Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 1, No. 2, 2022

Multi-Stage Transformer 3D Object Detection Method

Yanfei Liu [1], Kanglin Ning [1, *]

[1] Department of Basic Courses, High-tech Institute of Xi'an, Xi'an, 710025, Shanxi Province, China

[*] **Corresponding author**: Kanglin Ning (Email: ningkangl@icloud.com)

Abstract: With the development of autonomous driving, 3D object detection has experience great expectations. As the light detection and ranging (LiDAR) sensor can precisely measure the distance between environments and themselves, it has become the key component of current 3D object detection methods. However, the varing density and unstructure storage of LiDAR points cloud make it hard for feature learning. To tackle this problem, this paper proposes a multi-task transformer 3D object detection method.This method include a fast transformer based 3D encoder and a multi-stage transformer decoder. Extensive experiments demonstrate that our method can supress current other 3D object detection methods with a clear margin.

Keywords: 3D object detection; Point Cloud; 3D Vision; Deep Learning.

1. Introduction

3D Object Detection from the point cloud has experienced great flourish and has become a practical solution to robotic vision. Different from the image, the point cloud generated by LiDAR can precisely measure the distance from the LiDAR to the environment, which is crucial for a 3D object detection scenario. However, the varying density of LiDAR point cloud leads to challenging feature representation learning.

Like the detection methods on 2D images, these 3D detectors can also be divided into two groups: one-stage detectors– and two-stage detectors–, in terms of model structure. As two-stage detectors can take advantage of multi-stride by their region of interest (ROI) head, they can usually show better accuracy in the classification confidence and the box regression than those one-stage detectors. In these two-stage methods, their region proposal network first provides the proposal of bounding boxes. Then, based on those bounding boxes, the ROI head will perform the ROIPooling operation from the original point cloud and temporary voxel space to refine those provided proposals. Through the virtual point sampling strategy, these ROI heads could learn more valuable geometry information from objects far from the LiDAR sensors or severe occlusion. However, the computation brought by these ROI heads makes two-stage detectors challenging to meet the real-time requirement of the autonomous driving scene.

Different from the two-stage detectors, one-stage detectors can usually achieve better speeds. As the size of objects is invariant in the 3D scene, one-stage 3D detectors usually do not concern the pyramid features. Yet, some current works on 2D tiny object detection show that pyramid features can help the detectors perceive the difference between local and global tiny objects, thus helping these detectors achieve better accuracy. This can prove that their carelessness causes poor performance on small-size objects of one-stage 3D detectors with pyramid features. This motivates us to design a one-stage 3D detector to pay more attention to those pyramid features.

This paper presents a one-stage object detection framework consisting of a fast version transformer 3D encoder and a multi-stage transformer decoder module. Then a anchor-free centerpoint head is adopt to prediction the final bounding box prediction. In the proposed fast version transformer 3D encoder blocks, there is a fast version multi-head self-attention has been deployed. To help the proposed 3D detector achieve better performance, we have desin the multi-stage transformer 3D decoder module, which was consisted of fast version transformer decoder blocks. The proposed multi-stage transformer decoder module will collect the multi-stride information from the temporal output of transformer 3D encoder. Extensive experiments constructed on KITTI and Waymo Open Datasets show that our proposed methods can better balance speed and accuracy.

In summary, we make three-fold contribution:

We have proposed a one-stage 3D object detector, which uses the multi-stride residual 3D backbone and the path aggregation 2D backbones to help the detector make great use of pyramid features.

We design a voxel-level auxiliary network, which removes the need for voxel-to-point operations and achieves a better generality.

We conduct extensive experiments on both KITTI Dataset and Waymo Open Dataset. Its results show that our methods could achieve a great balance between accuracy and computation cost.

2. Methods

In this section, we first present the multi-path transformer 3D encoder. Then, a transformer decoder will be introducing to integrate multi-stage information from 3D encoder.

2.1. Fast Transformer 3D Encoder

Let $P = \{p_1, p_2, p_3, \dots, p_n\}$ denote a frame point input sequence, where $p_i = (x_i, y_i, z_i, r_i)$ denotes the i-th point. To save computation, we first subdivide the 3D space into equally spaced voxels. Let's define the voxelized point cloud sequence P as $V = \{v_1, v_2, v_3, \dots, v_n\}$ where $v_i = (c_i, d_i)$ represent a voxelized point cloud. The $d_i = (r_i, x_i - x_{ci}, y_i - y_{ci}, z_i - z_{ci})$ denote the feature vector for point p_i and c_i denote the centriod of voxel cells. As the transformer block do not concern the position information about input point cloud sequence, we should adopt a position embedding for them. As mentioned in [], the Fourier position can help deep learning module learn the high-dimension texture information in a low-dimension feature space. In this paper,

we adopt this Fourier position embedding method to help our transformer module to efficiently learn high-dimension information. For a give point p_i, the position embedding process can be described as:

$$\gamma_i = \left[cos\left(2\pi\sigma^{i/n}p_i\right), sin\left(2\pi\sigma^{i/n}p_i\right)\right] \quad (1)$$

Then, we will concat the fourier position vector with the vector feature to get the input feature vector set F. With these embedded feature sequences F and position V, the proposed fast transformer 3D encoder will be used to extract the abstract semantic feature. As shown in Fig. 1, our fast transformer 3D encoder consisted by a voxel level group block and a transformer encoder block. The transformer block can be described as follow:

$$\widehat{F_{l-1}} = FMSA\left(LN\left(GELU(F_{l-1})\right)\right) + F_{l-1}$$
$$F_l = MLP\left(LN\left(GELU(\widehat{F_{l-1}})\right)\right) + \widehat{F_{l-1}} \quad (2)$$

where F_{l-1} denote the input feature sequence, $\widehat{F_{l-1}}$ denote the temporal feature. The $FMSA$ denote the fast version mult-head self-attention module, LN denote the layer

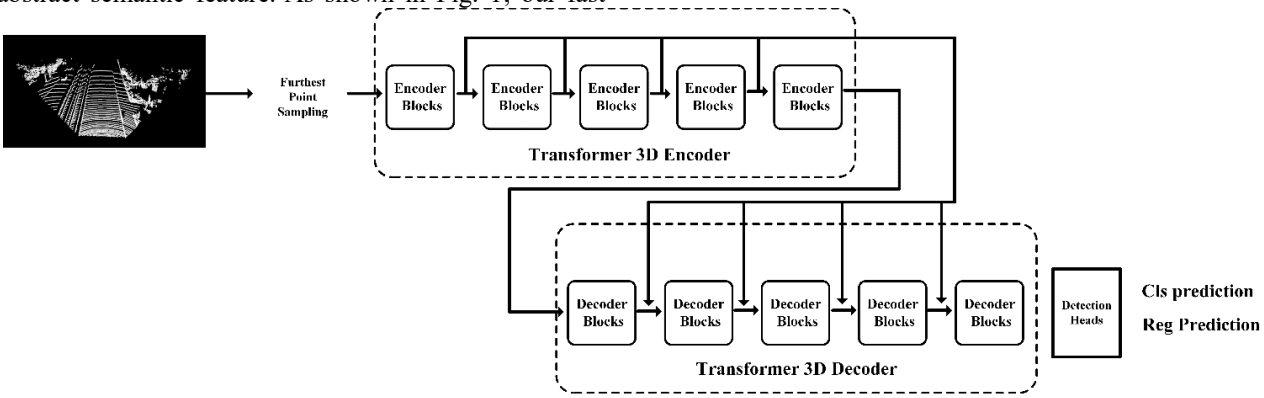

Figure 1. The pipeline of the proposed multi-stage transformer 3D object detection method

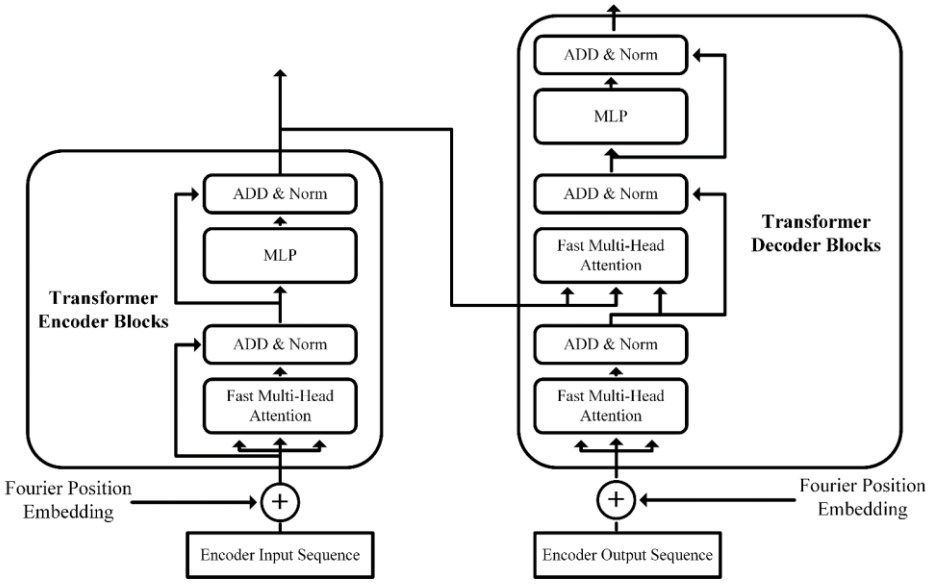

Figure 2. The architecture of the proposed fast version transformer encoder and decoder blocks.

Table 1. Performance Comparision on KITTI Validation Sets of 3D Average Precision

Method	Stage	Car			Pedestrian			Cyclist		
		Mod	Easy	Hard	Mod	Easy	Hard	Mod	Easy	Hard
Part A2	Two	82.79	89.69	78.92	65.42	71.23	60.33	72.21	90.03	69.54
PV RCNN	Two	83.37	90.23	80.89	55.64	63.69	50.59	71.07	88.51	66.64
VoTr-TSD	Two	84.85	92.04	82.56	54.84	66.13	51.69	72.33	89.42	67.42
Voxel RCNN	Two	84.64	92.48	81.14	58.28	65.85	53.15	71.33	90.26	66.87
PointPillars	One	74.13	85.51	69.21	48.33	53.32	43.62	61.65	79.10	57.92
CADNet	One	76.24	85.30	70.45	-	-	-	62.44	81.31	59.42
CIA-SSD	One	81.28	90.41	74.41	-	-	-	-	-	-
SECOND	One	78.36	87.42	75.40	51.64	54.99	45.41	61.82	79.51	57.59
CenterPoint	One	81.26	88.46	78.71	51.33	55.39	48.33	65.49	82.03	62.03
Our Method	One	81.22	89.54	79.11	56.22	58.32	51.11	69.30	84.71	64.33

Table 2. Performance Comparision on KITTI Validation Sets of Bird's Eye View Average Precision

Method	Stage	Car			Pedestrain			Cyclist		
		Mod	Easy	Hard	Mod	Easy	Hard	Mod	Easy	Hard
Part A2	Two	90.32	95.33	88.49	72.43	77.32	70.41	89.31	93.23	86.33
PV RCNN	Two	90.69	92.88	88.53	61.18	69.37	56.26	75.15	94.47	70.38
VoTr-TSD	Two	90.67	94.90	89.98	62.34	68.13	57.49	74.15	93.34	69.98
Voxel RCNN	Two	91.17	95.44	88.42	59.38	65.86	55.22	74.08	91.62	71.94
PointPillars	One	87.37	90.01	85.22	54.43	57.33	51.10	62.73	79.90	55.58
CADNet	One	88.02	90.41	86.30	-	-	-	65.12	79.51	58.25
CIA-SSD	One	89.28	91.59	85.47	-	-	-	-	-	-
SECOND	One	88.53	92.64	87.55	53.79	58.41	50.64	85.90	70.35	66.38
CenterPoint	One	91.30	94.45	88.59	59.32	63.42	56.49	71.44	90.04	70.03
Our Method	One	91.42	92.14	89.22	61.32	64.12	57.63	71.64	86.34	70.62

norm module, MLP denote the mult-layer preception module. The original version mult-head self-attention block can be described as:

$$q_l = F_l W_q, k_l = F_l W_k, v_l = F_l W_v$$
$$F_{out} = softmax\left(\frac{q_l k_l^T}{\sqrt{d_k}}\right) v_l \quad (3)$$

where W_q, W_k, W_v denote the feature map matrix, which is used to project the input feature sequence into a high-dimension space. As mentioned in [], the softmax operation consumes relatively large computing resources. To reduce the overhead of the model and improve the abaptability of the model to autonomous driving scenarios, this paper uses a cosine function to replace the softmax operation. Thus, the fast version mult-head self-attention module can be described as:

$$q_l = F_l W_q, k_l = F_l W_k, v_l = F_l W_v$$
$$F_{out} = cos(q_l, k_l)/\beta + B \quad (4)$$

where B is the relative position bias matrix of each point; β is a learnable scalar, non-shared across heads and layers. The initial value of β should be set larger than 0.01. The cosine function is naturally normalized, and thus can have milder attention values.

2.2. Ball Query Based Downsample Strategy

As the input point cloud sequence is too large, we adopt the point set aggregation operation like pointnet++. A set abstraction layer takes an $N \times (d + c)$ matrix as input, which is outputed by our fast version transformer 3D encoder. It outputs an $N^o \times (d + C^o)$ matrix of N^o subsampled points with d-dim coordinates and new C^o-dim feature vectors summarizing local context. Then, the output are groups of point sets of size $N^o \times K \times (d + C)$, where each group corresponds to a local region and K is the number of points in the neighborhood of centroid points. As shown in Fig. 1, the ball query downsample modules are connected behind each transformer 3D encoder blocks.

2.3. Mult-Stage Transformer 3D Decoder

As the size of objects is invariant in the 3D scene, one-stage 3D detectors usually do not concern the mult-stage features. In addition, current point-based method usually only adopt encoder only architecture to perform the 3D detection tasks. This motivate us to design a mult-stage transformer 3D encoder. The architecture of our mult-stage transformer 3D detector is shown in Fig. 2. The mult stage transformer 3D decoder is consisted by severals transformer decoder blocks. Different from encoder block, the transformer decoder block is consisted by two fast version mult-head self-attention modules. The first fast version mult-head self-attention head module will use the final output of transformer 3D encoder, while lthe second fast version mult-head self-attention head module will use the temporal output of 3D encoder as query and key input matrix.

3. Experiment

3.1. Experiment Setting

KITTI Dataset: The KITTI dataset provides 7481 training samples and 7518 test samples. The training samples are provided with annotated of car, pedestrain, and cyclist categories. In this paper, we following the common protocal of [], and dividing these trainig samples into training set and validation set. The training set contain 3712 samples while validation set contain 3769 samples. Each category objects are split into three levels based on occlusion and their distance from the LiDAR sensor. As the test set do not have the ground-truth labels, we need to submit the prediction results to the KITTI official test server to get the final result.

Implement Setting: We adopt the ADAM [] optimizer and onecycle learning rate shedule methods. We train our model 80 epoches on the KITTI dataset. The init learning rate is set at 0.003 and the lower-upper is set at 0.0000001. Our detector will contain 5 transformer encoder blocks and 5 transformer decoder blocks.

Experiment Environment: Our local server adopts a Core i9 10980XE with two RTX 3090 GPU. The local server deploys the Ubuntu 20.04 systems with python 3.8 and pytorch 1.8.1.

3.2. Comparision on KITTI

We compare our method with state-of-the-art 3D detector in KITTI validation set and test set. The results of validation set are come from our local experiment by running those detector's official release code. And these results are shown in Table1 and Table2. The results of test set are come from the KITTI official benchmark. The test results are shown in Table3. We adopt the 3D average precision (AP) and bird's-eye-view average precision (BEV AP) as metric.

From the result of Table 1, we can see that our methods could achieve a significantly better performance than other one-stage detectors. Even compared with some two-stage

methods, our method's accuracy is still having considerable competitiveness. In the car category objects with moderate difficulty level, our method outperforms the SECOND with xxx points 3D AP, outperform the pointpillar with xxx points 3D AP, and outperform the CIA-SSD with xxx points 3D AP. In the car category objects with hard level difficulty, our method outperforms the SECOND with xxx points 3D AP, outperform the pointpillar with xxx points 3D AP, and outperform the CIA-SSD with xxx points 3D AP. In the pedestrain category objects with moderate level difficulty, our method outperforms the SECOND with xxx points 3D AP, outperform the pointpillar with xxx points 3D AP, and outperform the CIA-SSD with xxx points 3D AP. In the cyclist category objects with moderate difficulty level, our method outperforms the SECOND with xxx points 3D AP, outperform the pointpillar with xxx points 3D AP, and outperform the CIA-SSD with xxx points 3D AP. In the results of Table 2, our method still has a clear margin with other listed one-stage 3D detectors. It proves that our proposed detector could stride a better balance between speeds and accuracy than other 3D detection methods.

4. Conclusion

In this paper, we proposed a multi-stage transformer 3D object detection method. In this framework, a fast version transformer 3D encoder block is proposed to combine the ball-query group downsample methods to learn feature from the origin point cloud space. Then, a mult-stage transformer 3D decoder is added to collect information from the 3D encoder's temporal output. Extensive experiments results show that our method hav a significant better performance than other one-stage detectors. In the future, we will attempt to integrate the multi-modal methods with this framework.

References

[1] Y. Zhou and O. 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, 2018, pp. 4490–4499.

[2] Y. Yan, Y. Mao, and B. Li, 'Second: Sparsely embedded convolutional detection', Sensors, vol. 18, no. 10, p. 3337, 2018.

[3] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, 'Pointpillars: Fast encoders for object detection from point clouds', in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 12697–12705.

[4] G. Zhang, S. Lu, and W. Zhang, 'CAD-Net: A context-aware detection network for objects in remote sensing imagery', IEEE Trans. Geosci. Remote Sens., vol. 57, no. 12, pp. 10015–10024, 2019.

[5] W. Zheng, W. Tang, S. Chen, L. Jiang, and C.-W. Fu, 'CIA-SSD: Confident IoU-Aware Single-Stage Object Detector From Point Cloud', ArXiv Prepr. ArXiv201203015, 2020.

[6] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, 'Structure aware single-stage 3d object detection from point cloud', in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11873–11882.

[7] Z. Yang, Y. Sun, S. Liu, and J. Jia, '3dssd: Point-based 3d single stage object detector', in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 11040–11048.

[8] R. Zhang, S. Tang, L. Liu, Y. Zhang, J. Li, and S. Yan, 'High Resolution Feature Recovering for Accelerating Urban Scene Parsing.', in IJCAI, 2018, pp. 1156–1162.

[9] T. Yin, X. Zhou, and P. Krahenbuhl, 'Center-based 3d object detection and tracking', in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 11784–11793.

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

[11] S. Shi, Z. Wang, X. Wang, and H. Li, 'Part-aˆ 2 net: 3d part-aware and aggregation neural network for object detection from point cloud', ArXiv Prepr. ArXiv190703670, vol. 2, no. 3, 2019.

[12] S. Shi et al., 'Pv-rcnn: Point-voxel feature set abstraction for 3d object detection', in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 10529–10538.

[13] S. Shi et al., 'PV-RCNN++: Point-voxel feature set abstraction with local vector representation for 3D object detection', ArXiv Prepr. ArXiv210200463, 2021.

[14] P. Bhattacharyya and K. Czarnecki, 'Deformable PV-RCNN: Improving 3D object detection with learned deformations', ArXiv Prepr. ArXiv200808766, 2020.

[15] Z. Li, Y. Yao, Z. Quan, W. Yang, and J. Xie, 'Sienet: spatial information enhancement network for 3d object detection from point cloud', ArXiv Prepr. ArXiv210315396, 2021.

[16] S. Pang, D. Morris, and H. Radha, 'CLOCs: Camera-LiDAR object candidates fusion for 3D object detection', in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020, pp. 10386–10393.

[17] A. Mahmoud, J. S. Hu, and S. L. Waslander, 'Dense Voxel Fusion for 3D Object Detection', ArXiv Prepr. ArXiv220300871, 2022.

[18] X. Wu et al., 'Sparse Fuse Dense: Towards High Quality 3D Detection with Depth Completion', ArXiv Prepr. ArXiv220309780, 2022.

[19] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, 'Voxel R-CNN: Towards High Performance Voxel-based 3D Object Detection', ArXiv Prepr. ArXiv201215712, 2020.

[20] H. Sheng et al., 'Improving 3d object detection with channel-wise transformer', in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 2743–2752.

[21] J. S. Hu, T. Kuai, and S. L. Waslander, 'Point density-aware voxels for lidar 3d object detection', in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 8469–8478.

[22] Z. Li, F. Wang, and N. Wang, 'Lidar r-cnn: An efficient and universal 3d object detector', in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 7546–7555.

[23] C. Lee et al., 'Interactive Multi-Class Tiny-Object Detection', in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 14136–14145.