arXiv:2101.02672v3 [cs.CV] 17 Mar 2021

SA-Det3D: Self-Attention Based Context-Aware 3D Object Detection

Prarthana Bhattacharyya, Chengjie Huang and Krzysztof Czarnecki

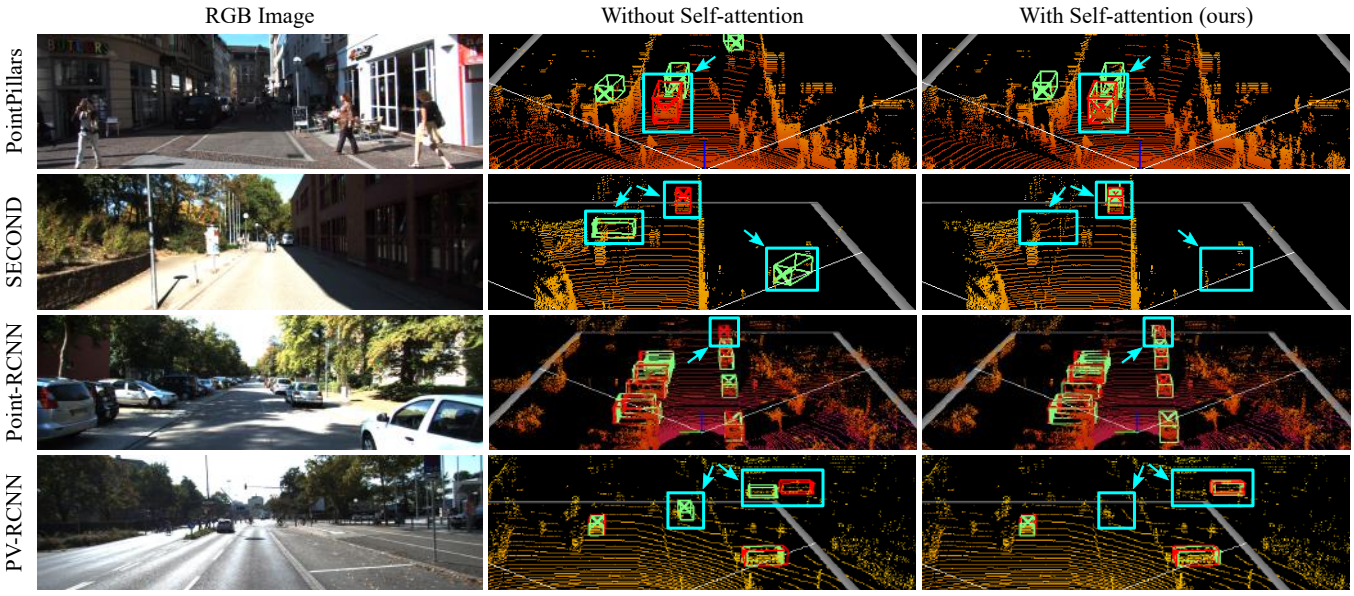


Fig. 1: Performance illustrations on KITTI *val*. Red bounding box is ground truth, green is detector outputs. From left to right: (a) RGB images (b) Result of state-of-the-art methods: PointPillars [8], SECOND [9], Point-RCNN [10] and PV-RCNN [11]. (c) Result of our full self-attention (FSA) augmented baselines. Our method identifies missed detections and removes false positives.

***Abstract*— 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, non-local neural networks and self-attention for 2D vision have shown that explicitly modeling long-range interactions can lead to more robust and competitive models. In this paper, we propose 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 of up to 1.5 3D AP while simultaneously reducing their parameter footprint and computational cost by 15-80% and 30-50%, respectively, on the KITTI validation set. We next 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 show our proposed method improves 3D object detection performance on KITTI, nuScenes and Waymo Open datasets. Code is available at https://github.com/AutoVision-cloud/SA-Det3D.**

[1]Prarthana Bhattacharyya, Chengjie Huang and Krzysztof Czarnecki are with the Faculty of Engineering, University of Waterloo, 200 University Avenue, Waterloo, ON, Canada. Email: (p6bhatta@uwaterloo.ca, c.huang@uwaterloo.ca, k2czarne@uwaterloo.ca)

I. Introduction

3D object detection has been receiving increasing attention in the computer vision and graphics community, driven by the ubiquity of LiDAR sensors and its widespread applications in autonomous driving and robotics. Point-cloud based 3D object detection has especially witnessed tremendous advancement in recent years [8], [9], [10], [11], [16], [17], [18], [19], [20], [56]. Grid-based methods first transform the irregular point-clouds to regular representations such as 2D bird's-eye view (BEV) maps or 3D voxels and process them using 2D/3D convolutional networks (CNNs). Point-based methods sample points from the raw point-cloud and query a local group around each sampled point to define convolution-like operations [7], [54], [55] for point-cloud feature extraction.

Both 2D/3D CNNs and point-wise convolutions process a local neighbourhood and aggregate global context by applying 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

Method	Task	Modality	Context	Scalability	Attention + Convolution Combination	Stage Added
HG-Net [21]	detection	points	global-static	-	gating	Attention modules are added at the end.
PCAN [22]	place-recognition	points	local-adaptive	-	gating	
Point-GNN [14]	detection	points	local-adaptive	-	-	Attention modules fully replace convolution and set-abstraction layers.
GAC [33]	segmentation	points	local-adaptive	-	-	
PAT [36]	classification	points	global-adaptive	randomly sample points subset	-	
ASCN [37]	segmentation	points	global-adaptive	randomly sample points subset	-	
Pointformer [38]	detection	points	global-adaptive	sample points subset and refine	-	
MLCVNet [35]	detection	points	global-static	-	residual addition	Attention modules are inserted into the backbone.
TANet [13]	detection	voxels	local-adaptive	-	gating	
PMPNet [12]	detection	pillars	local-adaptive	-	gated-recurrent-unit	
SCANet [34]	detection	BEV	global-static	-	gating	
A-PointNet [39]	detection	points	global-adaptive	attend sequentially to small regions	gating	
Ours (FSA/DSA)	detection	points, voxels, pillars, hybrid	global-adaptive	attend to salient regions using learned deformations	residual addition	Attention modules are inserted into the backbone.

TABLE I: Properties of recent attention-based models for point-clouds

to coordinate the optimization of parameters across multiple layers to capture patterns in the data [49].

In addition, point-cloud based 3D object detectors 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 [1] 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 to augment the standard convolutional features—Full Self-Attention (FSA) and Deformable Self-Attention (DSA). 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. Our experiments show that we can improve the performance of current 3D object detectors with our proposed FSA/DSA blocks while simultaneously promoting parameter and compute efficiency.

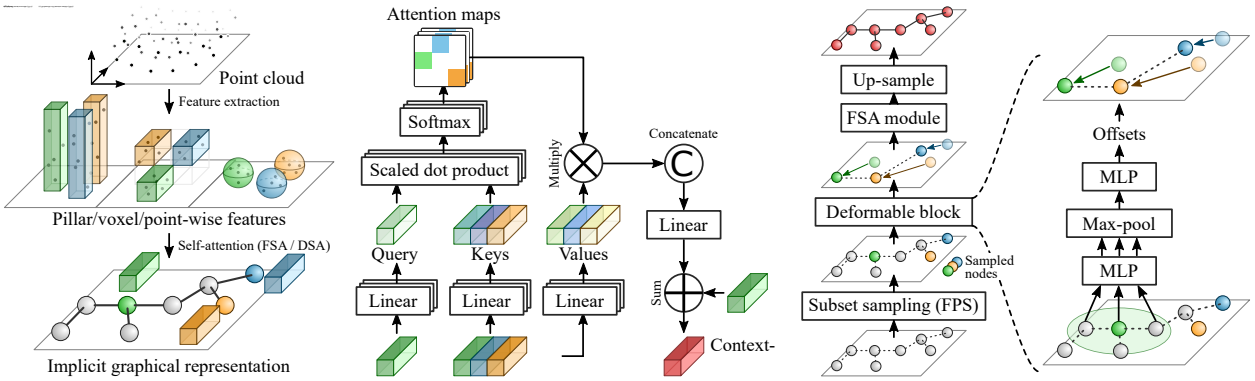
Contributions

- We propose the first generic globally-adaptive context aggregation module that can be applied across a range of modern architectures including BEV [8], voxel [9], point [10] and point-voxel [11] based 3D detectors. We show that we can outperform strong baseline implementations by up to 1.5 3D AP (average precision) while simultaneously reducing parameter and compute cost by 15-80% and 30-50%, respectively, 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-scale point-clouds like nuScenes and Waymo Open dataset.
- Extensive experiments demonstrate the benefits of our proposed FSA/DSA modules by consistently improving the performance of state-of-the-art detectors on KITTI [2], nuScenes [3] and Waymo Open dataset [4].

II. Related Works

a) 3D Object Detection: Current 3D object detectors include BEV, voxel, point or hybrid (point-voxel) methods. *BEV-based* methods like MV3D [5] fuse multi-view representations of the point-cloud and use 2D convolutions for 3D proposal generation. PointPillars [8] proposes a more efficient BEV representation and outperforms most fusion-based approaches while being 2-4 times faster. *Voxel-based* approaches, on the other hand, divide the point-cloud into 3D voxels and process them using 3D CNNs [20]. SECOND [9] introduces sparse 3D convolutions for efficient 3D processing of voxels, and CBGS [15] extends it with multiple heads. *Point-based* methods are inspired by the success of PointNet [6] and PointNet++ [7]. F-PointNet [16] first applied PointNet for 3D detection, extracting point-features from point-cloud crops that correspond to 2D camera-image detections. Point-RCNN [10] segments 3D point-clouds using PointNet++, and uses the segmentation features to better refine box proposals. *Point-Voxel-based* methods like STD [18], PV-RCNN [11] and SA-SSD [19] leverage both voxel and point-based abstractions to produce more accurate bounding boxes.

Relationship to current detectors: Instead of repeatedly stacking convolutions, we propose a simple, scalable, generic and permutation-invariant block called FSA/DSA to adaptively aggregate context information from the entire point-



(a) Self-attention induced graph

Fig. 2: Architectures of the proposed FSA and DSA modules.

cloud. This allows remote regions to directly communicate and can help in learning relationships across objects. This module is flexible and can be applied in parallel to convolutions within the backbone of modern point-cloud based detector architectures.

b) Attention for Context Modeling: Self-attention [1] has been instrumental to achieving state-of-the-art results in machine translation and combining self-attention with convolutions is a theme shared by recent work in natural language processing [57], image recognition [32], 2D object detection [59], activity recognition [31], person re-identification [58] and reinforcement learning [44].

Using self-attention to aggregate global structure in point-clouds for 3D object detection remains a relatively unexplored domain. PCAN [22], TANet [13], Point-GNN [14], GAC [33], PMPNet [12] use local context to learn context-aware discriminative features. However relevant contextual information can occur anywhere in the point-cloud and hence we need global context modeling. HGNet [21], SCANet [34], MLCVNet [35] use global scene semantics to improve performance of object detection, but the global context vector is shared across all locations and channels and does not adapt itself according to the input features leading to a sub-optimal representation. PAT [36], ASCN [37], Pointformer [38] build globally-adaptive point representations for classification, segmentation and 3D detection. But because they use the costly pairwise self-attention mechanism, the self-attention does not scale to the entire point-cloud. Consequently, they process a randomly selected subset of points, which may be sensitive to outliers. To process global context for 3D object detection and scale to large point-clouds, Attentional PointNet [39] uses GRUs [40] to sequentially attend to different parts of the point-cloud. Learning global context by optimizing the hidden state of a GRU is slow and inefficient, however.

In contrast, our method can processes context *adaptively* for each location from the entire point-cloud, while also *scaling* to large sets using learned deformations. Since the global context is fused with local-convolutional features, the training is stable and efficient as compared to GRUs or stand-alone attention networks [42]. Table I compares our work with recent point-cloud based attention methods.

III. 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. Next, inspired by 2D deformable convolutions [41] we introduce a variant of FSA called Deformable Self-Attention (DSA). DSA can reduce the quadratic computation time of FSA and scale to larger and denser point-clouds. The two proposed modules are illustrated in Figure 2.

A. Formulation

For the input set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_n\}$ of n correlated features and $i \in \{1, ...n\}$, we propose to use self-attention introduced by Vaswani et al. [1] 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.

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 2 (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 2 (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 for which many mechanisms like self-attention exist.

B. 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 2(b)). The

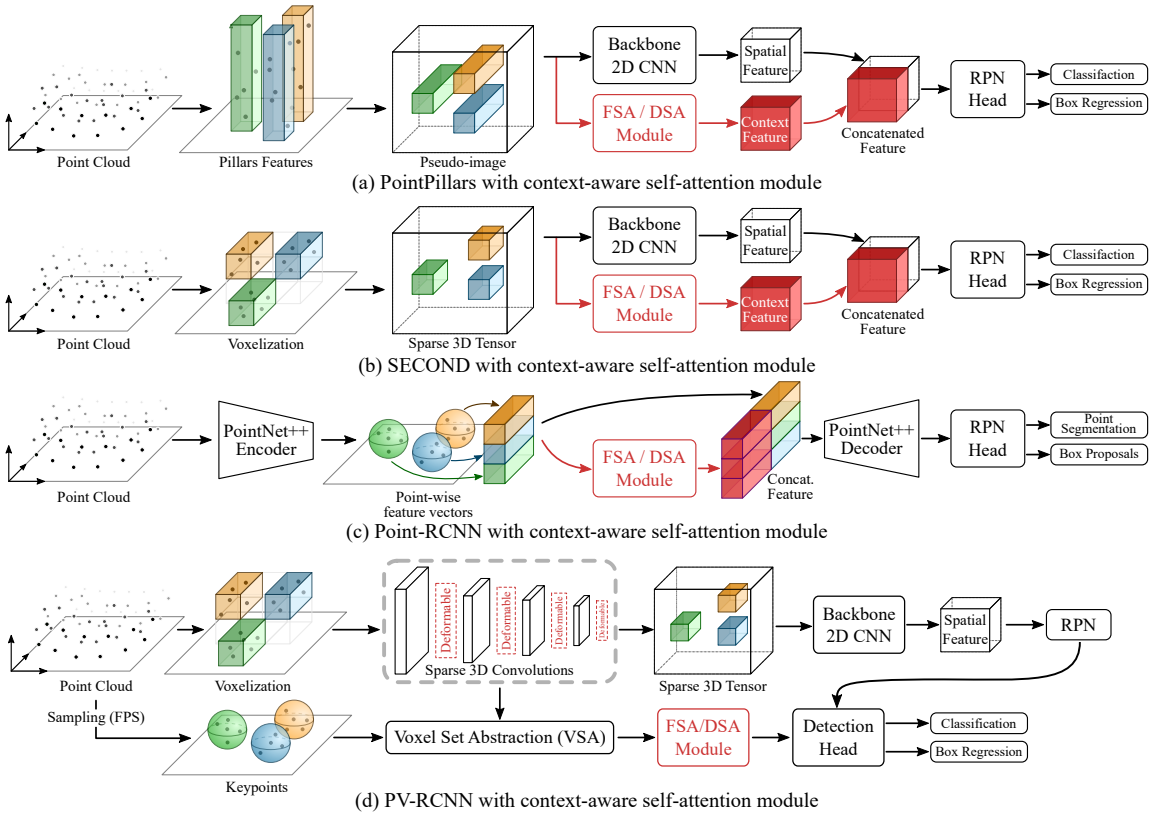


Fig. 3: Proposed FSA/DSA module augmented network architectures for different backbone networks.

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 group normalization [43] and summing it with $\mathbf{x}_i$ (residual connection).

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-invariant. 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 $\mathcal{O}(n^2d)$ 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 necessary to trade accuracy for computational efficiency 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.

C. 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. We describe the up-sampling process in the supplementary section. The complexity of this operation is $\mathcal{O}(m^2d)$, where $m << n$ 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 [41] 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 [45], [46]. 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:

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

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

Method	PointPillars [8]				SECOND [9]				Point-RCNN [10]				PV-RCNN [11]			
	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 II: Performance comparison for moderate difficulty Car class on KITTI *val* split with 40 recall positions

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:

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

Next, the m adaptively aggregated features $\{\mathbf{x}'_1 .... \mathbf{x}'_m\}$ are then passed into a full self-attention (FSA) module to model relationships between them. 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 2(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.

IV. Experiments

A. Network Architectures

We train and evaluate our proposed FSA and DSA modules on four state-of-the-art architecture backbones: PointPillars [8], SECOND [9], Point-RCNN [10], and PV-RCNN [11]. The architectures of the backbones are illustrated in Figure 3. The augmented backbones can be trained end-to-end without additional supervision.

For the KITTI dataset, the detection range is within [0,70.4] m, [-40,40] m and [-3,1] m for the XYZ axes, and we set the XY pillar resolution to (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 and the XY pillar resolution is (0.2, 0.2) m. For the Waymo Open dataset, the detection range is [-75.2, 75.2] m for the X and Y axes and [-2, 4] m for the Z-axis, and we set the voxel size to (0.1, 0.1, 0.15) m. Additionally, the deformation radius is set to 3 m, and the feature interpolation radius is set to 1.6 m 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 2,048 sampled points for KITTI and 4,096 sampled points for nuScenes and Waymo Open Dataset. We use standard data-augmentation for point clouds. For baseline models, we reuse the pre-trained checkpoints provided by OpenPCDet [48]. More architectural details are provided in the supplementary.

B. Implementation Details

KITTI: KITTI benchmark [2] is a widely used benchmark with 7,481 training samples and 7,518 testing samples. We follow the standard split [5] 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 [47] and one cycle learning rate schedule [53]. We also use the same batch size and learning rates as the baseline models.

nuScenes nuScenes [3] is a more recent large-scale benchmark for 3D object detection. In total, there are 28k, 6k, 6k, annotated frames for training, validation, and testing, respectively. The annotations include 10 classes with a long-tail distribution. 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 300 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 8 using Adam optimizer [47] and one cycle learning rate schedule [53].

Waymo Open Dataset Waymo Open Dataset [4] is currently the largest dataset for 3D detection for autonomous driving. There are 798 training sequences with 158,081 LiDAR samples, and 202 validation sequences with 39,987 LiDAR samples. The objects are annotated in the full 360°field of view. We train and evaluate a DSA model with SECOND as the backbone architecture. The model was trained on 4 NVIDIA Tesla V100 GPUs for 50 epochs with a batch size of 8 using Adam optimizer [47] and one cycle learning rate schedule [53].

V. Results

A. 3D Detection on the KITTI Dataset

On KITTI, we report the performance of our proposed model 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 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.

Comparison with state-of-the-art: Table II shows the results for car class on KITTI *val* split. For all four state-of-the-art models augmented with DSA and FSA, both variants were able to achieve performance improvements over strong baselines with significantly fewer parameters and FLOPs. On KITTI *test* split, we evaluate PV-RCNN+DSA and compare it with the models on KITTI benchmark. The results are shown in Table III. On the car class DSA shows an improvement of 0.15 3D AP on the hard setting,

Model	Car - 3D			Car - BEV			Cyclist - 3D			Cyclist - BEV		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
MV3D [5]	74.97	63.63	54.00	86.62	78.93	69.80	-	-	-	-	-	-
PointPillars [8]	82.58	74.31	68.99	90.07	86.56	82.81	77.10	58.65	51.92	79.90	62.73	55.58
SECOND [9]	83.34	72.55	65.82	89.39	83.77	78.59	71.33	52.08	45.83	76.50	56.05	49.45
PointRCNN [10]	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 [18]	87.95	79.71	75.09	94.74	89.19	**86.42**	78.69	61.59	55.30	81.36	67.23	59.35
3DSSD [17]	88.36	79.57	74.55	92.66	89.02	85.86	**82.48**	64.10	56.90	**85.04**	67.62	61.14
SA-SSD [19]	88.75	79.79	74.16	**95.03**	**91.03**	85.96	-	-	-	-	-	-
TANet [13]	83.81	75.38	67.66	-	-	-	73.84	59.86	53.46	-	-	-
Point-GNN [14]	88.33	79.47	72.29	93.11	89.17	83.90	78.60	63.48	57.08	81.17	67.28	59.67
PV-RCNN [11]	**90.25**	81.43	76.82	94.98	90.65	86.14	78.60	63.71	57.65	82.49	68.89	62.41
PV-RCNN + DSA (Ours)	88.25	**81.46**	**76.96**	92.42	90.13	85.93	82.19	**68.54**	**61.33**	83.93	**72.61**	**65.82**

TABLE III: Performance comparison of 3D detection on KITTI *test* split with AP calculated with 40 recall positions. The **best** and second-best performances are highlighted across all datasets.

Model	Mode	mAP	NDS	Car	Truck	Bus	Trailer	CV	Ped	Moto	Bike	Tr. Cone	Barrier
PointPillars [8]	Lidar	30.5	45.3	68.4	23.0	28.2	23.4	4.1	59.7	27.4	1.1	30.8	38.9
WYSIWYG [23]	Lidar	35.0	41.9	79.1	30.4	46.6	40.1	7.1	65.0	18.2	0.1	28.8	34.7
PointPillars+ [24]	Lidar	40.1	55.0	76.0	31.0	32.1	36.6	11.3	64.0	34.2	14.0	45.6	56.4
PMPNet [12]	Lidar	45.4	53.1	79.7	33.6	47.1	43.0	**18.1**	**76.5**	40.7	7.9	58.8	48.8
SSN [25]	Lidar	46.3	56.9	80.7	37.5	39.9	43.9	14.6	72.3	**43.7**	20.1	54.2	56.3
Point-Painting [24]	RGB + Lidar	46.4	58.1	77.9	35.8	36.2	37.3	15.8	73.3	41.5	**24.1**	**62.4**	**60.2**
PointPillars + DSA (Ours)	Lidar	**47.0**	**59.2**	**81.2**	**43.8**	**57.2**	**47.8**	11.3	73.3	32.1	7.9	60.6	55.3

TABLE IV: Performance comparison of 3D detection with PointPillars backbone on nuScenes *test* split. "CV", "Ped" , "Moto", "Bike", "Tr. Cone" indicate construction vehicle, pedestrian, motorcycle, bicycle and traffic cone respectively. The values are taken from the official evaluation server https://eval.ai/web/challenges/challenge-page/356/leaderboard/1012.

Difficulty	Method	Vehicle	
		3D AP	3D APH
L1	StarNet [26]	53.7	-
	PointPillars [8]	56.6	-
	PPBA [27]	62.4	-
	MVF [5]	62.9	-
	AFDet [28]	63.7	-
	CVCNet [29]	65.2	-
	Pillar-OD [30]	69.8	-
	†SECOND [9]	70.2	69.7
	PV-RCNN [11]	70.3	69.7
	SECOND + DSA (Ours)	**71.1**	**70.7**
L2	†SECOND [9]	62.5	62.0
	PV-RCNN [11]	**65.4**	**64.8**
	SECOND + DSA (Ours)	63.4	63.0

TABLE V: Comparison on Waymo Open Dataset *validation* split for 3D vehicle detection. Our DSA model has 52% fewer parameters and 32% fewer FLOPs compared to SECOND and 80% fewer parameters and 41% fewer FLOPs compared to PV-RCNN. †Re-implemented by [48]

while for the smaller cyclist class we achieve significantly better performance than all other methods with upto 4.5 3D AP improvement on the moderate setting. Overall, the results consistently demonstrate that adding global contextual information benefits performance and efficiency, especially for the difficult cases with smaller number of points.

B. 3D Detection on the nuScenes Dataset

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 state-of-the-art: We first compare our PointPillars+DSA model with PointPillars+ [24], a class-balanced re-sampled version of PointPillars inspired by [15]. DSA achieves about 7% improvement in mAP and 4.2% improvement in NDS compared to PointPillars+, even for some small objects, such as pedestrian and traffic cone. Compared with other attention and fusion-based methods like PMPNet and Point-Painting, DSA performs better in the main categories of traffic scenarios such as Car, Truck, Bus and Trailer etc. Overall, our model has the highest mAP and NDS score compared to state-of-the-art PointPillars-based 3D detectors.

C. 3D Detection on the Waymo Open Dataset

We also report performance on the large Waymo Open Dataset with our SECOND+DSA model to further validate its effectiveness. The objects in the dataset are split into two levels based on the number of points in a single object, where LEVEL1 objects have at-least 5 points and the LEVEL2 objects have at-least 1 point inside. For evaluation, the average precision (AP) and average precision weighted by heading (APH) metrics are used. The IoU threshold is 0.7 for vehicles.

Comparison with the state-of-the-art: Table V shows that our method outperforms previous state-of-the-art PV-RCNN with a 0.8%AP and 1%APH gain for 3D object detection while having 80% fewer parameters and 41% fewer FLOPs on LEVEL1. This supports that our proposed DSA is able to effectively capture global contextual information for improving 3D detection performance. Better performance in terms

Model	$N_{filters}$	N_h	N_l	N_{keypts}	r_{def}	r_{up}	3D AP	Params	FLOPs
baseline	(64,128,256)	-	-	-	-	-	78.39	4.8M	63.4G
	(64,64,128)	-	-	-	-	-	78.07	1.5M	31.5G
(A)	(64,64,64)	2	2	-	-	-	78.67	1.0M	31.3G
		4	1	-	-	-	78.34	1.0M	31.5G
		4	2	-	-	-	79.04	1.0M	31.7G
		4	4	-	-	-	78.56	1.0M	32.0G
(B)	(64,64,64)	4	2	512	3	1.6	78.70	1.1M	32.4G
				1024			78.95	1.1M	32.4G
				2048			78.94	1.1M	32.4G
				4096			78.90	1.1M	32.4G
(C)	(64,64,64)	4	2	2048	2	1.6	78.93	1.1M	32.4G
					1.4	1.6	78.22	1.1M	32.4G
					3	2	78.10	1.1M	32.4G
					3	1	78.96	1.1M	32.4G
(D)	(64,128,256)	4	2	2048	2	1	79.80	5.1M	73.5G

TABLE VI: Ablation of model components with PointPillars backbone on KITTI moderate Car class of *val* split.

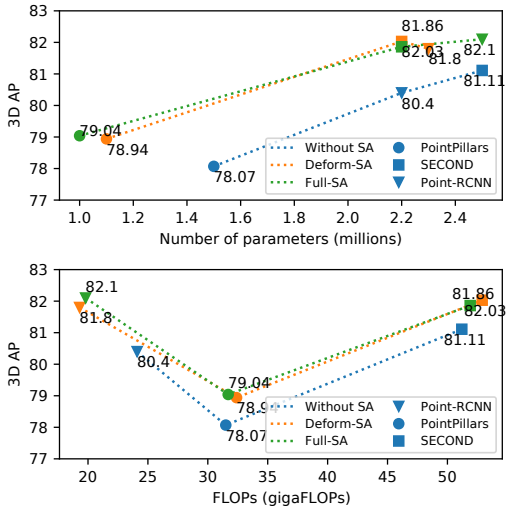


Fig. 4: 3D AP on moderate Car class of KITTI val split (R40) vs. number of parameters (Top) and GFLOPs (Bottom) for baseline models and proposed baseline extensions with Deformable and Full SA.

of APH also indicates that context helps to predict more accurate heading direction for the vehicles. On LEVEL2, we outperform the SECOND baseline by 0.9% AP and 1.0% APH. Overall SECOND+DSA provides the better balance between performance and efficiency as compared to PV-RCNN. The experimental results validate the generalization ability of FSA/DSA on various datasets.

D. Ablation studies and analysis

Ablation studies are conducted on the KITTI validation split [5] for moderate Car class using AP@R40, in order to validate our design choices.

a) Model variations: In our ablation study with PointPillars backbone in Table VI, we represent the number of 2D convolution filters as $N_{filters}$, self-attention heads as N_h, self-attention layers as N_l, sampled points for DSA as N_{keypts}, deformation radius as r_{def} and the up-sampling radius as r_{up}.

Effect of number of filters: We note that both FSA and DSA

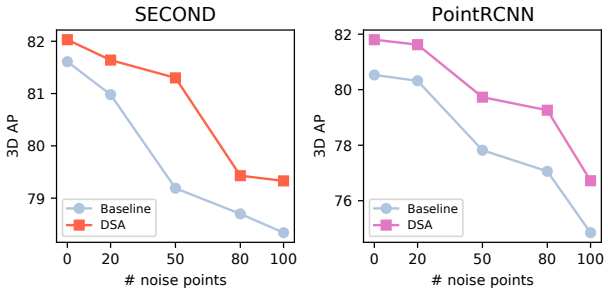


Fig. 5: 3D AP of SECOND-DSA (orange) and Point-RCNN-DSA (violet) vs. SECOND and Point-RCNN baseline (light-steel-blue) for noise-points per ground-truth bounding box, varying from 0 to 100 on KITTI *val* moderate

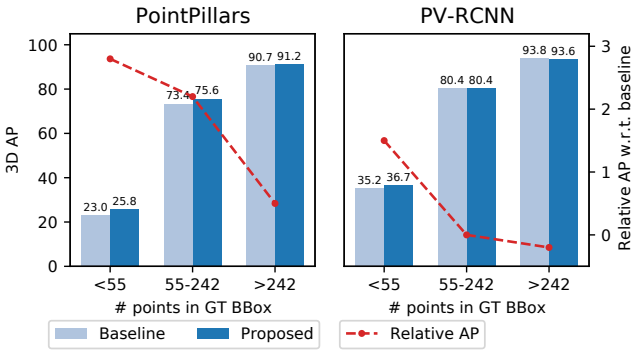


Fig. 6: 3D AP of PointPillars-FSA, PV-RCNN-FSA and respective baselines vs.number of points in the ground-truth bounding box on KITTI *val*

outperform not only the models with similar parameters by 0.97% and 0.87% respectively, but also the state-of-the-art models with 80% more parameters by 0.65% and 0.55%. This indicates that our modules are extremely parameter efficient. Finally, we also note that if the number of parameters and compute are kept roughly the same as the baseline(Row-D), DSA outperforms the baseline by a large margin of 1.41%. We also illustrate consistent gains in parameter and computation budget across backbones in Figure 4.

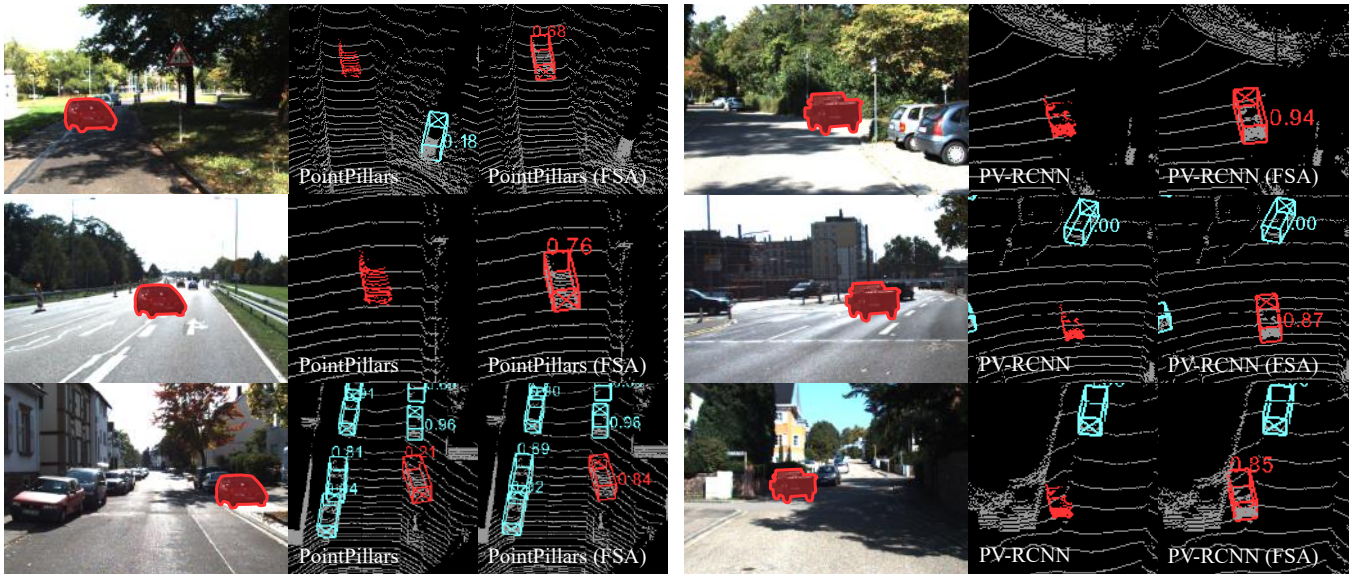
Effect of number of self-attention heads and layers (Row-A): We note that increasing heads from 2 to 4 leads to an improvement of 0.37% for PointPillars. Since increasing number of self-attention layers beyond a certain value can lead to over-smoothing [52], we use 2 FSA/DSA layers in the backbone and 4 heads for multi-head attention.

Effect of number of sampled points (Row-B): For DSA, we also vary the number of keypoints sampled for computation of global context. We note that the performance is relatively robust to the number of sampled points.

Effect of deformation and upsampling radius (Row-C): For DSA, we note that the performance is generally robust to the deformation radius upto a certain threshold, but the up-sampling radius needs to be tuned carefully. Generally an up-sampling radius of 1.6m in cars empirically works well.

b) Effect of noise on performance: We introduce noise points to each object similar to TANet [13], to probe the

(a) Object in the original scene



(b) Object inserted into another scene

Fig. 7: (a) RGB images and point-clouds of cars on KITTI-*val* in which addition of context via FSA had the largest increase in the detection confidence. (b) We use a simple *copy and paste* method on these cars to create new point-clouds for testing our attention-based context aggregation detector for Point-Pillars and PV-RCNN backbone. We find that our FSA-based detector is more accurate and robust across scenes compared to the baseline.

robustness of representations learned. As shown in Figure 5, 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.

c) Effect of number of object points on performance: We sort the cars based on the numbers of points in them in increasing order, and divide them into 3 groups based on the sorted order. Then we calculate the 3D AP across every group. As shown in Figure 6, the effect of the self-attention module becomes apparent as the number of points on the cars decreases. For objects with very few points, FSA can increase the 3D AP for PointPillars by 2.8% and PV-RCNN by 1.5%.

d) Qualitative results: In Figure 1, we first show that our FSA-based detector identifies missed detections and eliminates false positives across challenging scenes for different backbones. Next, we identify objects for which addition of self-attention shows the largest increase in detection confidences as shown in Figure 7(a). We then copy-paste the point-clouds for these cars into different scenes. Our expectation is that the FSA is a more robust detector and can detect these examples even when randomly transplanted to different scenes. The first two rows of Figure 7(b) show that FSA is capable of detecting the copy-pasted car in different scenes while the baseline consistently misses them. This supports our motivation that adding contextual self-attention features to convolutional maps results in a more accurate and robust feature extractor. In the third row of Figure 7(b), we show cases for Point-Pillars and PV-RCNN where the orientation is flipped for our FSA-based detector even though the detection confidence remains high. We expect that this confusion occurs because FSA aggregates context from nearby high-confidence detections thereby correlating their orientations.

VI. 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] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in 2017 Advances in Neural Information Processing Systems (NIPS), 2017.

[2] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012.

[3] H. Caesar et al., "nuScenes: A Multimodal Dataset for Autonomous Driving," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020

[4] P. Sun et al., "Scalability in Perception for Autonomous Driving: Waymo Open Dataset," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[5] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[6] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[7] R. Q. Charles, H. Su, and L.J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in 2017 Advances in Neural Information Processing Systems (NIPS), 2017.

[8] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[9] Y.Yan, Y. Mao and B. Li, "SECOND: Sparsely embedded convolutional detection," in 2018 Sensors, vol. 18, no. 10, 2018.

[10] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[11] S. Shi et al., "PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[12] J. Yin, J. Shen, C. Guan, D. Zhou, and R. Yang, "LiDAR-based online 3D video object detection with graph-based message passing and spatiotemporal transformer attention," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[13] Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, and X. Bai, "TANet: Robust 3D object detection from point clouds with Triple Attention," Proc. Conf. AAAI Artif. Intell., vol. 34, no. 07, 2020.

[14] W. Shi and R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[15] B. Zhu, Z. Jiang, X. Zhou, Z. Li, and G. Yu, "Class-balanced grouping and sampling for point cloud 3D object detection," arXiv [cs.CV], 2019.

[16] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D object detection from RGB-D data," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[17] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3DSSD: Point-based 3D single stage object detector," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020

[18] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "STD: Sparse-to-dense 3D object detector for point cloud," in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

[19] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, "Structure aware single-stage 3D object detection from point cloud," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[20] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[21] J. Chen, B. Lei, Q. Song, H. Ying, D. Z. Chen, and J. Wu, "A hierarchical graph network for 3D object detection on point clouds," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[22] W. Zhang and C. Xiao, "PCAN: 3D attention map learning using contextual information for point cloud based retrieval," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[23] P. Hu, J. Ziglar, D. Held, and D. Ramanan, "What you see is what you get: Exploiting visibility for 3D object detection," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[24] S. Vora, A. H. Lang, B. Helou, and O. Beijbom, "PointPainting: Sequential fusion for 3D object detection," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[25] X. Zhu, Y. Ma, T. Wang, Y. Xu, J. Shi, and D. Lin, "SSN: Shape signature networks for multi-class object detection from point clouds," in Computer Vision – ECCV 2020, Cham: Springer International Publishing, 2020.

[26] J. Ngiam et al., "StarNet: Targeted computation for object detection in point clouds," arXiv [cs.CV], 2019.

[27] S. Cheng et al., "Improving 3D object detection through progressive population based augmentation," in Computer Vision – ECCV 2020, Cham: Springer International Publishing, 2020.

[28] R. Ge et al., "AFDet: Anchor free one stage 3D object detection," arXiv [cs.CV], 2020.

[29] Q. Chen, L. Sun, E. Cheung and A. Yuille, "Every View Counts: Cross-View Consistency in 3D Object Detection with Hybrid-Cylindrical-Spherical Voxelization," in Advances in Neural Information Processing Systems (NeurIPS), 2020.

[30] Y. Wang et al., "Pillar-based object detection for autonomous driving," in Computer Vision – ECCV 2020, Cham: Springer International Publishing, 2020.

[31] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local Neural Networks," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[32] I. Bello, B. Zoph, Q. Le, A. Vaswani, and J. Shlens, "Attention Augmented Convolutional Networks," in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

[33] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[34] H. Lu, X. Chen, G. Zhang, Q. Zhou, Y. Ma, and Y. Zhao, "Scanet: Spatial-channel attention network for 3D object detection," in ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019.

[35] Q. Xie et al., "MLCVNet: Multi-Level Context VoteNet for 3D Object Detection," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[36] J. Yang et al., "Modeling point clouds with self-attention and Gumbel subset sampling," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[37] S. Xie, S. Liu, Z. Chen, and Z. Tu, "Attentional ShapeContextNet for Point Cloud Recognition," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018.

[38] X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, "3D Object Detection with Pointformer," arXiv [cs.CV], 2020.

[39] A. Paigwar, O. Erkent, C. Wolf, and C. Laugier, "Attentional PointNet for 3D-object detection in point clouds," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019.

[40] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," arXiv [cs.NE], 2014.

[41] J. Dai et al., "Deformable Convolutional Networks," in 2017 IEEE International Conference on Computer Vision (ICCV), 2017.

[42] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-Alone Self-Attention in Vision Models", in Advances in Neural Information Processing Systems (NeurIPS), 2019.

[43] Y. Wu and K. He, "Group Normalization," in Computer Vision – ECCV 2018, Cham: Springer International Publishing, 2018.

[44] V. Zambaldi, D. Raposo, A. Santoro, V. Bapst, Y. Li, I. Babuschkin, K. Tuyls, D.P. Reichert, T. Lillicrap, E. Lockhart, M. Shanahan, V. Langston, R. Pascanu, M. Botvinick, O. Vinyals, and P. Battaglia, "Deep reinforcement learning with relational inductive biases," in International Conference on Learning Representations (ICLR), 2019.

[45] C. Qin, H. You, L. Wang, C.-C. J. Kuo, and Y. Fu, "PointDAN: A multi-scale 3D domain adaption network for point cloud representation," in Advances in Neural Information Processing Systems (NeurIPS), 2019.

[46] G. Gkioxari, J. Johnson, and J. Malik, "Mesh R-CNN," in 2019

IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
[47] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", International Conference on Learning Representations(ICLR), 2015.
[48] OpenPCDet Development Team, "OpenPCDet: An Open-source Toolbox for 3D Object Detection from Point Clouds", https://github.com/open-mmlab/OpenPCDet, 2020.
[49] Z. Han, I.J. Goodfellow, D. Metaxas and A. Odena, "Self-Attention Generative Adversarial Networks," in International Conference on Machine Learning (ICML), 2019.
[50] J. Lee, Y. Lee, J. Kim, A.R. Kosiorek, S. Choi, and Y. Teh, "Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks", in International Conference on Machine Learning (ICML), 2019.
[51] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," arXiv [cs.LG], 2019.
[52] Y. Rong, W. Huang, T. Xu, and J. Huang, "DropEdge: Towards deep Graph Convolutional Networks on node classification," in International Conference on Learning Representations (ICLR), 2019.
[53] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay," arXiv [cs.LG], 2018.
[54] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "KPConv: Flexible and Deformable Convolution for Point Clouds," in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
[55] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep Convolutional Networks on 3D Point Clouds," in 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
[56] C. R. Qi, O. Litany, K. He, and L. Guibas, "Deep Hough voting for 3D object detection in point clouds," in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
[57] X. Wu, Y. Cai, Q. Li, J. Xu, and H.-F. Leung, "Combining contextual information by self-attention mechanism in convolutional neural networks for text classification," in Web Information Systems Engineering – WISE 2018, Cham: Springer International Publishing, 2018, pp. 453–467.
[58] Z. Zhang, C. Lan, W. Zeng, X. Jin, and Z. Chen, "Relation-Aware Global Attention for Person Re-Identification," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
[59] O. Oktay et al., "Attention U-Net: Learning where to look for the pancreas," arXiv [cs.CV], 2018.

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. We then showcase the flexibility and robustness of our models through extended results on the the KITTI [2] 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.

VII. Network Architectures and Training Details

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

A. Architectural details

The detailed specification of the various layers in our FSA and DSA augmented baselines—PointPillars [8], SECOND [9], Point-RCNN [10] and PV-RCNN [11]—is documented in Table VIII, Table IX, Table X, and Table XI, 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.

B. Experimental settings

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 [1]. For the *test* submissions to KITTI and nuScenes official servers, we retain the full parameterization of the original baselines. For the nuScenes *test* submission, we follow the configuration for PP described in Table VIII, while adding the DSA module with 4 heads, 2 layers, 64-dimensional context, 2 m deformation radius, 4096 sampled pillars and an up-sampling method described in the following subsection. For the Waymo Open dataset *validation* evaluation, we use the configuration for DSA-SECOND as described in Table IX, except that we use 4096 sampled keypoints, 2 m deformation radius and 1 m interpolation radius. We use Pytorch [51] and the recently released OpenPCDet [48] repository for our experiments. Our models are trained from scratch in an end-to-end manner with the ADAM optimizer [47]. The learning rates used for the different models are given in Table VII. For the proposal refinement stage in two-stage networks [10], [11], 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 [8], [9] and the top-100 proposals generated from two stage approaches [10], [11] 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.

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

TABLE VII: Batch size and learning rate configurations for each backbone model on KITTI benchmark

C. Up-sampling for Deformable Self-Attention (DSA)

Given the features for m sampled, deformed and attended points, we explore *two* up-sampling methods to distribute the accumulated structural information back to all n node locations. We first test the feature propagation method proposed in PointNet++ [7] to obtain point features for all the original nodes. This works well for most of our experiments, especially on the KITTI and the Waymo Open Dataset. While this is simple and easy to implement, a draw-back is that the interpolation radius has to be chosen empirically. To avoid choosing an interpolation radius for the diverse classes present in the nuScenes dataset, we explore an attention-based up-sampling method as proposed in [50]. The set of m points is attended to by the n node features to finally produce a set of n elements. This up-sampling method works well for the nuScenes dataset.

VIII. Detailed Results

We provide additional experimental details on the validation split of the KITTI [2] data-set in this section. In Table XIII, we first show the 3D and BEV AP for moderate difficulty on the *Cyclist* 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. We then proceed to list the 3D AP and BEV performances with respect to distance from the ego-vehicle in Table XIV. 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 XII, 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 [8], which loses a lot of information due to pillar-based discretization of points, can supplement this loss with fine-grained context information.

Attribute	PP [8]	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 VIII: Architectural details of PointPillars [8], our reduced parameter PointPillars version, proposed FSA-PointPillars and DSA-PointPillars

Attribute	SECOND [9]	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 IX: Architectural details of SECOND [9], our reduced parameter SECOND version, and proposed FSA-SECOND and DSA-SECOND

IX. Qualitative Results

A. 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 8). 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 [8], SECOND [9], Point-RCNN [10] and PV-RCNN [11]. 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).

B. Visualization of Attention Weights

We also visualize the attention weights for FSA-variant for the SECOND [9] backbone in Figure 9. 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 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

Attribute	Point-RCNN [10]	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 X: Architectural details of Point-RCNN [10], our reduced parameter Point-RCNN version, proposed FSA-Point-RCNN and DSA-Point-RCNN

Attribute	PV-RCNN [11]	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 XI: Architectural details of PV-RCNN [11], and proposed FSA-PVRCNN and DSA-PVRCNN

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 9 that it can aggregate local

Model	Modality	Params (M)	GFLOPs	Car 3D AP Easy	Moderate	Hard
PP [8]	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 [9]	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 [10]	Points	4.0	27.4	**91.94**	80.52	78.31
Point-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 XII: Detailed comparison of 3D AP with baseline on KITTI *val* split with 40 recall positions

Model	3D	BEV
PV-RCNN [11]	70.38	74.5
PV-RCNN + DSA	**73.03**	**75.45**
PV-RCNN + FSA	71.46	74.73

TABLE XIII: Performance comparison for moderate difficulty cyclist class on KITTI *val* split.

Distance	Model	Car	Cyclist	Pedestrian
0-30m	PV-RCNN [11]	91.71	73.76	56.82
	DSA	91.65	**74.89**	59.61
	FSA	**93.44**	74.10	**61.65**
30-50m	PV-RCNN [11]	50.00	35.15	-
	DSA	52.02	**47.00**	-
	FSA	**52.76**	39.74	-

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

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 9. 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 9. Our visualizations thus indicate that semantically meaningful patterns emerge through the self-attention based context-aggregation module.

X. 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 [8] and voxel grids [9] for 3D object detection. Point-wise feature learning methods like PointNet++ [7] have also been successful in directly utilizing sparse, irregular points for 3D object detection [10].

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:

$$f(\hat{p}) = \sum_{l \in \Omega_1} x_{\hat{p}+l} w_l \tag{4}$$

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:

$$f(\hat{p}) = \max_{l \in \Omega_2} h(x_l) \tag{5}$$

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.

a) 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.

Fig. 8: 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 [8], SECOND [9], Point-RCNN [10] and PV-RCNN [11]; Performance of proposed FSA module-augmented detectors. Viewed best when enlarged.

Fig. 9: Visualization of attention maps produced by our proposed FSA-variant on SECOND [9] backbone. We analyze the implications of the produced attention maps in Section 3.2.