GAM : Gradient Attention Module of Optimization for Point Clouds Analysis

Haotian Hu[1], **Fanyi Wang**[2*], **Zhiwang Zhang**[3], **Yaonong Wang**[1], **Laifeng Hu**[1], **Yanhao Zhang**[2]

[1] Zhejiang Leapmotor Technology CO., LTD.
[2] OPPO Research Institute
[3] The University of Sydney
{hu_haotian, wang_yaohong, hu_laifeng}@leapmotor.com, {wangfanyi,zhangyanhao}@oppo.com,
zhiwang.zhang@sydney.edu.au

Abstract

In point cloud analysis tasks, the existing local feature aggregation descriptors (LFAD) are unable to fully utilize information in the neighborhood of central points. Previous methods rely solely on Euclidean distance to constrain the local aggregation process, which can be easily affected by abnormal points and cannot adequately fit with the original geometry of the point cloud. We believe that fine-grained geometric information (FGGI) is significant for the aggregation of local features. Therefore, we propose a gradient-based local attention module, termed as Gradient Attention Module (GAM), to address the aforementioned problem. Our proposed GAM simplifies the process that extracts gradient information in the neighborhood and uses the Zenith Angle matrix and Azimuth Angle matrix as explicit representation, which accelerates the module by 35X. Comprehensive experiments were conducted on five benchmark datasets to demonstrate the effectiveness and generalization capability of the proposed GAM for 3D point cloud analysis. Especially on S3DIS dataset (Armeni et al. 2016), GAM achieves the best performance among current point-based models with mIoU/OA/mAcc of 74.4%/90.6%/83.2%, respectively.

Introduction

In recent years, point cloud analysis has become a hot topic in academia and industry due to the rapid development of autonomous driving and indoor robotics. Considering that point cloud is unordered, sparse, and irregular, traditional methods for 2D image processing cannot be directly applied to point clouds. PointNet (Qi et al. 2017a) is a pioneering work that uses Multi-Layer Perceptron (MLP) to learn point features independently. Qi et al. proposed PointNet++ (Qi et al. 2017b), which introduces local features to point cloud analysis models for further performance improvement of point cloud analysis models. Recent works present some promising results, using convolutional layers (Boulch 2020; Thomas et al. 2019; Xiang et al. 2021), graph structures (Wang et al. 2019b; Xu et al. 2020), MLP (Ma et al. 2022), or attention mechanisms (Guo et al. 2021; Zhao et al. 2021) for point cloud analysis. Among them, local feature aggregation descriptors (LFAD) play an important role.

However, existing LFAD cannot effectively distinguish points in a point cloud neighborhood, and thus are unable to learn finer semantic information of the point cloud. This observation motivates us to consider the attention mechanism within a point cloud neighborhood. The previous works treat all points in the neighborhood as equally important (Wang et al. 2019b), or only use distance information to constrain aggregation process (Lan et al. 2019; Thomas et al. 2019; Ma et al. 2022), ignoring deeper geometric relationships within the neighborhood. These operations include too much outlier information in the local feature aggregation process and impede the model to conform the original geometry of the point cloud. Therefore, we propose a novel gradient attention module (GAM) that utilizes neighboring gradient information to better constrain the aggregation process of the neighborhood features. As shown in Figure 1, with gradient information, our proposed method is enabled to predict clearer object boundaries.

In addition, we find that the gradient calculation method (Pauly 2003) based on the local surface fitting method is very slow, and hinders real-time inference after adding gradient information. To solve this problem, we propose to simplify the calculation of gradient information in the neighborhood, by converting gradient information to an explicit representation of the Zenith Angle and Azimuth Angle between the center point and its neighboring points. Our proposed method can accelerate computation speed by 35 times. As shown in Figure 2, GAM is a plug-and-play module, which effectively improves the performance of baseline methods while maintaining a similar inference speed.

Our proposed GAM is portable and can be added to previous state-of-the-art methods with a few lines of code. We conduct experiments on 3D semantic segmentation task using S3DIS dataset (Armeni et al. 2016), 3D shape classification task using ScanObjectNN dataset (Uy et al. 2019) and ModelNet40 dataset (Wu et al. 2015), 3D part segmentation task using ShapeNet (Yi et al. 2016), and 3D object detection task on KITTI dataset (Geiger et al. 2013). Experiment results demonstrate that GAM is an effective module and applicable to a wide range of point cloud analysis tasks with good performance improvements for various models.

The contributions of this paper are summarised as follows:

- We propose a lightweight and efficient gradient attentive

*Corresponding author

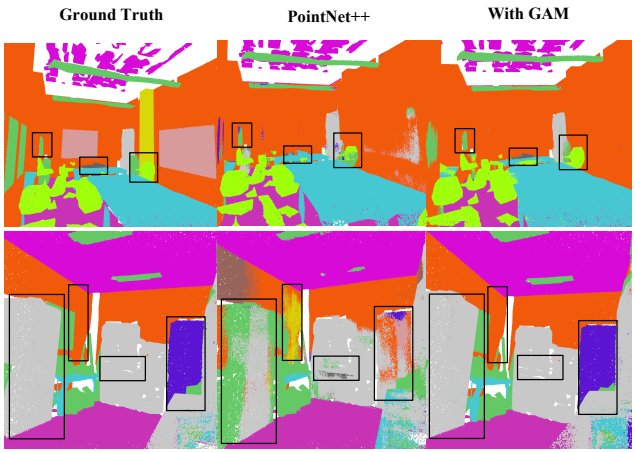

Figure 1: S3DIS benchmark visualization results, from left to right are ground truth, PointNet++ (Qi et al. 2017b) and the results after adding the gradient attention module (GAM).

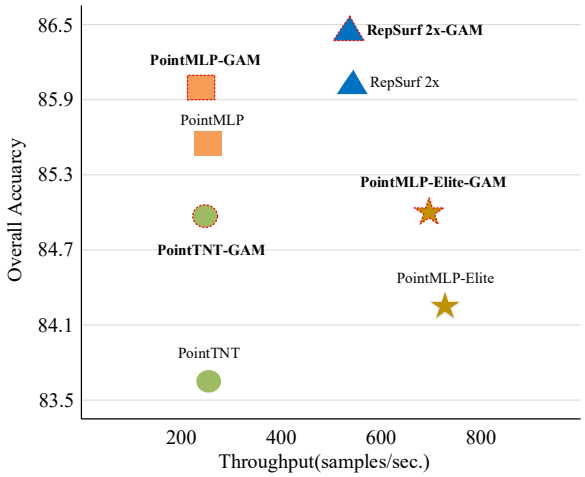

Figure 2: Overall Accuracy (OA) and throughput comparison plots of 3D shape classification experiment results on PointMLP (Ma et al. 2022), PointMLP-Elite (Ma et al. 2022), PointTNT (Berg, Oskarsson, and O'Connor 2022) and RepSurf 2x (Ran, Liu, and Wang 2022) models before and after adding GAM in ScanObjectNN (Uy et al. 2019).

module (GAM). To the best of our knowledge, gradient information is the first time to be introduced into the vector of locally aggregated descriptors of point cloud neighborhood features.

- The relationship between gradient information and zenith angle and azimuth angle in the point cloud neighborhood is established through mathematical representation. The gradient calculation process is simplified to the calculation of the zenith angle and azimuth angle of neighborhood points. Thus, the computation speed of GAM is effectively improved.
- Comprehensive experiments on five benchmarks demonstrate that our proposed gradient attention module can effectively boost performances of state-of-the-art methods within limited additional memory consumption and inference time. In addition, our proposed GAM can be used in various 3D tasks such as 3D semantic segmentation, 3D shape classification, 3d object detection, and 3D part segmentation.

Related Work

The point cloud analysis task starts with learning the embedding of each point, then extracts global embedding from the whole point cloud using local aggregation methods, and finally feeds global embedding to branches of each task. Due to the disorderly nature of point clouds, some previous works attempt to project point clouds into regular voxels (Zhou and Tuzel 2018; Wu et al. 2015; Yan, Mao, and Li 2018) or multiple views (Su et al. 2015; Wei, Yu, and Sun 2020; Liang et al. 2018). These methods significantly improve the computational speed, but lose information during the projection process and undermine model accuracy severely. In contrast, point-based methods directly use original point cloud information as input and employ various well-designed local feature aggregation descriptors (Thomas et al. 2019; Lan et al. 2019; Yang et al. 2019; Komarichev, Zhong, and Hua 2019). Meanwhile, some previous works (Wang et al. 2019a; Chen, Luca, and Antonios 2021; Wang et al. 2022; Cui et al. 2021; Wu et al. 2022) use the attention mechanism to extract the feature of the point cloud.

Multi-view and Voxel-based Approaches

Early works (Su et al. 2015; Wei, Yu, and Sun 2020) project unstructured point clouds into multiple 2D views, extract features from different views using 2D convolution, and then use sophisticated methods to fuse features from multiple views. MVCNN (Su et al. 2015) is a pioneering work that uses a maximum pooling layer to aggregate multi-view information into global features. But the maximum pooling layer retains only the largest elements, which inevitably leads to information loss. To address this problem, Wei et al. proposed View-GCN (Wei, Yu, and Sun 2020) using directed graphs to find the relationship between individual views. Each individual view is regarded as a graph node, and the global shape descriptor is obtained by max pooling graph nodes of all levels.

Voxel-based approaches divide the point cloud into a uniform 3D spatial grid and use 3D convolutional neural networks for feature extraction. Zhou et al. introduced VoxelNet (Zhou and Tuzel 2018) for robust 3D target detection. Although this method has achieved high detection performance, computation time and memory consumption sharply increase as the point cloud resolution is enhanced. To solve this problem, SECOND (Yan, Mao, and Li 2018) proposed 3D sparse convolution that effectively reduces memory and computation costs, but most devices still have difficulty affording such a large amount of computation.

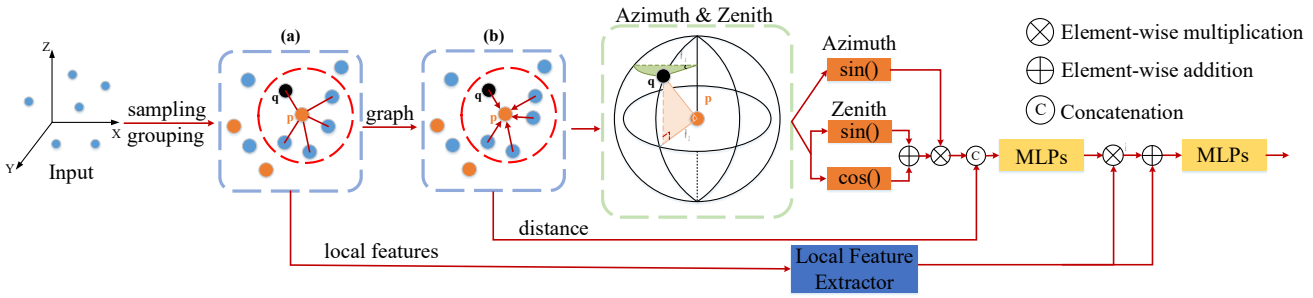

Figure 3: Schematic diagram of GAM structure, where p represents one center point, q represents a particular neighbor point of p, and LFE denotes the local feature extractor of different baselines. Module inputs are original positional information of the point cloud and features. Step (a) searches the center points of the point cloud and their corresponding neighborhood points; step (b) builds a directed graph to obtain the relative position vector between the center point and its neighborhood points. Then GAM calculates the zenith and azimuth angles in the neighborhood and uses them to construct a gradient attention matrix.

Point-based Approaches

PointNet (Qi et al. 2017a) has paved the way for relevant research studies on the point cloud, using MLP and max-pooling to extract and aggregate global features. PointNet++ (Qi et al. 2017b) introduces the concept of local features into 3D point cloud analysis. It uses Farthest Point Sampling (FPS) and ball query to perform centroid sampling, and neighborhood point search on point clouds to obtain various levels of local-global features. Subsequent work on 3D point cloud analysis focuses on the study of point cloud local feature aggregation descriptors. DGCNN (Wang et al. 2019b) builds a directed graph between centroids and neighboring points, extracts features of each edge using EdgeConv, and finally aggregates local features through max-pooling layer. In Geo-CNN (Lan et al. 2019), the edge feature of each direction is weighted by a learnable matrix that relates to the direction. Then local features are aggregated according to the angle between the relative vector and three axes.

Different from the above methods, the proposed GAM utilizes fine-grained geometric information to aggregate finer local features, which helps to improve the accuracy of subsequent tasks. Besides, compared to sophisticated local feature aggregation descriptors that lead to inefficiency in point cloud analysis models, the proposed GAM does not burden the computing device.

Proposed Method

The proposed gradient attention module (GAM) uses both gradient information and distance information between the center point and its neighboring points to generate corresponding importance weights of each neighboring point. The mathematical representation of gradient information of point cloud neighborhoods is given in Section 3.1, followed by the overall structure of GAM in Section 3.2.

Mathematical Representation of Point Cloud Neighborhood Gradients

We use a center point and a point in its neighborhood as an example to illustrate the mathematical relation between gradient information, zenith angle, and azimuth angle. Given a set of N cloud points $\{p_i\} \in \mathbb{R}^{N\times 3}$, where $i = 1, 2, ..., N$. A central point p_i has the coordinate of (x_i, y_i, z_i), and another point q_j is the neighborhood point of p_i, whose coordinate is (x_j, y_j, z_j). In the range image of the point cloud, two points p_i, q_j can be represented as discrete points $f(u_i, v_i) = z_i$ and $f(u_j, v_j) = z_j$, and the conversion equation is written as follows.

$$\begin{cases} u_j = \dfrac{l}{d} x_j + u_0 \\ v_j = \dfrac{l}{d} y_j + v_0 \end{cases} \tag{1}$$

where d is the depth of current point, l is the camera focal length, u_0 and v_0 are the X, Y coordinates of center point of the range image respectively.

Traditional method uses the difference of pixel value in depth map between current pixel and its adjacent pixels in the X and Y axis directions to represent depth gradient of the point. However, we focus on the association between the center point p_i and one of its neighboring points q_j in 3D space. Hence the point q_j is regarded as a neighboring pixel of p_i in the range image. We calculate the pixel value difference of these two points in the direction of an edge vector $\vec{\boldsymbol{b}} = (u_j - u_i, v_j - v_i)$ to represent depth gradient of the point. The depth gradient ∇d_b is defined as $\nabla d_b = \frac{z_{ji}}{\sqrt{u_{ji}^2 + v_{ji}^2}}$. And depth gradient along X, Y axes ∇d_x, ∇d_y are defined as follows,

$$\begin{cases} \nabla d_x = \dfrac{z_{ji}}{\sqrt{u_{ji}^2 + v_{ji}^2}} * \dfrac{u_{ji}}{\sqrt{u_{ji}^2 + v_{ji}^2}} \\ \nabla d_y = \dfrac{z_{ji}}{\sqrt{u_{ji}^2 + v_{ji}^2}} * \dfrac{v_{ji}}{\sqrt{u_{ji}^2 + v_{ji}^2}}. \end{cases} \tag{2}$$

where u_{ij}, v_{ij}, z_{ij} denote $u_j - u_i$, $v_j - v_i$, $z_j - z_i$, respectively. Combining Eq. 1 and Eq. 2, components of depth gradient along X and Y axis are defined as∇d_x, ∇d_y.

$$\begin{cases} \nabla d_x = \dfrac{d}{f}\dfrac{z_{ji}x_{ji}}{x_{ji}^2+y_{ji}^2}, \\ \nabla d_y = \dfrac{d}{f}\dfrac{z_{ji}y_{ji}}{x_{ji}^2+y_{ji}^2}. \end{cases} \tag{3}$$

Since we explore the geometric structure of point clouds in 3D space, it is necessary to convert depth gradient to the world coordinate system, in order to obtain gradients ∇z_x and ∇z_y respectively.

$$\begin{cases} \nabla z_x = \dfrac{d}{f}\nabla d_x = \dfrac{x_{ji}z_{ji}}{x_{ji}^2+y_{ji}^2}, \\ \nabla z_y = \dfrac{d}{f}\nabla d_y = \dfrac{y_{ji}z_{ji}}{x_{ji}^2+y_{ji}^2}. \end{cases} \tag{4}$$

The approximate gradient $\vec{\boldsymbol{g}}$ of the neighboring point q_j is $\vec{\boldsymbol{g}} = (g_x, g_y, g_z)$, where

$$\begin{cases} g_x &= \dfrac{z_{ji}}{\sqrt{x_{ji}^2+y_{ji}^2+z_{ji}^2}}\dfrac{x_{ji}}{\sqrt{x_{ji}^2+y_{ji}^2}}, \\ g_y &= \dfrac{z_{ji}}{\sqrt{x_{ji}^2+y_{ji}^2+z_{ji}^2}}\dfrac{y_{ji}}{\sqrt{x_{ji}^2+y_{ji}^2}}, \\ g_z &= \dfrac{\sqrt{x_{ji}^2+y_{ji}^2}}{\sqrt{x_{ji}^2+y_{ji}^2+z_{ji}^2}}. \end{cases} \tag{5}$$

Where $\frac{z_{ji}}{\sqrt{x_{ji}^2+y_{ji}^2+z_{ji}^2}}$ and $\frac{\sqrt{x_{ji}^2+y_{ji}^2}}{\sqrt{x_{ji}^2+y_{ji}^2+z_{ji}^2}}$ denote sine and cosine of the zenith angle of the neighborhood point respectively, $\frac{x_{ji}}{\sqrt{x_{ji}^2+y_{ji}^2}}$ and $\frac{y_{ji}}{\sqrt{x_{ji}^2+y_{ji}^2}}$ denote sine and cosine of the azimuth angle of the neighborhood point respectively.

We simplify gradient calculation of neighborhood points by using the zenith angles and azimuth angles, and effectively reduce computation time.

Gradient Attention Module

In this section we introduce our proposed gradient attention module (GAM) based on the zenith and azimuth angles of neighborhood points, which includes gradient information of neighborhood points during neighborhood aggregation process. Therefore the model is enabled to capture more accurate local features, by using more fine-grained geometric information in the local feature aggregation descriptors. Details of our proposed GAM are given in Algorithm 1.

As shown in Figure 3, there are a set of N points $\boldsymbol{P} = \{\boldsymbol{p}_i | i = 1, ..., N\} \in \mathbb{R}^{N\times 3}$ in the Cartesian coordinate system (x, y, z), with their corresponding features $\boldsymbol{F} = \{\boldsymbol{f}_i | i = 1, ..., N\} \in \mathbb{R}^{N\times C}$. We use the baseline method (Qi et al. 2017b; Wang et al. 2019b) to find center point and search for neighbor points. $\boldsymbol{P}^{center} = \{\boldsymbol{p}_s^{center} | s = 1, ..., N_s\} \in \mathbb{R}^{N_s\times 3}$ represents the selected set of center point. For each center point, K neighboring points are searched. In total there are $K * N_s$ neighboring points $\boldsymbol{Q}^{NBR} = \{\boldsymbol{q}_{s,j}^{NBR} | s = 1, ..., N_s, j = 1, ..., K\} \in \mathbb{R}^{N_s\times K\times 3}$ with corresponding features $\boldsymbol{F}^{NBR} = \{\boldsymbol{f}_{s,j}^{NBR} | s = 1, ..., N_s, j = 1, ..., K\} \in \mathbb{R}^{N_s\times K\times C}$, where C denotes the number of channels of input features.

After establishing the directed graph between center points and each of its neighboring points, the relative position matrix is represent as $\boldsymbol{E} = \{\boldsymbol{e}_{s,j} | s = 1, ..., N_s, j = 1, ..., K\}$. $\boldsymbol{e}_{s,j}$ is represented as $\boldsymbol{q}_{s,j}^{NBR} - \boldsymbol{p}_s^{center}$, where $\boldsymbol{p}_s^{center}$ is the center point and $\boldsymbol{q}_{s,j}^{NBR}$ is one of its neighboring points. Vector length $\boldsymbol{d_{s,j}}$ is also expressed as follows,

$$\begin{cases} \boldsymbol{e}_{s,j} &= (\overrightarrow{\boldsymbol{x}_{s,j}}, \overrightarrow{\boldsymbol{y}_{s,j}}, \overrightarrow{\boldsymbol{z}_{s,j}}) \\ \boldsymbol{d}_{s,j} &= \sqrt{(\overrightarrow{\boldsymbol{x}_{s,j}})^2 + (\overrightarrow{\boldsymbol{y}_{s,j}})^2 + (\overrightarrow{\boldsymbol{z}_{s,j}})^2} \end{cases} \tag{6}$$

where $(\overrightarrow{\boldsymbol{x}_{s,j}}, \overrightarrow{\boldsymbol{y}_{s,j}}, \overrightarrow{\boldsymbol{z}_{s,j}})$ represent the relative position vector in the Cartesian coordinate system.

Then the sum of azimuthal sine and cosine values of each neighboring point is calculated, and multiplied with sine of zenith angle, to represent the gradient information of the neighborhood points $\boldsymbol{g_{s,j}}$ written as follows,

$$\boldsymbol{g}_{s,j} = (\frac{\overrightarrow{\boldsymbol{z}_{s,j}}}{\boldsymbol{d}_{s,j}}\frac{\overrightarrow{\boldsymbol{x}_{s,j}} + \overrightarrow{\boldsymbol{y}_{s,j}}}{\sqrt{(\overrightarrow{\boldsymbol{x}_{s,j}})^2 + (\overrightarrow{\boldsymbol{y}_{s,j}})^2}}). \tag{7}$$

Our proposed GAM uses MLP to fuse gradient information and distance information of neighborhood points to obtain attentive weight calculated as follows,

$$a_{s,j} = Sigmoid(MLP([\boldsymbol{g}_{s,j}; \boldsymbol{d}_{s,j}])). \tag{8}$$

where Sigmoid is the activation function and [;] denotes the concatenation operation. The weight matrix is represented as $\boldsymbol{A} = \{a_{s,j} | s = 1, ..., N_s, j = 1, ..., K\}$

After obtaining the weight matrix $\boldsymbol{A}$, it is multiplied with the input point cloud features. Thus GAM can extract deep features of the point cloud according to importance of each

---

Algorithm 1: Gradient Attention Module

Input: point cloud $\boldsymbol{P} = \{\boldsymbol{p}_i | i = 1, ..., N\} \in \mathbb{R}^{N\times 3}$, with corresponding features $\boldsymbol{F} = \{\boldsymbol{f}_i | i = 1, ..., N\} \in \mathbb{R}^{N\times C}$
Parameter: local feature extractor $\phi(\cdot)$, balanced weights λ, sampling radius r, number of centroid samples N_s, number of neighborhood point samples K
Output: output features $\boldsymbol{F}^{out}$

1: Sample N_s points as the center points of the point cloud, with corresponding coordinates denoted as $\{\boldsymbol{p}_s^{center} | s = 1, ..., N_s\} \in \mathbb{R}^{N_s\times 3}$.
2: Search K points for each center point as its neighborhood, with corresponding coordinates $\boldsymbol{Q}^{NBR} = \{\boldsymbol{q}_{s,j}^{NBR} | s = 1, ..., N_s, j = 1, ..., K\} \in \mathbb{R}^{N_s\times K\times 3}$, and corresponding features $\boldsymbol{F}^{NBR} = \{\boldsymbol{f}_{s,j}^{NBR} | s = 1, ..., N_s, j = 1, ..., K\} \in \mathbb{R}^{N_s\times K\times C}$.
3: Create a directed graph in the neighborhood of each center point and compute relative position vector, distance information $\boldsymbol{d}_{s,j}$ (Eq. 6) and gradient information $\boldsymbol{g}_{s,j}$ (Eq. 7) of the neighborhood points.
4: Calculate weighted score matrix of neighborhood points $\boldsymbol{A}$ by using fine-grained geometric information $\boldsymbol{d}_{s,j}$ and $\boldsymbol{g}_{s,j}$.(Eq. 8)
5: Local features $\boldsymbol{F}^{NBR}$ are fed to local feature extractor $\phi(\cdot)$, multiplied with the corresponding weight score matrix $\boldsymbol{A}$ and then weighted by λ to obtain $\boldsymbol{F}^{out}$ (Eq. 9).
6: **Return** $\boldsymbol{F}^{out}$

Method	mIoU	OA	mAcc	TP
PointNet (Qi et al. 2017a)	47.6	78.5	66.2	-
PointCNN (Li et al. 2018)	65.4	88.1	75.6	-
PointWeb (Zhao et al. 2019)	66.7	87.3	76.2	-
RandLA-Net (Hu et al. 2020)	70.0	88.0	82.0	-
KPConv (Thomas et al. 2019)	70.6	-	79.1	-
BAAF-Net (Qiu et al. 2021)	72.2	88.9	83.1	-
PointNet++ (Qi et al. 2017b)	54.5	81.0	67.1	130
+ GAM	56.6	81.8	71.7	127
DGCNN (Wang et al. 2019b)	56.1	84.1	-	32
+ GAM	58.8	85.5	69.1	31
Point Trans. (Zhao et al. 2021)	73.5	90.2	81.9	26
+ GAM	73.9	89.9	83.0	25
+ GAM*	**74.4**	**90.6**	**83.2**	-

Table 1: Comparison results on S3DIS dataset for 3D semantic segmentation with 6-fold cross-validation. Mean Inter-over-Union (mIoU), overall accuracy (OA) and mean accuracy (mAcc) are used as evaluation metrics. * represents the voting strategy and throughput using the test result in Area5.

Method	OA	mAcc	TP
PointNet++ (Qi et al. 2017b)	77.9	75.4	-
DGCNN (Wang et al. 2019b)	78.1	73.6	-
GBNet (Qiu, Anwar, and Barnes 2022)	80.5	77.8	-
PRA-Net(1K) (Cheng et al. 2021)	81.0	77.9	1152
+ GAM	81.6	77.9	1094
Point-TNT (Berg et al. 2022)	83.6	82.3	240
+ GAM	85.0	82.8	238
PointMLP-elite (Ma et al. 2022)	84.4	82.6	749
+ GAM	84.8	88.24	708
PointMLP (Ma et al. 2022)	85.7	84.4	213
+ GAM	86.1	84.7	204
RepSurf-2x (Ran, Liu, and Wang 2022)	86.0	-	420
+ GAM	86.4	-	418
PointNeXt-s (Qian et al. 2022)	88.1	86.4	1628
+ GAM	**88.4**	**86.5**	1544

Table 2: Comparison result on ScanObjectNN dataset for 3D shape classification. For a fair comparison, all methods in the table use 1024 points as the input. Overall accuracy (OA) and mean accuracy (mAcc) are used as evaluation metrics.

neighborhood point. A complete local feature aggregation process can be expressed as follows,

$$\boldsymbol{F}^{out} = MLP(\frac{\lambda\phi(\boldsymbol{F}^{NBR}) \cdot \boldsymbol{A} + \phi(\boldsymbol{F}^{NBR})}{1+\lambda}) \quad (9)$$

where $\boldsymbol{F}^{out} \in \mathbb{R}^{N_s \times K \times C_{out}}$, C_{out} denotes the number of channels of output features. λ is the balance weight and $\cdot$ represents the element-wise multiplication. $\phi(\cdot)$ denotes the local feature extractor used in previous works (Ma et al. 2022; Qi et al. 2017b; Wang et al. 2019b) to extract deeply aggregated features.

Experiments

To fully evaluate the effectiveness and generality of our proposed GAM, we apply our proposed method on several state-of-the-art methods for 3D shape classification, 3D part segmentation, 3D semantic segmentation, and 3D object detection. Experiments are conducted on S3DIS dataset (Armeni et al. 2016), ScanObjectNN dataset (Uy et al. 2019), ShapeNet dataset (Yi et al. 2016), KITTI dataset (Geiger et al. 2013) and ModelNet40 dataset (Wu et al. 2015), respectively. The same training strategy used in each baseline is employed in our experiments, except that only GAM is added in each downsampling layer of the model. And λ is set to 1. The number of channels of the two-layer MLP in GAM is set to (1,16), (16,1). Experiments are run on NVIDIA GTX 3090 GPU and AMD EPYC 7402 CPU.

Results on 3D Semantic Segmentation

We conduct the 3D semantic segmentation experiment on S3DIS dataset (Armeni et al. 2016), which contains 3D scanned point clouds of six interior regions with 272 rooms in total. Each point belongs to one of the 13 semantic categories containing wood panels, bookcases, chairs, ceilings, etc. We compare the proposed GAM with state-of-the-art methods, including PointNet (Qi et al. 2017a), PointCNN (Li et al. 2018), PointWeb (Zhao et al. 2019), RandLA-Net (Hu et al. 2020), KPConv (Thomas et al. 2019), BAAF-Net (Qiu, Anwar, and Barnes 2021), PointNet++ (Qi et al. 2017b), DGCNN (Wang et al. 2019b), and Point Trans. (Zhao et al. 2021). We select PointNet++, DGCNN, and Point Trans. as baselines to evaluate the effectiveness of adding our proposed method GAM.

In Table 1, we report mean Inter-over-Union (mIoU), overall accuracy (OA) and mean accuracy (mAcc), and throughput(TP) for the 6-fold cross-validation of S3DIS dataset. We can find that after adding GAM into the three baselines PointNet++, DGCNN, and Point Trans, mIoU scores increase 2.1%, 2.7%, and 0.4% respectively, OA scores and mAcc scores are also significantly increased. After using Voting, GAM with Point Trans achieves the best performance, where mIoU score, OA score, and mAcc score are 74.4%, 90.6%, and 83.2%, respectively. Inference time of three baselines after adding GAM only increases by 2.3%, 3.1%, and 3.8%, respectively. Increment on computational costs in terms of throughput is almost negligible. These experimental results demonstrate that our proposed GAM is lightweight and efficient. Figure 4 shows visualization result of 3D semantic segmentation of S3DIS dataset.

3D Shape Classification Experimental Results on ScanObjectNN

We conduct the 3D shape classification experiment on ScanObjectNN dataset (Uy et al. 2019) that contains 15,000 objects of 15 different classes. The most difficult and commonly used variant of ScanObjectNN PB_T50_RS is implemented for experiments. We compare our proposed GAM with state-of-the-art methods, including PointNet++ (Qi et al. 2017b), DGCNN (Wang et al. 2019b), GBNet (Qiu, Anwar, and Barnes 2022), PRA-Net(1k) (Cheng et al. 2021), Point-TNT (Berg, Oskarsson, and O'Connor 2022), PointMLP-elite (Ma et al. 2022), PointMLP (Ma et al. 2022), RepSurf-2x (Ran, Liu, and Wang 2022) and PointNeXt-s (Qian et al. 2022). Among these meth-

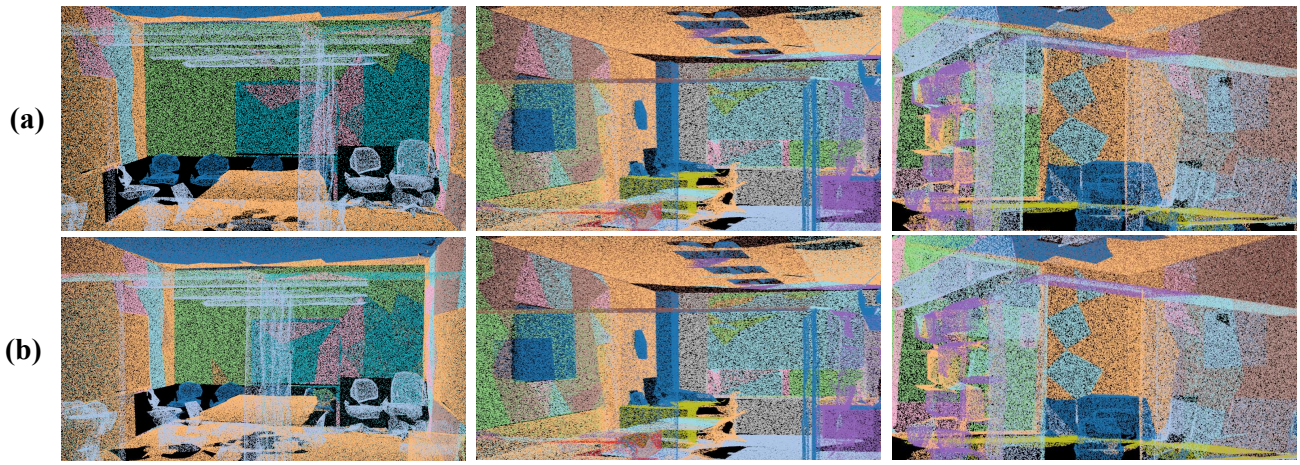

Figure 4: 3D semantic segmentation experiment visualization results. (a) represents ground truth and (b) represents forecast results of point transformer (Zhao et al. 2021) after GAM is inserted.

Method	Cars	Pedestrians	Cyclists
IA-SSD (Zhang et al. 2022)	**79.57**	58.91	71.24
+GAM	79.16	**59.31**	**72.58**

Table 3: Comparison results of 3D object detection on KITTI 3D target detection validation set. Mean Inter-over-Union (mIoU) score is used as the evaluation metric.

ods, PRA-Net(1k), Point-TNT, PointMLP-elite, PointMLP, RepSurf-2x, and PointNeXt-s are used as baselines to evaluate the effectiveness of adding our proposed GAM.

In Table 2, we report the overall accuracy (OA) score, mean accuracy (mAcc), and throughput (TP) on the ScanObjectNN dataset. The voting strategy is not used in each baseline. After adding GAM, the OA score and mAcc score of each baseline method are increased significantly. Experiment results demonstrate that our proposed GAM has the potential to be widely used in the point cloud domain. Besides, results in terms of throughput (TP) indicate that computation cost barely increases after adding GAM.

Result on 3D Object Detection

For 3D object detection task, experiments are conducted on the KITTI dataset (Geiger et al. 2013), which has three detection categories, cars, pedestrians, and bicycles. Each category has three subsets, "easy", "medium" and "difficult", basing on the detection difficulty. The "medium" subset is the most commonly used for evaluation.

In Table 3, we report mean Inter-over-Union (mIoU) score on validation set of the KITTI dataset. After adding GAM, mIoU scores of pedestrian and bicycle category are improved by 0.4% and 1.34% respectively, while for car category mIoU score decreases by 0.41%. These results indicate that detection performance for small targets can be effectively improved with GAM.

Method	Input	OA	mAcc
KPConv (Thomas et al. 2019)	7k	92.9	-
Point Trans. (Zhao et al. 2021)	1k	93.7	90.6
CurveNet (Xiang et al. 2021)	1k	94.2	-
PointNet++(S) (Qi et al. 2017b)	1k	92.2	89.1
+ GAM	1k	92.8	91.5
PointNet++(M) (Qi et al. 2017b)	1k+N	92.8	90.7
+ GAM	1k+N	93.3	91.4
DGCNN (Wang et al. 2019b)	1k	92.9	90.2
+ GAM	1k	93.3	90.5
PointMLP* (Ma et al. 2022)	1k	94.5	91.4
+ GAM*	1k	**94.7**	**91.9**

Table 4: Comparison results on ModelNet40 dataset on 3D shape classification tasks. N indicates that the input point cloud contains normal vector information, * indicates that voting is used, S represents Single-Scale Grouping, and M represents Multi-Scale Grouping.

3D Shape Classification Experimental Results on ModelNet40

We conduct 3D shape classification experiments on ModelNet40 dataset (Wu et al. 2015), which has 12311 CAD samples, including 9843 training samples and 2468 test samples. The proposed GAM is compared with the state-of-the-art methods, which are KPConv (Thomas et al. 2019), Point Transformer (Zhao et al. 2021), CurveNet (Xiang et al. 2021), PointNet++(SSG) (Qi et al. 2017b), PointNet++(MSG) (Qi et al. 2017b), DGCNN (Wang et al. 2019b), and PointMLP (Ma et al. 2022). We select PointNet++(SSG), PointNet++(MSG), DGCNN, and PointMLP as baselines to evaluate effectiveness of our proposed method.

In Table 4, we report overall accuracy (OA) score and mean accuracy (mAcc) on the ModelNet40 dataset. After adding GAM, OA score and mAcc are increase significantly.

Method	Ins	TP
PointNet (Qi et al. 2017a)	83.7	-
Point Tran. (Zhao et al. 2021)	86.6	-
PointMLP (Ma et al. 2022)	86.1	-
PointNet++ (Qi et al. 2017b)	85.1	370
+ GAM	85.5	368
DGCNN (Wang et al. 2019b)	85.2	257
+ GAM	85.5	235
CurveNet (Xiang et al. 2021)*	86.8	104
+ GAM*	**87.0**	99

Table 5: Comparison results on the ShapeNet dataset for 3D part segmentation on different classes. Ins. represent the instance average Inter-over-Union. * denotes the use of voting.

		PointNet++		PointMLP	
Distance	Gradient	OA	mA	OA	mA
✗	✗	86.2	84.3	85.7	84.4
✓	✗	86.1	84.2	84.3	82.4
✗	✓	86.5	84.5	85.8	84.6
✓	✓	**87.0**	**85.8**	**86.1**	**84.7**

Table 6: Ablation study of our proposed GAM using different kinds of information on ScanObjectNN dataset.

Result on 3D Part Segmentation

We conduct 3D part segmentation experiments on the ShapeNet dataset (Yi et al. 2016) that has 16881 3D objects with 50 different categories of segmentation masks. We compare our GAM with the state-of-the-art methods, including PointNet (Qi et al. 2017a), Point Tran. (Zhao et al. 2021), PointMLP (Ma et al. 2022), PointNet++ (Qi et al. 2017b), DGCNN (Wang et al. 2019b) and CurveNet (Xiang et al. 2021). Each object was sampled to 2048 points.

In Table 5, we report the instruction mIou (Ins.) and throughput (TP) for each baseline method before and after adding GAM. The Ins. scores of baseline methods (i.e. PointNet++, DGCNN, and SOTA CurveNet) increase by 0.4%, 0.3%, and 0.2% after adding GAM, respectively.

Ablation Study

In order to verify the effectiveness of gradient information, two variants of GAM are created, one with distance information and the other with gradient information. Results of ablation experiments conducted on the ScanObjectNN dataset and S3DIS dataset are present in Table 6 and Table 7, respectively.

As shown in Table 6, after adding the GAM variant with only distance information for PointNet++ and PointMLP, OA scores decrease by 0.1% and 1.3%. After adding the GAM variant with only gradient information, the OA score increases by 0.3% and 0.1%, indicating that gradient information can constrain the local feature aggregation process more effectively. While with both distance and gradient information, the OA score can be significantly increased than single fine-grained information. Therefore, we conclude that the combination of various fine-grained geometric information is more effective in constraining the local aggregation process by using multiple geometric dimensions.

		PointNet++		DGCNN	
Distance	Gradient	mIoU	OA	mIoU	OA
✗	✗	53.5	83.0	47.9	83.6
✓	✗	54.1	**83.2**	49.3	84.4
✗	✓	54.5	83.0	49.7	84.5
✓	✓	**54.8**	**83.2**	**50.0**	**84.6**

Table 7: Ablation study of our proposed GAM using different kinds of information on S3DIS dataset Area5.

In Table 7, we report the results of the ablation study for our proposed GAM using different kinds of information on the S3DIS dataset Area5. It also demonstrates the superiority of gradient information over distance information, and the effectiveness of combining various fine-grained geometric information.

	Normal	Zenith & Azimuth
time (ms)	18.6	0.522

Table 8: Computation time of a single run before and after GAM simplification. Normal represents direct calculation of the normal vector for each neighborhood point, Zenith & Azimuth represents calculation of the explicit representation using the zenith angles and azimuth angles.

In three-dimensional space, the gradient of a three-dimensional function and the normal vector of a three-dimensional isosurface have different geometric meanings, but they are essentially the same. Therefore, we modify the local surface fitting method (Pauly 2003) to calculate the normal vector of a plane, which is formed by the projection of the center point on the X and Y circles and neighboring points. It is the same as the object meaning of the gradient calculated in GAM. The results are shown in Table 8. After simplification, calculation speed is about 35 times faster than the original method, which effectively alleviates the problem of slow inference speed after adding GAM.

Conclusion

In this paper, we propose an efficient, lightweight, and plug-and-play gradient attention module (GAM), in which gradient information is introduced into the local feature aggregator for the first time. Our proposed GAM solves the problem of the different importance of each neighborhood point in the local feature aggregation process, and brings fine-grained geometric information to the local aggregation process. The effective and efficient performance of our proposed GAM is verified by conducting comparison experiments on four tasks, including 3D point cloud shape classification, 3D part segmentation, 3D semantic segmentation, and 3D object detection. It is our expectation that this work can promote further research on local feature aggregation descriptors.

Acknowledgements

This work was supported by a grant from Zhejiang Leapmotor Technology CO., LTD, China.

References

Armeni, I.; Sener, O.; Zamir, A. R.; Jiang, H.; Brilakis, I.; Fischer, M.; and Savarese, S. 2016. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1534–1543.

Berg, A.; Oskarsson, M.; and O'Connor, M. 2022. Points to Patches: Enabling the Use of Self-Attention for 3D Shape Recognition. *arXiv preprint arXiv:2204.03957*.

Boulch, A. 2020. ConvPoint: Continuous convolutions for point cloud processing. *Computers & Graphics*, 88: 24–34.

Chen, C.; Luca, Z. F.; and Antonios, T. 2021. GAPointNet: Graph attention based point neural network for exploiting local feature of point cloud. *Neurocomputing*, 122–132.

Cheng, S.; Chen, X.; He, X.; Liu, Z.; and Bai, X. 2021. Pranet: Point relation-aware network for 3d point cloud analysis. *IEEE Transactions on Image Processing*, 30: 4436–4448.

Cui, Y.; Liu, X.; Liu, H.; Zhang, J.; Zare, A.; and Fan, B. 2021. Geometric attentional dynamic graph convolutional neural networks for point cloud analysis. *Neurocomputing*, 300–310.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11): 1231–1237.

Guo, M.-H.; Cai, J.-X.; Liu, Z.-N.; Mu, T.-J.; Martin, R. R.; and Hu, S.-M. 2021. Pct: Point cloud transformer. *Computational Visual Media*, 7(2): 187–199.

Hu, Q.; Yang, B.; Xie, L.; Rosa, S.; Guo, Y.; Wang, Z.; Trigoni, N.; and Markham, A. 2020. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11108–11117.

Komarichev, A.; Zhong, Z.; and Hua, J. 2019. A-cnn: Annularly convolutional neural networks on point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 7421–7430.

Lan, S.; Yu, R.; Yu, G.; and Davis, L. S. 2019. Modeling local geometric structure of 3d point clouds using geo-cnn. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, 998–1008.

Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; and Chen, B. 2018. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31.

Liang, M.; Yang, B.; Wang, S.; and Urtasun, R. 2018. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European conference on computer vision (ECCV)*, 641–656.

Ma, X.; Qin, C.; You, H.; Ran, H.; and Fu, Y. 2022. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. *arXiv preprint arXiv:2202.07123*.

Pauly, M. 2003. *Point primitives for interactive modeling and processing of 3D geometry*. Citeseer.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

Qian, G.; Li, Y.; Peng, H.; Mai, J.; Hammoud, H. A. A. K.; Elhoseiny, M.; and Ghanem, B. 2022. PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies. *arXiv preprint arXiv:2206.04670*.

Qiu, S.; Anwar, S.; and Barnes, N. 2021. Semantic segmentation for real point cloud scenes via bilateral augmentation and adaptive fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1757–1767.

Qiu, S.; Anwar, S.; and Barnes, N. 2022. Geometric back-projection network for point cloud classification. *IEEE Transactions on Multimedia*, 24: 1943–1955.

Ran, H.; Liu, J.; and Wang, C. 2022. Surface Representation for Point Clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 18942–18952.

Su, H.; Maji, S.; Kalogerakis, E.; and Learned-Miller, E. 2015. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, 945–953.

Thomas, H.; Qi, C. R.; Deschaud, J.-E.; Marcotegui, B.; Goulette, F.; and Guibas, L. J. 2019. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, 6411–6420.

Uy, M. A.; Pham, Q.-H.; Hua, B.-S.; Nguyen, T.; and Yeung, S.-K. 2019. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, 1588–1597.

Wang, J.; Cui, Y.; Guo, D.; Li, j.; Liu, Q.; and Shen, C. 2022. PointAttN: You Only Need Attention for Point Cloud Completion. *arXiv preprint arXiv:2203.08485*.

Wang, L.; Huang, Y.; Hou, Y.; Zhang, S.; and Jie, S. 2019a. Graph attention convolution for point cloud semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10296–10305.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019b. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5): 1–12.

Wei, X.; Yu, R.; and Sun, J. 2020. View-gcn: View-based graph convolutional network for 3d shape analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1850–1859.

Wu, H.; Deng, J.; Wen, C.; Li, X.; Cheng, W.; and Li, J. 2022. CasA: A Cascade Attention Network for 3-D Object Detection From LiDAR Point Clouds. *IEEE Transactions on Geoscience and Remote Sensing*, 1–11.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1912–1920.

Xiang, T.; Zhang, C.; Song, Y.; Yu, J.; and Cai, W. 2021. Walk in the cloud: Learning curves for point clouds shape analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 915–924.

Xu, Q.; Sun, X.; Wu, C.-Y.; Wang, P.; and Neumann, U. 2020. Grid-gcn for fast and scalable point cloud learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5661–5670.

Yan, Y.; Mao, Y.; and Li, B. 2018. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10): 3337.

Yang, J.; Zhang, Q.; Ni, B.; Li, L.; Liu, J.; Zhou, M.; and Tian, Q. 2019. Modeling point clouds with self-attention and gumbel subset sampling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 3323–3332.

Yi, L.; Kim, V. G.; Ceylan, D.; Shen, I.-C.; Yan, M.; Su, H.; Lu, C.; Huang, Q.; Sheffer, A.; and Guibas, L. 2016. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6): 1–12.

Zhang, Y.; Hu, Q.; Xu, G.; Ma, Y.; Wan, J.; and Guo, Y. 2022. Not all points are equal: Learning highly efficient point-based detectors for 3d lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 18953–18962.

Zhao, H.; Jiang, L.; Fu, C.-W.; and Jia, J. 2019. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 5565–5573.

Zhao, H.; Jiang, L.; Jia, J.; Torr, P. H.; and Koltun, V. 2021. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 16259–16268.

Zhou, Y.; and Tuzel, O. 2018. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4490–4499.