

GeoFormer: Geometry Point Encoder for 3D Object Detection with Graph-based Transformer

Xin Jin^{♣1,3♥} Haisheng Su^{♣2,3✉} Cong Ma^{♣3} Kai Liu³ Wei Wu³ Fei Hui^{1✉} Junchi Yan^{2✉}
¹Chang’an University, ²Shanghai Jiao Tong University, ³SenseAuto Research
 {suhaiseng, yanjunchi}@sjtu.edu.cn, {jinxin, feihui}@chd.edu.cn

Abstract

Lidar-based 3D detection is one of the most popular research fields in autonomous driving. 3D detectors typically detect specific targets in a scene according to the pattern formed by the spatial distribution of point clouds. However, existing voxel-based methods usually adopt MLP and global pooling (e.g., PointNet, CenterPoint) as voxel feature encoder, which makes it less effective to extract detailed spatial structure information from raw points, leading to information loss and inferior performance. In this paper, we propose a novel graph-based transformer to encode voxel features by condensing the full and detailed point’s geometry, termed as GeoFormer. We first represent points within a voxel as a graph, based on relative distances to capture its spatial geometry. Then, We introduce a geometry-guided transformer architecture to encode voxel features, where the adjacent geometric clues are used to re-weight point feature similarities, enabling more effective extraction of geometric relationships between point pairs at varying distances. We highlight that GeoFormer is a plug-and-play module which can be seamlessly integrated to enhance the performance of existing voxel-based detectors. Extensive experiments conducted on three popular outdoor datasets demonstrate that our GeoFormer achieves the start-of-the-art performance on both effectiveness and robustness comparisons.

1. Introduction

LiDAR sensors have been widely used in the field of Autonomous Driving (AD), and high-precision 3D object detection has received a great deal of attention [36, 37, 42]. In contrast to image-based 2D object detection, point cloud data encompasses a larger spatial range and is disordered yet sparse. Therefore, how to effectively extract the patterns and rules constructed by the 3D point cloud from it has been the key to studying the 3D perception task.

♣ Equal Contribution. ✉ Corresponding Authors.
 ♥ Work done during an internship at SenseAuto Research.

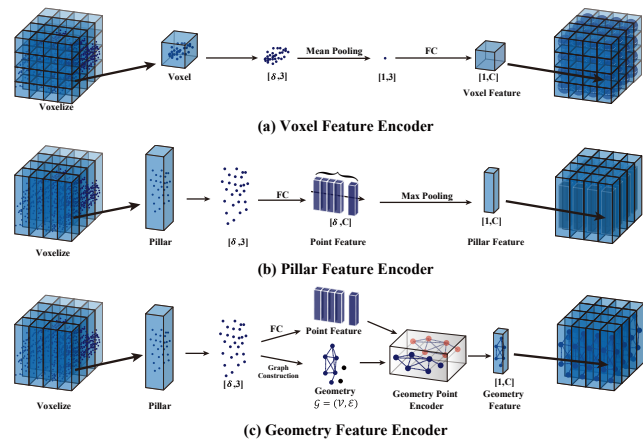


Figure 1. (a) **Voxel Feature Encoder** [34]: average pooling then MLP encoding (for smaller voxels); (b) **Pillar Feature Encoder** [21]: maximum pooling after MLP encoding (for larger voxels); (c) **Geometry Feature Encoder** (pillar version): geometric-guided feature extraction of spatial structure.

Existing methods can generally be classified into two mainstreams, the first one represents point-based methods [4, 30, 31, 33, 50], which employ PointNet [30, 31] to directly process the raw point clouds in the down-sampled set. The other stream of voxel-based methods [5–7, 24, 32] usually grid the point cloud space to directly apply convolution for feature extraction. Point-based methods require a massive number of point set abstraction operations in 3D space, leading to difficulties in meeting the demands of outdoor detection tasks in real-time. And voxel-based methods are widely used due to their computational efficiency.

Current voxel-based approaches mainly focus on how to effectively utilize voxel features and various architectural innovations have been made at the 3D Backbone, including CNN [53], Transformer [43], and RNN [26, 54]. However, how to encoding voxel features effectively is also a crucial task that should not be neglected [19, 22, 25], as it directly handles the structural information of raw point clouds. Most of existing methods either adopt global pooling or utilizes a

simple PointNet [30, 31] for voxel feature encoding, without considering finer-grained spatial local details. Moreover, extensive pooling operations inevitably result in the loss of structural information [58], especially with larger voxel size. This limitation constrains the perceptual performance upper-limit for voxel-based detectors. As shown in Fig. 1, previous methods either [6, 20, 29, 34, 35] directly conduct average pooling (Fig. 1(a)) over the intra-voxel points or perform [21, 43, 49, 59] point-level feature projection through a fully connected layer followed by voxel-level max pooling (Fig. 1(b)), causing inferior performance owing to the missing details of spatial structure. There are also some attempts [16, 19, 22, 25, 57] aiming to encode voxels more effectively, but they adopt either point-wise or channel-wise CNN attention to enhance the point feature representation implicitly, without explicitly modeling the voxel-wise geometry structure, which is crucial for subsequent perceptual tasks.

Therefore, on the purpose of encoding geometric relations from point clouds, we propose **GeoFormer** based on Transformer architecture [41]. CNN-based approaches are limited by the dynamic number of input points within voxels, however, in the field of NLP, the Transformer has been proven to specialize in handling the indefinite-length input sequences. Moreover, benefiting from Transformer’s inherent permutation invariance, it is well-suited to capturing complex geometric relationships, as the order of points does not affect their underlying geometry.

To better capture the spatial structure of point clouds, we organize them with a geometry graph and propose a geometry-guided transformer (GeoFormer) to model the relations of intra-voxel points from the aspects of explicit distance prior and implicit feature similarity, which is intuitive yet effective for LiDAR 3D detection. Specifically, we first construct the voxel graph based on the relative distances among intra-voxel points, which is used as input for the GeoFormer. Inspired by the Graph Transformer [10], we use normalized adjacent geometric matrix to re-weight the calculated feature similarity, explicitly reducing the negative impact of irrelevant points, as points that are farther apart generally lack clear geometric connections. Thanks to GeoFormer’s powerful encoding capabilities and flexible application, we can seamlessly integrate it into a wide range of existing voxel-based detectors. Despite the diverse 3D backbones employed by various detectors for feature extraction, GeoFormer consistently delivers performance enhancements, underscoring its intuitive yet highly effective design. Additionally, the voxel features encoded by GeoFormer demonstrate strong robustness to varying voxel sizes. We conduct experiments on three widely recognized datasets respectively. Finally, GeoFormer achieves 72.4 NDS and 68.2 mAP on the nuScenes dataset, and obtains a significant improvement on the larger-scale Waymo Open

Dataset and Argoverse 2 Dataset as well.

In sum, our main contributions are three folds as follows:

- We propose a plug-and-play Geometry Point Encoder to enhance the voxel feature extraction with geometry-guided interaction modeling, which can be seamlessly integrated into existing voxel-based detectors for better 3D detection performance.
- We design a GeoFormer to encode both spatial geometry and feature similarity of paring points within each voxel, thus facilitating further voxel relation modeling without spatial structure loss.
- Extensive experiments are conducted on three popular datasets (nuScenes, Waymo Open, Argoverse 2), showcasing the superior performance and great robustness of our GeoFormer for LiDAR-based 3D object detection.

2. Related Work

2.1. 3D Object Detection Based on Point Clouds

Previous 3D object detectors based on point cloud can be generalized into two categories, point-based and voxel-based methods. Benefiting from the PointNet series [30, 31] network, the former [4, 33, 47, 50, 55] directly extracts geometric features from the downsampled point cloud set, but the downsampling and the multi-scale group aggregation bring about a huge amount of computation. Therefore, the latter method [1, 5, 12, 21, 24, 34, 35, 48, 59] is usually adopted in outdoor self-driving scenarios, which divides the point cloud into voxels, and then convolution can be applied for global feature extraction. SECOND [48] introduces sparse convolution, which greatly improves the processing efficiency, and FocalConv [5] proposes a new sparse convolution operator, which learns the positional importance in sparse convolution; Largekernel3D [6] designed a large convolution kernel in 3D space, demonstrating the feasibility of large kernels in 3D vision tasks. However, all these methods proceed to enhance the feature representation of the point cloud after voxelization, neglecting the information lost during the voxelization process. To address this issue, we propose a novel voxel feature encoding module to enhance the upper bound of detection accuracy, starting from encoding the geometric representation of the point cloud in the original voxel.

2.2. 3D Object Detection with Transformer

With the successful development of transformer in 2D tasks as well as NLP, many works [8, 20, 39, 43, 44, 49, 56, 60] explore its capability in 3D point cloud object detection tasks. VoTr [29] implements a fast voxel query to construct local attention and dilated attention. conQueR [61] employs query contrast to enhance DETR-based [3] sparse detection, optimizing the two-stage refinement process by efficiently merging multi-scale contextual information with

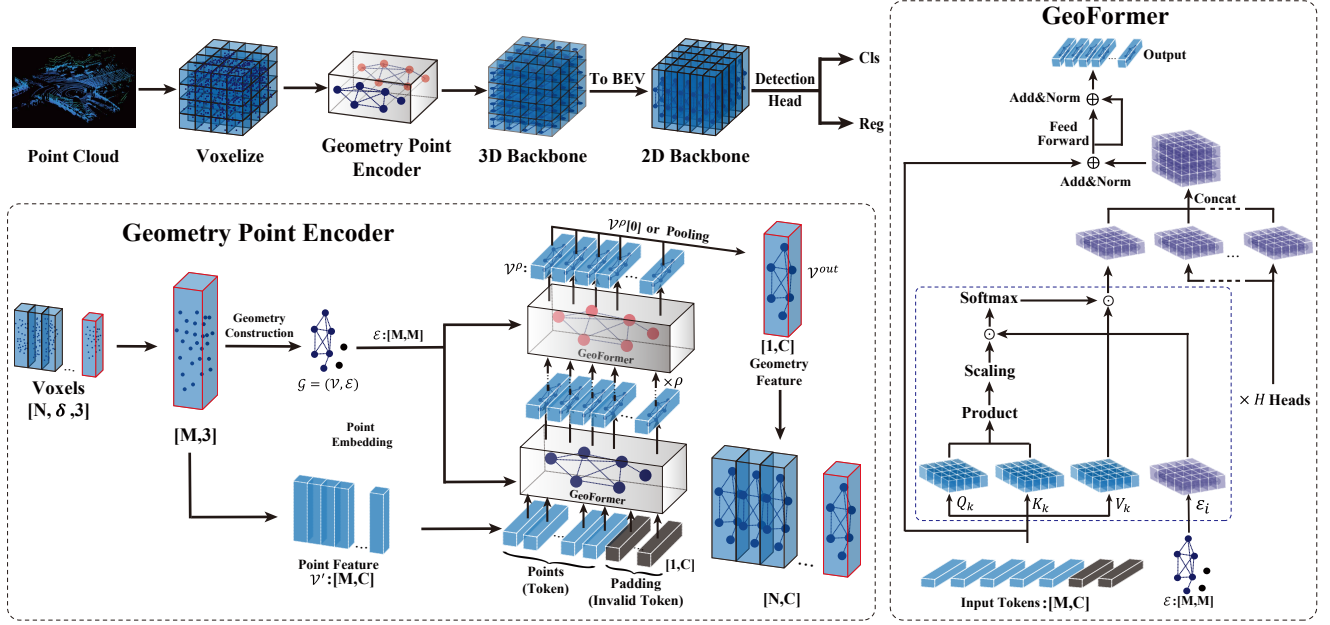


Figure 2. The overall architecture of our proposed GeoFormer. The input point clouds are voxelized and fed into the Geometry Point Encoder to extract the geometric information of raw points within each voxel. Subsequently, the voxel features (or pillar features) encoding high-dimensional geometric information can be integrated with any existing voxel-based framework to obtain final detection results.

a redesigned channel decoding module; SphereFormer [20] expands the receptive field by dividing a narrow window over a long window to improve the performance of sparse distant points; DSVT [43] proposes a dynamic sparse window attention that applies a rotated set of partitions to improve the feature interactions. Although all of these explorations make Transformer achieve exciting results in point cloud processing, they make efforts in 3D backbone and two-stage refinement without considering the relationship of the original point in space and shape information.

3. Method

In this section, we introduce the technical details of GeoFormer. The overall framework is depicted in Fig. 2, which consists of two main core components. Firstly, the Geometry Point Encoder (GPE) proposes to use graph structure to represent spatial relations of pairing points with each voxel. The other core design is GeoFormer, which introduces an geometry-guided multi-head attention mechanism to better extract the geometric structure details and preserve the spatial connections rather than global pooling directly.

3.1. Geometry Point Encoder

Geometry Construction. The point clouds $P \in \mathbb{R}^{n \times 3}$ are partitioned into structured voxels of size $W \times H$, a set of the voxels $\Psi \in \mathbb{R}^{N \times \delta \times 3}$, where N indicates the number of non-empty voxels $\Psi = \{\mathcal{V}_i, i \in [1, N]\}$, and each voxel $\mathcal{V} : \{v_j : (x_j, y_j, z_j), j \in [1, \delta]\}$ contains δ points. Due to

the varying number of point clouds contained in each voxel \mathcal{V}_i , we define the max number for each voxel as M . And voxel \mathcal{V}_i is padded if the number of points in \mathcal{V}_i is less than M . Subsequently, we construct the points within \mathcal{V}_i as a graph $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)$, where $\mathcal{V}_i \in \mathbb{R}^{M \times 3}$ represents the set of nodes from the graph \mathcal{G}_i , which is equivalent to the i -th voxel. $\mathcal{E}_i : \{e_{jk} \in [0, 1], j, k \in [1, M]\} \in \mathbb{R}^{M \times M}$ denotes the connection relationships between nodes, where the padded points $\mathcal{V}_i^p : \{v_j, j \in [\delta, M - \delta]\}$ are not connected to other points, thus $\{e_{jk}, e_{kj}, j \in [\delta, M - \delta], k \in [1, M]\}$ in \mathcal{E}_i are set as σ , typically, σ is a very small positive value. The edge values for other nodes are set as $[0, 1]$ according to the relative distance between two nodes. The graph construction strategy is defined as:

$$e_{ij} = \begin{cases} 1, & d_{ij} < \theta_{min} \\ \frac{d_{ij} - \theta_{max}}{\theta_{min} - \theta_{max}}, & d_{ij} \in [\theta_{min}, \theta_{max}] \\ 0, & d_{ij} > \theta_{max} \end{cases} \quad (1)$$

where $d_{ij} = \|v_i - v_j\|$ denotes the relative distance between v_i and v_j . θ is a preset distance threshold. To better represent the geometric structure of the local voxel, the coordinates of all points will be transformed to the relative coordinate system (rx, ry, rz) based on the voxel center, which is defined as:

$$\bar{\mathcal{V}}_i = \{\bar{v}_j = [(\bar{x}_j, \bar{y}_j, \bar{z}_j)]\}, j \in [1, \delta] \\ (\bar{x}_j, \bar{y}_j, \bar{z}_j) = (x_j, y_j, z_j) - (c_x, c_y, c_z), \quad (2)$$

where (cx_i, cy_i, cz_i) refers to the coordinates of the center point corresponding to the i -th non-empty voxel \mathcal{V}_i .

Point Feature Encoding. Given Graph $\mathcal{G}_i = \{\mathcal{V}_i, \mathcal{E}_i\}$, $i \in [1, N]$. All of the nodes $\bar{\mathcal{V}}_i = \{\bar{v}_j\} \in \mathbb{R}^{M \times 3}$ are projected to a high-dimensional feature space, obtaining Point Feature $\mathcal{V}'_i \in \mathbb{R}^{M \times c}$, which is formulated as:

$$\mathcal{V}' = \left\{ \mathcal{V}'_i \mid \mathcal{V}'_i = FC_{3 \rightarrow c}(\bar{\mathcal{V}}_i) \right\} i \in [1, N] \quad (3)$$

where \mathcal{V}'_i denotes the graph node corresponding to the i -th non-empty voxel, and FC denotes the fully connected layer used for feature upscaling, which contains two layers and a GELU [17] activation. $\bar{\mathcal{V}}_i = \{\bar{v}_j\}$, $j \in [1, M]$ are utilized for the inputs of Transformer Encoder as a token sequence.

GeoFormer. We utilize transformer encoder architecture to extract geometric information represented by the graph structure \mathcal{G} corresponding to each voxel, enhancing the understanding of the point clouds scene. Specifically, taking the graph structure $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)$ corresponding to voxel \mathcal{V}_i as an example, we treat each node in $v' \in \mathcal{V}_i$ as a token input to a sequence of consecutive transformer blocks. Simultaneously, *the edges \mathcal{E}_i is used to dynamically control the attention connections between different nodes.* Inspired by PCT [15], we refrain from using additional positional encoding, assuming that the coordinate data of the point clouds already encompasses its positional information. All of the nodes \mathcal{V}_i are fed to GeoFormer. The computation of the multi-head attention in the first layer of the transformer can be summarized as follows:

$$\{Q, K, V\} = \{Lin_q(\mathcal{V}_i), Lin_k(\mathcal{V}_i), Lin_v(\mathcal{V}_i)\} \quad (4)$$

where $Q, K, V \in \mathbb{R}^{M \times c}$ are obtained by processing \mathcal{V}_i through three linear layers $Lin_*(\cdot)$, which indicates the query, key and value matrices, respectively. Q, K, V are split to n heads $\mathbb{R}^{M \times [h \times d]}$ to obtain $Q_k, K_k, V_k \in \mathbb{R}^{M \times h \times d}$, representing the features of the k -th head, respectively. Inspired by Graph Transformer [10], for the k -th head, we calculate the attention weight matrix $\phi(Q_k, K_k, \mathcal{E}_i) \in \mathbb{R}^{M \times d}$, which is defined as:

$$\phi(Q_k, K_k, \mathcal{E}_i) = softmax\left(\frac{Q_k \cdot K_k^T}{\sqrt{d}} \cdot \mathcal{E}_i\right) \quad (5)$$

where $\mathcal{E}_i \in \mathbb{R}^{M \times M}$ indicates the geometry structure of graph \mathcal{G}_i as mentioned in Eq.1, and then the output of multi-attention head $Y \in \mathbb{R}^{M \times d}$ is calculated as:

$$\begin{aligned} Y &= Lin_{out}(Concat[y_1, y_2, \dots, y_{h-1}]) \\ y_k &= \phi(Q_k, K_k, \mathcal{E}_i) \cdot V_k \end{aligned} \quad (6)$$

where $y_k \in \mathbb{R}^{M \times \frac{d}{h}}$ refers to the weighted feature output of the k -th head. Then the computation of a complete transformer block can be summarized as follows:

$$\begin{aligned} \mathcal{V}_i^0 &= \mathcal{V}_i, & \mathcal{V}_i &\in \mathcal{G}_i \\ \widehat{\mathcal{V}}_i^l &= MSA(LN(\mathcal{V}_i^{l-1}), E_i) + \mathcal{V}_i^{l-1}, & l &\in [1, \rho] \\ \mathcal{V}_i^l &= MLP(LN(\widehat{\mathcal{V}}_i^l)) + \widehat{\mathcal{V}}_i^l, & l &\in [1, \rho] \\ \mathcal{V}_i^{out} &= LN(\mathcal{V}_i^\rho[0]) \end{aligned} \quad (7)$$

where $LN(\cdot)$ denotes the layer norm, \mathcal{V}_i^0 represents the input of the initial transformer, ρ denotes the number layers of transformer blocks, $MSA(\cdot)$ is the standard Multi-head Self-Attention layer, $\widehat{\mathcal{V}}_i^l$ and \mathcal{V}_i^l denote the output of the $MSA(\cdot)$ and $MLP(\cdot)$ modules of the l -th block, respectively. The outputs of GPE are represented as $\mathcal{V}_i^\rho[j]$, $j \in [0, M-1]$. Similar to ViT [9], we consider that the points encoded by the transformer contain enough contextual information, so we simply take the output of the $\mathcal{V}_i^\rho[0]$ position to characterize the geometric information of the voxel \mathcal{V}_i , which is denoted as \mathcal{V}_i^{out} . And then, all of voxels are extracted features one-by-one and we obtain a series of geometric feature of voxels $\mathcal{V}^{out} : \{\mathcal{V}_{i1}^{out}, \mathcal{V}_{i2}^{out}, \dots, \mathcal{V}_N^{out}\}$.

3.2. Model Variant

As the number of point clouds contained within the voxel and the pillar usually differs significantly, two model variants are provided for training purposes. The first one is a base model for pillar-based detectors. It employs only one Geometry Point Encoder for encoding, i.e., for each non-empty voxel $\mathcal{V} \in \Psi$, it is encoded by the same GPE, yielding the output $\mathcal{V}^{out} = GPE(\mathcal{V})$, where $GPE(\cdot)$ denotes the Geometry Point Encoder.

Whereas, since voxels are usually smaller in size than pillars, and the point cloud is very sparse, the number of point clouds inside most non-empty voxels is less than 5 (about 80%). Therefore, we propose a simple partitional structure using two GPEs for training to avoid all voxels being trained with the maximum number of points M , which results in a large number of invalid tokens involved in the computation, and to improve the training efficiency. Specifically, the set of the input non-empty voxels \mathcal{V} , is divided into two voxel sets according to the number δ of voxel internal point clouds: $\mathcal{V}^1 = \{\mathcal{V}_i \mid \delta \leq \delta_{min}\}$, $\mathcal{V}^2 = \{\mathcal{V}_i \mid \delta > \delta_{min}\}$, where δ_{min} denotes the threshold of the number of point clouds used for partition. Subsequently, these two voxel sets are encoded by two GPEs respectively, and the final outputs are defined as follows:

$$\mathcal{V}^{out} = MLP([GPE^1(\mathcal{V}^1), GPE^2(\mathcal{V}^2)]) \quad (8)$$

where $GPE^1(\cdot)$, $GPE^2(\cdot)$ denotes two Geometry Point Encoders, and $MLP(\cdot)$ denotes a multilayer perceptron to unify the feature outputs of different encoders.

Table 1. Comparison experiments on the nuScenes without using any test-time augmentation.

| Method | Present at | mAP | NDS | Car | Truck | Bus | Trailer | C.V. | Ped. | Motor. | B.C. | T.C. | Barrier |
|-------------------------|------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Results on the val set | | | | | | | | | | | | | |
| PointPillars ([21]) | CVPR'2019 | 28.2 | 46.8 | 75.5 | 31.6 | 44.9 | 23.7 | 4.0 | 49.6 | 14.6 | 0.4 | 8.0 | 30.0 |
| CenterPoint ([51]) | CVPR'2021 | 59.2 | 66.5 | 84.9 | 57.4 | 70.7 | 38.1 | 16.9 | 85.1 | 59.0 | 42.0 | 69.8 | 68.3 |
| Transfusion-L ([1]) | CVPR'2022 | 65.5 | 70.1 | 86.9 | 60.8 | 73.1 | 43.4 | 25.2 | 87.5 | 72.9 | 57.3 | 77.2 | 70.3 |
| SASA ([4]) | AAAI'2022 | 45.0 | 61.0 | 76.8 | 45.0 | 66.2 | 36.5 | 16.1 | 69.1 | 39.6 | 16.9 | 29.9 | 53.6 |
| VoxelNeXt ([7]) | CVPR'2023 | 60.5 | 66.7 | 83.9 | 55.5 | 70.5 | 38.1 | 21.1 | 84.6 | 62.8 | 50.0 | 69.4 | 69.4 |
| PVT-SSD ([49]) | CVPR'2023 | 53.6 | 65.0 | 79.4 | 43.8 | 62.1 | 34.2 | 21.7 | 79.8 | 53.4 | 38.2 | 56.6 | 67.1 |
| PillarNeXt ([24]) | CVPR'2023 | 62.2 | 68.4 | 85.0 | 57.4 | 67.6 | 35.6 | 20.6 | 86.8 | 68.6 | 53.1 | 77.3 | 69.7 |
| SAFDNet ([53]) | CVPR'2024 | 66.3 | 71.0 | 87.6 | 60.8 | 78.0 | 43.5 | 26.6 | 87.8 | 75.5 | 58.0 | 75.0 | 69.7 |
| SEED ([27]) | ECCV'2024 | 66.2 | 71.2 | - | - | - | - | - | - | - | - | - | - |
| VoxelMamba ([54]) | NIPS'2024 | 67.5 | 71.9 | 87.9 | 62.8 | 76.8 | 45.9 | 24.9 | 89.3 | 77.1 | 58.6 | 80.1 | 71.5 |
| GeoFormer (Ours) | - | 68.2 | 72.4 | 88.2 | 64.5 | 78.1 | 45.7 | 27.4 | 89.7 | 74.8 | 59.6 | 80.3 | 73.8 |
| Results on the test set | | | | | | | | | | | | | |
| 3DSSD ([50]) | CVPR'2020 | 42.6 | 56.4 | 81.2 | 47.2 | 61.4 | 30.5 | 12.6 | 70.2 | 36.0 | 8.6 | 31.1 | 47.9 |
| AFDetV2 ([18]) | AAAI'2022 | 62.4 | 68.5 | 86.3 | 54.2 | 62.5 | 58.9 | 26.7 | 85.8 | 63.8 | 34.3 | 80.1 | 71.0 |
| PillarNet ([32]) | ECCV'2022 | 66.0 | 71.4 | 87.6 | 57.5 | 63.6 | 63.1 | 27.9 | 87.3 | 70.1 | 42.3 | 83.3 | 77.2 |
| Focals Conv ([5]) | CVPR'2022 | 63.8 | 70.0 | 86.7 | 56.3 | 67.7 | 59.5 | 23.8 | 87.5 | 64.5 | 36.3 | 81.4 | 74.1 |
| LargeKernel3D ([6]) | CVPR'2023 | 65.4 | 70.6 | 85.5 | 53.8 | 64.4 | 59.5 | 29.7 | 85.9 | 72.7 | 46.8 | 79.9 | 75.5 |
| HEDNet ([52]) | NIPS'2023 | 67.7 | 72.0 | 87.1 | 56.5 | 70.0 | 63.5 | 33.6 | 87.9 | 70.4 | 44.8 | 85.1 | 78.1 |
| SAFDNet ([53]) | CVPR'2024 | 68.3 | 72.3 | 87.3 | 57.3 | 68.0 | 63.7 | 37.3 | 89.0 | 71.1 | 44.8 | 84.9 | 79.5 |
| VoxelMamba ([54]) | NIPS'2024 | 69.0 | 73.0 | 86.8 | 57.1 | 68.0 | 63.2 | 35.4 | 89.5 | 74.7 | 50.8 | 86.9 | 77.3 |
| GeoFormer (Ours) | - | 69.8 | 73.7 | 87.5 | 60.3 | 69.8 | 65.8 | 36.3 | 90.2 | 73.3 | 49.5 | 86.7 | 78.5 |

3.3. Detection Head and Loss

GeoFormer is a generalized voxel-based point cloud feature extraction backbone that can be applied to most voxel-based detectors. Therefore, based on previous experience, for the nuScenes [2] and Waymo Open dataset [38], we use Transfusion’s detection head [1] and Centerpoint’s detection head [51] respectively. And for Argoverse 2 [45], we use the heatmap-based sparse center detection head consistent with SAFDNet [53]. The final loss function is defined as:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{cls} + \lambda_2 \mathcal{L}_{reg} + \lambda_3 \mathcal{L}_{iou} \quad (9)$$

where $\lambda_1, \lambda_2, \lambda_3$ denotes the weighting factor of the various loss items, \mathcal{L}_{cls} uses the binary cross-entropy loss, \mathcal{L}_{reg} is the regression loss calculated using L1, and \mathcal{L}_{iou} is the IOU loss of the prediction boxes. Additionally, following [18, 32], we add the IoU-aware-rectification scheme that takes into account the IoU information in the confidence score.

4. Experiments

4.1. Datasets and Metrics

nuScenes [2] is a challenging outdoor dataset that provides 1000 scenarios, of which 700 are used for training, 150 for validation, and 150 for testing. It provides a variety of 10 different categories of object annotations covering most of the traffic participants. The size varies greatly between categories, with large objects like trailers and small objects like traffic cones, bringing great challenges to detection. We

evaluate the performance of our approach using officially given evaluation tools, reporting NDS and mAP, taking into account the average detection accuracy and other aspects of the error (e.g., direction, speed).

Waymo Open Dataset (WOD) [38] is a large-scale dataset, which consists of 798 training scenarios and 202 validation scenarios, containing 160K and 40K samples respectively. The mAP and mAPH are provided to evaluate the detection performance, where mAPH refers to the mean average precision weighted by heading. Two different detection difficulties are defined according to the sparsity of the point cloud in the bounding box: LEVEL_1 (L1) is the bounding box with more than 5 points, and LEVEL_2 (L2) is the bounding box with 1-5 points.

Argoverse 2 [45] is receiving increasing attention for its large detection range ($200m \times 200m$). It contains a total of 1000 sequences, of which 700 sequences are used for training and 150 sequences for validation. Moreover, it contains 26 classes of annotations, posing great challenges. As in previous approaches [26, 53], we use mAP as the evaluation metric.

4.2. Implementation Details

Network Implementation. For nuScenes, we use the grid size of $[0.3, 0.3, 0.25]$ to voxelize the scene in the range $[-54.0, -54.0, -5.0, 54.0, 54.0, 3.0]$. We set the number of nodes $M = 32$ of each Voxel-generated graph, and random sampling will be performed for the cases where the points within the voxel are larger than M . Each node is dimension-

Table 2. Comparison experiments on the Waymo Open validation set (100% training data). All of these models are trained with single-frame inputs and no additional test-time augmentation is added. The best results in each category are shown in **bold**. ‘‘Ped.’’ denotes Pedestrian, and ‘‘Cyc.’’ denotes Cyclist.

| Method | Present at | Stages | Vehicle(L1) mAP/mAPH | Vehicle(L2) mAP/mAPH | Ped.(L1) mAP/mAPH | Ped.(L2) mAP/mAPH | Cyc.(L1) mAP/mAPH | Cyc.(L2) mAP/mAPH |
|---------------------|--------------|--------|-------------------------|-------------------------|----------------------|----------------------|----------------------|----------------------|
| PointPillar ([21]) | CVPR’2019 | one | 70.43/69.83 | 62.18/61.64 | 66.21/46.32 | 58.18/40.64 | 55.26/51.75 | 53.18/49.80 |
| SECOND ([48]) | Sensors’2018 | one | 70.96/70.34 | 62.58/62.02 | 65.23/54.24 | 57.22/47.49 | 57.13/55.62 | 54.97/53.53 |
| CenterPoint ([51]) | CVPR’2021 | two | 76.70/76.20 | 68.80/68.30 | 79.00/72.90 | 71.00/65.30 | -/- | -/- |
| Voxset ([16]) | CVPR’2022 | one | 74.50/74.00 | 66.00/65.60 | 80.00/72.40 | 72.50/65.40 | 71.60/70.30 | 69.00/67.70 |
| SST ([11]) | CVPR’2022 | one | 75.13/74.64 | 66.61/66.17 | 80.07/72.12 | 72.38/65.01 | 71.49/70.20 | 68.85/67.61 |
| AFDetV2 ([18]) | AAAI’2022 | one | 77.64/77.14 | 69.68/69.22 | 80.19/72.62 | 72.16/66.95 | 73.72/72.74 | 71.06/70.12 |
| PillarNet ([32]) | ECCV’2022 | one | 79.09/78.59 | 70.92/70.46 | 80.59/74.01 | 72.28/66.17 | 72.29/71.21 | 69.72/68.67 |
| CenterFormer ([60]) | ECCV’2022 | two | 75.00/74.40 | 69.90/69.40 | 78.00/72.40 | 73.10/67.70 | 73.80/72.70 | 71.30/70.20 |
| PV_RCNN++ ([35]) | IJCV’2023 | two | 79.25/78.78 | 70.61/70.18 | 81.83/76.28 | 73.17/68.00 | 73.72/72.66 | 71.21/70.19 |
| SWFormer ([39]) | ECCV’2022 | one | 77.80/77.30 | 69.20/68.80 | 80.90/72.70 | 72.50/64.90 | -/- | -/- |
| VoxelNext ([7]) | CVPR’2023 | one | 78.20/77.70 | 69.90/69.40 | 81.50/76.30 | 73.50/68.60 | 76.10/74.90 | 73.30/72.20 |
| OcTr ([56]) | CVPR’2023 | two | 79.20/78.70 | 70.80/70.40 | 82.20/76.30 | 74.00/68.50 | 73.90/72.80 | 71.10/69.20 |
| DSVT-P ([43]) | CVPR’2023 | one | 79.30/78.80 | 70.90/70.50 | 82.80/77.00 | 75.20/69.80 | 76.40/75.40 | 73.60/72.70 |
| MsSVT++ ([23]) | TPAMI’2023 | one | 78.96/78.39 | 70.57/70.01 | 80.64/73.78 | 73.12/66.80 | 75.98/74.73 | 72.97/71.82 |
| PTv3 ([46]) | CVPR’2024 | one | -/- | 71.20/70.80 | -/- | 76.30/70.40 | -/- | 71.50/70.40 |
| VoxelMamba ([54]) | NIPS’2024 | one | 80.80/80.30 | 72.60/72.20 | 85.00/80.80 | 77.70/73.60 | 78.60/77.60 | 75.70/74.80 |
| GeoFormer(ours) | - | one | 80.48/79.98 | 72.21/71.75 | 85.57/81.20 | 78.32/74.03 | 79.54/78.27 | 76.42/75.61 |

Table 3. Comparison experiments on the Argoverse 2 val set.

| Method | mAP | Veh. | Bus | Ped. | Truck | Bicyclist |
|--------------------|-------------|-------------|-------------|-------------|-------------|-------------|
| CenterPoint ([51]) | 22.0 | 67.6 | 38.9 | 46.5 | 22.1 | 20.1 |
| FSDv1 ([12]) | 28.2 | 68.1 | 40.9 | 59.0 | 21.1 | 33.4 |
| VoxelNeXt ([7]) | 30.7 | 72.7 | 38.8 | 63.2 | 16.9 | 32.4 |
| HEDNet ([52]) | 37.1 | 78.2 | 47.7 | 67.6 | 21.6 | 38.7 |
| FSDv2 ([13]) | 37.6 | 77.0 | 47.6 | 70.5 | 24.0 | 45.9 |
| SAFDNet ([53]) | 39.7 | 78.5 | 49.4 | 70.7 | 23.6 | 42.7 |
| LION-Mamba ([26]) | 41.5 | 75.1 | 43.6 | 73.9 | 19.0 | 47.3 |
| GeoFormer (Ours) | 41.7 | 77.4 | 50.7 | 73.7 | 24.6 | 48.0 |

Table 4. Effectiveness of GeoFormer. The results are obtained using 20% of the training data.

| Method | 3D AP/APH(L2) | | |
|--------------------|--------------------|--------------------|--------------------|
| | Vehicle | Pedestrian | Cyclist |
| baseline (MLP) | 61.02/60.55 | 62.85/52.38 | 61.22/60.03 |
| Mamba ([14]) | 61.90/60.84 | 64.78/55.42 | 62.79/61.34 |
| Transformer ([41]) | 62.33/61.46 | 66.02/56.96 | 64.19/62.70 |
| GeoFormer (Ours) | 62.95/61.87 | 67.67/58.80 | 65.04/63.59 |

ally expanded to $c = 128$ through a fully connected layer and serves as the input for the GeoFormer. We use two layers of GeoFormer, where the number of heads is 8. The detection framework and hyperparameters follow [26, 53].

Training & inference. Our experiments are conducted using the OpenPCDet framework [40]. The training optimizer utilizes AdamW [28], and we employ the OneCycle learning rate update strategy with initial learning rate, weight decay, and learning rate decay set to 0.003, 0.05, and 0.1, respectively. The remaining hyperparameters are kept consistent with [26]. We trained 48 epochs, 24 epochs, and 24 epochs for the nuScenes, Waymo Open, and Argoverse 2,

with batch sizes set to 2, respectively.

4.3. Comparison with State-of-the-art Methods

Results on nuScenes. As shown in Tab. 1, GeoFormer exhibits robust performance in the nuScenes. In the validation set, it achieves a 68.2 mAP and 72.4 NDS. Compared with TransFusion-L, which utilizes the same detection head, GeoFormer achieves a significant enhancement, with a higher NDS of 2.3 and a higher mAP of 2.7. It also demonstrates enhancement when compared to the newly published novel DETR-based detector SEED[27] and the sparse detector SAFDNet [53], with mAP improvements of 2.0 and 1.9, respectively. In the classification comparison with SAFDNet, GeoFormer achieves significant outperformance in the important objects of Car, Pedestrian, and Trailer, achieving 0.6, 1.9, and 2.2 improvements, respectively. These results demonstrate the importance and effectiveness of GeoFormer for detailed geometric information extraction.

Results on Waymo. As shown in Tab. 2, we present the comparative experimental results on the WOD. It can be observed that GeoFormer achieves state-of-the-art performance on WOD. For the newly proposed transformer-based architecture PTv3 [46], we outperform L2.mAP by 1.01, 2.02, and 4.92 for the Vehicle, Pedestrian, and Cyclist, respectively. Even for the For the two-stage detector OcTr [56], GeoFormer also demonstrates even better performance, with L2.mAP 4.32, and 5.32 higher for Pedestrian and Cyclist, which are more challenging objects, respectively. This indicates the effectiveness of our proposed model on challenging scenarios, demonstrating that GeoFormer, by learning geometric features of raw point clouds,

Table 5. Impact of GPE on the performance of existing Voxel-based detectors.

| Method | mAP | NDS |
|-----------------------|-------------------|-------------------|
| PillarNet ([32]) | 55.3 | 64.2 |
| PillarNet+GPE | 58.9(+3.6) | 66.1(+1.9) |
| DSVT ([43]) | 66.2 | 70.9 |
| DSVT+GPE | 66.5(+0.3) | 71.1(+0.2) |
| HEDNet ([52]) | 66.5 | 70.9 |
| HEDNet+GPE | 67.0(+0.5) | 71.4(+0.5) |
| SAFDNet ([53]) | 66.2 | 70.8 |
| SAFDNet+GPE | 66.7(+0.5) | 71.4(+0.6) |
| VoxelMamba ([54]) | 67.2 | 71.5 |
| VoxelMamba+GPE | 68.2(+1.0) | 72.4(+0.9) |

can retain more fine-grained information. Even for objects with sparser point clouds, the remaining points can effectively showcase their geometric structures.

Results on Argoverse 2. As shown in Tab. 3, we present the evaluation results of GeoFormer on Argoverse 2, demonstrating the effectiveness of our method in ultra-long range detection scenarios. Among all previous detectors, our method shows the best performance. Compared to SAFDNet [53] in which we use the same sparse detection head, GeoFormer achieves 2.0 mAP improvement, demonstrating GeoFormer’s powerful detection capability.

4.4. Ablation Experiments

4.4.1. Effectiveness of GeoFormer

To demonstrate the effectiveness of our proposed GeoFormer in extracting geometric information, we constructed four different feature extractors for voxels for a comprehensive comparison: MLP, Mamba, Transformer, and GeoFormer. In these comparisons, MLP, Mamba, and Transformer represent direct inputs of tokens, formed by points within voxels after Point Embedding. As shown in Tab. 4, GeoFormer achieved the best results. We observe that using more powerful feature encoders consistently outperforms the original MLP-based method, proving that encoding finer-grained voxel features is crucial for subsequent detection tasks. Due to its inherent RNN architecture, Mamba’s performance is slightly inferior to Transformer, as it relies on the input sequence’s order. And compared to the original Transformer architecture, GeoFormer surpasses it in three categories of AP by 0.62, 1.65, and 0.85, respectively. These results demonstrate that GeoFormer enhances the expression of geometric information by introducing edge weighting.

4.4.2. Enhancements to Voxel-based Methods

To demonstrate the effectiveness of our model, we applied GPE to detection architectures with various backbone networks, including PillarNet, DSVT, HEDNet, SAFDNet,

Table 6. Ablation Experiments of Different Graph Construction Thresholds.

| θ_{min} | θ_{max} | Voxel Size | mAP | NDS |
|----------------|----------------|---------------|-------------|-------------|
| $Max(d_{ij})$ | - | | 66.4 | 70.8 |
| 0.2 | 0.5 | | 66.0 | 70.2 |
| 0.5 | 1.5 | (0.3,0.3,8.0) | 66.3 | 70.5 |
| 0.5 | 2.0 | | 66.7 | 71.4 |
| 0.8 | 2.0 | | 66.5 | 71.0 |

Table 7. Ablation Experiments of Previous Voxel Encoder. PP indicates CenterPoint-Pillar [51].

| Method | mAP | NDS | Latency | Param. |
|-----------------|-------------|-------------|--------------|----------------|
| PP ([21]) | 48.6 | 57.3 | 30 ms | 68.70 M |
| PP+Dyn ([32]) | 48.9 | 57.8 | 29 ms | 68.70 M |
| PP+TANet ([25]) | 49.9 | 58.5 | 34 ms | 68.77 M |
| PP+GPE | 51.8 | 60.5 | 38 ms | 69.70 M |

and VoxelMamba. To ensure a fair comparison, only the voxel feature coding module in both baselines is replaced with GPE, while the rest of the architecture remains unchanged. The results presented in Tab. 5 demonstrate that the incorporation of GPE leads to substantial performance enhancements across various detector architectures, including Transformer-based (e.g., DSVT), CNN-based (e.g., SAFDNet), and SSM-based (e.g., VoxelMamba) models. Specifically, in the PillarNet detector, a 3.6 mAP improvement is achieved. This highlights the generalizability of GPE as a “plug-and-play” module, capable of enhancing the performance of various voxel-based detectors.

4.4.3. Ablation Experiments of Different Graph Construction Thresholds

We compare different graph construction thresholds on the nuScenes. As shown in Tab. 6, GeoFormer achieves the highest accuracy when the threshold is set between [0.5, 2.0]. The term $Max(d_{ij})$ represents a fully connected graph, where all points are interconnected and the edges have a weight of 1. However, when the θ_{max} is reduced, many points become disconnected, leading to a decline in performance. This outcome suggests that long-range dependencies are essential for effectively extracting geometric information. Additionally, when the θ_{max} threshold remains fixed at 2, lowering the θ_{min} further enhances performance, indicating that considering relative distances in the computation of attention scores is beneficial.

4.4.4. Ablation Experiments of Previous Voxel Encoder

We compare our method with previous voxel encoders on the nuScenes, as shown in Tab. 7. GPE consistently outperforms current methods, exceeding the previous best by 1.9 mAP and 2.0 NDS on nuScenes. These results strongly validate the effectiveness of GeoFormer in extracting features by modeling the geometric relationships within voxel

Table 8. Ablation studies on the number of layers of the GeoFormer.

| Method | mAP | NDS | Latency | Param. |
|--------|-------------|-------------|--------------|-----------------|
| 1 | 51.2 | 60.0 | 36 ms | 68.93 MB |
| 2 | 51.8 | 60.5 | 38 ms | 69.70 MB |
| 3 | 51.8 | 60.5 | 41 ms | 70.47 MB |
| 4 | 51.7 | 60.4 | 46 ms | 71.24 MB |

Table 9. Experimental results at larger voxels. Here, “size” refers to the voxel size, “PP” refers to CenterPoint-Pillar [51].

| Method | Size | 3D AP/APH(L2) | | |
|--------------------|------|--------------------|----------------------|----------------------|
| | | Vehicle | Pedestrian | Cyclist |
| PP | | 56.91/56.36 | 58.66/48.19 | 57.01/55.66 |
| PP+GPE | 0.32 | 59.02/58.57 | 63.07/54.51 | 62.14/60.91 |
| Improvement | | +2.11/+2.21 | +4.41/+6.32 | +5.13/+5.25 |
| PP | | 57.69/57.05 | 50.24/39.30 | 44.66/41.63 |
| PP+GPE | 0.64 | 60.18/59.64 | 60.68/52.91 | 60.75/59.51 |
| Improvement | | +2.49/+2.59 | +10.44/+13.61 | +16.09/+17.88 |

point clouds, demonstrating a clear advantage over earlier approaches that rely on simply stacking attention layers.

4.4.5. Analysis of Model Efficiency

To evaluate the runtime efficiency of Geometry Point Encoder (GPE), we compared it with existing voxel encoders in terms of parameter count and latency. As shown in Tab. 7, GPE significantly enhances model performance while maintaining similar parameter counts and inference speeds to other encoders. Compared to TNet [25], GPE achieves a 1.9 mAP improvement with only a 4 ms increase in latency and an additional 0.93 MB in parameters. These results demonstrate that GPE can seamlessly replace existing voxel encoders without introducing significant computational overhead or parameter increases.

4.4.6. Ablations on the number of layers of GeoFormer

We analyzed the impact of different numbers of GeoFormer layers in GPE on model performance, and the results are shown in Tab. 8. We can notice that 2 layers are enough to model the geometric relations among intra-voxel points. Increasing the number of layers further does not significantly improve performance and instead introduces a substantial computational overhead.

4.4.7. Ablation Experiments of Different Voxel Sizes

We conducted ablation experiments to evaluate the impact of varying voxel sizes. As detailed in Tab. 9, we observe a pronounced accuracy decline in PP as the voxel size increases from 0.32 to 0.64, particularly for Pedestrian and Cyclist. However, the integration of our GPE significantly enhances performance, with AP improvements of 10.44 for Pedestrians and 16.09 for Cyclists at a voxel size of 0.64. These findings emphasize the critical role of robust voxel encoders in detection tasks and validate GPE’s efficacy in preserving intricate geometric structure information.



(a)



(b)

Figure 3. Visualization results on the WOD validation set.

4.5. Visualization

We visualize the detection results of Waymo Open datasets in Fig. 3. As shown in the red box in the figure, thanks to GeoFormer’s ability in efficiently encoding the geometric relations of raw point clouds, more fine-grained local spatial details are preserved, allowing it to effectively detect some hard cases.

5. Conclusion

In this paper, we propose GeoFormer, a 3D detector based on Graph-based Transformer. To better understand the point cloud scene, we propose the Geometry Point Encoder (GPE), a novel voxel feature extraction strategy to capture the geometric structure of points within voxels. We use graph representations to capture the relationships between points within voxels. And we introduce GeoFormer, which enhances attention scores by incorporating edge features, effectively preserving the crucial edge information. Additionally, GPE is an effective component in enhancing the capabilities of existing voxel-based methods, and it significantly outperforms previous voxel encoders. Finally, experimental results on several outdoor point cloud benchmarks demonstrate the effectiveness and robustness of GeoFormer. In the future, we are committed to exploring the potential of GeoFormer to further enhance feature extraction and improve the performance of 3D Detection.

6. Acknowledgments

This work was supported in part by the following grants: 1) National Natural Science Foundation of China (No. 52172380); 2) Key Research and Development Program of Shaanxi Province (No. 2024GX-ZDCYL-01-25); 3) Key Research and Development Program of Shaanxi Province (No. 2024CY-JJQ-55); 4) Shanghai Municipal Science and Technology Major Project (No. 2021SHZDZX0102).

References

- [1] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1090–1099, 2022. 2, 5
- [2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multi-modal dataset for autonomous driving. In *CVPR*, 2020. 5
- [3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 2
- [4] Chen Chen, Zhe Chen, Jing Zhang, and Dacheng Tao. Sasa: Semantics-augmented set abstraction for point-based 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 221–229, 2022. 1, 2, 5
- [5] Yukang Chen, Yanwei Li, Xiangyu Zhang, Jian Sun, and Jiaya Jia. Focal sparse convolutional networks for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5428–5437, 2022. 1, 2, 5
- [6] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Largekernel3d: Scaling up kernels in 3d sparse cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13488–13498, 2023. 2, 5
- [7] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Voxelnxt: Fully sparse voxelnet for 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21674–21683, 2023. 1, 5, 6
- [8] Yilun Chen, Zhiding Yu, Yukang Chen, Shiyi Lan, Anima Anandkumar, Jiaya Jia, and Jose M Alvarez. Focalformer3d: focusing on hard instance for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8394–8405, 2023. 2
- [9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4
- [10] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*, 2021. 2, 4
- [11] Lue Fan, Ziqi Pang, Tianyuan Zhang, Yu-Xiong Wang, Hang Zhao, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Embracing single stride 3d object detector with sparse transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8458–8468, 2022. 6
- [12] Lue Fan, Feng Wang, Naiyan Wang, and ZHAO-XIANG ZHANG. Fully sparse 3d object detection. *Advances in Neural Information Processing Systems*, 35:351–363, 2022. 2, 6
- [13] Lue Fan, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Fsd v2: Improving fully sparse 3d object detection with virtual voxels. *arXiv preprint arXiv:2308.03755*, 2023. 6
- [14] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. 6
- [15] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7:187–199, 2021. 4
- [16] Chenheng He, Ruihuang Li, Shuai Li, and Lei Zhang. Voxel set transformer: A set-to-set approach to 3d object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8417–8427, 2022. 2, 6
- [17] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 4
- [18] Yihan Hu, Zhuangzhuang Ding, Runzhou Ge, Wenxin Shao, Li Huang, Kun Li, and Qiang Liu. Afdetv2: Rethinking the necessity of the second stage for object detection from point clouds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 969–979, 2022. 5, 6
- [19] Xin Jin, Kai Liu, Cong Ma, Ruining Yang, Fei Hui, and Wei Wu. Swiftpillars: High-efficiency pillar encoder for lidar-based 3d detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2625–2633, 2024. 1, 2
- [20] Xin Lai, Yukang Chen, Fanbin Lu, Jianhui Liu, and Jiaya Jia. Spherical transformer for lidar-based 3d recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17545–17555, 2023. 2, 3
- [21] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019. 1, 2, 5, 6, 7
- [22] Zhaoqi Leng, Pei Sun, Tong He, Dragomir Anguelov, and Mingxing Tan. Pvtransformer: Point-to-voxel transformer for scalable 3d object detection. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4238–4244. IEEE, 2024. 1, 2
- [23] Jianan Li, Shaocong Dong, Lihe Ding, and Tingfa Xu. Mssvt++: Mixed-scale sparse voxel transformer with center voting for 3d object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–17, 2023. 6
- [24] Jinyu Li, Chenxu Luo, and Xiaodong Yang. Pillarnxt: Rethinking network designs for 3d object detection in lidar

- point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17567–17576, 2023. 1, 2, 5
- [25] Zhe Liu, Xin Zhao, Tengting Huang, Ruolan Hu, Yu Zhou, and Xiang Bai. Tanet: Robust 3d object detection from point clouds with triple attention. In *Proceedings of the AAAI conference on artificial intelligence*, pages 11677–11684, 2020. 1, 2, 7, 8
- [26] Zhe Liu, Jinghua Hou, Xinyu Wang, Xiaoqing Ye, Jingdong Wang, Hengshuang Zhao, and Xiang Bai. Lion: Linear group rnn for 3d object detection in point clouds. *arXiv*, 2024. 1, 5, 6
- [27] Zhe Liu, Jinghua Hou, Xiaoqing Ye, Tong Wang, Jingdong Wang, and Xiang Bai. Seed: A simple and effective 3d detr in point clouds. In *ECCV*, 2024. 5, 6
- [28] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6
- [29] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3164–3173, 2021. 2
- [30] 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. 1, 2
- [31] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 1, 2
- [32] Guangsheng Shi, Ruifeng Li, and Chao Ma. Pillarnet: Real-time and high-performance pillar-based 3d object detection. In *European Conference on Computer Vision*, pages 35–52. Springer, 2022. 1, 5, 6, 7
- [33] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779, 2019. 1, 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. 1, 2
- [35] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection. *International Journal of Computer Vision*, 131(2):531–551, 2023. 2, 6
- [36] Haisheng Su, Feixiang Song, Cong Ma, Wei Wu, and Junchi Yan. Robosense: Large-scale dataset and benchmark for ego-centric robot perception and navigation in crowded and unstructured environments. *arXiv preprint arXiv:2408.15503*, 2024. 1
- [37] Haisheng Su, Wei Wu, and Junchi Yan. Difs: Ego-centric fully sparse paradigm with uncertainty denoising and iterative refinement for efficient end-to-end autonomous driving. *arXiv preprint arXiv:2409.09777*, 2024. 1
- [38] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020. 5
- [39] Pei Sun, Mingxing Tan, Weiyue Wang, Chenxi Liu, Fei Xia, Zhaoqi Leng, and Dragomir Anguelov. Swformer: Sparse window transformer for 3d object detection in point clouds. In *European Conference on Computer Vision*, pages 426–442. Springer, 2022. 2, 6
- [40] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. <https://github.com/open-mmlab/OpenPCDet>, 2020. 6
- [41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2, 6
- [42] Dequan Wang, Coline Devin, Qi-Zhi Cai, Philipp Krähenbühl, and Trevor Darrell. Monocular plan view networks for autonomous driving. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2876–2883. IEEE, 2019. 1
- [43] Haiyang Wang, Chen Shi, Shaoshuai Shi, Meng Lei, Sen Wang, Di He, Bernt Schiele, and Liwei Wang. Dsvt: Dynamic sparse voxel transformer with rotated sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13520–13529, 2023. 1, 2, 3, 6, 7
- [44] Junyi Wang and Yue Qi. Multi-task learning and joint refinement between camera localization and object detection. *Computational Visual Media*, 10(5):993–1011, 2024. 2
- [45] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*, 2023. 5
- [46] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xi-hui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler, faster, stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 6
- [47] Mingye Xu, Zhipeng Zhou, Yali Wang, and Yu Qiao. Towards robustness and generalization of point cloud representation: A geometry coding method and a large-scale object-level dataset. *Computational Visual Media*, 10(1):27–43, 2024. 2
- [48] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 2, 6
- [49] Honghui Yang, Wenxiao Wang, Minghao Chen, Binbin Lin, Tong He, Hua Chen, Xiaofei He, and Wanli Ouyang. Pvt-ssd: Single-stage 3d object detector with point-voxel transformer.

- In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13476–13487, 2023. [2](#), [5](#)
- [50] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048, 2020. [1](#), [2](#), [5](#)
- [51] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793, 2021. [5](#), [6](#), [7](#), [8](#)
- [52] Gang Zhang, Chen Junnan, Guohuan Gao, Jianmin Li, and Xiaolin Hu. HEDNet: A hierarchical encoder-decoder network for 3d object detection in point clouds. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. [5](#), [6](#), [7](#)
- [53] Gang Zhang, Junnan Chen, Guohuan Gao, Jianmin Li, Si Liu, and Xiaolin Hu. Safdnet: A simple and effective network for fully sparse 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14477–14486, 2024. [1](#), [5](#), [6](#), [7](#)
- [54] Guowen Zhang, Lue Fan, Chenhang He, Zhen Lei, Zhaoxiang Zhang, and Lei Zhang. Voxel mamba: Group-free state space models for point cloud based 3d object detection. *arXiv preprint arXiv:2406.10700*, 2024. [1](#), [5](#), [6](#), [7](#)
- [55] Yifan Zhang, Qingyong Hu, Guoquan Xu, Yanxin Ma, Jianwei Wan, and Yulan Guo. 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*, pages 18953–18962, 2022. [2](#)
- [56] Chao Zhou, Yanan Zhang, Jiabin Chen, and Di Huang. Octree-based transformer for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5166–5175, 2023. [2](#), [6](#)
- [57] Sifan Zhou, Zhi Tian, Xiangxiang Chu, Xinyu Zhang, Bo Zhang, Xiaobo Lu, Chengjian Feng, Zequn Jie, Patrick Yin Chiang, and Lin Ma. Fastpillars: A deployment-friendly pillar-based 3d detector. *arXiv preprint arXiv:2302.02367*, 2023. [2](#)
- [58] Sifan Zhou, Zhihang Yuan, Dawei Yang, Xubin Wen, Xing Hu, Yuguang Shi, Ziyu Zhao, and Xiaobo Lu. Pillarhist: A quantization-aware pillar feature encoder based on height-aware histogram. *arXiv preprint arXiv:2405.18734*, 2024. [2](#)
- [59] 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](#)
- [60] Zixiang Zhou, Xiangchen Zhao, Yu Wang, Panqu Wang, and Hassan Foroosh. Centerformer: Center-based transformer for 3d object detection. In *European Conference on Computer Vision*, pages 496–513. Springer, 2022. [2](#), [6](#)
- [61] Benjin Zhu, Zhe Wang, Shaoshuai Shi, Hang Xu, Lanqing Hong, and Hongsheng Li. Conquer: Query contrast voxel-detr for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9296–9305, 2023. [2](#)