

Accurate and Real-Time 3D Pedestrian Detection Using an Efficient Attentive Pillar Network

Duy Tho Le , Graduate Student Member, IEEE, Hengcan Shi , Hamid RezaTofighi ,
and Jianfei Cai , Fellow, IEEE

Abstract—Efficiently and accurately detecting people from 3D point cloud data is of great importance in many robotic and autonomous driving applications. This fundamental perception task is still very challenging due to (i) significant deformations of human body pose and gesture over time and (ii) point cloud sparsity and scarcity for pedestrian objects. Recent efficient 3D object detection approaches rely on pillar features. However, these pillar features do not carry sufficient expressive representations to deal with all the aforementioned challenges in detecting people. To address this shortcoming, we first introduce a stackable Pillar Aware Attention (PAA) module to enhance pillar feature extraction while suppressing noises in point clouds. By integrating multi-point-channel-pooling, point-wise, channel-wise, and task-aware attention into a simple module, representation capabilities of pillar features are boosted while only requiring little additional computational resources. We also present Mini-BiFPN, a small yet effective feature network that creates bidirectional information flow and multi-level cross-scale feature fusion to better integrate multi-resolution features. Our proposed framework, namely PiFeNet, has been evaluated on three popular large-scale datasets for 3D pedestrian Detection, i.e. KITTI, JRDB, and nuScenes. It achieves state-of-the-art performance on KITTI Bird-eye-view (BEV) as well as JRDB, and competitive performance on nuScenes. Our approach is a real-time detector with 26 frame-per-second (FPS).

Index Terms—Deep learning for visual perception, recognition, 3D object detection, LiDAR, real-time.

I. INTRODUCTION

FAST and reliable pedestrian detection in 3D world is a fundamental step in a perception system, conducive to many robotic and autonomous driving applications such as human-robot interactions, service robots, and autonomous navigation in crowded human environments. The recent 3D pillar-based object detectors, e.g. [27], [30], [55] integrate pillar features, a down-sampled representation of point clouds, into their frameworks to detect objects. These approaches have shown auspicious results in detecting rigid and large objects such as vehicles. However, their effectiveness in detecting pedestrians is limited because of the two challenges.

Challenge 1: Weak expressive capability of pillar features for pedestrians. Pedestrians are not rigid bodies, hence they can

Manuscript received 22 August 2022; accepted 13 December 2022. Date of publication 30 December 2022; date of current version 20 January 2023. This letter was recommended for publication by Associate Editor S. Aakur and Editor C. Cadena Lerma upon evaluation of the reviewers' comments.

The authors are with the Department of Data Science and AI, Monash University, Clayton, VIC 3800, Australia (e-mail: duy.le1@monash.edu; hengcan.shi@monash.edu; hamid.rezatofighi@monash.edu; jianfei.cai@monash.edu).

The code for our PiFeNet is available at <https://github.com/ldtho/PiFeNet>
Digital Object Identifier 10.1109/LRA.2022.3233234

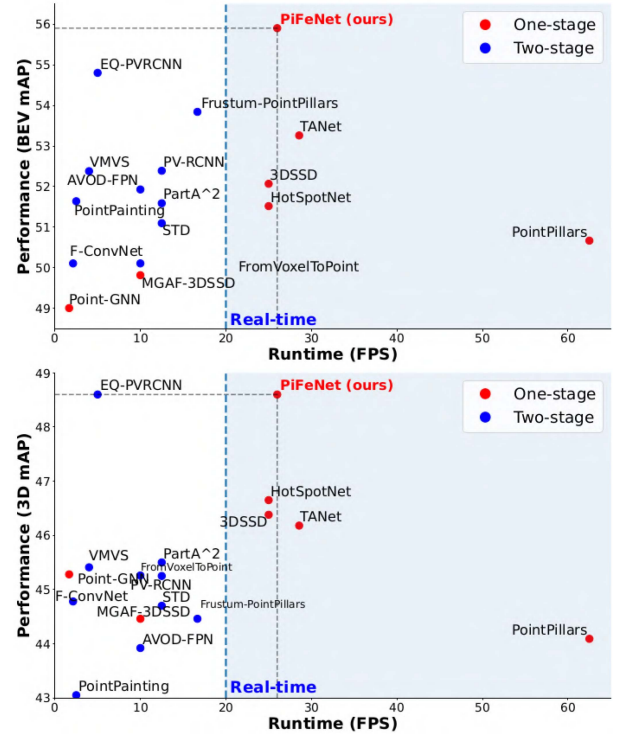


Fig. 1. The performance of our PiFeNet on KITTI [11] official test split on both pedestrian Birds-eye-view and 3D detection versus inference speed in FPS. Our approach achieved SOTA performance in detecting pedestrians. The comparison details are presented in Table 1.

take numerous poses (running, bowing, sitting, laying, etc.), as shown in Fig. 2 A. Previous methods such as Pointpillars [21] struggle distinguishing pedestrians from poles and trees (Fig. 7), due to 3D spatial information loss during quantisation process. They necessitate more representative pillar features to correctly distinguish pedestrians from other similarly-looking objects. To address this limitation, TANet [27] has incorporated a triple attention module to enhance pillar feature extraction. However, this approach only uses max pooling to capture context features, suppressing a lot of information from the points inside the pillar, which are informative for better person detection and localisation.

To retain important information while generalising to various pedestrian representations, detectors require to adopt comprehensive attention mechanisms in pillar features. To this end, we propose a **Pillar Aware Attention (PAA) module** that performs attention on pillars utilising improved attention

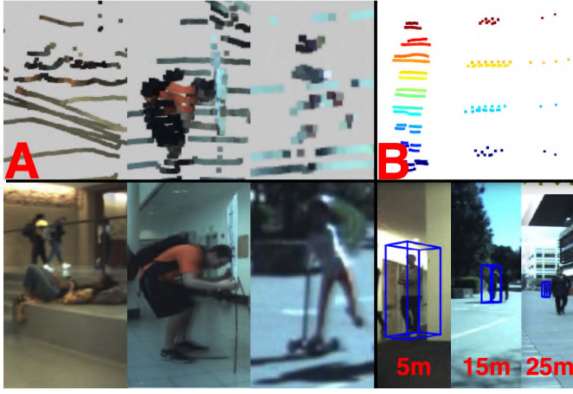


Fig. 2. Point cloud density of pedestrian instances with challenging postures (A), and at increasing distances to the LiDAR sensor (B) captured in the JRDB [29] dataset (point clouds are zoomed and coloured for better visibility). Blue bounding boxes are instances of interest.

mechanisms. The PAA module includes multi-point-channel pooling, point-wise, channel-wise, and task-aware attention techniques to enhance pillar representations. Multi-point-channel pooling uses various pooling strategies to grasp context information from all points in a pillar. Point- and channel-wise attentions retain proper information during the feature extraction by suppressing redundant information. Task-aware attention switches the channels on or off based on their contributions to downstream tasks by selecting the most representative channels. We will show that PAA module is a key in boosting the performance of pedestrian detection.

Challenge 2: Fewer pillars enclosing pedestrians after quantization as a result of their small occupancy and point sparsity in 3D point cloud. In comparison with vehicles, human are relatively small, and thus occupy fewer pillars after the quantisation step. As demonstrated in Fig. 2 B, distant pedestrians are really small and hard to detect, due to their size and the sparsity of the sensor’s laser beams. This becomes more challenging in crowded and/or cluttered areas, where people tend to stand or move in dense groups (*cf.* Fig. 6). Therefore, detecting pedestrians may require more than just extracting a naive pillar feature, e.g. to enrich feature representations in both low- and high-resolution point clouds.

Thus, we introduce **Mini-BiFPN**, a lightweight feature network inspired by [39] that bidirectionally connects and fuses multi-resolution features. The learnable weights in Mini-BiFPN fusion gates can be adaptively adjusted based on the contribution of features at multiple resolutions. Also, the module is composed of three convolutional blocks with a minimal number of layers, making it lightweight and efficient.

Combining the aforementioned modules, we present **PiFeNet**, a novel end-to-end trainable 3D object detector capable of extracting important features in real-time, which benefits from the reduced inference latency of single-stage methods and the improved feature extraction mechanisms from attention. Our contributions are as follows:

- 1) We introduce a stackable Pillar Aware Attention Module, modelling various attention on pillars for a better feature learning.
- 2) We propose Mini-BiFPN, a lightweight network that efficiently performs weighted multi-scale feature fusion and provides bidirectional information flows.

- 3) We comprehensively evaluate the performance and generalisability of our framework on three large-scale 3D detection benchmarks, including KITTI [11], JRDB [29] and Nuscenes [4]. Compared with the published counterparts, PiFeNet achieves SOTA performance on KITTI (BEV) and JRDB, as well as competitive performance on nuScenes.

II. RELATED WORK

3D object detection from LiDAR point cloud: In general, 3D detectors for LiDAR point cloud can be classified into two types: one-stage and two-stage. Two-stage detectors [6], [24], [33], [34], [45], [51] first generate a series of anchors by the region proposal network (RPN), and then predict proposals based on these anchors. Many two-stage works [6], [33], [34] enhance RPN outputs, while others [24], [45], [51] refine point cloud representations by multi-modal feature fusion from RGB images and 3D point clouds. Recently, Transformer-based two-stage approaches EQ-PVRCNN [47] and CasA [44] leverage the Query-Key mechanism to refine intermediate representations/proposals, serving as inputs to the latter stage.

Although two-stage approaches provide more accurate predictions than one-stage detectors, they are significantly slower than one-stage counterparts due to the high computation overhead. One-stage approaches make a trade off between detection speed and accuracy. The recent progress on one-stage detectors [12], [48], [53] have been shown a very competitive performance compared to the two-stage approaches. They apply fully convolutional network on modified point cloud representations such as pillars [21] and voxels [46], [54], to directly regress proposals. SA-SSD [12] further transforms convolutional feature maps to point-level representations to capture more structure-aware features. 3D-SSD [48] proposes a real-time network by eliminating upsampling layers and the refining stage. CIA-SSD [1] uses one fusion gate to link multi-scale features, while our Mini-BiFPN module presents four fusion gates. Most recently, SE-SSD [53] proposes a shape-aware augmentation method to train a pair of teacher-student SSDs, which is encouraged to infer entire object forms.

These 3D detectors achieve great performance in detecting general object categories. Nevertheless, they do not perform well in detecting pedestrians due to aforementioned challenges. TANet [27] proposes a triple attention module, which can better extract discriminating features for pedestrians by attention mechanisms. However, this technique only considers the maximum values of the pillars’ channels as encoded representations, and these maximum values do not reflect rich representations of all the points inside that pillar, which is essential for precise object detection and localisation. In this letter, we address this weakness by proposing PAA module with multiple attention mechanisms and improved pooling strategies.

Attention mechanisms in object detection: Attention mechanisms have been shown to be advantageous for a variety of computer vision applications, including image classification [2], [12], [14], segmentation [10], and object detection [38]. Particularly in the detection of small objects, the technique is applied to guide the model’s attention to very small details without incurring a considerable computational cost. Following the feature maps generated by convolutional layers, the attention masks are then computed to capture the spatial (Non-local Net [41] and

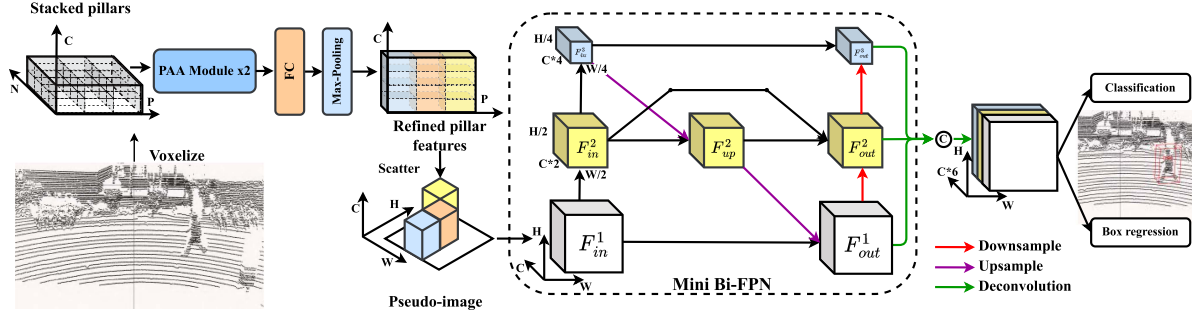


Fig. 3. The PiFeNet's architecture. Input point clouds are first converted to pillars, and then passed through stacked PAA modules for feature extraction. The most expressive features are chosen and converted into a pseudo image, which serves as the input of Mini-BiFPN for generating 3D bounding boxes.

Criss-cross Net [17]) and channel-related information (SE [14] and AiA [8]). Also, some techniques such as CBAM [43] and CA [13] combine the two for improved information fusion. In 3D vision tasks, Point-attention [9] proposes a self-attention structure and skip-connection to capture long-range dependencies of points for point cloud segmentation. Point-Transformer [52] learns the attention weights for the neighbour points, by using local geometric relationships between the center point and its surrounding.

Following the success of pillar-based approaches [21], [27], PiFeNet is a novel end-to-end trainable 3D object detector capable of extracting important features of small objects in real-time, which benefits from the reduced inference latency of single-stage methods and the improved feature extraction mechanisms from attention.

III. OUR APPROACH

This section introduces pillar-based PiFeNet for 3D object detection, followed by the formulas of multi-point-aware pooling, channel-wise, point-wise, task-aware attention mechanisms in the PAA module, and the Mini-BiFPN's structure.

A. Pillar-Feature Network (PiFeNet)

As shown in Fig. 3, PiFeNet firstly takes a point cloud as input and quantises it into pillars by a pillar generator module similar to [21]. Next, we stack our PAA modules to extract pillar features with reduced information loss. Then the attention-weighted pillar features are scattered onto a pseudo-image, which is run through our Mini-BiFPN to fuse features on different resolutions. Finally, we use a simple detection head including classification and box regression branches to produce detection results.

B. 3D Object Detection

3D object detection expects to predict a 3D bounding box $(c_x, c_y, c_z, w, l, h, \theta)$ for each object. c_x, c_y, c_z denote the x, y, and z centers of the box, respectively. w, l, h are the box's dimensions and θ is the heading angle. The input of the model is a set of points (point cloud), defined as $T = \{p_i = [x_i, y_i, z_i, r_i]^T \in \mathbb{R}\}_{i=1,2,3,\dots,M}$. x_i, y_i, z_i are the x, y, and z coordinates of points, respectively. r_i is the points' reflection intensity, and M is the total number of points. Assume the dimension of a point cloud T is defined as L^* for length (x axis), W^* for width (y-axis), and H^* for height (z-axis). T is then equally divided into pillars [21] (z-axis stacked voxels) along the x and y axes. Due to the sparsity and unequal distribution of the point clouds,

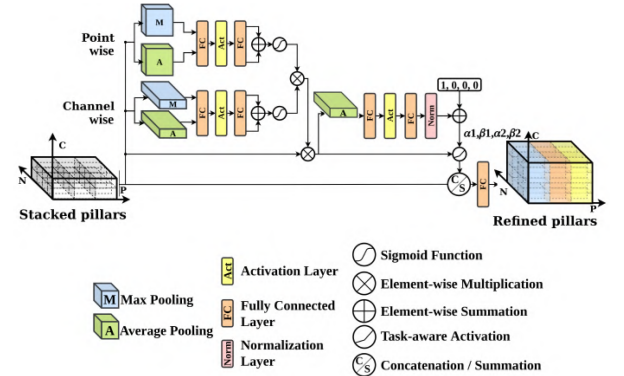


Fig. 4. Detailed architecture of the Pillar Aware Attention module.

each pillar has a varying amount of points. Let N be a pillar's maximum number of points. Let C be the number of channels of each point, and P be the amount of pillars in the pillar grid G , where $G = \{p_1, p_2, \dots, p_j\}_{j \leq P}$. Each pillar's dimensions are $[p_W^*, p_L^*, p_H^*]$, where $p_H^* = H^*$ since we do not divide the point cloud along z-axis. Let the number of output classes be N_c , and N_a is the number of anchors per pillar. Our PiFeNet directly takes a raw point cloud as input and produces $P \times N_a \times N_c$ prediction boxes and their classifications.

C. Pillar Aware Attention (PAA) Module

As shown in Fig. 4, our PAA module begins with multi-point-channel pooling followed by two branches of point- and channel-wise attention. The outputs are then integrated by element-wise multiplication before going through the task-aware attention sub-module.

Multi-point-channel pooling: To capture the context of all points and channels in a pillar, we present a multi-point-channel pooling, which applies both max and average pooling to the point- and channel-wise dimensions in point clouds. Consider a pillar grid $G \in \mathbb{R}^{P \times N \times C}$, where N is the maximum number of points, C is the number of channels in a pillar, and P is the maximum amount of pillars $\{p_1, p_2, \dots, p_j\}_{j \leq P}$ in the grid. We perform channel-wise average and max pooling on the pillar grid G to aggregate channel information. The outputs are two distinct channel context representations F_c^{mean} and F_c^{max} , where $F_c^{max}, F_c^{mean} \in \mathbb{R}^{P \times 1 \times C}$. The same strategies are also applied to the point-wise dimension to aggregate the point information F_p^{mean} and F_p^{max} , where $F_p^{max}, F_p^{mean} \in \mathbb{R}^{P \times N \times 1}$.

Channel- and Point-wise attention: To encapsulate the global information, F_c^{mean} and F_c^{max} are passed through a shared multi-layer perceptron (MLP) with two fully-connected layers, an activation function δ , and a reduction ratio r (the same applied to F_p^{mean} and F_p^{max}). The outputs of the MLP are summed element-wise to generate the final attention score vectors $A_c \in \mathbb{R}^{P \times 1 \times C}$ and $A_p \in \mathbb{R}^{P \times N \times 1}$:

$$A_{c/p} = \sigma \left(w_1 \left(w_0 \delta \left(F_{c/p}^{mean} \right) \right) + w_1 \left(w_0 \delta \left(F_{c/p}^{max} \right) \right) \right) \quad (1)$$

where σ is Sigmoid function. The weights in the two fully-connected layers are $w_0 \in \mathbb{R}^{N/r \times N}$ and $w_1 \in \mathbb{R}^{N \times N/r}$ in case of point-wise attention, or $w_0 \in \mathbb{R}^{C/r \times C}$ and $w_1 \in \mathbb{R}^{C \times C/r}$ for channel-wise attention. The full attention matrix is then achieved by combining channel-wise attention A_c and point-wise attention A_p using element-wise multiplication $M_j = A_p \times A_c$, where $M_j \in \mathbb{R}^{N \times P \times C}$. By multiplying M_j with the original pillar P_j , we can obtain attention-weighted features in both channel-wise and point-wise dimensions.

After the point-wise and channel-wise attention sub-modules, the pillar features become more expressive and sensitive to all points in the point cloud and their channel features (x, y, z locations, centres).

Task-aware attention: The task-aware attention sub-module is chained at the end to dynamically control the activation of each channel. The output pillar features are then reorganised into distinct activations in response to the needs of various downstream tasks (i.e. detection and classification). Similar to [7], given the feature map $F_c \in \mathbb{R}^{1 \times N}$ of the pillar p_j and the channel C_k , the activation is calculated as follows:

$$A_T^k(F_c) = \max \left(\alpha_k^1 \cdot F_c + \beta_k^1, \alpha_k^2 \cdot F_c + \beta_k^2 \right) \quad (2)$$

where $A_T^k(F_c)$ is the task-aware weighted pillar feature map of the channel C_k . $[\alpha_k^1, \beta_k^1, \alpha_k^2, \beta_k^2] = \theta(\cdot)$ is a hyper-function to control channel-wise activation thresholds. It is constructed from one global pooling, two fully-connected layers and a shifted sigmoid layer $f(x) = 2\sigma(x) - 1$ to normalise the output to $[-1, 1]$ before multiplying with the original features. Note that $\theta(\cdot)$ is initialised with $[1, 0, 0, 0]$, indicating that (2) is originally a ReLU function $\max\{F_c, 0\}$. Finally, the task-aware weighted features are concatenated or summed back to the input features.

Stackable PAA module: Our approach stacks two PAA modules to better leverage multi-level feature attention. The first module combines nine task-aware weighted features with the original ones. The resulting 18 channels are then used as input to the second PAA module. The PAA results are element-wise added to the input of a fully-connected layer. The fully-connected layer increases the feature dimension to 64 for greater expressiveness. Eventually, a point-wise max pooling is performed to extract the strongest features, serving as inputs of Mini-BiFPN module, as shown in Fig. 3.

D. Mini-Bifpn

Given the small occupancy of pedestrians in point clouds, the feature network needs more comprehensive connections, so that features from multiple resolutions can complement and compensate each other. Therefore, we propose Mini-BiFPN detection head, a faster variant of BiFPN [39] that significantly improves the performance in 3D object detection with minimal efficiency trade-offs. Mini-BiFPN firstly inputs a pseudo-image

into convolutional blocks B_1, B_2, B_3 to produce a list of features $\bar{F}_{in} = (F_{in}^1, F_{in}^2, F_{in}^3)$, with resolutions $1/2^{i-1}$ of input pseudo-image where $i \in (1, 2, 3)$. Next, the multi-scale features are aggregated by repeatedly applying top-down and bottom-up bidirectional feature fusions, as shown in Fig. 3. Trainable weights are added to adjust the fusion weights accordingly. Our Mini-BiFPN can be formalised as follows:

$$F_{up}^2 = \text{conv} \left(\delta \left(\frac{(w'_1 \cdot F_{in}^2 + w'_2 \cdot \text{upsample}(F_{in}^3))}{w'_1 + w'_2 + \varepsilon} \right) \right) \quad (3)$$

$$F_{out}^2 = \text{conv} \left(\delta \left(\frac{(w''_1 \cdot F_{in}^2 + w''_2 \cdot F_{up}^2 + w''_3 \cdot \text{downsample}(F_{out}^1))}{w''_1 + w''_2 + w''_3 + \varepsilon} \right) \right) \quad (4)$$

where δ is Swish activation function adopted from [39], and F_{up}^2 is the fusion result of F_{in}^2 and F_{in}^3 . Then F_{out}^2 is calculated by F_{up}^2 , F_{out}^1 , and F_{in}^2 . $w'_{1,...,i}$ and $w''_{1,...,i}$ are trainable parameters where i is the nu [32]. F_{out}^1 and F_{out}^3 are computed similarly to F_{out}^2 . The final feature representation is the concatenation of F_{out}^1 , F_{out}^2 , and F_{out}^3 . It is then passed into a simple SSD [26] detection head including classification and box regression branches for final predictions.

Loss function: Similar to SECOND [46], Focal loss [25] and L1 loss are used for bounding box classification and regression, respectively.

IV. EXPERIMENTS

Our proposed PiFeNet is evaluated on KITTI [11], JRDB [29] and Nuscenes [4] benchmarks. We choose CBGS-PP (Pointpillars backbone) as a baseline for Nuscenes and PointPillars for KITTI and JRDB. We tried training PiFeNet on KITTI using pretrained weights on JRDB and observed that the pretraining-finetuning strategy is not necessary considering the domain shift, sensor setup, and LiDAR resolution differences among datasets. Similar behaviours are also seen in [18]. Therefore, all models in this work are trained from scratch (random weight initialisation).

KITTI has 7481 and 7518 samples for training and testing, respectively. The training set is further split into training and validating set with a ratio of 85:15. We conduct experiments on the ‘‘Pedestrian’’ category and use average precision (AP) with an (IoU) threshold of 0.5 as an evaluation metric. The detection range is $[0, 47.36]$, $[-19.84, 19.84]$, $[-2.5, 0.5]$ metres in X and Y and Z coordinates, respectively. Pillar size is $[0.16, 0.16, 4]$ metres in width, length, and height. Also, ground truths are mixed with sampled objects from other point clouds, together with random global rotation, scaling, and translation.

JRDB is a novel human-centric dataset comprised of more than 1.8 million 3D cuboids surrounding all pedestrians in both indoor and outdoor settings. This dataset contains 54 sequences, 27 for training and 27 for testing. During training, we select 5 from 27 sequences for validation, as suggested in [29], and we use AP with an IoU of 0.5 to monitor the training process. The detection range is $[-39.68, 39.68]$ in X and Y, and $[-2, 2]$ in Z LiDAR coordinate. Random pedestrians are cut-and-pasted to the scene ensuring at least 30 people per input point cloud. Random rotation, flipping and scaling are also applied. The top 300 proposals after Non-max suppression (NMS) with IoU of 0.5 are kept as final predictions.

Nuscene: We adopt CBGS [55] augmentations strategies (global translation, scaling, and rotation) and multi-task heads. The detection range is $[-51.2, 51.2]$ metres for X and Y, and

TABLE I

COMPARISON WITH PREVIOUS APPROACHES FOR 3D OBJECT DETECTION TASK ON THE KITTI OFFICIAL TEST SPLIT FOR PEDESTRIAN ON BIRDS-EYE-VIEW (BEV) AND 3D DETECTION TASKS. THE AP IS CALCULATED USING 40 RECALL POSITIONS, AND 3D MAP REPRESENTS THE MEAN AVERAGE PRECISION OF THE THREE DIFFICULTIES. “*” DENOTES RUN-TIME IS CITED FROM KITTI OFFICIAL WEBSITE

Stage	Method	Venue	Performance (mAP)								Speed & Computation	
			Birds-eye-view				3D Detection				fps	GPU
			Easy	Mod.	Hard	Overall	Easy	Mod.	Hard	Overall		
Two	PointPainting [40]	CVPR’20	58.7	49.93	46.29	51.64	50.32	40.97	37.87	43.05	3*	Unknown
	AVOD-FPN [19]	IROS’18	58.49	50.32	46.98	51.93	50.46	42.27	39.04	43.92	10*	Titan X
	STD [49]	ICCV’19	60.02	48.72	44.55	51.10	53.29	42.47	38.35	44.70	13*	Titan V
	F-PointPillars [30]	ICCV’21	60.98	52.23	48.3	53.84	51.22	42.89	39.28	44.46	17*	1080
	VMVS [20]	IROS’19	60.34	50.34	46.45	52.38	53.44	43.27	39.51	45.41	4*	Titan X
	VoxelToPoint [23]	ACMMM’21	56.54	48.15	45.63	50.11	51.8	43.28	40.71	45.26	10	V100
	PV-RCNN [33]	CVPR’20	59.86	50.57	46.74	52.39	52.17	43.29	40.29	45.25	13*	1080
	PartA2 [35]	TPAMI’20	59.04	49.81	45.92	51.59	53.1	43.35	40.06	45.50	13	V100
	F-ConvNet [42]	IROS’19	57.04	48.96	44.33	50.11	52.16	43.38	38.8	44.78	2*	Unknown
One	EQ-PVRCNN [47]	CVPR’22	61.73	52.81	49.87	54.80	55.84	47.02	42.94	48.60	5	3090
	VoxelNet [54]	CVPR’18	57.73	39.48	33.69	43.63	39.48	33.69	31.51	34.89	4*	Unknown
	SECOND [46]	Sensors’18	55.1	46.27	44.76	48.71	51.07	42.56	37.29	43.64	20*	1080
	PointPillars [21]	CVPR’19	57.6	48.64	45.78	50.67	51.45	41.92	38.89	44.09	63	V100
	MGAFFSSD [22]	ACMMM’21	56.09	48.46	44.9	49.82	50.65	43.09	39.65	44.46	10	V100
	Point-GNN [36]	CVPR’20	55.36	47.07	44.61	49.01	51.92	43.77	40.14	45.28	2	V100
	3DSSD [48]	CVPR’20	60.54	49.94	45.73	52.07	54.64	44.27	40.23	46.38	25*	Titan V
	TANet [27]	AAAI’20	60.85	51.38	47.54	53.26	53.72	44.34	40.49	46.18	29*	Titan V
	HotSpotNet [5]	ECCV’20	57.39	50.53	46.65	51.52	53.1	45.37	41.47	46.65	25	V100
	PiFeNet (ours)		63.25	53.92	50.53	55.90	56.39	46.71	42.71	48.60	26	V100

$[-5, 3]$ for Z LiDAR coordinates, pillar size is set at $[0.2, 0.2, 8]$ in width, length, and height respectively. We train the model for 20 epochs with a batch size of 2. During testing, top 83 predictions with confidence scores greater than 0.1 are kept, and rotated NMS with IOU threshold of 0.02 is applied.

AdamW [28] optimiser and One Cycle policy [37] with max LR at 0.003 are used in all experiments.

A. Comparison With State-of-The-Arts

KITTI: As shown in Table I, PiFeNet outperforms all the existing state-of-the-art approaches in BEV detection. Our model outperforms TANet [27] by 2.64 mAP in BEV and 2.22 mAP in 3D detection. It also surpasses the current strongest BEV pillar-based detector Frustum-PointPillars [30] by 2.06 mAP in BEV pedestrian detection.¹ These significant improvements are caused by the enhanced expressiveness of the pillars, together with better communication between pillars at multiple scales. While achieving SOTA performance, PiFeNet’s inference speed is 26 FPS (*cf.* Table VI), making it one of the most efficient detectors on KITTI.

JRDB and nuScenes: We further evaluate our model on JRDB-3D detection benchmark. Compared to other published methods, our model is ranked first on JRDB 2019² in multiple AP thresholds (i.e. 0.3, 0.5, and 0.7). Our model also achieves the best performance on the recently launched JRDB 2022³ 3D detection leaderboard. We also present in Table II the performance of our approach on nuScenes. Compared with our

¹[Online]. Available: cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=bev

²[Online]. Available: <https://jrdb.erc.monash.edu/leaderboards/detection>

³[Online]. Available: <https://jrdb.erc.monash.edu/leaderboards/detection22>

TABLE II

PIFENET PERFORMANCE ON OFFICIAL NUSCENES TEST SET COMPARED TO OTHER APPROACHES. BOLD VALUES ARE THE BEST PERFORMANCE

Method	AP (pedestrian)	mAP (all classes)	NDS
PointPillars+ [21]	0.640	0.401	0.550
WYSIWYG [15]	0.650	0.350	0.419
SARPNET [50]	0.694	0.324	0.484
SA-Det3D [3]	0.733	0.470	0.592
PiFeNet (ours)	0.745	0.478	0.606

TABLE III

PIFENET PERFORMANCE ON OFFICIAL THE JRDB 2019 AND NEWLY LAUNCHED JRDB 2022 TEST SETS COMPARED TO OTHER APPROACHES.

*REPRODUCED RESULT

	Method	AP@0.3	AP@0.5	AP@0.7
JRDB 2019	F-Pointnet [31]	38.205	6.378	0.081
	EPNet [16]	59.252	16.845	0.418
	TANet++ [27]	63.922	27.991	1.842
	PiFeNet (ours)	74.284	42.617	4.886
JRDB 2022	PointPillars* [21]	69.209	33.677	2.209
	PiFeNet (ours)	70.724	39.838	4.59

baseline Pointpillars++[21], there is a significant improvement of 0.11 AP in pedestrian detection.

In Fig. 5, we present our qualitative findings which include both detection results (Fig. 5 A and 5 C) as well as some typical failure cases (Fig. 5 B and 5 D). Fig. 5 C shows the

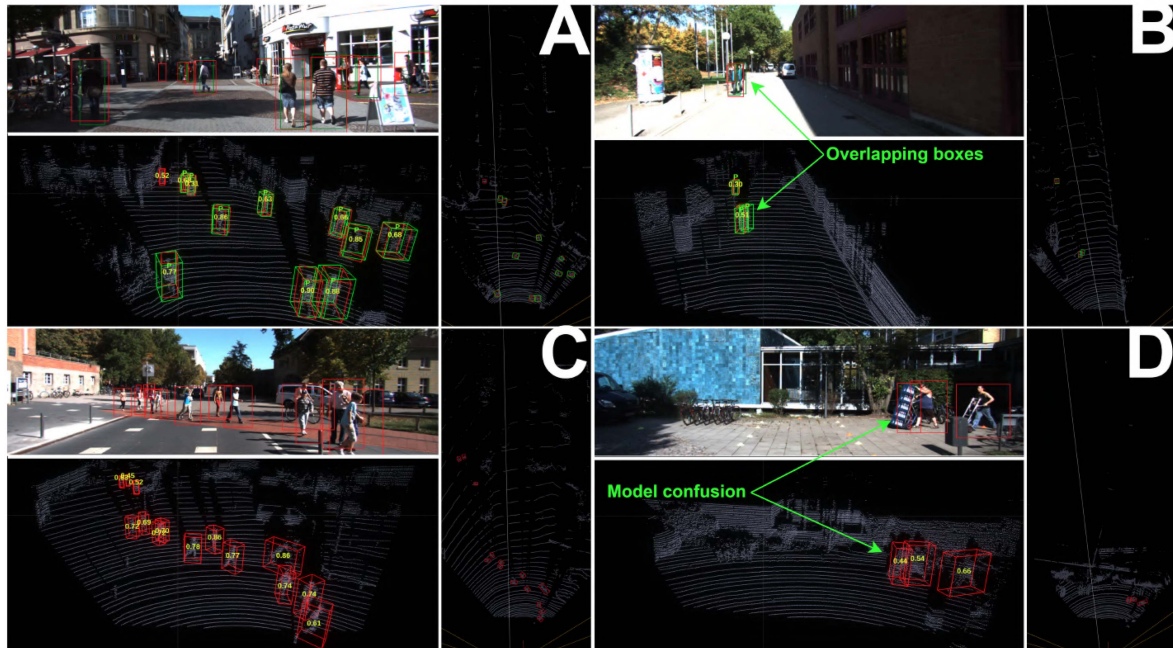


Fig. 5. Visualization on KITTI val set (A, B) and test set (C, D). Each visualization includes BEV (right), 3D point clouds (top), and 2D image (bottom). The ground truth boxes are green, the predicted boxes are red with yellow confident scores. B and D are typical val and test set failures.

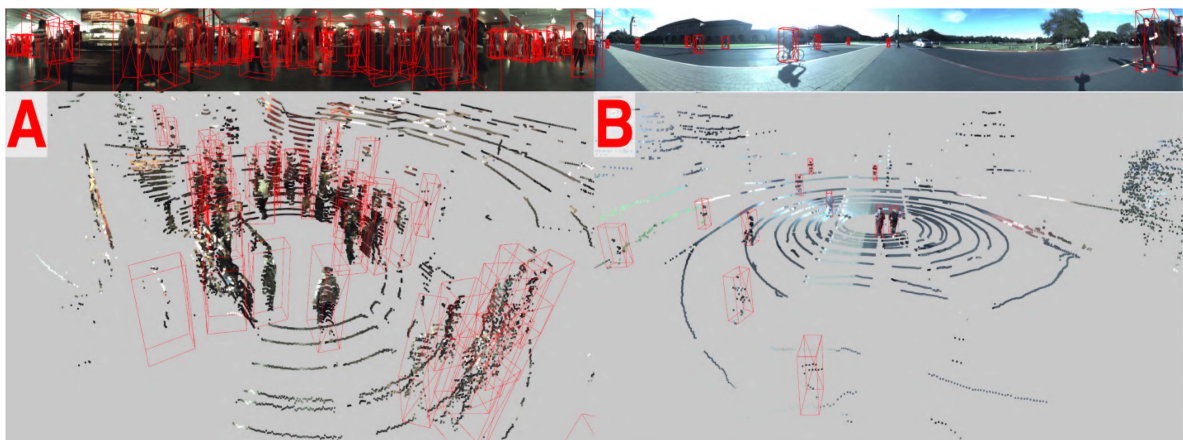


Fig. 6. Qualitative results on JRDB 2022 official test set. Very crowded indoor (A) and outdoor (B) settings are visualised (2D panorama image at the top, 3D point cloud at the bottom). Prediction bounding boxes (with red cross showing heading direction) are shown in red. Best viewed in color.

robustness of our model when it can accurately detect all pedestrians in the picture, even at very long distances. Compared to Pointpillars [21], we successfully overcome the lack of expressiveness in the extracted pillar features, which usually leads to confusion between narrow vertical features of trees/poles and pedestrians. Given the fact that we rarely see the model misclassify a tree/pole as a pedestrian (see Fig. 7), we did find an instance where the model was confused when encountering a stack of soda crates as in Fig. 5 D. The second difficulty arises when pedestrians are too close together (Fig. 5 B), preventing the model from accurately locating both items. However, as shown in Fig. 6 A, PiFeNet can elegantly distinguish people from background objects in a very crowded indoor scene, thanks to the improved representation and multi-scale communication between pillars.

V. ABLATION STUDIES

We provide extensive ablation experiments to evaluate the efficacy and contribution of each component in PiFeNet. These experiments are conducted on the official KITTI *val* split. We use Pointpillars [21] as our baseline. Our baseline without any attention mechanisms reaches 61.43 overall mAP, and achieves 66.73, 61.06, and 56.5 AP at three difficulty levels (easy, moderate, and hard).

Attention module analysis: We use both average and max pooling as a default setting, then we sequentially test each of the PAA module's components and their combinations. According to Table IV, channel-wise and task-aware attention alone does not significantly improve the performance. However, combining them results in a significant 2.2% mAP increase.

TABLE IV
ABLATION STUDY ON PAA MODULE'S COMPONENTS

Pooling		Modules			Pedestrian				
Avg	Max	Point-	Channel-	Task-	Easy	Moderate	Hard	mAP	FPS
✓	✓	✓			66.7	61.1	56.5	61.4	41.5
✓	✓		✓		69.6	63.4	58.1	63.7	36.3
✓	✓			✓	67.3	62.1	56.4	61.9	35.0
✓	✓	✓			68.2	61.4	55.9	61.8	31.2
✓	✓	✓	✓		67.3	61.4	56.5	61.8	32.3
✓	✓	✓		✓	69.3	63.3	58.1	63.6	28.1
✓	✓	✓	✓	✓	68.3	62.4	56.6	62.4	28.8
✓	✓	✓	✓	✓	68.2	62.2	57.3	62.6	28.8
✓	✓	✓	✓	✓	69.6	61.9	55.4	62.3	27.9
✓	✓	✓	✓	✓	69.7	64.0	58.7	64.2	27.4

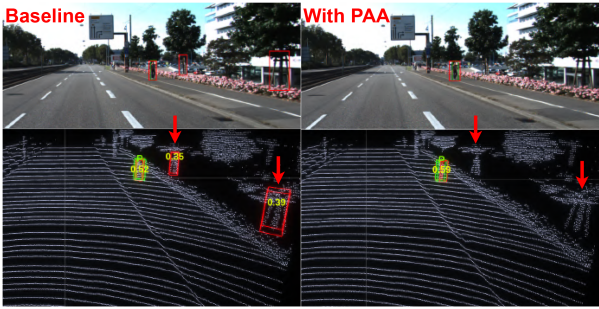


Fig. 7. Qualitative improvement of the model with PAA modules (without Mini-BiFPN) over the baseline PointPillars. The figure shows that, without PAA modules, the trees on the right are wrongly classified as pedestrians, while the model with the PAA modules correctly classifies them as background. Best viewed in color.

TABLE V
ABLATION STUDY ON THE NUMBER OF STACKED PAA MODULES

Modules	Pedestrian				
	Easy	Moderate	Hard	mAP	FPS
Baseline	66.7	61.1	56.5	61.4	41.5
+ 1 PAA	68.29	62.36	56.64	62.43	34.2
+ 2 PAA	69.70	64.02	58.74	64.2	27.4
+ 3 PAA	70.37	64.35	58.92	64.55	19.3

Moreover, integrating point-wise attention with either channel-wise or task-aware slightly improves the performance. Therefore, while standalone point-wise attention (2.3% mAP increase) can significantly increase model performance, channel-wise and task-aware sub-modules must be employed together for better accuracy. Additionally, Fig. 7 shows the qualitative result of models with and without PAA modules.

Number of stacked PAA modules: Table V shows how the number of PAA modules affects the performance. Firstly, more PAA modules improve the accuracy, which demonstrates the effectiveness of our PAA module. Nevertheless, when we stack three PAA modules or more, the performance improvement rate slightly decelerates and the model requires more time for running. To balance the accuracy and speed, we use two PAA modules in other experiments.

Pooling mechanisms analysis: In the last 3 rows of Table IV, we show that each pooling method improves the performance (point-, channel-, and task-aware sub-modules are included as default). The z-axis related channels may be used to extract the

TABLE VI
CONTRIBUTION OF MODULES TO PiFeNet

	Pedestrian				FPS
	Easy	Moderate	Hard	mAP	
Baseline	66.73	61.06	56.5	61.43	41.5
+ PAA	69.70	64.02	58.74	64.2	27.4
+ Mini-BiFPN	68.52	62.30	57.19	62.67	36.2
+ PAA + Mini-BiFPN	73.78	67.69	61.51	67.66	25.7

maximum height of points inside a pillar, while the x and y-axis related channels can be used to extract the mean position of points inside a pillar. While average pooling outperforms max pooling in detecting moderate and hard objects, max pooling is better in detecting easy ones. Concurrent usage of both pooling methods increases mAP to 64.2, showing that both pooling mechanisms are essential.

Mini-BiFPN modules analysis: Table VI also presents the outcome of the Mini-BiFPN module. By gathering more information from other stages of the feature network, our Mini-BiFPN's gate-controlled cross-scale connections and bidirectional information flow provide more accurate predictions. As in Table VI, after switching from the baseline feature network to the Mini-BiFPN module, we achieve a significant improvement in all APs (7.05, 6.63, and 5.01 increase in easy, moderate, and hard APs) as well as 6.23 increase in mAP compared to the baseline. Thus, both PAA and Mini-BiFPN modules are important components of our model..

Run-time Analysis: At inference time, PiFeNet can reach an average speed of 26 FPS (39.16 ms latency), which breaks down into 5.84 ms for data pre-processing, 18.12 ms for PAA module, 0.60 ms for scattering the features to a pseudo-image, 13.26 ms for Mini-BiFPN, and 1.34 ms for post-processing the predictions. The experiment is conducted on the KITTI dataset and a single NVIDIA V100 GPU. Following previous works, we also compare the inference speed (in FPS) between PiFeNet and other published approaches. While this comparison cannot be easily benchmarked due to hardware/computation variations, it still demonstrates the efficiency of our detector for running in real-time.

VI. CONCLUSION

Our PiFeNet has been presented, addressing two main issues in 3D pedestrian detection. First, we introduced the Pillar Aware Attention Module, which combines multi-point-channel-pooling, point-wise, channel-wise, and task-aware attention to better extract pillar features. Next is the Mini-BiFPN module, a lightweight feature network that leverages cross-scale feature fusion and bidirectional connections, enriching information flow in the feature network. Our method achieves SOTA performance on both large-scale benchmarks KITTI [11] and JRDB [29], and competitive results on Nuscen [4].

REFERENCES

- [1] W. Zheng et al., "CIA-SSD: Confident iou-aware single-stage object detector from point cloud," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 3555–3562.
- [2] I. Bello, B. Zoph, A. Vaswani, and J. Shlens, "Attention augmented convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3286–3295.

- [3] P. Bhattacharyya, C. Huang, and K. Czarnecki, "SA-Det3D: Self-attention based context-aware 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 3022–3031.
- [4] H. Caesar et al., "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11621–11631.
- [5] Q. Chen, L. Sun, Z. Wang, K. Jia, and A. Yuille, "Object as hotspots: An anchor-free 3D object detection approach via firing of hotspots," in *Proc. IEEE Eur. Conf. Comput. Vis.*, 2020, pp. 68–84.
- [6] Y. Chen, S. Liu, X. Shen, and J. Jia, "Fast point R-CNN," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9775–9784.
- [7] Y. Chen, X. Dai, M. Liu, D. Chen, L. Yuan, and Z. Liu, "Dynamic relu," in *Proc. IEEE Eur. Conf. Comput. Vis.*, 2020, pp. 351–367.
- [8] P. Fang, J. Zhou, S. K. Roy, P. Ji, L. Petersson, and M. Harandi, "Attention in attention networks for person retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 4626–4641, Sep. 2022.
- [9] M. Feng, L. Zhang, X. Lin, S.Z. Gilani, and A. Mian, "Point attention network for semantic segmentation of 3D point clouds," *Pattern Recognit.*, vol. 107, 2020, Art. no. 107446.
- [10] J. Fu et al., "Dual attention network for scene segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3146–3154.
- [11] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.
- [12] C. He, H. Zeng, J. Huang, X.S. Hua, and L. Zhang, "Structure aware single-stage 3d object detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11873–11882.
- [13] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 13713–13722.
- [14] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.
- [15] P. Hu, J. Ziglar, D. Held, and D. Ramanan, "What you see is what you get: Exploiting visibility for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11001–11009.
- [16] T. Huang, Z. Liu, X. Chen, and X. Bai, "EPNet: Enhancing point features with image semantics for 3D object detection," in *Proc. IEEE Eur. Conf. Comput. Vis.*, 2020, pp. 35–52.
- [17] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "CCNet: Criss-cross attention for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 603–612.
- [18] D. Jia, A. Hermans, and B. Leibe, "2D vs. 3D LiDAR-based person detection on mobile robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 3604–3611.
- [19] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–8.
- [20] J. Ku, A. D. Pon, S. Walsh, and S. L. Waslander, "Improving 3D object detection for pedestrians with virtual multi-view synthesis orientation estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 3459–3466.
- [21] A.H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12697–12705.
- [22] J. Li, H. Dai, L. Shao, and Y. Ding, "Anchor-free 3D single stage detector with mask-guided attention for point cloud," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 553–562.
- [23] J. Li, H. Dai, L. Shao, and Y. Ding, "From voxel to point: IoU-guided 3D object detection for point cloud with voxel-to-point decoder," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 4622–4631.
- [24] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun, "Multi-task multi-sensor fusion for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7345–7353.
- [25] T.Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.
- [26] W. Liu et al., "SSD: Single shot multibox detector," in *Proc. IEEE Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
- [27] Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, and X. Bai, "TANet: Robust 3D object detection from point clouds with triple attention," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11677–11684.
- [28] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, *arXiv:1711.05101*.
- [29] R. Martin-Martin et al., "JRDB: A dataset and benchmark of ego-centric robot visual perception of humans in built environments," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Apr. 2, 2021, doi: 10.1109/TPAMI.2021.3070543.
- [30] A. Paigwar, D. Sierra-Gonzalez, O. Erkent, and C. Laugier, "Frustum-PointPillars: A multi-stage approach for 3D object detection using rgb camera and lidar," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2926–2933.
- [31] C.R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointNets for 3D object detection from RGB-D data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 918–927.
- [32] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," 2017, *arXiv:1710.05941*.
- [33] S. Shi et al., "PV-RCNN: Point-voxel feature set abstraction for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10529–10538.
- [34] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 770–779.
- [35] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 8, pp. 2647–2664, Aug. 2021.
- [36] W. Shi and R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1711–1719.
- [37] L.N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," *Proc. Artif. Intell. Mach. Learn. Multi-Domain Oper. Appl.*, 2019, vol. 11006, Art. no. 1100612.
- [38] A. Srinivas, T. Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, "Bottleneck transformers for visual recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 16519–16529.
- [39] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10781–10790.
- [40] S. Vora, A.H. Lang, B. Helou, and O. Beijbom, "PointPainting: Sequential fusion for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4604–4612.
- [41] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7794–7803.
- [42] Z. Wang and K. Jia, "Frustum convNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 1742–1749.
- [43] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. IEEE Eur. Conf. Comput. Vis.*, 2018, pp. 3–19.
- [44] H. Wu, J. Deng, C. Wen, X. Li, C. Wang, and J. Li, "CasA: A cascade attention network for 3-D object detection from LiDAR point clouds," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–11, 2022.
- [45] L. Xie et al., "PI-RCNN: An efficient multi-sensor 3D object detector with point-based attentive cont-conv fusion module," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 12460–12467.
- [46] Y. Yan, Y. Mao, and B. Li, "SECOND: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, 2018, Art. no. 3347.
- [47] Z. Yang, L. Jiang, Y. Sun, B. Schiele, and J. Jia, "A unified query-based paradigm for point cloud understanding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8541–8551.
- [48] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3DSSD: Point-based 3D single stage object detector," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11040–11048.
- [49] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "STD: Sparse-to-dense 3D object detector for point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1951–1960.
- [50] Y. Ye, H. Chen, C. Zhang, X. Hao, and Z. Zhang, "SARPNET: Shape attention regional proposal network for LiDAR-based 3D object detection," *Neurocomputing*, vol. 379, pp. 53–63, 2020.
- [51] J.H. Yoo, Y. Kim, J. Kim, and J. W. Choi, "3D-CVF: Generating joint camera and LiDAR features using cross-view spatial feature fusion for 3D object detection," in *Proc. IEEE Eur. Conf. Comput. Vis.*, 2020, pp. 720–736.
- [52] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16259–16268.
- [53] W. Zheng, W. Tang, L. Jiang, and C. W. Fu, "SE-SSD: Self-ensembling single-stage object detector from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 14494–14503.
- [54] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3d object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4490–4499.
- [55] B. Zhu, Z. Jiang, X. Zhou, Z. Li, and G. Yu, "Class-balanced grouping and sampling for point cloud 3D object detection," 2019, *arXiv:1908.09492*.