showed good performance, as they dealt with raw points directly. However, PointNet did not consider the dependence of points in information aggregation. Yang *et al.* [46] proposed a two-stage fusion method STD, combining PointNet based methods and volumetric convolution based methods. However, the two-stage process might learn some unmatched features for object detection. VoteNet [29] proposed a new voting method, predicting the object centers with the features learned which helped aggregate distant semantic information. However, the local shape information was not well accounted for in the VoteNet. Since there can be a variety of objects, the features needed for detecting different objects may not be in an identical distribution. In other words, semantics of multi-levels may be needed for identifying different objects.

2.2. Spatial-based Graph Convolution Networks

Graph convolution networks (GCNs) can be divided into two types: spatial-based [26, 3, 28] and spectral-based [12, 6, 15, 10]. Spatial-based methods are mainly based on the spatial relations of vertices in graphs, and are widely used on point clouds. Thus, we focus on reviewing these methods. The first spatial-based GCN was proposed in [26], by summing up the neighborhood information of vertices directly. Later, an inductive feature aggregation algorithm (GraphSAGE, including *Mean aggregator*, *LSTM aggregator*, and *Pooling aggregator*) was proposed in [10] to replace the transductive learning. Strictly speaking, GraphSAGE is not a kind of GCN, but it embodied the ideas of GCNs. Graph Attention Networks [40] employed attention mechanisms in learning relative weights among neighboring vertices, and showed attractive performance over previous works. In addition, many attention based GCNs [18, 1, 25] were proposed. GINs [45] assigned different weights for the *central vertex* and its neighboring vertices. For 3D data, Li *et al.* [21] introduced the dilated GCNs, which better balanced the receptive fields and computation. Feature-Steered GConv [41] verified that GConvs could capture shape information by modelling the geometric positions of the points, and outperformed the traditional shape descriptors. Wang *et al.* presented a dynamic edge convolution method for semantic segmentation, called EdgeConv [42], which aimed to capture the relationship of points but neglected the importance of the relative geometric positions of points.

3. Hierarchical Graph Network

3.1. Motivation and Overview

We aim to develop a new effective method for 3D object detection on point clouds. Different from 2D image data, point clouds often do not present clear object shape information (e.g., corners and edges), and thus some shape-attentive feature extractors are needed to process point clouds. Even though the previous work [42, 32, 41] im-

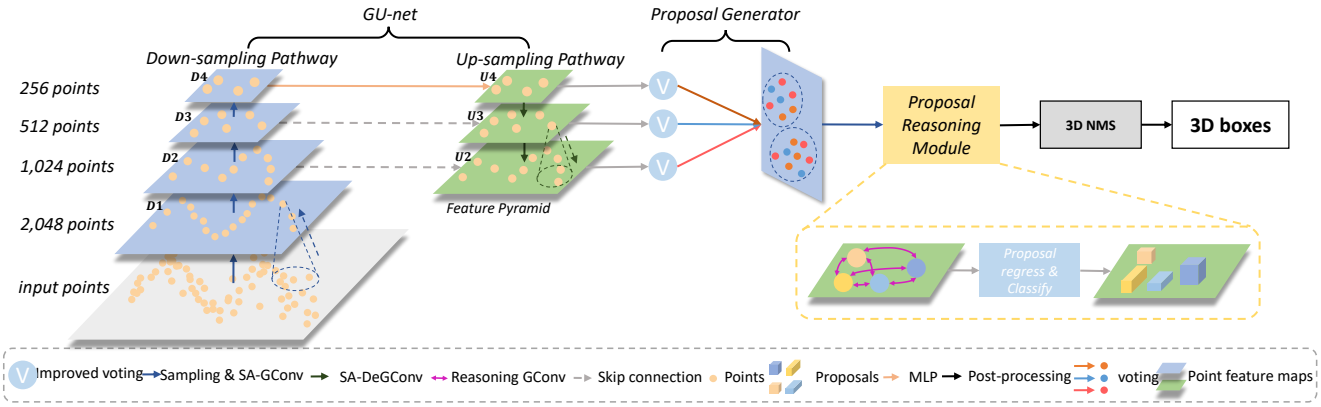

Figure 2. An overview of our HGNet framework, which contains GU-net, Proposal Generator, and ProRe Module. The counts of points are indicated on the left. In inference, 3D non-maximum suppression (NMS) is utilized and predicted 3D boxes are finally produced.

plicitly used the positions of points, it is more efficient to explicitly model the geometric positions of points, which better describe the shapes of objects. In addition, the multi-level semantics were proved beneficial [43, 39, 34, 22] to detecting objects of various sizes. Also, points can be sparse on the surfaces of objects, and thus the semantics of different levels may provide complementary information for one another. Many previous studies for 3D object detection did not sufficiently utilize multi-level semantics, which was inefficient to tackle point clouds with objects of various sizes and point sparsity.

In this work, we develop an end-to-end hierarchical graph network (HGNet) for 3D object detection on point clouds, as shown in Fig. 2. The entire HGNet contains three main parts: a GConv based U-shape network (GU-net), a Proposal Generator, and a Proposal Reasoning Module (ProRe Module). A new shape-attentive GConv is proposed to capture the local shape semantics. GU-net generates the multi-level semantics, which are aggregated to generate proposals by the Proposal Generator. Finally, the ProRe Module reasons on the proposals to help predict the bounding boxes by leveraging the global scene semantics.

Below we will discuss in detail the novel Shape-attentive (De)GConv in Sec. 3.2, GU-net in Sec. 3.3, the Proposal Generator in Sec. 3.4, the ProRe Module in Sec. 3.5, and the loss functions in Sec. 3.6.

3.2. Shape-attentive Graph (De)Convolutions

Point clouds usually do not present the object shapes clearly, yet shape information is important for 3D object detection. One might describe the local shape around a point using the relative geometric positions of its neighboring points. In this section, we will present a novel Shape-attentive GConv, which captures object shapes by modelling the geometric positions of points.

Shape-attentive Graph Convolution. Consider a point set $X = \{x_i \in \mathbb{R}^{D+3}\}_{i=1}^{n}$, where a point $x_i = [f_i, p_i]$, $p_i \in \mathbb{R}^3$ is the geometric position and $f_i \in \mathbb{R}^D$ is the D-dimensional feature. From X, we want to generate a point set $X' = \{x_i \in \mathbb{R}^{D'+3}\}_{i=1}^{n'}$, $n' < n$. Here we design a GConv to aggregate features from X to X'. Similar to the sampling layer in PointNet++, we first sample n' points from n points. Typically, *k Nearest Neighbors (kNN)* or *ball-query* [32] with respect to the geometric positions of points is used to construct a local region after sampling a point $x_i \in X$ as the *central point* for feature aggregation. In this paper, we use *kNN* as example.

Our shape-attentive GConv (SA-GConv) models the point positions by an independent term. Consider two points x_i and x_j in a local region, where x_i is the *central point* and x_j is one of the neighboring points of x_i. The relative geometric position vector, $e_{ij} = p_i - p_j$, can well express the relative geometric direction and the relative geometric distance between points x_i and x_j. Usually, a local region contains dozens of points, which are sufficient for local shape description in the 3-dimensional space if x_j enumerates all the points in the local region except x_i. To model the relative geometric positions of points and adaptively aggregate the point features, we define a directed GConv, SA-GConv, in an attractively simple way as:

$$f_i = \max_{x_j \in kNN(x_i)} \mathbf{g}(p_i - p_j) \cdot \mathbf{f}(x_i, x_j) \qquad (1)$$

We model the relative geometric positions by a learnable function $\mathbf{g} : \mathbb{R}^3 \rightarrow \mathbb{R}^1$, and the point features (including geometric positions) are addressed by $\mathbf{f} : \mathbb{R}^{D+3} \times \mathbb{R}^{D+3} \rightarrow \mathbb{R}^{D'}$. Without loss of generality, we employ the max-pooling operation to finally aggregate the features. In particular, we can implement $\mathbf{g}$ by a simple one-by-one convolution with the Sigmoid activation function, and implement $\mathbf{f}$ by $\mathbf{f}(x_i, x_j) = \mathbf{MLP}([x_i, x'_j])$, where $x'_j = x_j - x_i$, $\mathbf{MLP}(\cdot)$ is a multi-layer perceptron with batch normalization and ReLU activation, and $[\cdot, \cdot]$ indicates channel wise concatenation. The operation is illustrated as in Fig. 3. In

Fig. 3, the blue point x_i is sampled from a point set as *central point*, and the corresponding local region contains 3 nearest neighbors of x_i (including the orange, green, yellow points); the features of the 3 nearest neighbors of x_i are aggregated to x_i following Eq. (1). Our proposed SA-GConv has the property of *permutation invariance*, as the max-pooling operation is symmetric with respect to the input.

This shape-attentive operation is different from the simple MLP based operations (e.g., EdgeConv [42]). Eq. (1) explicitly computes the shape information by an independent function $\mathbf{g}$ while MLP based methods used learned weights. Three dimensions (e.g., for geometric positions) in a high dimensional feature space have very limited impacts if one co-treats all features (including "positions") using merely an MLP. Beside, as shown in Fig. 7, the function $\mathbf{g}$ is highly responsive to the shape information, and such shape description is beneficial to object detection.

Shape-attentive Graph De-Convolution.
In processing grid-structured data, an effective up-sampling operation often pads the feature maps (e.g., by interpolation) and then performs a convolution, as shown in the left part of Fig. 4. Generalizing this operation to irregular data, we propose the Shape-attentive graph De-Convolution (SA-DeGConv), which performs the inverse operation of SA-GConv. SA-DeGConv provides a method to propagate the features from certain points to more points in an adaptive way, as shown in the right part of Fig. 4.

The SA-DeGConv is performed in three steps. (1) Padding the points. As shown in Fig. 2, if we up-sample the features in the *point feature maps* $U4$ to generate $U3$, we should pad the points on $U3$ by following the positions of points on $D3$, as the points on $D3$ and $U3$ shall be positionally aligned. (2) Feature Initialization. As $\{p_i^{(4)}\} \subset \{p_i^{(3)}\}$, and $p_i^{(3)}, p_i^{(4)}$ indicate the geometric positions of the i-th point on $U3$ and $U4$, respectively. Thus, for the points on $U3$, we use arithmetic average to initialize the features by $f_i^* = \sum_{j=1}^{k} f_j^{(4)}/k$, where $f_j^{(4)}$ indicates the features of the j-th k positionally neighboring points on $U4$. (3) Feature aggregation. We use SA-GConv (Eq. (1)) to update the features of all the points on $U3$, as illustrated in Fig. 4.

3.3. GU-net

Effectively detecting objects needs to use abundant semantics. Previous methods (e.g., PointNet based methods) barely utilized semantics of multi-levels, which was not very beneficial to detecting objects of various sizes, as discussed in [22, 24]. Besides, as points can be sparse and even missing on the surfaces of objects, using multi-level semantics provides abundant information for object detection. To capture the multi-level semantics, we propose a new U-shape network called GU-net, based on SA-(De)GConv. We design a *down-sampling module*, and repeatedly stack it 4 times to form the down-sampling pathway, while an *up-sampling module* is repeatedly stacked twice to make up the up-sampling way. Similar to FPN [22], GU-net generates a feature pyramid with three *point feature maps* (see Fig. 2).

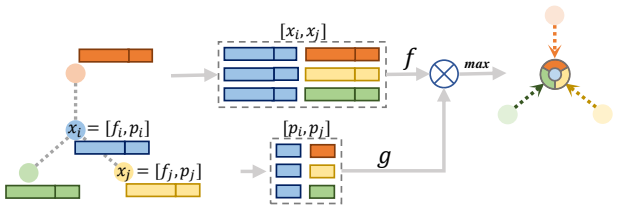

Figure 3. An illustration of the Shape-attentive GConv operation. The blue point x_i indicates a sampled point whose feature is updated by aggregating the features from other points (x_j, including the orange, yellow, and green points). p indicates the geometric position. The aggregation follows from Eq. (1).

Down-sampling Module. Given a *point feature map* with N points, we first sample a subset containing N' ($N' < N$) points by the farthest point sampling (FPS) [27, 7, 32]. Then we construct the local regions by *kNN* or *ball-query* around the sampled points, and then update the features of sampled points by performing the SA-GConv. In this way, a *point feature map* is processed to generate a higher-level *point feature map* with fewer points (e.g., $D4$ is generated from $D3$).

Up-sampling Module. The process of the up-sampling module is inverse of the process of the down-sampling module, mainly performed by SA-GConv. The skip connections are also used to bridge the corresponding *point feature maps* (e.g., $U3$ and $D3$) by channel-wise concatenation, except for the top-most *point feature map* $U4$. $U4$ and $D4$ is connected by **MLP**. Thus, the GU-net outputs a feature pyramid with three *point feature maps* (see Fig. 2).

3.4. Proposal Generator

Three *point feature maps* are generated by GU-net (see Fig. 2), containing the multi-level semantics. Some previous methods (e.g., VoteNet [29]) used only one feature map for object prediction. Even though the higher-level features are computed by fusing the lower-level features in the up-sampling pathway, it is more beneficial to use the multi-level features together for proposal generation as the features of different levels provide various semantics. To this end, we propose the Proposal Generator to predict the object centers (shown in Fig. 1) with an improved voting module as the main structure, which transforms the multi-level features into an identical feature space.

Improved Voting Module. The voting module in

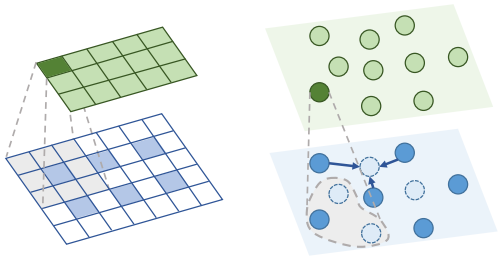

Figure 4. An illustration of the up-sampling operation (left) to the grid-structured data, and SA-DeGConv (right). In the right part, the dashed circles denote the padded points and the blue arrows indicate the arithmetic average for feature initialization in performing SA-DeGConv.

VoteNet [29] was proposed to predict the object centers and centralize the object features. In our paper, we perform the voting operation on all the *point feature maps* in the feature pyramid. Thus, the improved voting module also helps to transform the multi-level features (of different feature spaces) into an identical feature space (as shown in Fig. 1), which can be further utilized directly to generate proposals. We implement the improved voting module with SA-GConv, since SA-GConv is more efficient. The voting process is specified by:

$$\begin{aligned} [f_v, p_v] &= [f, p] + [\Delta f, \Delta p] \\ [\Delta f, \Delta p] &= \textbf{SA-GConv}([f, p]) \end{aligned} \tag{2}$$

where $\mathbf{f} \in \mathbb{R}^F$ and $p \in \mathbb{R}^3$ are the features and the geometric positions of the points in feature pyramid, and $f_v \in \mathbb{R}^{F_v}$ and $p_v \in \mathbb{R}^3$ are the features and geometric positions of the votes. **SA-GConv**$(\cdot)$ follows from Eq. (1). We use SA-GConv to implement the improved voting module by adding three additional channels to predict the geometric shifts.

Generating Proposals. By performing the improved voting module, the features in the feature pyramid are transformed into an identical feature space. To aggregate the features, we retain N_p votes by FPS and aggregate the features of all the votes into them, similar to VoteNet ($N_p = 256$ as default). Thus, the features of multi-levels are fully fused to predict bounding boxes and categories.

3.5. Proposal Reasoning Module

With the structures presented above, the local semantics and multi-level semantics are captured and fully fused. On one hand, these semantics are learned in the local receptive fields, yet the global scene semantics are not used in object detection. On the other hand, some objects contain very few points on their external surfaces (e.g., see the point clouds of the SUN RGB-D dataset in Fig. 6), and it can be hard to detect those objects with such limited information. Hence, we propose a new GConv based Proposal Reasoning Module (ProRe Module) to reason on proposals by leveraging the global scene information. The features of the proposals are updated by a new GConv, incorporating the global semantics and using the relative positions of the proposals as an attention map. We formulate the relation of the proposals as a directed graph $\mathcal{G}_g = (\mathcal{V}_g, \mathcal{E}_g)$. $\mathcal{V}_g$ denotes the vertex set, and each vertex is for a proposal presenting as high dimensional features. The edges $\mathcal{E}_g$ in $\mathcal{G}_g$ are initially set as fully-connected with self-loops.

Formally, given a proposal set in which the features of the proposals lie in an F-dimensional space, we consider a proposal-feature tensor $\mathbf{H}_p \in \mathbb{R}^{n \times F}$ and a tensor $\mathbf{P} \in \mathbb{R}^{n \times n \times 3}$ recording the relative positions of the proposals. In $\mathbf{P}$, an element $P_{i,j,k} = p_{i,k} - p_{j,k}$, where $p_{i,k}$ and $p_{j,k}$ are the k-th dimension ($k \in \{x, y, z\}$) of the geometric positions of the i-th and j-th proposals, respectively. The reasoning procedure can be specified as:

$$\mathbf{H}'_p = \Phi(\mathbf{P}, \mathbf{H}_p) = \gamma(\mathbf{P}) \odot \Psi_c(\Psi_v(\mathbf{H}_p)^T + \mathbf{H}_p^T)^T \tag{3}$$

where "$+\mathbf{H}_p^T$" indicates a residual connection [11], $\odot$ denotes the Hadamard product, and the operation Ψ_i ($i \in \{c, v\}$) is mainly implemented by one-dimensional convolutions, operating along the vertex-wise and channel-wise directions, respectively. The vertex-wise operation Ψ_v incorporates features and propagates information among vertices (proposals), and the channel-wise operation Ψ_c updates the features of proposals. $\mathbf{H}'_p \in \mathbb{R}^{n \times F'}$ denotes the proposal-feature tensor after reasoning. Different from the previous GConvs, ProRe considers the relative geometric positions among proposals in feature aggregation using γ, which transforms $\mathbf{P}$ into size $n \times F'$ for Hadamard production. After the reasoning, the 3D bounding boxes and corresponding categories are predicted as in VoteNet [29].

3.6. Loss Functions

The improved voting process on the feature pyramid is under the guidance of $\mathcal{L}_{\text{voting}}$, as:

$$\mathcal{L}_{\text{voting}} = \sum_m \left(\frac{1}{M_m} \sum_i |\Delta p_i - \Delta p_i^*| \mathbb{1}[x_i \text{ on object}]\right) \tag{4}$$

where $\mathbb{1}[x_i \text{ on object}]$ indicates whether a point x_i is on an object surface. M_m is the point number on a certain object in the m-th level *point feature maps* of feature pyramid, and $|\cdot|$ denotes the L_1 loss. The other loss terms $\mathcal{L}_{\text{obj-cls}}, \mathcal{L}_{\text{boxes}}, \mathcal{L}_{\text{sem-cls}}$ also follow VoteNet. The loss function of the entire framework is defined by:

$$\mathcal{L} = \mathcal{L}_{\text{voting}} + \lambda_1 \mathcal{L}_{\text{obj-cls}} + \lambda_2 \mathcal{L}_{\text{boxes}} + \lambda_3 \mathcal{L}_{\text{sem-cls}} \tag{5}$$

where $\lambda_1 = 0.5$, $\lambda_2 = 1$, and $\lambda_3 = 0.1$ as default.

4. Experiments

To evaluate our method, two key questions should be addressed by the experiments of HGNet.

$\mathbf{Q}_1$: How does HGNet compare to the state-of-the-art methods for 3D object detection on point clouds?
$\mathbf{Q}_2$: How to analyze the performance of SA-(De)GConv (**for local shape semantics**), GU-net with Proposal Generator (**for semantics of multi-levels**), and the ProRe Module (**for global semantics**)?

4.1. Implementation Details

The entire HGNet in Fig. 2 is trained end-to-end. We implement our framework using PyTorch 1.0 on Python 3.6. The framework is trained on 1 GeForce RTX 2080Ti GPU. We train HGNet with the Adam optimizer. With a batch size of 8, the learning rate is 10^{-3} initially, is reduced by $10\times$ after 80 epochs, and is reduced by $10\times$ again after 120 epochs. Training the whole framework to convergence takes about 18 hours on SUN RGB-D and about 5 hours on ScanNetV2. In our experiments, the evaluation metrics follow those in [29], using the average precision (AP). In addition to the mean average precision (mAP) for evaluating the performance of the frameworks compared, we also use the **coefficient of variation for AP** (cvAP) to show the adaptability of the frameworks to detect various objects, defined as

$$\text{cvAP} = [\frac{\sum_i^{N_c}(\text{AP}_i - \text{mAP})^2}{N_c \cdot \text{mAP}^2}]^{\frac{1}{2}} \tag{6}$$

where N_c indicates the numer of the object categories. The lower cvAP is, the better a framework is.

4.2. Datasets

SUN RGB-D [36] is a single-view dataset showing indoor scenes, with 37 object categories in total (but 10 most common categories are used). The whole dataset contains $\sim$5K RGB-D images with 5,285 images for training. All the images are annotated with oriented 3D bounding boxed and the categories. We convert the depth images into point cloud data before model processing.
ScanNet-V2 [5] is a dataset of indoor scenes, containing RGB-D scans of about 1.5K scenes. To compare with the state-of-the-art frameworks, we prepare the data as in [13].
Input Data. Similar to PointNet [31], we use raw points as input, after randomly sampling 20,000 points from a point cloud in SUN RGB-D or 40,000 points from a 3D scan in ScanNet-V2. We only use the height features and geometric positions as in VoteNet [29], without RGB cues. For data augmentation, we randomly flip point clouds along the x-axis and y-axis, and randomly scale the point clouds by s times, $s \sim U(0.9, 1.1)$.

4.3. Evaluation Results

Comparison with State-of-the-art Methods. To answer question $\mathbf{Q}_1$, we compare on SUN RGB-D and ScanNet-V2 with various state-of-the-art methods: Deep sliding shapes (DSS) [37], 3D-SIS [13], 2D-driven [16], F-PointNet [30], GSPN [47], Cloud of gradients descriptor (COGD) [33], and VoteNet [29]. The experimental results are shown in Table 1 and Table 2. The performance results of the previous methods are obtained from either the original papers or [29].
The experimental results show that our HGNet outperforms all the previous methods by a large margin without RGB cues. Specifically, HGNet promotes the AP scores for large objects compared to VoteNet [29], such as desk and bathtub, which puzzled VoteNet, as shown in Table 1. Note that HGNet has less bias than the previous methods (even reducing cvAP by $\sim$9% on SUN RGB-D), which illustrates that HGNet is more adaptive to various objects. This likely is due to the proposed feature pyramid and our hierarchical graph modelling (SA-GConv, GU-net, and ProRe Module). It is worth noting that the AP scores cannot completely show the power of HGNet, and this will be discussed in the next paragraph. Besides, the difference of the inference time per point cloud between VoteNet and HGNet is within 0.001s on our GPUs, on both SUN RGB-D and SCanNet.
Visualization Results. Fig. 6 gives some visualization examples of point clouds, comparing the predicted bounding boxes and ground truth boxes. These examples show that HGNet has good performance on various objects. Besides, HGNet often detects some objects in the scenes that are not annotated by the ground truth (see the first and second rows for SUN RGB-D in Fig. 6). This implies that the indicator AP might underestimate the ability of HGNet.

4.4. Ablation Analysis

Ablation Experiments. To answer question Q_2, we evaluate the contributions of SA-GConv, GU-net, and the ProRe Module via ablation experiments on the SUN RGB-D dataset. Some quantitative results are shown in Table 3. We compare SA-GConv with a simple GConv (SGConv) by $\mathbf{SGConv}(x_i, x_j) = \mathbf{f}(x_i, x_j)$, eliminating the position modelling term $\mathbf{g}(p_i - p_j)$. Also, we compare De-GConv with the arithmetic interpolation (Inter.), which is the initialization method of De-GConv (described in Sec. 3.2). We compare feature pyramid with $U2$ (as shown in Fig. 2). The first row in Table 3 is for the *baseline*. As one can see in Table 3, SA-GConv, ProRe Module, and feature pyramid contributes $\sim$2%, respectively. Besides, SA-DeGConv also contributes 0.4%. It is clear that these proposed components are useful. Below we further discuss the effects of ProRe module and SA-GConv.
Local Shape Information Capturing. To further illustrate the performance of SA-(De)GConv, we compare it with the set abstraction module (SA) of PointNet++. We replace SA-GConv by SA in HGNet, and compare the precision of the voting results on SUN RGB-D. The voting results show the power of feature capturing. We define a smaller box with

	Input style	bathtub	bed	bookshelf	chair	desk	dresser	nightstand	sofa	table	toilet	mAP	cvAP
DSS	XYZ + RGB	44.2	78.8	11.9	61.2	20.5	6.4	15.4	53.5	50.3	78.9	42.1	0.61
COGD	XYZ + RGB	58.3	63.7	31.8	62.2	**45.2**	15.5	27.4	51.0	51.3	70.1	47.7	0.35
2D-driven	XYZ + RGB	43.5	64.5	31.4	48.3	27.9	25.9	41.9	50.4	37.0	80.4	45.1	0.36
F-PointNet	XYZ + RGB	43.3	81.1	33.3	64.2	24.7	32.0	58.1	61.1	51.1	90.9	54.0	0.38
VoteNet	XYZ	74.4	83.0	28.8	**75.3**	22.0	29.8	**62.2**	64.0	47.3	90.1	57.7	0.40
HGNet	XYZ	**78.0**	**84.5**	**35.7**	75.2	34.3	**37.6**	61.7	**65.7**	**51.6**	**91.1**	**61.6**	**0.31**

Table 1. 3D object detection performance results on the SUN RGB-D V1 dataset. The average precision with a 3D IoU threshold of 0.25 is used. Only the 10 most common categories are shown. cvAP is defined in Eq. (6).

	Input style	mAP@0.25	mAP@0.50	cvAP@0.25	cvAP@0.5
DSS	XYZ + RGB	15.2	6.8	-	-
F-PointNet	XYZ + RGB	19.8	10.8	-	-
GSPN	XYZ + RGB	30.6	17.7	-	-
3D-SIS	XYZ	27.6	16.0	0.65	1.25
3D-SIS	XYZ+5 views	32.2	24.7	0.97	1.03
VoteNet	XYZ	58.6	33.5	0.40	0.84
HGNet	XYZ	**61.3**	**34.4**	**0.38**	**0.82**

Table 2. 3D object detection results on the ScanNet-V2 dataset with 3D IoU thresholds of 0.25 and 0.5, respectively. "-" means "not applicable", since the corresponding data were not available.

SGConv				GU-net		ProRe	mAP
SA-GConv	SGConv	SA-DeGConv	Inter.	FP	$U2$		
	✓		✓		✓		57.3
	✓		✓	✓			59.5
✓			✓		✓		59.7
✓		✓			✓		60.1
	✓		✓		✓	✓	58.9
✓		✓		✓			60.8
✓		✓		✓		✓	61.6

Table 3. Quantitative ablation experiments on SUN RGB-D. "FP" indicates the feature pyramid.

the same center in the bounding box of an object, and the lengths of the small box are only 30% of those of the bounding box. We define "precise votes" if the votes lie in the small box. We calculate the ratio of "precise votes" over the votes from $U2$ (as in Fig. 2). As shown in Table 4, it can be seen that the points are better clustered (by over 6% in the "precise votes" ratio) to the object centers with SA-GConv. Note that the proposals are generated from the votes, and thus the voting results are very important.

To demonstrate the **shape information** capturing capability of SA-GConv, we let $\textbf{SA-GConv}_g(x) = \max_{x_j \in kNN(x_i)}\{g(p_i - p_j)\}$, with $\mathbf{f}(x_i, x_j) \equiv 1$ in Eq. (1). The $\mathbf{g}$ parameters of $\textbf{SA-GConv}_g(x)$ are inherited from the $\mathbf{g}$ parameters of the first SA-GConv in GU-net (see Fig. 2). Then we operate $\textbf{SA-GConv}_g(x)$ on the SUN RGB-D point clouds. As illustrated in Fig. 7, the object parts that have obvious shape information (e.g, corners, edges) are highly responsive. Besides, the response hot maps are similar among the objects in the same category. This obviously verifies that our SA-GConv (especially $\mathbf{g}$) well captures the shape information by modelling the geometric positions.

The ProRe Module helps the features propagate among the proposals. This module might not be so useful if the features for detecting an object had been adequately learned; but it helps in detecting an object with very few points (e.g., the points can be sparse or missing on some objects). In each category of SUN RGB-D, we sort the objects based on the numbers of points on them in increasing order, and divide the objects into 10 groups based on the sorted order. Then we calculate the total average recall (AR) across the categories in every percentile range (group). As shown in Fig. 5, as the number of points on the objects decreases, the impact of the ProRe Module is gradually becoming apparent. For objects with very few points, the ProRe Module can promote the recall rate by even over 12%.

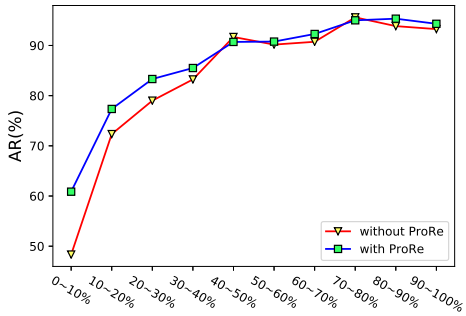

Figure 5. The x-axis is for the percentile ranges in the sorted order of the objects, and the y-axis is for AR with respect to the objects.

5. Conclusions

For 3D object detection on point clouds, we proposed a novel framework HGNet, learning the semantics via hierarchical graph modelling. Specifically, we proposed the novel and light Shape-attentive (De)GConv to capture the local shape semantics, which aggregates the features considering the relative geometric positions of points. We built GU-net based on SA-GConv and SA-DeGConv, generating the feature pyramid containing the multi-level semantics. The points on the feature pyramid vote to be at the corresponding object centers and the semantics of multi-levels are further aggregated to generate proposals. Then a ProRe Module is employed to incorporate and propagate the features among the proposals, promoting the detection

Ratio of "precise votes" (%)	bathtub	bed	bookshelf	chair	desk	dresser	nightstand	sofa	table	toilet	average
HGNet + SA	32.5	51.2	14.8	46.3	26.5	24.7	32.1	44.4	36.3	55.4	36.4
HGNet + SA-GConv	42.1	59.3	19.6	48.3	31.7	32.6	38.9	53.7	41.4	57.0	42.5

Table 4. Comparison of voting results between SA-GConv and SA module in HGNet on SUN RGB-D dataset.

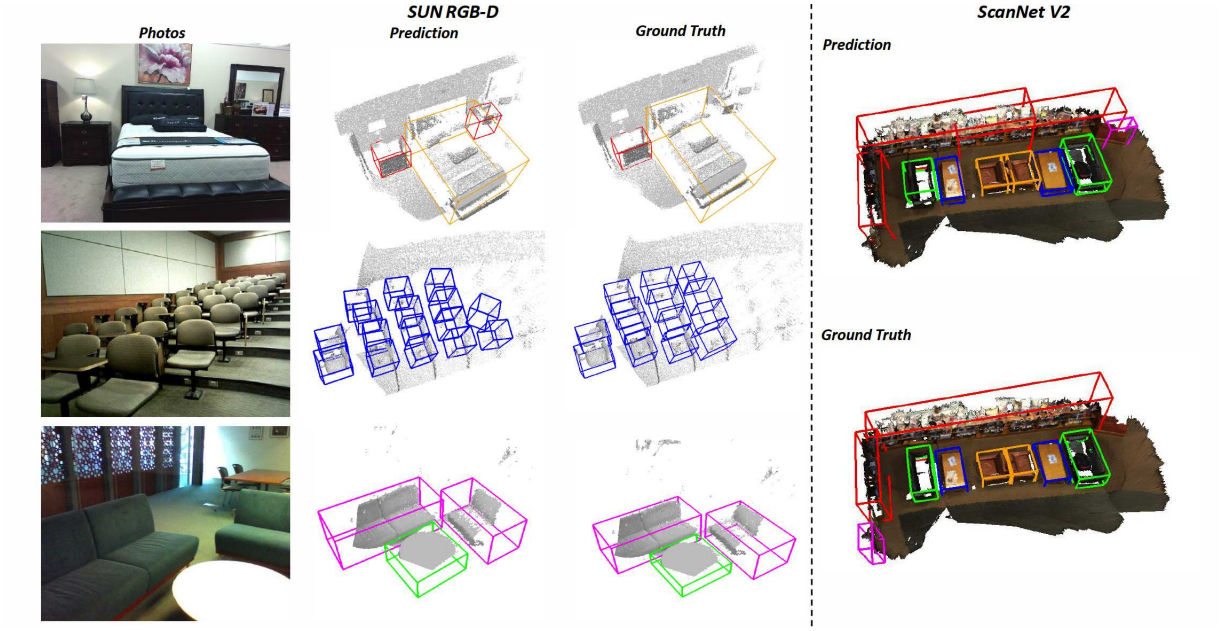

Figure 6. Comparison between the predicted bounding boxes and ground truth boxes on SUN RGB-D and ScanNet-V2.

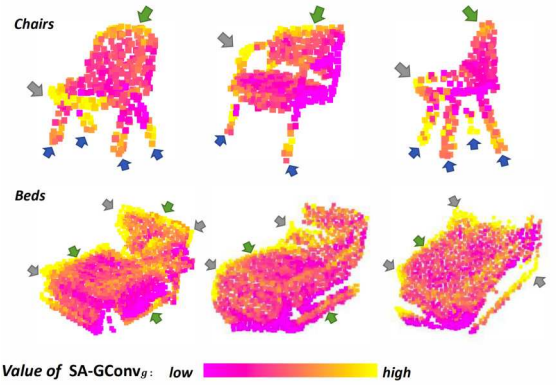

Figure 7. Visualization examples of the response values of **SA-GConv**$_g$ on some objects of the SUN RGB-D dataset. One can see that the edges (green arrows), corners (grey arrows), and sharp parts (blue arrows) of the objects are highly responsive.

performance by leveraging the global scene semantics. Finally, the bounding boxes and the categories are predicted. Different from the previous methods, HGNet attains better performance by carefully considering the shape information and aggregating the semantics of multi-levels.

6. Acknowledgements.

The research of Real Doctor AI Research Centre was partially supported by the Zhejiang University Education Foundation under grants No.K18-511120-004, No.K17-511120-017, and No.K17-518051-021, the National Natural Science Foundation of China under grant No.61672453, the National key R&D program sub project "large scale cross-modality medical knowledge management" under grant No.2018AAA0102100, the Zhejiang public welfare technology research project under grant No.LGF20F020013, the National Key R&D Program Project of "Software Testing Evaluation Method Research and its Database Development on Artificial Intelligence Medical Information System" under the Fifth Electronics Research Institute of the Ministry of Industry and Information Technology (No.2019YFC0118802), and The National Key R&D Program Project of "Full Life Cycle Detection Platform and Application Demonstration of Medical Artificial Intelligence Product" under the National Institutes for Food and Drug Control (No.2019YFB1404802), and the Key Laboratory of Medical Neurobiology of Zhejiang Province. D. Chen's research was supported in part by NSF Grant CCF-1617735. We like to thank three anonymous reviewers for their professional suggestions. We also like to thank Maosen Li in SJTU and Wenting Zhang in CSU for their helpful suggestions.

References

[1] Sami Abu-El-Haija, Bryan Perozzi, Rami Al-Rfou, and Alexander A Alemi. Watch Your Step: Learning Node Embeddings via Graph Attention. In *NeurIPS*, 2018.

[2] Eduardo Arnold, Omar Y Al-Jarrah, Mehrdad Dianati, Saber Fallah, David Oxtoby, and Alex Mouzakitis. A Survey on 3D Object Detection Methods for Autonomous Driving Applications. *T-ITS*, 2019.

[3] James Atwood and Don Towsley. Diffusion-Convolutional Neural Networks. In *NeurIPS*, 2016.

[4] Jorge Beltrán, Carlos Guindel, Francisco Miguel Moreno, Daniel Cruzado, Fernando Garcia, and Arturo De La Escalera. BirdNet: A 3D Object Detection Framework from LiDAR Information. In *ITSC*, 2018.

[5] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *CVPR*, 2017.

[6] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NeurIPS*, 2016.

[7] Yuval Eldar, Michael Lindenbaum, Moshe Porat, and Yehoshua Y Zeevi. The Farthest Point Strategy for Progressive Image Sampling. *IEEE Transactions on Image Processing*, 1997.

[8] Martin Engelcke, Dushyant Rao, Dominic Zeng Wang, Chi Hay Tong, and Ingmar Posner. Vote3Deep: Fast Object Detection in 3D Point Clouds Using Efficient Convolutional Neural Networks. In *ICRA*, 2017.

[9] Di Feng, Lars Rosenbaum, and Klaus Dietmayer. Towards Safe Autonomous Driving: Capture Uncertainty in the Deep Neural Network for Lidar 3D Vehicle Detection. In *ITSC*, 2018.

[10] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. In *NeurIPS*, 2017.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.

[12] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep Convolutional Networks on Graph-structured Data. *arXiv preprint arXiv:1506.05163*, 2015.

[13] Ji Hou, Angela Dai, and Matthias Nießner. 3D-SIS: 3D Semantic Instance Segmentation of RGB-D Scans. In *CVPR*, 2019.

[14] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent Slice Networks for 3D Segmentation of Point Clouds. In *CVPR*, 2018.

[15] Thomas N Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*, 2017.

[16] Jean Lahoud and Bernard Ghanem. 2D-Driven 3D Object Detection in RGB-D Images. In *ICCV*, 2017.

[17] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast Encoders for Object Detection From Point Clouds. In *CVPR*, 2019.

[18] John Boaz Lee, Ryan Rossi, and Xiangnan Kong. Graph Classification Using Structural Attention. In *KDD*, 2018.

[19] Bo Li. 3D Fully Convolutional Network for Vehicle Detection in Point Cloud. In *IROS*, 2017.

[20] Bo Li, Tianlei Zhang, and Tian Xia. Vehicle Detection from 3D Lidar Using Fully Convolutional Network. *arXiv preprint arXiv:1608.07916*, 2016.

[21] Guohao Li, Matthias Müller, Ali Thabet, and Bernard Ghanem. DeepGCNs: Can GCNs Go as Deep as CNNs? In *ICCV*, 2019.

[22] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *CVPR*, 2017.

[23] Or Litany et al. ASIST: Automatic Semantically Invariant Scene Transformation. *Computer Vision and Image Understanding*, 2017.

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single Shot Multibox Detector. In *ECCV*, 2016.

[25] Ziqi Liu, Chaochao Chen, Longfei Li, Jun Zhou, Xiaolong Li, Le Song, and Yuan Qi. Geniepath: Graph Neural Networks with Adaptive Receptive Paths. In *AAAI*, 2019.

[26] Alessio Micheli. Neural Network for Graphs: A Contextual Constructive Approach. *IEEE Transactions on Neural Networks*, 2009.

[27] Carsten Moenning and Neil A Dodgson. Fast Marching Farthest Point Sampling. Technical report, University of Cambridge, Computer Laboratory, 2003.

[28] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning Convolutional Neural Networks for Graphs. In *ICML*, 2016.

[29] Charles R. Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep Hough Voting for 3D Object Detection in Point Clouds. In *ICCV*, 2019.

[30] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum PointNets for 3D Object Detection from RGB-D Data. In *CVPR*, 2018.

[31] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *CVPR*, 2017.

[32] Charles R. Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NeurIPS*, 2017.

[33] Zhile Ren and Erik B Sudderth. Three-Dimensional Object Detection and Layout Prediction Using Clouds of Oriented Gradients. In *CVPR*, 2016.

[34] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *MICCAI*, 2015.

[35] Martin Simon, Stefan Milz, Karl Amende, and Horst-Michael Gross. Complex-YOLO: An Euler-Region-Proposal for Real-Time 3D Object Detection on Point Clouds. In *ECCV*, 2018.

[36] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. SUN RGB-D: A RGB-D Scene Understanding Benchmark Suite. In *CVPR*, 2015.

[37] Shuran Song and Jianxiong Xiao. Deep Sliding Shapes for Amodal 3D Object Detection in RGB-D Images. In *CVPR*, 2016.

[38] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view Convolutional Neural Networks for 3D Shape Recognition. In *ICCV*, 2015.

[39] Zhi Tian et al. Fcos: Fully Convolutional One-stage Object Detection. In *ICCV*, 2019.

[40] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph Attention Networks. In *ICLR*, 2017.

[41] Nitika Verma et al. Feastnet: Feature-steered Graph Convolutions for 3D Shape Analysis. In *CVPR*, 2018.

[42] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic Graph CNN for Learning on Point Clouds. *TOG*, 2019.

[43] Kun Wei et al. Adversarial Fine-Grained Composition Learning for Unseen Attribute-Object Recognition. In *ICCV*, 2019.

[44] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. Squeeze-Seg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D Lidar Point Cloud. In *ICRA*, 2018.

[45] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful are Graph Neural Networks? In *ICLR*, 2019.

[46] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: Sparse-to-Dense 3D Object Detector for Point Cloud. In *ICCV*, 2019.

[47] Li Yi, Wang Zhao, He Wang, Minhyuk Sung, and Leonidas J Guibas. GSPN: Generative Shape Proposal Network for 3D Instance Segmentation in Point Cloud. In *CVPR*, 2019.