PointBi-FPN: An Extention to Pointpillars for LiDAR 3D Object Detection in Autonomous Vehicles using Bi-directional Feature Pyramid Network

SOUMYA A[1], Krishna Mohan C[2], Linga Reddy Cenkeramaddi[3], Sobhan Babu[4]
[1,2,4]*Department of Computer Science and Engineering, Indian Institute of Technology, Hyderabad, India*
[3]*Department of Information and Communication Technology, University of Agder, Grimstad, Norway*

TENCON 2024 - 2024 IEEE Region 10 Conference (TENCON) | 979-8-3503-5082-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/TENCON61640.2024.10902951

***Abstract*—Autonomous vehicles increasingly rely on accurate three-dimensional (3D) object detection for safe navigation. While two-dimensional (2D) methods offer computational efficiency, the shift to 3D detection enhances precision in understanding environments. Point-based and voxel-based approaches are accurate but computationally intensive for onboard deployment. Pillar-based methods like PointPillars provide efficiency but may lack detection accuracy compared to voxel-based approaches. In this paper, we focus on enhancing the performance of PointPillars, a popular pillar-based detector, using LiDAR data. The proposed approach (PointBi-FPN) employs a bi-directional feature pyramid network (Bi-FPN) as a backbone, that aggregates multiscale features in the input data, providing a holistic view of the environment, which is crucial for detecting objects of varying sizes and distances accurately. Bi-FPN improves the model's understanding of complex scenes, making it more robust to occlusions. Through extensive experimentation on the KITTI dataset, the PointBi-FPN approach demonstrates better performance across all three detection benchmarks: BEV (Bird's Eye View), 3D (Three-Dimensional), and AOS (Average Orientation Similarity). Notably, the proposed approach exhibits significant improvements, particularly in accurately detecting objects labeled as "hard" difficulty.**

***Index Terms*—Deep Learning, Object detection, Bi-directional feature pyramid network, and Computer Vision**

I. Introduction

In three-dimensional (3D) object detection, LiDAR emerges as a predominant sensor choice, as it provides accurate depth information, allowing autonomous vehicles to precisely perceive the distance between objects and their position, which is essential for safe navigation. A standard LiDAR sensor emits laser pulses that bounce off objects and return to the sensor. It calculates the distance based on the return time, creating a three-dimensional (3D) map of space. This 3D map of space is referred to as point cloud or LiDAR data. Since point clouds consist of an unordered collection of 3D points, they are invariant to the order of their points. This poses challenges for using convolutional neural networks (CNNs), as traditional convolution operations rely on fixed grid structures. Utilizing CNNs for point clouds faces two main challenges: sensitivity to point ordering and loss of shape information. To tackle these challenges, numerous strategies have been devised to generate point cloud feature representations, specifically tailored for 3D object detection. Some methodologies involve projecting point clouds into a perspective view and then employing image-based feature extraction techniques [9], [12]. On the other hand, alternative approaches focus on converting point clouds into a 3D voxel grid and encoding each voxel with handcrafted features [4], [5].

Building upon these challenges in existing methodologies, we propose PointBi-FPN, which integrates a Bidirectional Feature Pyramid Network (Bi-FPN) into the PointPillars framework for enhanced 3D object detection. Our approach converts point clouds into sparse pseudo-images, which are processed by the Bi-FPN backbone to extract high-level representations. A detection head then constructs 3D bounding boxes to identify and locate objects. The central idea of this paper is to integrate Bi-FPN as the backbone network. Bi-FPN enhances feature fusion across multiple scales through bi-directional pathways, improving feature representation and detection accuracy.

The contributions of our paper include the integration of Bi-FPN as the backbone network. Bi-FPN effectively captures multiscale features through both top-down and bottom-up pathways, integrating contextual information from different levels of abstraction. These enriched feature representations enable accurate detection of objects of varying sizes, shapes, and orientations in complex driving environments. Our method uses weighted feature fusion to combine features from different resolutions of LiDAR point cloud data, enhancing robustness against occlusions, clutter, and variations in object appearance. This ensures accurate and robust object detection in autonomous driving scenarios. Our PointBi-FPN framework demonstrated significant performance enhancements for the car category, with increases of +5.28%, +5.36%, and +0.8% in average precision across the bird's-eye-view (BEV), 3D, and average orientation similarity (AOS) benchmarks at the "hard" difficulty level. Our model reliably predicts 3D bounding boxes for cars, pedestrians, and cyclists under varying lighting conditions.

II. Related Work

Several methods have been suggested to take advantage of the complementary data from different sensors, including camera images and LiDAR, to enhance object detection in three-dimensional space. One notable method, MV3D [4] introduced 3D object anchors refined with features from both LiDAR and camera images in bird's-eye-view (BEV). AVOD [8] fused these modalities to improve proposal generation. PIXOR [18] analyzed sparse BEV point clouds efficiently. PointPillars [1] utilized a pillar feature encoder for 2D convolution on pseudo-images. VoxelNet [19] partitioned point clouds into 3D voxels, while SECOND [17] enhanced VoxelNet with sparse convolutions. PointNet [13] and PointRCNN [14] revolutionized by directly processing raw point clouds for detection. 3D-SSD [11] introduced

furthest-first-point sampling (F-FPS) but faces computational challenges.

Existing approaches for 3D object detection in autonomous driving face various limitations. Pseudo-lidar generated from RGB images lacks depth information, which impacts performance. Methods that convert point clouds into 2D image-like representations can lose 3D geometric details. On the other hand, approaches based on handcrafted features may be laborious and might not fully capture the complexity of real-world objects. These limitations motivate the exploration of our approach, PointBi-FPN, which employs BiFPN to enhance object detection accuracy by improving the model's ability to understand complex scenes and making it more robust to occlusions.

III. Proposed Approach

This study proposes an enhanced 3D object detection framework named PointBi-FPN, which integrates the Bidirectional feature pyramid network (Bi-FPN) into the PointPillars [1] architecture. This novel approach aims to improve detection performance by leveraging advanced multi-scale feature fusion techniques. The PointBi-FPN architecture in Fig. 1 consists of three main phases. First, a feature encoder network transforms the point cloud into a sparse pseudo-image. Then, the bidirectional feature pyramid network serves as the backbone for processing the pseudo-image into a higher-level representation. Third, the detection head network is designed for identifying and locating the objects within the point clouds.

A. Architecture

Our architecture processes point clouds as input and predicts oriented 3D bounding boxes for the objects. It comprises three primary stages, as depicted in Figure 1: The first stage involves a pillar feature network (PFN) that takes in point clouds and converts them into sparse pseudo-images. This encoding process enables the representation of the point clouds in a structured format resembling images. In the second stage, a backbone network that operates on two-dimensional convolution is applied to the pseudo-images generated in the previous step. Bi-FPN as a backbone enhances feature fusion by incorporating both top-down and bottom-up pathways. The top-down pathway down-samples higher-resolution features and combines them with lower-resolution features, enhancing spatial awareness. Conversely, the bottom-up pathway extracts features at different resolutions, capturing fine details of nearby objects and broader context for distant ones, which helps in capturing multi-scale features more effectively. Finally, in the third stage, a detection head is employed to detect and construct three-dimensional bounding boxes based on the high-level representations obtained from the backbone network.

The pillar feature network processes the sparse pseudo-images generated from the point clouds, extracting features from individual points or pillars. Let (x, y, z) represent the coordinates and r denote the reflection intensity of each LiDAR data point l_0. Initially, the scene is partitioned in the x and y directions with a particular step size of [0.16, 0.16]. Note that the scene is not partitioned along the z direction. This process generates a set of pillars, denoted as A. Each point within a pillar is encoded as a nine-dimensional vector E, parameterized by $(x, y, z, r, x_0, y_0, z_0, x_d, y_d)$. Here, (x_0, y_0, z_0) represents the geometric center of all points within the pillar, while (x_d, y_d) indicates the distance of each point from this geometric center. Points are randomly sampled when they exceed the limit to maintain a fixed number of points P = 32 points in each pillar. This results in a tensor of shape (E, A, P), fed to the pillar feature encoder to extract the features. The E is expanded to F = 64 during feature extraction. Consequently, a feature tensor of shape (F, A, P) is obtained.

The output from the PFN is fed into the Bi-FPN backbone. Here, Bi-FPN operates on the extracted features, enhancing them with multi-scale context information through its top-down and bottom-up pathways. The bottom-up pathway takes the input feature map F_{in} and generates a set of feature maps at different resolutions. The top-down pathway takes the feature maps generated by the bottom-up pathway and propagates them upward through a series of fusion and refinement steps. This step allows for integrating information from different resolutions, enabling better feature representation across various scales.

The feature maps obtained from the Bi-FPN are then passed on to the detection head, which performs further processing to generate object detection predictions. These output feature maps are typically fed into convolutional layers to produce final detection outputs, such as bounding box coordinates and object class probabilities. The detection head follows the single shot detection method, it incorporates two anchors placed at the center of each pillar with angles of 0 and 90 degrees. For evaluating the overlap between predicted and ground truth bounding boxes, we use Intersection over Union (IoU) with a rotated IoU, as it has been found to offer the highest accuracy.

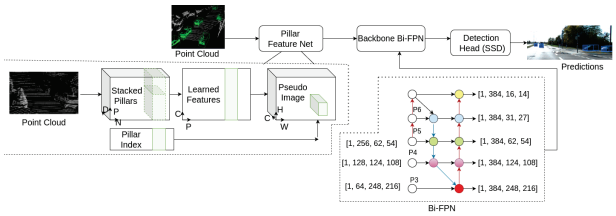

Fig. 1. Proposed PointPillar network with Bi-FPN as backbone

IV. Implementation Details

A. Dataset

The KITTI [6] 3D object detection benchmark dataset is used for the experiments conducted. This dataset comprises 7,467 training data points and 7,506 test data points. The KITTI benchmark focuses on detecting objects in specific categories, including cars, pedestrians, and cyclists. Among the training data points, 3706 are used for actual training and 3761 for the validation of models. The evaluation of test results in the KITTI dataset utilizes official evaluation metrics, including bird's eye view (BEV), 3D, and average orientation similarity. The KITTI dataset provides data in three difficulty levels: easy, moderate, and hard.

B. Evaluation Metrics

The average orientation similarity (AOS) metric [6] evaluates the accuracy of orientation estimation in 3D object detection algorithms, crucial for self-driving scenarios using datasets like KITTI. It measures how well the predicted orientation aligns with ground truth (GT) labels by computing the average cosine similarity between their unit vectors. The cosine similarity ranges from -1 to 1, where 1 indicates flawless alignment, 0 indicates no alignment, and -1 indicates perfect misalignment. To compute the AOS, the predicted orientation is compared to

the ground truth orientation for each object, and the highest cosine similarity across all possible orientations is selected. This maximum similarity is averaged over all objects to derive the final AOS score.

$$AOS = \frac{1}{11} \sum_{i \in 0,0.1,\ldots 1} \max_{\tilde{i}:\tilde{i} \geq i} cs(\tilde{i})$$

Here, i represents the recall as the ratio of True Positives (TP) to the sum of True Positives and False Negatives (FN), where the detected bounding boxes in two dimensions are considered to be accurate if there is a minimum of 50% overlap with the ground truth bounding box.

The AOS metric is commonly used in conjunction with other evaluation metrics, such as average precision (AP) and average recall (AR), to comprehensively evaluate object detection performance, particularly in accurately localizing and orienting objects in 3D space.

C. Loss function

For our implementation, the angle localization loss function in [17] is used in extension with focal loss in [10]. Ground truth boxes and anchors are defined by $(x, y, z, w, p, q, \theta)$. The total localization loss is

$$\mathrm{L}_{loc} = \sum_{r \in x,y,z,w,p,q,\theta} SmoothL1(\Delta r)$$

To differentiate flipped boxes, we use a softmax classification loss on discretized directions (L_{dir}). For object classification, we apply focal loss:

$$\mathrm{L}_{cls} = -0.25(1 - cp^i)^{0.2} \log(cp^i)$$

where cp^i is the class probability of an anchor. $A = 0.25$ and $\zeta = 0.2$ are the parameters for focal loss. Therefore, the total loss is formulated as

$$\mathrm{L} = \frac{1}{N_{pos}}(2L_{loc} + L_{cls} + 0.2L_{dir})$$

here N_{pos} is the count of positive anchors. $\beta_{loc} = 2$, $\beta_{cls} = 1$, and $\beta_{dir} = 0.2$ are considered as constant coefficients in our model loss calculation. To optimize the loss function, we employ the Adam optimizer with an initial learning rate set to 0.001.

D. Settings

MMDetection3D is a widely used open-source framework for three-dimensional object detection in computer vision applications. We adopted a xy resolution of 0.16 m, allowing for precise localization. Our system can handle up to 12,000 pillars, each representing a distinct region of interest. Additionally, each pillar has a maximum limit of 100 points. To ensure consistency, our anchor matching criteria are as follows: Each anchor is characterized by its width, length, height, and z-center. We evaluate it in two orientations: 0 and 90 degrees. The matching process involves comparing anchors to the GT using two-dimensional IoU. An anchor is considered an ideal match if the threshold is reached or the IoU with the GT is at its maximum. During inference, the non-maximum suppression (NMS) overlap threshold is set to 0.5 of intersection over union. Sample settings for the class car are given below.

Car: The car objects in our system are confined within a 3D space defined by the x, y, and z ranges of [(0, 70.39), (-40.1, 40.1), and (-3.01, 1.1)] meters, respectively. To capture the car characteristics, we employ a car anchor with specific dimensions: a width of 1.61 meters, a length of 3.92 meters, and a height of 1.53 meters. The z-center of this anchor is positioned at -1 meter. For matching the anchors to the ground truth, we utilize a positive threshold of 0.6 and a negative threshold of 0.45.

V. Results

We showcase the outcomes achieved through our enhanced PointPillars approach and comprehensively compare them with the existing literature. The evaluation of all detection results follows the standard KITTI evaluation metrics, encompassing bird's eye view (BEV), three-dimension (3D), two-dimension, and average orientation similarity (AOS). Our PointBi-FPN approach, with Bi-FPN as the backbone, achieved better AP in all three benchmarks. Our network achieved a BEV AP of (89.41, 87.19, and 84.93) shown in Table I, higher than all the literature mentioned across all three difficulty categories. Similarly, in the 3D detection benchmark, our network attained an AP of 86.28, 76.74, and 73.86. The easy and hard categories in 3D saw an improvement of +7.33% and +5.36% respectively, as shown in Table II. Even in the AOS benchmark (90.37, 88.77, and 87.18), the hard category AP is increased, as shown in Table III. Figure. 2 illustrates our model's ability to detect objects, displaying 3D bounding boxes for cars, pedestrians, and cyclists across different lighting conditions, as demonstrated using the KITTI dataset. Our PointBi-FPN with Bi-FPN as the backbone improved the network's overall performance for the car category. Our network showed superior performance, especially in the "hard" difficulty level, with +5.28%, +5.36%, and +0.8% increase in AP across the BEV, 3D, and AOS benchmarks, respectively.

Method	Modality	Car			Pedestrian			Cyclist		
		Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D [3]	Mono	5.22	5.19	4.13	N/A	N/A	N/A	N/A	N/A	N/A
VoxelNet [20]	LiDAR	89.35	79.26	77.39	46.13	40.74	38.11	66.70	54.76	50.55
SECOND [17]	LiDAR	88.07	79.37	77.95	55.10	46.27	44.76	73.67	56.04	48.78
AVOD-FPN [8]	LiDAR&Image	88.53	83.79	77.90	58.75	51.05	47.54	68.09	57.48	50.77
Vote3D [15]	LiDAR	56.80	47.99	42.56	44.48	35.74	33.72	41.43	31.24	28.60
PointPillars [1]	LiDAR	88.32	85.70	79.65	58.66	50.23	47.19	79.14	62.25	56.00
PointPillar with Bi-FPN	LiDAR	**89.41**	**87.19**	**84.93**	57.36	**50.69**	**49.23**	**80.17**	61.87	**56.14**

TABLE I
PERFORMANCE COMPARISON IN BEV DETECTION.

Method	Modality	Car			Pedestrian			Cyclist		
		Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D [3]	Mono	2.53	2.31	2.31	N/A	N/A	N/A	N/A	N/A	N/A
BirdNet [2]	LiDAR	14.75	13.44	12.04	14.31	11.80	10.55	18.35	12.43	11.88
VoxelNet [20]	LiDAR	77.47	65.11	57.73	39.48	33.69	31.5	61.22	48.36	44.37
AVOD-FPN [8]	LiDAR&Image	81.94	71.88	66.38	50.80	42.81	40.88	64.00	52.18	46.61
SECOND [17]	LiDAR	83.13	73.66	66.20	51.07	42.56	37.29	70.51	53.85	46.90
PointPillars [1]	LiDAR	78.95	75.02	68.50	52.08	43.53	41.49	75.78	59.07	52.92
PointPillar with Bi-FPN	LiDAR	**86.28**	**76.74**	**73.86**	51.87	**44.86**	41.12	75.46	**61.13**	50.18

TABLE II
PERFORMANCE COMPARISON IN 3D DETECTION

Method	Modality	Car			Pedestrian			Cyclist		
		Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D [3]	Mono	2.53	2.31	2.31	N/A	N/A	N/A	N/A	N/A	N/A
3DVP [16]	LiDAR	86.92	74.59	64.11	N/A	N/A	N/A	N/A	N/A	N/A
VeloFCN [9]	LiDAR	70.58	52.84	46.14	N/A	N/A	N/A	N/A	N/A	N/A
LSVM-MDPM-sv [7]	LiDAR	67.27	55.77	43.59	43.58	35.49	32.42	27.54	22.07	21.45
BirdNet [2]	LiDAR	50.85	35.81	34.90	21.34	17.26	16.67	41.48	30.76	28.66
PointPillars [1]	LiDAR	90.19	88.76	86.38	58.05	49.66	47.88	82.43	68.16	61.96
PointPillar with Bi-FPN	LiDAR	**90.37**	**88.77**	**87.18**	**58.87**	48.56	46.12	**82.76**	64.13	60.18

TABLE III
PERFORMANCE COMPARISON IN AOS DETECTION

Fig. 2. Predictions of the proposed PointBi-FPN model: Car (blue bounding box), Pedestrian (green), Cyclist (red). Left top: Predictions in crowded areas with overlapping objects. Left bottom: Predictions at shadow, Top right: Predictions at long distance.

VI. Conclusion

We aimed to improve 3D object detection in autonomous vehicles using LiDAR data. Building on PointPillars, we integrated Bi-FPN as the backbone network, resulting in significant improvements. Our experiments on the KITTI dataset showed that our PointBi-FPN outperformed the original PointPillars model across all detection benchmarks (BEV, 3D, and AOS), especially in challenging "hard" difficulty labels. This demonstrates Bi-FPN's effectiveness in multiscale feature aggregation, making our approach promising for safe and efficient autonomous driving in complex urban environments.

Acknowledgment

This work was supported by the project entitled LiDAR and Camera Sensors Data-based Deep Learning Algorithms for Autonomous Driving System (EEQ/2021/000864), Department of Science and Technology (DST), India.

References

[1] Alex H. Lang and, Sourabh Vora and, H.C.a.L.Z.a.J.Y., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR. pp. 12697–12705. Computer Vision Foundation / IEEE (2019)

[2] Beltrán, J., Guindel, C., Moreno, F.M., Cruzado, D., Garcia, F., De La Escalera, A.: Birdnet: a 3d object detection framework from lidar information. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). pp. 3517–3523. IEEE (2018)

[3] Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., Urtasun, R.: Monocular 3d object detection for autonomous driving. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2147–2156 (2016)

[4] Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3d object detection network for autonomous driving. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 1907–1915 (2017)

[5] Engelcke, M., Rao, D., Wang, D.Z., Tong, C.H., Posner, I.: Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 1355–1361. IEEE (2017)

[6] Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: 2012 IEEE conference on computer vision and pattern recognition. pp. 3354–3361. IEEE (2012)

[7] Geiger, A., Wojek, C., Urtasun, R.: Joint 3d estimation of objects and scene layout. Advances in Neural Information Processing Systems (2011)

[8] Ku, J., Mozifian, M., Lee, J., Harakeh, A., Waslander, S.L.: Joint 3d proposal generation and object detection from view aggregation. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1–8. IEEE (2018)

[9] Li, B., Zhang, T., Xia, T.: Vehicle detection from 3d lidar using fully convolutional network. arXiv preprint arXiv:1608.07916 (2016)

[10] Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on computer vision. pp. 2980–2988 (2017)

[11] Luo, Q., Ma, H., Tang, L., Wang, Y., Xiong, R.: 3d-ssd: Learning hierarchical features from rgb-d images for amodal 3d object detection. Neurocomputing **378**, 364–374 (2020)

[12] Premebida, C., Carreira, J., Batista, J., Nunes, U.: Pedestrian detection combining rgb and dense lidar data. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (2014)

[13] Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation supplementary material (2017)

[14] Shi, S., Wang, X., Li, H.: Pointrcnn: 3d object proposal generation and detection from point cloud. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 770–779 (2019)

[15] Wang, D.Z., Posner, I.: Voting for voting in online point cloud object detection. In: Robotics: science and systems. vol. 1, pp. 10–15. Rome, Italy (2015)

[16] Xiang, Y., Choi, W., Lin, Y., Savarese, S.: Data-driven 3d voxel patterns for object category recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1903–1911 (2015)

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

[18] Yang, B., Luo, W., Urtasun, R.: Pixor: Real-time 3d object detection from point clouds. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 7652–7660 (2018)

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

[20] Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018. pp. 4490–4499. Computer Vision Foundation / IEEE Computer Society (2018). https://doi.org/10.1109/CVPR.2018.00472