Self-Attention Based Context-Aware 3D Object Detection

Prarthana Bhattacharyya Chengjie Huang Krzysztof Czarnecki
University of Waterloo, Canada
{p6bhatta, c.huang, k2czarne}@uwaterloo.ca

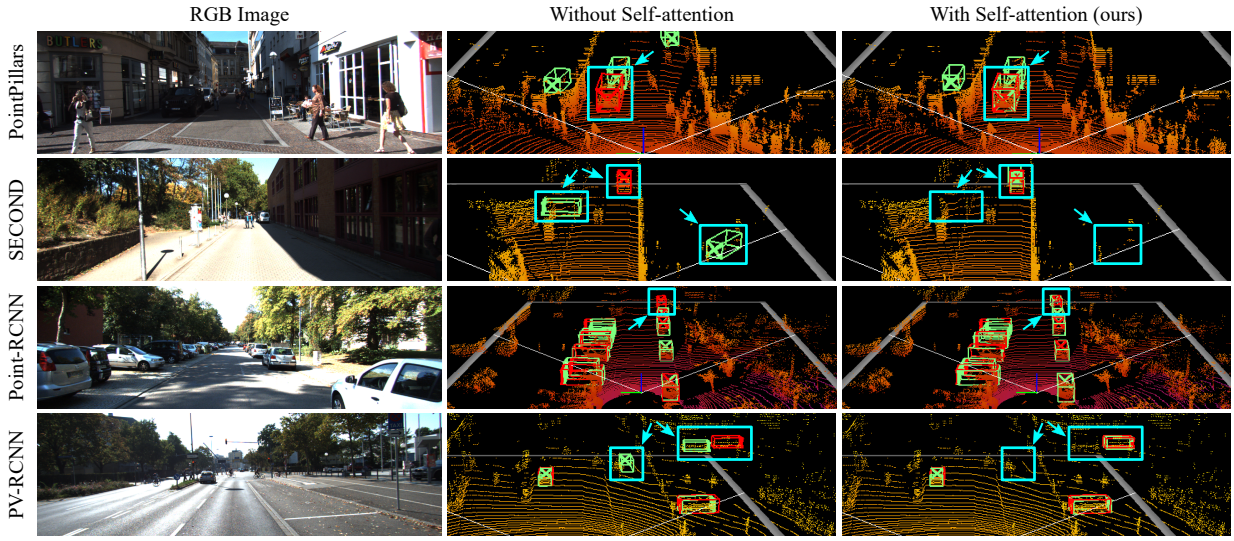


Figure 1: Performance illustrations on KITTI *val*. Red bounding box represents ground truth; green represents detector outputs. From left to right: (a) RGB image of challenging scenes. (b) Result of the state-of-the-art methods: PointPillars [18], SECOND [48], Point-RCNN [35], PV-RCNN [34]. (c) Result of our full self-attention (FSA) augmented baselines, which uses significantly fewer parameters and FLOPs. FSA attends to the entire point-cloud to produce global context-aware feature representations. Our method identifies missed detections and removes false positives.

Abstract

Most existing point-cloud based 3D object detectors use convolution-like operators to process information in a local neighbourhood with fixed-weight kernels and aggregate global context hierarchically. However, recent work on non-local neural networks and self-attention for 2D vision has shown that explicitly modeling global context and long-range interactions between positions can lead to more robust and competitive models. In this paper, we explore two variants of self-attention for contextual modeling in 3D object detection by augmenting convolutional features with self-attention features. We first incorporate the pairwise self-attention mechanism into the current state-of-the-art BEV, voxel and point-based detectors and show consistent improvement over strong baseline models while simultaneously significantly reducing their parameter footprint and computational cost. We also propose a self-attention variant that samples a subset of the most representative features by learning deformations over randomly sampled locations. This not only allows us to scale explicit global contextual modeling to larger point-clouds, but also leads to more discriminative and informative feature descriptors. Our method can be flexibly applied to most state-of-the-art detectors with increased accuracy and parameter and compute efficiency. We achieve new state-of-the-art detection performance on KITTI and nuScenes datasets. Code is available at https://github.com/AutoVision-cloud/SA-Det3D.

1. Introduction

3D object detection has been receiving increasing attention in the computer vision and graphics community, driven particularly by the ubiquity of LiDAR sensors and

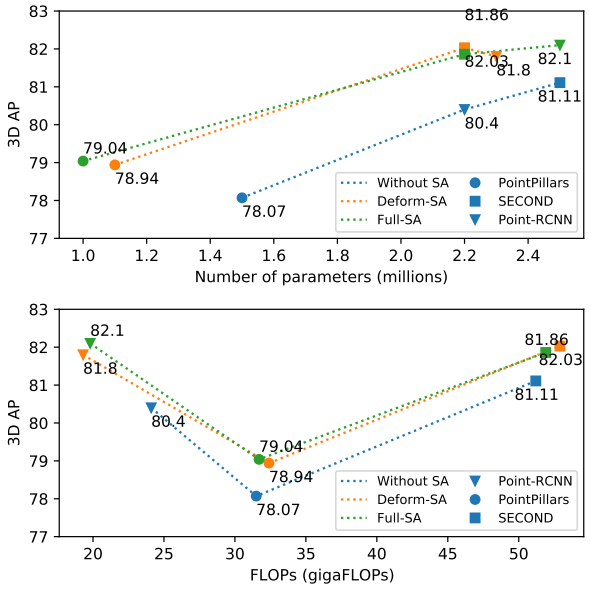


Figure 2: For similar number of parameters and FLOPs, self-attention (SA) systematically improves 3D object detection across state-of-the-art 3D detectors (PointPillars [18], SECOND [48], Point-RCNN [35]). AP on moderate Car class of KITTI val split (R40) vs. the number of parameters (Top) and GFLOPs (Bottom) for baseline models and proposed baseline extensions with Deformable and Full SA.

its widespread applications in autonomous driving and robotics. Point-cloud based 3D object detection has especially witnessed tremendous advancement in recent years with several milestone methods having been proposed [59, 50, 49, 48, 4, 28, 18, 35, 27, 47, 34, 53, 52, 13, 36]. Grid-based methods first transform the irregular point-clouds to regular representations such as 2D bird's-eye view maps or 3D voxels and then process them using 2D/3D convolutional networks (CNN) to extract features for detection. The recent point-based methods sample points from the raw point-cloud and query a local group around the each sampled point to define convolution-like operations [45, 40, 23, 16, 30] for point-cloud feature extraction.

Both 2D/3D CNNs and point-wise convolutions process a local neighbourhood and aggregate global context by applying these feature extractors hierarchically across many layers. This has several limitations: the number of parameters scales poorly with increased size of the receptive field; learned filters are stationary across all locations; and it is challenging to coordinate the optimization of parameters across multiple layers to capture patterns in the data [56].

In addition to the limitations of the feature aggregation mechanism, point-cloud based 3D object detectors also have to deal with missing/noisy data and a large imbalance in points for nearby and faraway objects. This motivates the need for a feature extractor that can learn global point-cloud correlations to produce more powerful, discriminative and robust features. For example, there is a strong correlation between the orientation features of cars in the same lane and this can be used to produce more accurate detections especially for distant cars with fewer points. High-confidence false positives produced by a series of points that resemble a part of an object can be also be eliminated by adaptively acquiring context information at increased resolutions.

Self-attention [41] has recently emerged as a basic building block for capturing long-range interactions in many applications. The key idea of self-attention is to acquire global information as a weighted summation of features from all positions to a target position where the corresponding weight is calculated dynamically via a similarity function between the features in an embedded space at these positions. The number of parameters is independent of the scale at which self-attention processes long-range interactions. Inspired by this idea, we propose two self-attention based context-aware modules—Full Self-Attention (FSA) and Deformable Self-Attention (DSA)—to augment the standard convolutional features. Our FSA module computes pairwise interactions among all non-empty 3D entities, and the DSA module scales the operation to large point-clouds by computing self-attention on a representative and informative subset of features.

We test our method on KITTI [9] and the large scale nuScenes [5] dataset, across a wide range of architectures at different computational budgets. We demonstrate in Figure 2 that reducing the total number of convolutional filters and adding the FSA and DSA modules lead to parameter and compute efficient models that consistently outperform their baseline implementations with similar parameters, while also outperforming the original top-performing implementations (Table 1). In Figure 1, we provide qualitative examples to show the benefits of context modeling. Finally, we also show state-of-the-art results on the KITTI and nuScenes test set.

Contributions

- We propose the first generic self-attention based context aggregation module for 3D object detectors that can be applied across a range of modern architectures including BEV [18], voxel [48], point [35] and point-voxel [34] based detectors. We show that we can outperform strong baseline implementations with significantly fewer parameters and computational cost on the KITTI validation set.

- We design a scalable self-attention variant that learns to deform randomly sampled locations to cover the most representative and informative parts and aggregate context on this subset. This allows us to aggregate global context in large point-clouds like nuScenes [5].

- Extensive experiments demonstrate the benefits of contextual information aggregation using our proposed methods for 3D object detection. Our network outperforms state-of-the-art methods on both KITTI [9] and nuScenes [5] datasets.

2. Related Works

2.1. 3D Object Detection

Current 3D object detectors can be divided into bird's-eye view (BEV), voxel and point-cloud based methods.

BEV-based methods MV3D [6] is a seminal work on multi-view fusion with BEV features and others [20, 17, 42] extend MV-3D with improved multi-sensor fusion strategies. More efficient BEV representations followed [18, 49, 50, 4]. PointPillars [18] outperforms most fusion-based approaches and runs 2-4 times faster, making it suitable for real-time applications.

Voxel-based methods Voxel-based approaches, on the other hand, divide the point-cloud into 3D voxels and process them using 3D CNNs. VoxelNet [59] first proposed to produce an end-to-end trainable framework with 3D CNNs. SECOND [48] introduced sparse 3D convolutions [12] for efficient 3D processing of voxels. CBGS [60] extends this work by adding multiple heads to the detector.

Point-based methods Inspired by the success of PointNet [29], F-PointNet [28] first applied point features to 3D detection in point cloud crops that correspond to the 2D camera-image detections. Point-RCNN [35] segments 3D point-clouds and uses the segmentation features to better refine box proposals. 3DSSD [52] is a pioneering work that introduces a fusion sampling strategy that allows removing the refinement module to achieve real-time speed.

Point-Voxel based methods STD [53], PV-RCNN [34] and SA-SSD [13] leverage both voxel and point-based abstractions to produce more accurate bounding boxes.

While a lot of recent progress has been made, these approaches mostly use stacks of convolutions to perform local reasoning. Different from these approaches, we propose a simple, scalable and generic 3D feature extraction framework that adaptively aggregates context information from the entire point-cloud and allows remote regions to directly communicate information during each stage of feature processing. This module is flexible and can be added on top of modern LiDAR-based 3D detector architectures.

2.2. Context Modeling

Attention mechanisms For discriminative feature extraction, we want the model to selectively focus on salient parts of objects. Attention has enjoyed widespread adoption as a computational module for computer vision because of its ability to capture long-range dependencies and bias computation towards the most informative components of an input signal [15, 44, 14]. Self-attention, which relates different elements of a representation, has been instrumental to achieving state-of-the-art results in machine translation [41]. Combining self-attention with convolutions is a theme shared by recent work in natural language processing [46], image recognition [3], 2D object detection [24], person re-identification [58], activity recognition [43] and reinforcement learning [55].

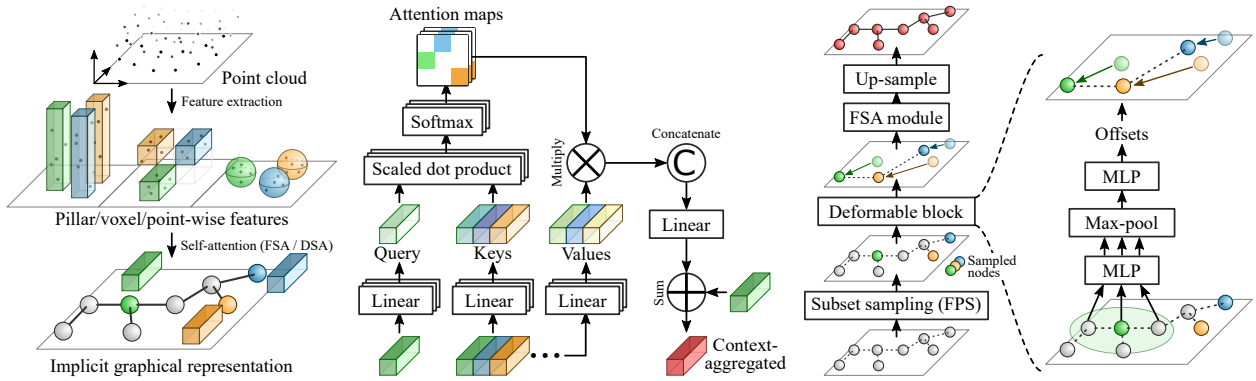
Attention for point-clouds Using self-attention to aggregate global structure in point-clouds for 3D object detection remains a relatively unexplored domain. PC-CAN [57] uses point-wise attention to aggregate multi-scale local context for point-cloud based retrieval. TA-Net [21] is a 3D object detector that proposes to use standard attention [2] to aggregate context from a local point-voxel neighbourhood by using a shared transformation function across all voxels. However these attention mechanisms use local context, whereas relevant contextual information can occur anywhere in the point-cloud and hence we need global context modeling.

PAT [51] uses self-attention to build point representations for end-to-end point-cloud classification. Recent research [32], however, shows that self-attention aggregation is more stable and effective when used in combination with convolutions. PMP-Net [54] uses self-attention for spatial and temporal context aggregation in large-scale point-clouds for 3D object detection, but because they use pairwise self-attention mechanism, it does not scale to the entire point-cloud. Consequently, they process k-nearest neighbours for context modeling .

To process global context for 3D object detection and scale to large point-clouds, Attentional PointNet [25] uses GRUs [7] to sequentially attend to different parts of the point-cloud. Learning global context by optimizing the hidden state of a GRU can be slow and inefficient however. In contrast, our method can processes context information adaptively for each location from the entire point-cloud while also scaling to large sets. Since the global context is fused with local-convolutional features, the training is stable and efficient.

3. Methods

In this section, we first introduce a Full Self-Attention (FSA) module for discriminative feature extraction in 3D object detection that aims to produce more powerful and robust representations by exploiting global context. Inspired by 2D deformable convolutions [8], we introduce a variant of FSA called Deformable Self-Attention (DSA). DSA can reduce computation from quadratic to linear in the number of context points and can scale to larger point-clouds. The



(a) Self-attention induced graph (b) Full self-attention module (FSA) (c) Deformable self-attention module (DSA)

Figure 3: Architectures of the proposed FSA and DSA modules.

two proposed modules are illustrated in Figure 1.

3.1. Formulation

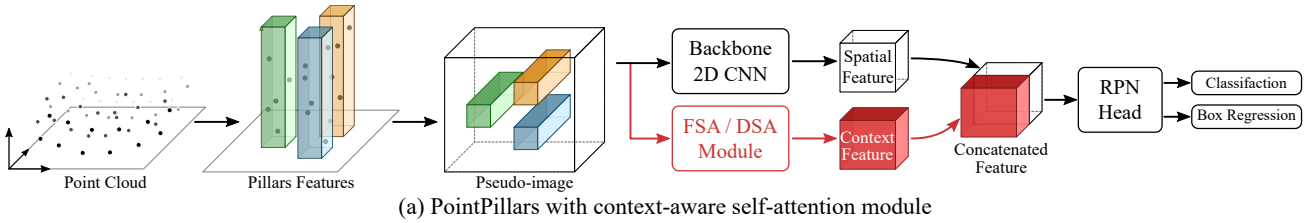
Attention-augmented architectures Generally, for the input set $\mathcal{X} = \{x_1, x_2, ...x_n\}$ of n correlated features, the goal of content-adaptive feature aggregation methods like attention [2] is to use spatial or channel-wise information to selectively emphasize some features and suppress less useful ones by learning a mask denoted as $a = (a_1, ..., a_n) \in R^n$. In detectors like TA-Net [21], the attention for a feature node is determined locally by applying a transformation function shared across all nodes. However, this does not fully exploit the correlations from a global view. The most common strategy to learn the attention value a_i of the i^{th} feature is to formulate it as a dot product between the original feature x_i and a learned global feature x_{global} obtained by using all the feature nodes (e.g. by concatenation or global average pooling) and passing them through a fully connected layer. In this construction unlike convolution, the aggregation weights can vary across different locations, depending on the content of the point-cloud. The attention-aggregated features are then fused with the convolutional features via addition, gating or concatenation to produce more powerful, flexible and robust representations that can model both local and long-range dependencies. This is used in 3D detectors like SCANet [22] and MLCVNet [47]. However x_{global} is shared across all locations and channels and does not adapt itself according to the input features, leading to a representation that is sub-optimal.

Self-Attention for 3D Object Detection In contrast, we propose to use self-attention introduced by Vaswani et al. [41], to exploit the pairwise similarities of the i^{th} feature node with all the feature nodes, and stack them to compactly represent the global structural information for the current feature node. Self-attention exploits a larger and more flexible receptive field to derive a different modulating feature x^i_{global} for individual feature positions.

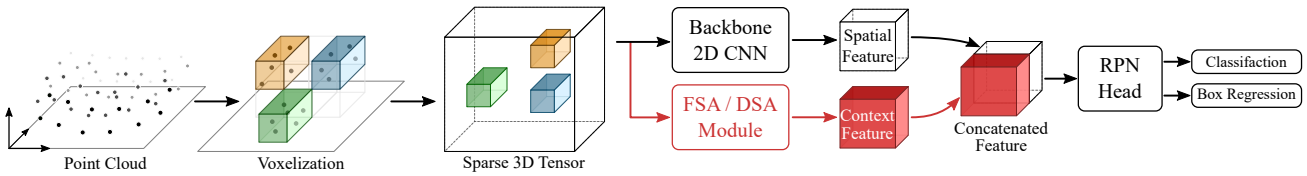
Mathematically, the set of pillar/voxel/point features and their relations are denoted by a graph $G = (\mathcal{V}, \mathcal{E})$, which comprises the node set $\mathcal{V} = \{\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_n \in R^d\}$, together with an edge set $\mathcal{E} = \{\mathbf{r}_{i,j} \in R^{N_h}, i = 1, ..., n$ and $j = 1, ..., n\}$. A self-attention module takes the set of feature nodes, and computes the edges (see Figure 1 (a)). The edge $\mathbf{r}_{i,j}$ represents the relation between the i^{th} node and the j^{th} node, and N_h represents the number of heads (number of attention maps in Figure 1 (b)) in the attention mechanism across d feature input channels as described below. We assume that N_h divides d evenly. The advantage of representing the processed point-cloud features as nodes in a graph is that now the task of aggregating global context is analogous to capturing higher order interaction among nodes by message passing on graphs [10], for which many mechanisms like self-attention exist.

3.2. Full Self-Attention Module

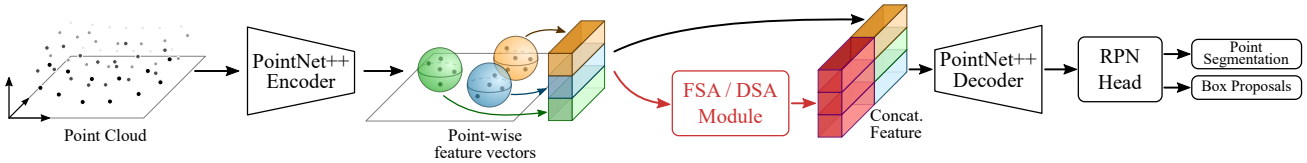
Our Full Self-Attention (FSA) module projects the features $\mathbf{x}_i$ through linear layers into matrices of query vectors Q, key vectors K, and value vectors V (see Figure 1(b)). The similarities between query $\mathbf{q}_i$ and all keys, $\mathbf{k}_{j=1:n}$, are computed by a dot-product, and normalized into attention weights $\mathbf{w}_i$, via a softmax function. The attention weights are then used to compute the pairwise interaction terms, $\mathbf{r}_{ij} = w_{ij}\mathbf{v}_j$. The accumulated global context for each node vector $\mathbf{a}_i$ is the sum of these pairwise interactions, $\mathbf{a}_i = \sum_{j=1:n} \mathbf{r}_{ij}$. As we mentioned in our formulation, we also use multiple attention heads, applied in parallel, which can pick up channel dependencies independently. The final output for the node i is then produced by concatenating the accumulated context vectors $\mathbf{a}_i^{h=1:N_h}$ across heads, passing it through a linear layer, normalizing it with layer normalization [1] and summing it with $\mathbf{x}_i$ (residual connection).



(a) PointPillars with context-aware self-attention module



(b) SECOND with context-aware self-attention module



(c) Point-RCNN with context-aware self-attention module

Figure 4: Proposed FSA/DSA module augmented network architectures for different backbone networks.

Advantages The important advantage of this module is that the resolution at which it gathers context is independent of the number of parameters and the operation is permutation-equivariant [19]. This makes it attractive to replace a fraction of the parameter-heavy convolutional filters at the last stages of 3D detectors with self-attention features for improved feature quality and parameter efficiency.

Complexity The pairwise similarity calculation is $\mathcal{O}(n^2 d)$ in nature. The inherent sparsity of point-clouds and the efficient matrix-multiplication based pairwise computation makes FSA a viable feature extractor in current 3D detection architectures. However, it is important to trade-off between accuracy and computational complexity in order to scale to larger point-clouds. In the next section, we propose our Deformable Self-Attention module to reduce the quadratic computation time of FSA.

3.3. Deformable Self-Attention Module

Our primary idea is to attend to a representative subset of the original node vectors in order to aggregate global context. We then up-sample this accumulated structural information back to all node locations using the feature propagation method proposed in PointNet++ [30]. The complexity of this operation is $\mathcal{O}(nmd)$, where m is the number of points chosen in the subset. In order for the subset to be representative, it is essential to make sure that the selected nodes cover the informative structures and common characteristics in 3D geometric space. Inspired by deformable convolution networks [8] in vision, we propose a geometry-guided vertex refinement module that makes the nodes self-adaptive and spatially recomposes them to cover locations which are important for semantic recognition. Our node offset-prediction module is based on vertex alignment strategy proposed for domain alignment [31, 11]. Initially m nodes are sampled from the point-cloud by farthest point sampling (FPS) with vertex features $\mathbf{x}_i$ and a 3D vertex position v_i. For the i^{th} node, the updated position v_i' is calculated by aggregating the local neighbourhood features with different significance as follows:

$$x_i^* = \frac{1}{k}\text{ReLU} \sum_{j \in \mathcal{N}(i)} W_{\text{offset}}(\mathbf{x}_i - \mathbf{x}_j) \cdot (v_i - v_j) \quad (1)$$

$$v_i' = v_i + \tanh(W_{\text{align}} x_i^*) \quad (2)$$

where $\mathcal{N}_i$ gives the i-th node's k-neighbors in the point-cloud and W_{offset} and W_{align} are weights learned end-to-end. The final node features are computed by a non-linear processing of the locally aggregated embedding as follows:

$$\mathbf{x}_i' = \max_{j \in \mathcal{N}(i)} W_{out} \mathbf{x}_j \quad (3)$$

This aggregated global information is then shared among all n nodes from the m representatives via up-sampling. We call this module a Deformable Self-Attention (DSA) module as illustrated in Figure 1(c).

Advantages The main advantage of DSA is that it can scalably aggregate global context for pillar/voxel/points. Another advantage of DSA is that it is trained to collect information from the most informative regions of the point-cloud, improving the feature descriptors.

Method	PointPillars [18]				SECOND [48]				Point-RCNN [35]				PV-RCNN [34]			
	3D	BEV	Param	FLOPs	3D	BEV	Param	FLOPs	3D	BEV	Param	FLOPs	3D	BEV	Param	FLOPs
Baseline	78.39	88.06	4.8 M	63.4 G	81.61	88.55	4.6 M	76.9 G	80.52	**88.80**	4.0 M	27.4 G	84.83	**91.11**	12 M	89 G
DSA	78.94	88.39	1.1 M	32.4 G	**82.03**	89.82	2.2 M	52.6 G	81.80	88.14	2.3 M	19.3 G	84.71	90.72	10 M	64 G
FSA	**79.04**	**88.47**	1.0 M	31.7 G	81.86	**90.01**	2.2 M	51.9 G	**82.10**	88.37	2.5 M	19.8 G	**84.95**	90.92	10 M	64.3 G
Improve.	**+0.65**	**+0.41**	**-79%**	**-50%**	**+0.42**	**+1.46**	**-52%**	**-32%**	**+1.58**	-	**-37%**	**-38%**	**+0.12**	-	**-16%**	**-27%**

Table 1: Performance comparison for moderate difficulty Car class on KITTI *val* split with 40 recall positions

4. Experiments

4.1. Network Architectures

We train and evaluate the proposed DSA and FSA modules using four state-of-the-art architecture backbones: PointPillars [18], SECOND [48], Point-RCNN [35], and PV-RCNN [34]. The architectures of the first three backbones are illustrated in Figure 4. We refer readers to our supplemental material for more details on network baselines, proposed variants and experiment preparation. The augmented backbones can be trained end-to-end without additional supervision.

4.2. Implementation Details

KITTI KITTI benchmark [9] is a widely used benchmark with 7,481 training samples and 7,518 testing samples. We follow the standard split [6] and divide the training samples into *train* and *val* split with 3,712 and 3,769 samples respectively. All models were trained on 4 NVIDIA Tesla V100 GPUs for 80 epochs with Adam optimizer and one cycle learning rate schedule [38]. We use standard data augmentation for point clouds. For baseline models, we reuse the pre-trained checkpoints provided by OpenPCDet.[1] We also use the same batch size and learning rates.

nuScenes nuScenes is a more recent large-scale benchmark for 3D object detection. The dataset contains 1,000 scenes (sequences of frames) resulting in a total of over 1.4 M annotated objects. Compared to KITTI, nuScenes is a much larger and more challenging dataset.

We train and evaluate a DSA model with PointPillars as the backbone architecture. All previous methods combine points from current frame and previous frames within 0.5 s, gathering about 400 k points per frame. FSA does not work in this case since the number of pillars in a point cloud is too large to fit the model in memory. In DSA, this issue is avoided by sampling a representative subset of pillars. The model was trained on 4 NVIDIA Tesla V100 GPUs for 20 epochs with a batch size of 16 using Adam optimizer. We use one cycle learning rate (LR) schedule with start LR of 0.0001 and max LR of 0.001.

[1] https://github.com/open-mmlab/OpenPCDet

5. Results

5.1. KITTI

On KITTI dataset, we report the performance of our proposed approaches on both *val* and *test* split. We focus on the average precision for moderate difficulty and two classes: car and cyclist. We calculate the average precision on *val* split with 40 recall positions using rotated IoU threshold of 0.7 for car class and 0.5 for cyclist class. The performance on *test* split is calculated using the official KITTI test server.

Val split Table 1 shows the results for car class on KITTI *val* split. We compare the performance of four state-of-the-art 3D object detectors modified with our proposed self-attention modules. For all four models, both DSA and FSA variants were able to achieve better or similar performance with fewer parameters and FLOPs.

Test split On KITTI *test* split, we evaluate our best performing models—PV-RCNN with DSA and FSA—and compare with the state-of-the-art models on KITTI benchmark. The results are shown in Table 2. Both DSA and FSA are able to achieve state-of-the-art performance on car class, only behind the best performing models by a small margin in a few cases. Our models perform well on moderate and hard difficulty cyclist class and achieve significantly better performance than all other methods in this comparison in terms of both 3D and BEV metrics.

5.2. nuScenes

To test the performance of our methods in more challenging scenarios, we evaluate PointPillars with DSA modules on the nuScenes benchmark using the official test server. In addition to average precision (AP) for each class, nuScenes benchmark introduces a new metric called nuScenes Detection Score (NDS). It is defined as a weighted sum between mean average precision (mAP), mean average errors of location (mATE), size (mASE), orientation (mAOE), attribute (mAAE) and velocity (mAVE).

Comparison with baseline We first compare our modified PointPillars + DSA model with vanilla PointPillars [18] and PointPillars+ [42], a class-balanced re-sampled version of PointPillars inspired by [60]. We report class AP and improvements in Table 3. For 6 out of 8 classes, our model is

Model	Car - 3D			Car - BEV			Cyclist - 3D			Cyclist - BEV		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
RGB + LiDAR												
MV3D [6]	74.97	63.63	54.00	86.62	78.93	69.80	-	-	-	-	-	-
ContFuse [20]	83.68	68.78	61.67	94.07	85.35	75.88	-	-	-	-	-	-
AVOD-FPN [17]	83.07	71.76	65.73	90.99	84.82	79.62	63.76	50.55	44.93	69.39	57.12	51.09
F-PointNet [28]	82.19	69.79	60.59	91.17	84.67	74.77	72.27	56.12	49.01	77.26	61.37	53.78
LiDAR only												
SECOND [48]	83.34	72.55	65.82	89.39	83.77	78.59	71.33	52.08	45.83	76.50	56.05	49.45
PointPillars [18]	82.58	74.31	68.99	90.07	86.56	82.81	77.10	58.65	51.92	79.90	62.73	55.58
PointRCNN [35]	86.96	75.64	70.70	92.13	87.39	82.72	74.96	58.82	52.53	82.56	67.24	60.28
STD [53]	87.95	79.71	75.09	94.74	89.19	**86.42**	78.69	61.59	55.30	81.36	67.23	59.35
PV-RCNN [34]	**90.25**	81.43	76.82	94.98	90.65	86.14	78.60	63.71	57.65	82.49	68.89	62.41
TANet [21]	83.81	75.38	67.66	-	-	-	73.84	59.86	53.46	-	-	-
Point-GNN [37]	88.33	79.47	72.29	93.11	89.17	83.90	78.60	63.48	57.08	81.17	67.28	59.67
SA-SSD [13]	88.75	79.79	74.16	**95.03**	**91.03**	85.96	-	-	-	-	-	-
3DSSD [52]	88.36	79.57	74.55	92.66	89.02	85.86	**82.48**	64.10	56.90	**85.04**	67.62	61.14
Ours												
PV-RCNN + DSA	88.25	**81.46**	**76.96**	92.42	90.13	85.93	82.19	**68.54**	**61.33**	83.93	**72.61**	**65.82**
PV-RCNN + FSA	88.01	81.31	76.75	92.30	89.87	85.71	80.68	65.20	59.14	81.86	69.67	63.32
Improvement	-	**+0.03**	**+0.14**	-	-	-	-	**+4.44**	**+3.68**	-	**+3.72**	**+3.41**

Table 2: Performance comparison on KITTI *test* split with AP calculated calculated with 40 recall positions

Model	Car	Truck	Bus	Trailer	Pedestrian	Moto	Tr. Cone	Barrier
PointPillars [18]	68.4	23.0	28.2	23.4	59.7	27.4	30.8	38.9
PointPillars+ [42]	76.0	31.0	32.1	36.6	64.0	**34.2**	45.6	**56.4**
PointPillars + DSA (ours)	**81.2**	**43.9**	**54.6**	**48.5**	**70.8**	27.8	**52.6**	49.8
Improvement	**+5.2**	**+12.9**	**+22.5**	**+11.9**	**+6.8**	-	**+7.0**	-

Table 3: Comparison with baseline models on nuScenes benchmark

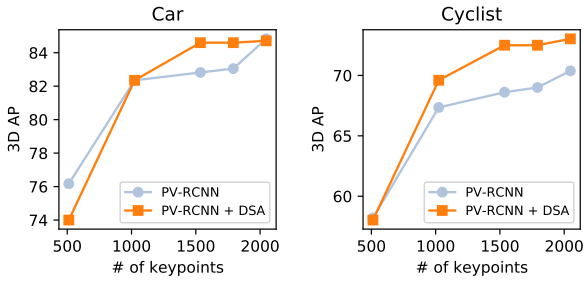


Figure 5: 3D AP of PV-RCNN (orange) vs. PV-RCNN-DSA (lightsteelblue) for a number of keypoints, varying from 512 to 2048 on KITTI *val*

able to achieve significantly higher AP than both PointPillars and PointPillars+. For *truck* and *trailer*, the improvement is over 10%, and for *bus*, over 20%.

Comparison with related works We add two related recent works to the comparison: 3DSSD [52] and PMP-Net [54]. Similar to the spirit of our DSA module, 3DSSD proposes a new sampling layer to retain enough representative points for accurate classification and regression. PMP-Net uses full self-attention in a local neighbourhood on pillar features for spatial and temporal context aggregation.

The results are shown in Table 4. In this comparison, we report the NDS score, mAP and inference errors. Overall, our model has the highest NDS score. The localization error (mATE) and orientation error (mAOE) are both lower than all other models in this comparison.

5.3. Ablation

Ablation studies are conducted on the KITTI validation split [6] for moderate Car class using AP@R40.

Model components To study the effect of FSA and DSA, we first construct a model for PointPillars and SECOND by removing 2D/3D convolutional filters to obtain a model with similar number of parameters and FLOPs as our DSA and FSA models. For DSA, we also vary the number of keypoints sampled for computation of global context. The results are shown in Table 5. We note that both FSA and DSA

Model	NDS ↑	mAP ↑	mATE ↓	mASE ↓	mAOE ↓	mAVE ↓	AAE ↓
PointPillars [18]	0.45	0.31	0.54	0.29	0.45	0.29	0.41
PointPillars+ [42]	0.55	0.40	0.39	0.27	0.48	0.27	**0.10**
3DSSD [52]	0.56	0.43	0.39	0.29	0.44	**0.22**	0.12
PMP-Net [54]	0.53	**0.45**	0.38	**0.25**	0.68	0.28	0.37
PointPillars + DSA (ours)	**0.58**	**0.45**	**0.32**	**0.25**	**0.41**	0.31	0.13

Table 4: Comparison with related recent works on nuScenes benchmark

Model	Params	FLOPs	Keypts	3D	BEV
PointPillars [18]	4.8M	63G	-	78.39	88.06
PointPillars	1.5M	32G	-	78.07	88.31
PP + DSA	1.1M	32G	1024	78.95	88.38
PP + DSA	1.1M	32G	2048	78.94	88.39
PP + DSA	1.1M	32G	4096	78.90	88.38
PP + FSA	1.0M	32G	-	**79.04**	**88.47**
SECOND [48]	4.6M	77G	-	81.61	88.55
SECOND	2.5M	51G	-	81.11	89.75
SEC + DSA	2.2M	52G	1024	82.03	89.79
SEC + DSA	2.2M	53G	2048	**82.03**	89.82
SEC + DSA	2.2M	53G	4096	82.00	89.79
SEC + FSA	2.2M	52G	-	81.86	**90.01**

Table 5: Ablation on model components

Model	Heads	Depth	3D	BEV
PointPillars [18]	-	-	78.07	88.11
PointPillars + FSA	2	2	78.67	88.19
PointPillars + FSA	4	1	78.34	88.00
PointPillars + FSA	4	2	**79.04**	**88.47**
PointPillars + FSA	4	4	78.56	88.01
SECOND [48]	-	-	81.11	88.75
SECOND + FSA	2	2	81.43	90.28
SECOND + FSA	4	2	**81.86**	**90.01**

Table 6: Ablation on hyper-parameters

outperform not only the models with similar parameters (3D AP improvement for PP: 0.97%, 0.87%; SECOND: 0.75%, 0.92%), but also the state-of-the-art implementations (PP: 0.65%, 0.55%; SECOND: 0.25%, 0.42%). We also note that for the car class, parameter efficiency can be improved by almost 70% for PointPillars and 30% for SECOND. For DSA, the performance is relatively robust to the number of sampled points, an observation also supported by Figure 5.

Hyper-parameters To study the effect of the number of self-attention layers and the number of attention heads, we conduct experiments with our constructed version of PointPillars and SECOND in Table 6. We note that increasing heads from 2 to 4 leads to an improvement of 0.37% for PointPillars and 0.43% for SECOND. Since increasing number of self-attention layers beyond a certain value can lead to over-smoothing [33], we use 2 FSA/DSA layers in the backbone and 4 heads for multi-head attention.

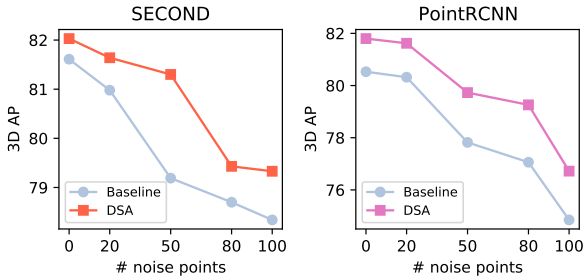


Figure 6: 3D AP of SECOND-DSA (orange) and Point-RCNN-DSA (violet) vs. SECOND and Point-RCNN baseline (light-steel-blue) for noise-points per ground-truth bounding box, varying from 0 to 100 on KITTI *val*

Robustness We introduce noise points to each object similar to TANet [21], to probe the robustness of representations learned. As shown in Figure 6, self-attention-augmented models are more robust to noise than the baseline. For example, with 100 noise points added, the performance of SECOND and Point-RCNN drops by 3.3% and 5.7% respectively as compared to SECOND-DSA and Point-RCNN-DSA which suffer a lower drop of 2.7% and 5.1% respectively.

6. Conclusions

In this paper, we propose a simple and flexible self-attention based framework to augment convolutional features with global contextual information for 3D object detection. Our proposed modules are generic, parameter and compute-efficient, and can be integrated into a range of 3D detectors. Our work explores two forms of self-attention: full (FSA) and deformable (DSA). The FSA module encodes pairwise relationships between all 3D entities, whereas the DSA operates on a representative subset to provide a scalable alternative for global context modeling. Quantitative and qualitative experiments demonstrate that our architecture systematically improves the performance of 3D object detectors.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization, 2016. 4

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2014. 3, 4

[3] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *ICCV*, 2019. 3

[4] Jorge Beltrán, Carlos Guindel, Francisco Miguel Moreno, Daniel Cruzado, Fernando Garcia, and Arturo De La Escalera. BirdNet: a 3d object detection framework from LiDAR information. In *ITSC*. IEEE, 2018. 2, 3

[5] 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 multimodal dataset for autonomous driving. In *CVPR*, 2020. 2, 3

[6] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017. 3, 6, 7

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014. 3

[8] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 3, 5

[9] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 2, 3, 6, 11

[10] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. *ICML*, 2017. 4

[11] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh R-CNN. In *ICCV*, 2019. 5

[12] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, 2018. 3

[13] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *CVPR*, 2020. 2, 3, 7

[14] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Andrea Vedaldi. Gather-Excite: Exploiting feature context in convolutional neural networks. In *NeurIPS*, 2018. 3

[15] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 3

[16] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *CVPR*, 2018. 2

[17] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *IROS*. IEEE, 2018. 3, 7

[18] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019. 1, 2, 3, 6, 7, 8, 11, 12, 14, 15, 16

[19] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set Transformer: A framework for attention-based permutation-invariant neural networks. In *ICML*. PMLR, 2019. 5

[20] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, 2018. 3, 7

[21] Zhe Liu, Xin Zhao, Tengteng Huang, Ruolan Hu, Yu Zhou, and Xiang Bai. TANet: Robust 3d object detection from point clouds with triple attention. *AAAI*, 2020. 3, 4, 7, 8

[22] Haihua Lu, Xuesong Chen, Guiying Zhang, Qiuhao Zhou, Yanbo Ma, and Yong Zhao. SCANet: Spatial-channel attention network for 3d object detection. In *ICASSP*. IEEE, 2019. 4

[23] Jiageng Mao, Xiaogang Wang, and Hongsheng Li. Interpolated convolutional networks for 3d point cloud understanding. In *ICCV*, 2019. 2

[24] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. Attention U-Net: Learning where to look for the pancreas, 2018. 3

[25] Anshul Paigwar, Ozgur Erkent, Christian Wolf, and Christian Laugier. Attentional pointnet for 3d-object detection in point clouds. In *CVPR Workshops*, 2019. 3

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 11

[27] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep Hough voting for 3d object detection in point clouds. In *ICCV*, 2019. 2

[28] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum PointNets for 3d object detection from RGB-D data. In *CVPR*, 2018. 2, 3, 7

[29] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 3

[30] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017. 2, 5, 15

[31] Can Qin, Haoxuan You, Lichen Wang, C-C Jay Kuo, and Yun Fu. PointDAN: A multi-scale 3d domain adaption network for point cloud representation. In *NeurIPS*, 2019. 5

[32] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, volume 32, 2019. 3

[33] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. DropEdge: Towards deep graph convolutional networks on node classification. In *ICLR*, 2019. 8

[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 *CVPR*, 2020. 1, 2, 3, 6, 7, 11, 14, 16

[35] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. PointRCNN: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 1, 2, 3, 6, 7, 11, 13, 14, 15, 16

[36] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *PAMI*, 2020. 2

[37] Weijing Shi and Raj Rajkumar. Point-GNN: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1711–1719, 2020. 7

[38] Leslie N Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay, 2018. 6

[39] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. https://github.com/open-mmlab/OpenPCDet, 2020. 11

[40] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. KPConv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019. 2

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 2, 3, 4, 11

[42] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *CVPR*, 2020. 3, 6, 7, 8

[43] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 3

[44] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: Convolutional block attention module. In *ECCV*, 2018. 3

[45] Wenxuan Wu, Zhongang Qi, and Li Fuxin. PointConv: Deep convolutional networks on 3d point clouds. In *CVPR*, 2019. 2

[46] Xin Wu, Yi Cai, Qing Li, Jingyun Xu, and Ho-fung Leung. Combining contextual information by self-attention mechanism in convolutional neural networks for text classification. In *WISE*. Springer, 2018. 3

[47] Qian Xie, Yu-Kun Lai, Jing Wu, Zhoutao Wang, Yiming Zhang, Kai Xu, and Jun Wang. MLCVNet: Multi-level context votenet for 3d object detection. In *CVPR*, 2020. 2, 4

[48] Yan Yan, Yuxing Mao, and Bo Li. SECOND: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 1, 2, 3, 6, 7, 8, 11, 12, 14, 15, 16, 17

[49] Bin Yang, Ming Liang, and Raquel Urtasun. HDNet: Exploiting HD maps for 3d object detection. In *Conference on Robot Learning*, 2018. 2, 3

[50] Bin Yang, Wenjie Luo, and Raquel Urtasun. PIXOR: Real-time 3d object detection from point clouds. In *CVPR*, 2018. 2, 3

[51] Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, and Qi Tian. Modeling point clouds with self-attention and gumbel subset sampling. In *CVPR*, 2019. 3

[52] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3DSSD: Point-based 3d single stage object detector. In *CVPR*, 2020. 2, 3, 7, 8

[53] Junbo Yin, Jianbing Shen, Chenye Guan, Dingfu Zhou, and Ruigang Yang. LiDAR-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention. In *CVPR*, 2020. 2, 3, 7

[54] Junbo Yin, Jianbing Shen, Chenye Guan, Dingfu Zhou, and Ruigang Yang. LiDAR-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention. In *CVPR*, 2020. 3, 7, 8

[55] Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, et al. Deep reinforcement learning with relational inductive biases. In *ICLR*, 2018. 3

[56] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *ICML*. PMLR, 2019. 2

[57] Wenxiao Zhang and Chunxia Xiao. PCAN: 3d attention map learning using contextual information for point cloud based retrieval. In *CVPR*, 2019. 3

[58] Zhizheng Zhang, Cuiling Lan, Wenjun Zeng, Xin Jin, and Zhibo Chen. Relation-aware global attention for person re-identification. In *CVPR*, 2020. 3

[59] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018. 2, 3

[60] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection, 2019. 3, 6

Supplementary Material: Self-Attention Based Context-Aware 3D Object Detection

In this document, we provide technical details and additional experimental results to supplement our main submission. We first discuss the implementation and training details used for our experiments. Code will be released upon acceptance. We then showcase the flexibility and robustness of our models through extended results on the the KITTI [9] dataset. We further qualitatively show the superiority of our proposed module-augmented implementations over the baseline across different challenging scenes. We also visualize the attention maps, where we observe the emergence of semantically meaningful behavior that captures the relevance of context in object detection. We finally briefly review existing standard feature extractors for 3D object detection to support our design.

7. Network Architectures and Training Details

In this section, we provide a more detailed description of the architectures used in our experiments.

The general pipeline for the proposed FSA/DSA PV-RCNN variants is illustrated in Figure 1. For both variants, the baseline PV-RCNN architecture is augmented with two instances of the FSA module—one for the region proposals and another one for the final predictions. Further, the DSA variant additionally includes deformable convolutions in its sparse 3D convolutional feature extractor.

The detailed specification of the various layers in our FSA and DSA augmented baselines—PointPillars [18], SECOND [48], Point-RCNN [35] and PV-RCNN [34]—is documented in Table 2, Table 3, Table 4, and Table 5, respectively. We also provide the details of a reduced parameter baseline that aims to compare the performance of the model with similar number of parameters and FLOPs compared to their FSA and DSA counterparts.

For the KITTI dataset, the detection range is within [0,70.4] m, [-40,40] m and [-3,1] m for the XYZ axes, with XY pillar resolution [0.16, 0.16] m and XYZ voxel-resolution of [0.05, 0.05, 0.1] m. For nuScenes, the range is [-50,50] m, [-50,50] m, [-5,3] m along the XYZ axes. The XY pillar resolution is [0.2, 0.2] m. Additionally for nuScenes, the deformation radius is set to 2 m, and the feature interpolation radius is set to 1m with 16 samples. The self-attention feature dimension is 64 across all models. We apply 2 FSA/DSA modules with 4 attention heads across our chosen baselines. For DSA, we use a subset of 2048 sampled points for KITTI and 6144 sampled points for nuScenes.

Backbone	Batch Size	Start LR	Max LR
PointPillars	16	0.0003	0.003
SECOND			
Point-RCNN	8	0.001	0.01
PV-RCNN			

Table 1: Batch size and learning rate configurations for each backbone model on KITTI benchmark

Additional details on encoding, training, and inference parameters are as follows. For pillar and voxel-based detection, we use absolute-position encoding for the full self-attention blocks [41]. For the test submissions, we retain the parameterization of the original baselines as mentioned in our main paper. We use Pytorch [26] and the recently released PC-Det [39] repository for our experiments. Our models are trained from scratch in an end-to-end manner with the ADAM optimizer. The learning rates used for the different models are given in Table 1. For the proposal refinement stage in two-stage networks [35], [34], we randomly sample 128 proposals with 1:1 ratio for positive and negative proposals. A proposal is considered positive if it has at-least 0.55 3D IoU with the ground-truth boxes, otherwise it is considered to be negative. For inference, we keep the top-500 proposals generated from single stage approaches [18], [48] and the top-100 proposals generated from two stage approaches [35], [34] with a 3D IoU threshold of 0.7 for non-maximum-suppression (NMS). An NMS classification threshold of 0.1 is used to remove weak detections.

8. Detailed Results

We provide additional experimental details on the validation split of the KITTI [9] data-set in this section. In Table 6, we first show the 3D and BEV AP for moderate difficulty on the *Cyclist* and *Pedestrian* class for PV-RCNN and its variants. The table shows that both our proposed modules improve on the baseline results. This showcases the robustness of our approach in also naturally benefiting smaller and more complicated objects like cyclists and pedestrians. We then proceed to list the 3D AP and BEV performances with respect to distance from the ego-vehicle in **??**. We find that the proposed blocks especially improve upon detection at further distances, where points become sparse and context becomes increasingly important. These results hold especially for the *cyclist* class—as opposed to the *car* class, which shows that context is possibly more important for smaller objects with reduced number of points available for detection. In Table 8, we provide results for all three difficulty categories for the *car* class. We see consistent improvements across backbones with various input modalities on the *hard* category. This is consistent with our premise that samples in the hard category can benefit more context information of surrounding instances. We also note that PointPillars [18], which loses a lot of information due to pillar-based discretization of points, can supplement this

Attribute	PP [18]	PP$_{red}$	FSA-PP	DSA-PP
Layer: 2D CNN Backbone				
Layer-nums	[3, 5, 5]	[3, 5, 5]	[3, 5, 5]	[3, 5, 5]
Layer-stride	[2, 2, 2]	[2, 2, 2]	[2, 2, 2]	[2, 2, 2]
Num-filters	[64, 128, 256]	[64, 64, 128]	[64, 64, 64]	[64, 64, 64]
Upsample-stride	[1, 2, 4]	[1, 2, 4]	[1, 2, 4]	[1, 2, 4]
Num-upsample-filters	[128, 128, 128]	[128, 128, 128]	[128, 128, 128]	[128, 128, 128]
Layer: Self-Attention				
Stage Added	-	-	Pillar feature	Pillar feature
Num layers	-	-	2	2
Num heads	-	-	4	4
Context Linear Dim	-	-	64	64
Num Keypoints	-	-	-	2048
Deform radius	-	-	-	3.0m
Feature pool radius	-	-	-	2.0m
Interpolation MLP Dim	-	-	-	64
Interpolation radius	-	-	-	1.6m
Interpolation samples	-	-	-	16

Table 2: Architectural details of PointPillars [18], our reduced parameter PointPillars version, proposed FSA-PointPillars and DSA-PointPillars

Attribute	SECOND [48]	SECOND$_{red}$	FSA-SECOND	DSA-SECOND
Layer: 3D CNN Backbone				
Layer-nums in Sparse Blocks	[1, 3, 3, 3]	[1, 3, 3, 2]	[1, 3, 3, 2]	[1, 3, 3, 2]
Sparse tensor size	128	64	64	64
Layer: 2D CNN Backbone				
Layer-nums	[5, 5]	[5, 5]	[5, 5]	[5, 5]
Layer-stride	[1, 2]	[1, 2]	[1, 2]	[1, 2]
Num-filters	[128, 256]	[128, 160]	[128, 128]	[128, 128]
Upsample-stride	[1, 2]	[1, 2]	[1, 2]	[1, 2]
Num-upsample-filters	[256, 256]	[256, 256]	[256, 256]	[256, 256]
Layer: Self-Attention				
Stage Added	-	-	Sparse Tensor	Sparse Tensor
Num layers	-	-	2	2
Num heads	-	-	4	4
Context Linear Dim	-	-	64	64
Num Keypoints	-	-	-	2048
Deform radius	-	-	-	4.0m
Feature pool radius	-	-	-	4.0m
Interpolation MLP Dim	-	-	-	64
Interpolation radius	-	-	-	1.6m
Interpolation samples	-	-	-	16

Table 3: Architectural details of SECOND [48], our reduced parameter SECOND version, and proposed FSA-SECOND and DSA-SECOND

Attribute	Point-RCNN [35]	Point-RCNN$_{red}$	FSA-Point-RCNN	DSA-Point-RCNN
Layer: Multi-Scale Aggregation				
N-Points	[4096, 1024, 256, 64]	[4096, 1024, 256, 64]	[4096, 1024, 256, 64]	[4096, 1024, 128, 64]
Radius	[0.1, 0.5], [0.5, 1.0], [1.0, 2.0], [2.0, 4.0]	[0.1, 0.5], [0.5, 1.0], [1.0, 2.0], [2.0, 4.0]	[0.1, 0.5], [0.5, 1.0], [1.0, 2.0], [2.0, 4.0]	[0.1, 0.5], [0.5, 1.0], [1.0, 2.0], [2.0, 4.0]
N-samples	[16, 32]	[16, 32]	[16, 32]	[16, 32]
MLPs	[16, 16, 32], [32, 32, 64], [64, 64, 128], [64, 96, 128] [128, 196, 256], [128, 196, 256] [256, 256, 512], [256, 384, 512]	[16, 32], [32, 64], [64, 128], [64, 128] [128, 256], [128, 256] [256, 512], [256, 512]	[16, 32], [32, 64], [64, 128], [64, 128] [128, 256], [128, 256] [256, 512], [256, 512]	[16, 32], [32, 64], [64, 128], [64, 128] [128, 256], [128, 256] [256, 512], [256, 512]
FP-MLPs	[128, 128], [256, 256], [512, 512], [512, 512]	[128, 128], [128, 128], [128, 128], [128, 512]	[128, 128], [128, 128], [128, 128], [128, 128]	[128, 128], [128, 128], [128, 128], [128, 128]
Layer: Self-Attention				
Stage Added	-	-	MSG-3 and MSG-4	MSG-3 and MSG-4
Num layers	-	-	2	2
Num heads	-	-	4	4
Context Linear Dim	-	-	64	64
Num Keypoints	-	-	-	(128, 64)
Deform radius	-	-	-	(2.0, 4.0)m
Feature pool radius	-	-	-	(1.0, 2.0)m
Interpolation MLP Dim	-	-	-	(64, 64)
Interpolation radius	-	-	-	(1.0, 2.0)m
Interpolation samples	-	-	-	(16, 16)

Table 4: Architectural details of Point-RCNN [35], our reduced parameter Point-RCNN version, proposed FSA-Point-RCNN and DSA-Point-RCNN

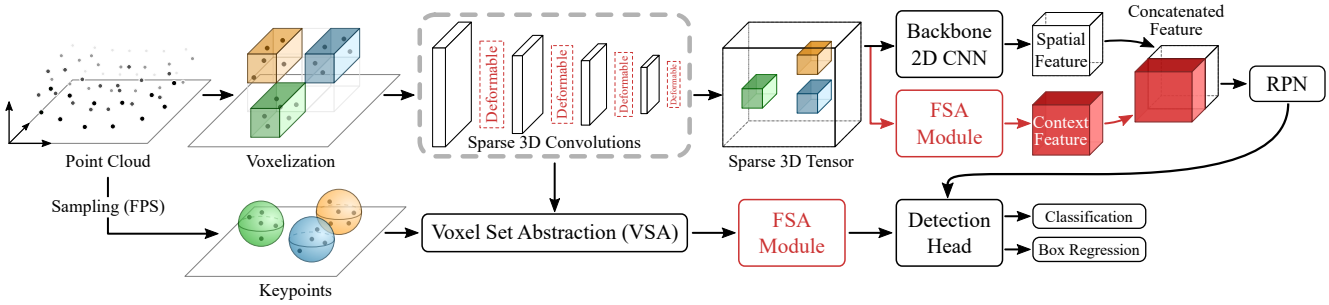


Figure 1: Architecture of the proposed FSA/DSA-PVRCNN.

Attribute	PV-RCNN [34]	FSA-PVRCNN	DSA-PVRCNN
Layer: 3D CNN Backbone			
Layer-nums in Sparse Blocks	[1, 3, 3, 3]	[1, 3, 3, 2]	[1, 3, 3, 3]
Sparse tensor size	128	64	128
Layer: 2D CNN Backbone			
Layer-nums	[5, 5]	[5, 5]	[5, 5]
Layer-stride	[1, 2]	[1, 2]	[1, 2]
Num-filters	[128, 256]	[128, 128]	[128, 256]
Upsample-stride	[1, 2]	[1, 2]	[1, 2]
Num-upsample-filters	[256, 256]	[256, 256]	[256, 256]
Layer: Self-Attention			
Stage Added	-	Sparse Tensor and VSA	VSA
Num layers	-	2	2
Num heads	-	4	4
Context Linear Dim	-	128	128
Num Keypoints	-	-	2048
Deform radius	-	-	[0.4, 0.8], [0.8, 1.2], [1.2, 2.4], [2.4, 4.8]
Feature pool radius	-	-	Multi-scale: (0.8, 1.6)m
Interpolation MLP Dim	-	-	Multi-scale: (64, 64)
Interpolation radius	-	-	Multi-scale: (0.8, 1.6)m
Interpolation samples	-	-	Multi-scale: (16, 16)

Table 5: Architectural details of PV-RCNN [34], and proposed FSA-PVRCNN and DSA-PVRCNN

	3D	BEV
PV-RCNN [34]	70.38	74.5
PV-RCNN + DSA	**73.03**	**75.45**
PV-RCNN + FSA	71.46	74.73

Table 6: Performance comparison for moderate difficulty cyclist class on KITTI *val* split.

Table 7: Comparison of nearby and distant-object detection on the moderate level of KITTI *val* split with AP calculated by 40 recall positions

Distance	Model	Car	Cyclist
0-30m	PV-RCNN [34]	91.71	73.76
	DSA	91.65	**74.89**
	FSA	**93.44**	74.10
30-50m	PV-RCNN [34]	50.00	35.15
	DSA	52.02	**47.00**
	FSA	**52.76**	39.74

loss with fine-grained context information.

9. Qualitative Results

9.1. Comparison with Baseline

In this section, we provide additional qualitative results across challenging scenarios from real-world driving scenes and compare them with the baseline performance (see Figure 2). The ground-truth bounding boxes are shown in *red*, whereas the detector outputs are shown in *green*. We show consistent improvement in identifying missed detections across scenes and with different backbones including PointPillars [18], SECOND [48], Point-RCNN [35] and PV-RCNN [34]. We note that we can better refine proposal bounding box orientations with our context-aggregating FSA module (Rows 1, 2, and 4). We also note that cars at distant locations can be detected by our approach (Rows 3, 4 and 6). Finally we analyze that cars with slightly irregular shapes even at nearer distances are missed by the baseline but picked up by our approach (Rows 7 and 8).

9.2. Visualization of Attention Weights

We also visualize the attention weights for FSA-variant for the SECOND [48] backbone in Figure 3. In this implementation, voxel features down-sampled by 8-times from the point-cloud space are used to aggregate context information through pairwise self-attention. We first visualize the voxel space, where the center point of each voxel

Model	Modality	Params (M)	GFLOPs	Car 3D AP		
				Easy	Moderate	Hard
PP [18]	BEV	4.8	63.4	87.75	78.39	75.18
PP_{red}	BEV	1.5	**31.5**	88.09	78.07	75.14
PP-DSA	BEV	1.1	32.4	89.37	78.94	75.99
PP-FSA	BEV	**1.0**	31.7	**90.10**	**79.04**	**76.02**
SECOND [48]	Voxel	4.6	76.7	90.55	81.61	78.61
$SECOND_{red}$	Voxel	2.5	**51.2**	89.93	81.11	78.30
SECOND-DSA	Voxel	2.2	52.6	**90.70**	**82.03**	**79.07**
SECOND-FSA	Voxel	**2.2**	51.9	89.05	81.86	78.84
Point-RCNN [35]	Points	4.0	27.4	**91.94**	80.52	78.31
$Point\text{-}RCNN_{red}$	Points	2.2	24.1	91.47	80.40	78.07
Point-RCNN-DSA	Points	**2.3**	**19.3**	91.55	81.80	79.74
Point-RCNN-FSA	Points	2.5	19.8	91.63	**82.10**	**80.05**

Table 8: Detailed comparison of 3D AP with baseline on KITTI *val* split with 40 recall positions

is represented as a yellow point against the black scene-background. We next choose the center of a ground-truth bounding box as a reference point. We refer this bounding box as the reference bounding box. The reference bounding box is shown in *yellow*, and the rest of the labeled objects in the scene are shown in *orange*. We next visualize the attention weights across all the voxel centers with respect to the chosen reference bounding box center. Of the 4 attention maps produced by the 4 FSA-heads, we display the attention map with the largest activation in our figures. We find that attention weights become concentrated in small areas of the voxel-space. These voxel centers are called attended locations and are represented by a thick cross in our visualizations. The color of the cross represents the attention weight at that location and the scale of attention weights is represented using a colorbar. The size of the cross is manipulated manually by a constant factor. In an effort to improve image-readability, we connect the chosen reference object to the other labelled objects in the scene that it pays attention to (with blue boxes and blue arrows) as inferred from the corresponding attended locations while aggregating context information.

In our paper, we speculate that sometimes for true-positive cases, CNNs (which are essentially a pattern matching mechanism) detect a part of the object but are not very confident about it. This confidence can be increased by looking at nearby voxels and inferring that the context-aggregated features resemble a composition of parts. We therefore first ask the question if our FSA module can adaptively focus on its own local neighbourhood. We show in Rows 1 and 2 of Figure 3 that it can aggregate local context adaptively. We also hypothesize that, for distant cars, information from cars in similar lanes can help refine orientation. We therefore proceed to show instances where a reference bounding box can focus on cars in similar lanes, in Rows 3 and 4 of Figure 3. We also show cases where FSA can adaptively focus on objects that are relevant to build structural information about the scene in Rows 5 and 6 of Figure 3. Our visualizations thus indicate that semantically meaningful patterns emerge through the self-attention based context-aggregation module.

10. Standard Feature Extractors for 3D Object Detection

In this section, we briefly review the standard feature extractors for 3D object detection to motivate our design. 2D and 3D convolutions have achieved great success in processing pillars [18] and voxel grids [48] for 3D object detection. Point-wise feature learning methods like PointNet++ [30] have also been successful in directly utilizing sparse, irregular points for 3D object detection [35].

Given a set of vectors $\{x_1, x_2, ...x_n\}$, which can represent pillars, voxels or points, with $x_i \in R^C$, one can define a function $f : \mathcal{X} \rightarrow R^{C'}$ that maps them to another vector. In this case, standard convolution at location $\hat{p}$ can be formulated as:

$$f(\hat{p}) = \sum_{l \in \Omega_1} x_{\hat{p}+l} w_l \tag{1}$$

where w is a series of C' dimensional weight vectors with kernel size $2m+1$ and $\Omega_1 = [l \in (-m, ...m)]$ representing the set of positions relative to the kernel center. Similarly, a point-feature approximator at $\hat{p}$ can be formulated as:

$$f(\hat{p}) = \max_{l \in \Omega_2} h(x_l) \tag{2}$$

where h is a C' dimensional fully connected layer, $\max$ denotes the max-pooling operator and Ω_2 denotes the k-nearest neighbors of $\hat{p}$. The operator f thus aggregates features with pre-trained weights, h and w, from nearby locations.

Figure 2: Qualitative comparisons of our proposed approach with the baseline on the KITTI validation set. *Red* represents Ground-Truth bounding box while *Green* represents detector outputs. From left to right: RGB images of scenes; Baseline performance across state-of-the-art detectors PointPillars [18], SECOND [48], Point-RCNN [35] and PV-RCNN [34]; Performance of proposed FSA module-augmented detectors. Viewed best when enlarged.

Limitations One of the disadvantages of this operator is that weights are fixed and cannot adapt to the content of the features or selectively focus on the salient parts. Moreover, since the number of parameters scales linearly with the size of the neighborhood to be processed, long range feature-dependencies can only be modeled by adding more layers, posing optimization challenges for the network. Since useful information for fine-grained object recognition and localization appears at both global and local levels of a point-cloud, our work looks for more effective feature aggregation mechanisms.

Figure 3: Visualization of attention maps produced by our proposed FSA-variant on SECOND [48] backbone. We analyze the implications of the produced attention maps in Section 3.2.