Spatial Pruned Sparse Convolution for Efficient 3D Object Detection

Anonymous Author(s)
Affiliation
Address
email

Abstract

3D scenes are dominated by a large number of background points, which is redundant for the detection task that mainly needs to focus on foreground objects. In this paper, we analyze major components of existing sparse 3D CNNs and find that 3D CNNs ignore the redundancy of data and further amplify it in the down-sampling process, which brings a huge amount of extra and unnecessary computational overhead. Inspired by this, we propose a new convolution operator named spatial pruned sparse convolution (SPS-Conv), which includes two variants, spatial pruned submanifold sparse convolution (SPSS-Conv) and spatial pruned regular sparse convolution (SPRS-Conv), both of which are based on the idea of dynamically determine crucial areas for performing computations to reduce redundancy. We empirically find that magnitude of features can serve as important cues to determine crucial areas which get rid of the heavy computations of learning-based methods. The proposed modules can easily be incorporated into existing sparse 3D CNNs without extra architectural modifications. Extensive experiments on the KITTI and nuScenes datasets demonstrate that our method can achieve more than 50% reduction in GFLOPs without compromising the performance.

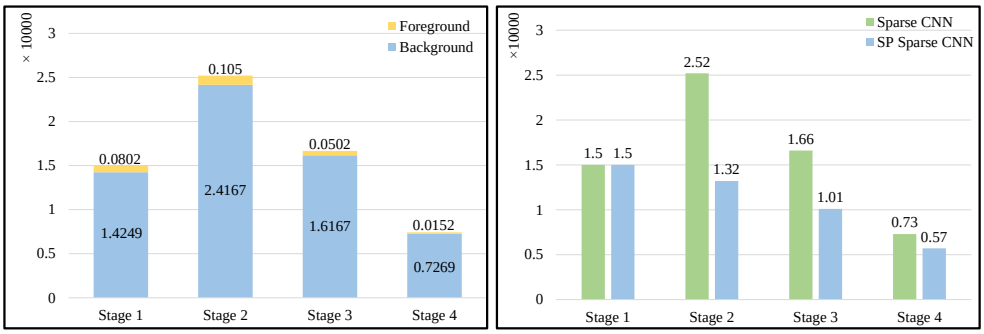
1 Introduction

3D object detection has always been a research field of great interest due to its wide applications in autonomous driving, virtual reality, and robotics. However, compared with the 2D detection task, LiDAR-based 3D detection is more challenging due to the inherent characteristics of LiDAR points such as disorders, irregularity, and non-uniformity. Recently, voxel-based methods [23, 10, 2, 36, 30, 18] with deep 3D sparse convolutional neural networks (CNNs) have become one of the major research streams to tackle 3D object detection problem due to its effectiveness and simplicity. However, the improved accuracy is often accompanied by increased computational costs [29, 35, 3], limiting its applicability in practical systems. This motivates us to investigate potential redundancies in the detection model that can be safely avoided to improve efficiency without sacrificing accuracy.

When delving into the task of 3D detection, we found that 3D data itself has high redundancy as shown in Fig. 1(a). Notably, compared with background points, the proportion of foreground points in the entire scene is extremely low (around 5%), which shows that non-informative background points occupy the major areas of a scene. However, the existing 3D sparse CNNs are applied uniformly to the whole scene, causing a considerable amount of computation on the background areas, which might be potentially redundant. Intuitively, if such areas can be identified and selectively skipped, the computational costs have the potential to be dramatically reduced without deteriorating the performance.

Besides the redundancy of the data, the model design itself also brings redundancy. To avoid aggressive down-sampling, 3D sparse CNNs also adopt a dilation-like design when applied to

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



(a) (b)

Figure 1: (a) Comparison of the foreground and background ratios. (b) By replacing SPRS-Conv with its counterpart in sparse CNN, unnecessary positions are effectively suppressed from being activated.

perform the down-sampling operation. Specifically, it will compute features for adjacent empty voxels as shown in Fig. 2 (b): the yellow voxels which are initially empty will be assigned with computed features. Consequently, after convolutional (stride > 1) down-sampling, the number of non-empty voxels might be increased rather than decreased as shown in Fig. 1 (b): the number of non-empty voxels even doubled (see stage 2) compared to the input (see stage1). This will undoubtedly increase unnecessary computational costs for follow-up stages.

The above observations prompt us to ask whether there is a way to identify redundant computations that can be pruned to improve efficiency. A solution that has been attempted in the 2D image domain [28, 11] is to add an additional learnable module to predict a soft mask [21, 27] that locates areas to be skipped for computational efficiency. The module often requires additional post-fine-tuning, auxiliary costs for integration, and incurs non-negligible computational overheads.

With the aforementioned considerations, we propose a new simple and efficient sparse convolution operator named Spatial Pruned Sparse Convolution (SPS-Conv), to alleviate redundancies caused by data and model design. The core idea of SPS-Conv is to find redundancy in the model dynamically. To avoid the overheads of learning-based methods for simplicity and efficiency, we investigate whether the features from the model itself contain useful cues that can be leveraged to identify redundancies. Fortunately, we find that the magnitude of features could be a robust signal to reflect the importance where locations with smaller magnitude are more likely to be redundant areas. This leads us to a magnitude-based sampling module which is incorporated into 3D convolution layers to reduce redundancy from data and model.

Specifically, to address data redundancy, the submanifold variant of SPS-Conv, termed as spatial pruned submanifold sparse convolution (SPSS-Conv) can adaptively calculate important positions in the light of the magnitude of features and perform convolutions on these locations, leaving features of redundant locations unchanged. As for model redundancy in the down-sampling process, another variant named spatial pruned regular sparse convolution (SPRS-Conv), which can dynamically determine the position that needs to be inflated. As Tab. 1(b) shows, our method effectively reduces the meaningless expansion caused by convolution (around 50% in stage2). In general, SPS-Conv only keeps the essence and discards the irrelevant ones for pursuing a higher efficiency without compromising performance.

In summary, we propose an efficient convolution operator for spatial redundancy pruning, which can be easily incorporated into existing sparse 3D CNNs for 3D detection. We conduct extensive experiments on both KITTI and nuScenes dataset. The result shows that we can achieve comparable performance with SOTA methods while enjoying 52.4% and 46.5% GFLOPs reduction.

2 Related Work

Sparse Convolution. Depending on how it is used, sparse convolution[15] can be subdivided into regular sparse convolution and submanifold sparse convolution. Regular sparse convolution is often used for the down-sample layer, which dilates all input features to its kernel-size neighbors,

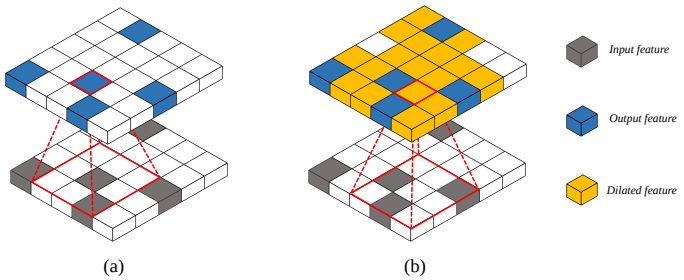


Figure 2: Illustration of conventional sparse convolution. (a) Submanifold sparse convolution (b) Regular sparse convolution.

sacrificing computational efficiency in exchange for a wider interaction of information. In contrast, the submanifold convolution is more like a simplified version of the regular sparse convolution, frequently used in residual blocks[17], only calculates on valid positions, avoiding meaningless expansion and achieving efficient computation.

Voxel-based Detectors. In order to process irregular point cloud data, voxel-based methods first convert the point cloud into regular voxel, and then use mature CNNs for feature extraction. However, the computational cost and memory requirement both increase cubically along with the voxel resolution. Thus, it is infeasible to train a voxel-based model with high-resolution inputs. Benjamin et al.[15] proposed a novel convolution operator named submanifold sparse convolution, by reducing calculation on invalid position, greatly improves the computing efficiency and memory.

Approaches for voxel-based detection methods can be grouped into two categories, i.e., single-stage[12, 16, 33, 34, 38, 39] and two-stage detectors[10, 23, 25, 22, 24]. Single-stage detectors are relatively simple, directly predict final bounding boxes based on the features extracted by sparse CNN. VoxelNet[40] utilizes a Sparse CNN to extract voxel features from a dense grid. SECOND[30] proposes 3D sparse convolutions to efficiently extract voxel features. HVNet[33] designs a convolutional network that attentively aggregate and project the multi-scale feature maps to achieve better performance. In contrast, two-stage detectors are more complex, but can get higher performance. Part-A^2[24] proposed a part-aware and aggregation module to exploit the intra-object part location. PV-RCNN[23] uses keypoints to extract voxel features for better boxes refinement.

Dynamic kernel shape design. Many literature work on dynamic kernel design for adapting various tasks. Deformable convolution [9] predicts offets for feature sampling. For 3D scene understanding, KPConv [26] constructs dynamic graph for kernel points. Deformable PV-RCNN [3] applies offset prediction for feature sampling in 3D object detection. Focals Conv [8] learns a dynamic spatially sparsity which enhances network for spatial modeling capabilities,

3 Background and Motivation

In this section, we will introduce the mechanism of sparse convolution, then analyze its redundancy and introduce our motivation.

3.1 Preliminary of Sparse Convolution

To better illustrate sparse convolution, we first introduce its general expression. We denote x_p as an input feature with c_{in} dimension at position p, $\mathrm{w} \in \mathbb{R}^{K^d \times \mathrm{c}_{\text{in}} \times \mathrm{c}_{\text{out}}}$ is the weight of convolution kernel, where d and K^d refers to the dimension of the spatial space and spatial size of the kernel respectively. Thus we can formulate the convolution process as follows.

$$\mathrm{y}_p \in P_{\text{out}} = \sum_{k \in K^d(p, P_{\text{in}})} \mathrm{w}_k \cdot \mathrm{x}_{\bar{p}_k}, \text{ where } K^d(p, P_{\text{in}}) = \left\{ k \,|\, p + k \in P_{\text{in}}, k \in K^d \right\}. \tag{1}$$

Here, P_{in} and P_{out} refers to the input and output feature space, k is the kernel offset that corresponds to all the valid locations in kernel space K^d, $\bar{p}_k = p + k$ denotes all non-empty neighbor voxels around center p. Due to the discrete data distribution in 3d space, we define $K^d(p, P_{\text{in}})$ as a subset of K^d, leaving out the empty position, which is constrained by position p and input feature space P_{in}.

There are two types of sparse convolutions, namely, regular sparse convolution [14] and submanifold sparse convolution [15], their major difference lies in the output position of the convolution. For regular sparse convolution, as shown in Fig. 2, the position related to the input will be activated during the convolution process. Specifically, these position will be assigned value and combined with the P_{in} to form a new P_{out}. This process can be easily formulated as

$$P_{\text{out}} = \bigcup_{p \in P_{\text{conv}}} Q\left(p, K^d\right), \text{where } Q\left(p, K^d\right) = \left\{p + k \mid k \in K^d\right\} \tag{2}$$

where P_{conv} is a subset of P_{in}, representing the position where the operation needs to be performed. $Q\left(p, K^d\right)$ refers to the position related to P_{conv} which is restricted by kernel size.

Regular sparse convolution can effectively expand the receptive field like 2D convolution which is beneficial for 3D sparse data. However, it also brings computation burden due to the generation of too many active locations, which can lead to a decrease in speed in subsequent convolution layers due to the large number of active points. In order to achieve a better trade-off between efficiency and receptive field, regular sparse convolution is often adopted in the down-sample layer in sparse CNNs.

In contrast, submanifold sparse convolution[15] restricts an output location to be active if and only if the corresponding input location is active, which avoid the dilation in regular sparse convolution, although the receptive field is limited. With a simple modification on Eq. 2, $P_{\text{in}} = P_{\text{out}}$, regular mode can be convert to submanifold mode.

3.2 Redundancy Analysis and Motivations

Despite the wide adoption of sparse CNNs in 3D object detection due to their generality and efficiency, spatial redundancy caused by sparse CNNs for this task is still under-explored. In Fig. 5, we can find that the most common component of sparse CNN is the submanifold convolution. Considering the large number of task-independent background points in the 3D scene, a lot of redundant computation will be generated in each stage. In addition, due to the expansion of regular convolution, the background points will become denser, bringing more computational burden to the following stages. Therefore, in response to these problems, we have targeted the improvement of the sparse convolution operator. Details are shown in Sec. 4

4 Spatial Pruned Convolution Network

In this section, we propose a new efficient sparse convolution operator, named spatial pruned sparse convolution (SPS-Conv). Intending to effectively bring down the unnecessary computation costs caused by the 3D spatial redundancy, two variants of SPS-Conv are investigated, i.e., spatial pruned submanifold sparse convolution (SPSS-Conv) and spatial pruned regular sparse convolution (SPRS-Conv), and both of them are based on the same idea that the redundancy should be dynamically ameliorated in 3D space according to the specific contexts of different individuals. In the following, the magnitude-based sampling strategy is presented in Sec. 4.1, followed by the introduction of SPSS-Conv and SPRS-Conv in Sec. 4.2 and Sec. 4.3 respectively. Then, Sec. 4.4 manifests the generalization ability by showing how our method is applied to generic sparse CNNs.

4.1 Magnitude-guided spatial sampling

In previous works [28, 11, 5], sampling module is often designed in a learnable manner, *i.e.*, a convolution layer is trained to yield soft masks for sampling, while, at the time of inference, the soft masks are converted to hard ones where elements with lower responses will be simply discarded. Although these methods have been empirically shown effective in 2D tasks, the following issues still exist: 1) Additional supervision is required on the prediction branch, otherwise it is easy to fall into trivial solutions, *i.e.*, the values in the soft mask are all close to 1. 2) The proportion of pruning is

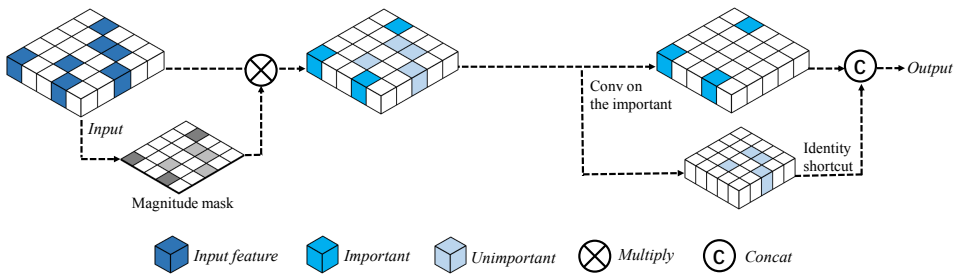


Figure 3: Illustration of magnitude-guided spatial sampling and spatial pruned submanifold sparse convolution (SPSS-Conv). Notably, during the process of SPSS-Conv, "conv on important" refers to perform convolution only at the positions with high magnitude value, and the neighborhood of the convolution still contains the unimportant features.

hard to be fixed, and it is instead automatically determined throughout the learning process, making it hard to satisfy the specific requirement with certain limited computation budget. 3) Two-stage fine-tuning is required to compensate for the performance loss of converting from soft mask to hard mask, consuming additional time and resources.

Recent literature has shown that the attention mechanism is able to reveal what the model really cares about. Following [37, 32], the feature magnitude is adopted for yielding the model's "attention" to highlight the informative elements. For the purpose of getting the magnitude-based attention mask, we first calculate the absolute mean values on different channels, and then the sigmoid function is applied to get the normalized output, which can be formulated as

$$M(F) = \text{Sigmoid}(G(F)), \text{where } G(F) = \frac{1}{C} \cdot \sum_{c=1}^{C} |F_c| \,. \tag{3}$$

where F and C denote the input feature and its channel. F_c is the feature of the c-th dimension. G and M refer to the spatial magnitude map and magnitude mask.

4.2 Spatial Pruned Submanifold Sparse Convolution

Considering the fact that the background regions often take the majority in 3D scenes, directly performing the submanifold convolution everywhere of the input volume inevitably involves substantial unnecessary calculations, leading to computational redundancy. Therefore, with a focus on alleviating this issue, we propose the spatial pruned submanifold sparse convolution (SPSS-Conv) that dynamically examines the regions of interest.

Specifically, with the guidance of the magnitude-based attention mask, we separate the elements P_{in} into two disjoint subsets, i.e., the important set P_{im} and the unimportant set P_{nim}, where $P_{\text{im}} \cup P_{\text{uim}} = P_{\text{in}}$. We note that various methods can be incorporated for accomplishing the division, such as using a fixed threshold or simply taking the elements with top-k scores. Then we re-weight the input feature map by multiplying it with the magnitude mask, applying submanifold convolution on important positions P_{im} according to Eq. 1, and concatenate it with the features of unimportant positions P_{nim} as a residual, as shown in Fig. 3. To this end, the reconstructed output feature can be written as:

$$\mathrm{y}_p \in P_{\text{out}} = \begin{cases} \sum\limits_{k \in K^d(p, P_{\text{in}})} (\mathrm{x}_{\bar{p}_k} \cdot M(\mathrm{x}_{\bar{p}_k})) \cdot \mathrm{w}_k, & p \in P_{\text{im}} \\ \mathrm{x}_{\bar{p}_k} \cdot M(\mathrm{x}_{\bar{p}_k}), & p \in P_{\text{nim}} \end{cases} \tag{4}$$

Since P_{im} is sparser than P_{in}, convolution enables task-beneficial locations to be highlighted. In contrast, those unimportant positions have relatively small activation values, thus the skip connection can not only reduce the computational overhead but also suppress the diffusion of redundant information.

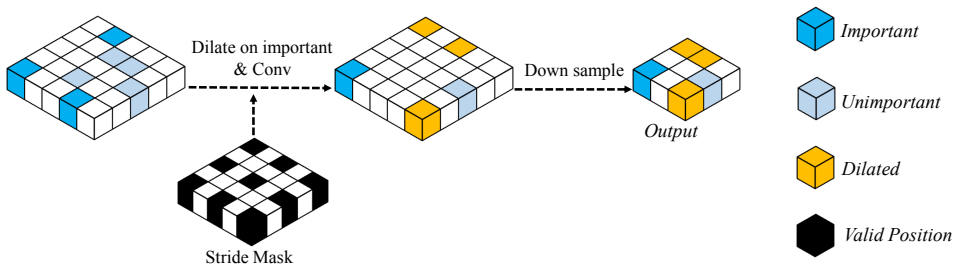


Figure 4: Illustration of spatial pruned regular sparse convolution (SPRS-Conv). The input of SPRS-Conv is generated by magnitude-guided spatial sampling. The figure shows the case of stride=2.

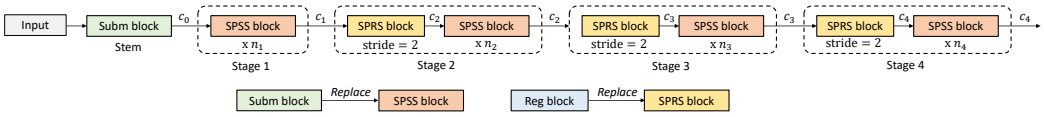


Figure 5: Framework overview. We substitute all the submanifold blocks and regular blocks with SPSS blocks and SPRS blocks, except the block in the stem layer. Each block contains a convolution layer followed by batch normalization and ReLU activation.

4.3 Spatial Pruned Regular Sparse Convolution

The "dilation" effect of the regular sparse convolution inevitably causes a large number of adjacent positions to be activated, which brings a greater computational burden to subsequent layers. Recent work [8] shows that the spatially dynamic sparsity in sparse convolution is essential for sophisticated 3D object detection. Motivated by this, SPRS-Conv is designed as a magnitude-based dynamic prediction down-sampling module, which can effectively suppress the amplification effect of spatial redundancy.

Similar to SPSS-Conv, P_{in} is first divided into two disjoint subsets, *i.e*, P_{im} and P_{nim}. Unlike the regular sparse convolution that dilates all possible neighborhood locations, leading to sub-optimal efficiency, we instead dynamically choose whether to expand for each position based on the magnitude mask yielded by Eq. 3. Further, we define P_{dilate}, which is a set composed of all P_{im} positions and the positions within their kernel range. The equation can be written as:

$$P_{\text{dilate}} = \left(\bigcup_{p \in P_{\text{im}}} P\left(p, K^d\right) \right) \cup P_{\text{im}} \tag{5}$$

Considering the problem of stride during down-sampling, we designed a stride mask $E\left(P\right)$ to ignore those that should be skipped by the stride, which can be formulated as below:

$$E(P) = \left\{ \left(\sum_{i=\{x,y,z\}} \left(p_i \,\%\, s\right) \right) = 0 \,|\, p \in P \right\} \tag{6}$$

where s refers to the stride of convolution, $\%$ denotes mod operation. With $E\left(P\right)$, we can further modify Eq. 2 as:

$$P_{\text{out}} = E\left(P_{\text{dilate}}\right) \cup E\left(P_{\text{nim}}\right) \tag{7}$$

As shown in Fig. 4, we operate at the specified positions according to Eq. 1 to get the output feature map. Then we down-sample it to the correct size according to the stride. Compared to the regular sparse convolution, our method dynamically prunes P_{out} conditioned on its specific content, effectively suppressing meaningless computations without compromising the performance.

Method	mAP	NDS	Car	Truck	Bus	Trailer	C.V.	Ped	Mot	Byc	T.C.	Bar
PointPillars [18]	30.5	45.3	68.4	23.0	28.2	23.4	4.1	59.7	27.4	1.1	30.8	38.9
3DSSD [31]	42.6	56.4	81.2	47.2	61.4	30.5	12.6	70.2	36.0	8.6	31.1	47.9
CBGS [41]	52.8	63.3	81.1	48.5	54.9	42.9	10.5	80.1	51.5	22.3	70.9	65.7
HotSpotNet [7]	**59.3**	66.0	83.1	50.9	56.4	53.3	**23.0**	81.3	**63.5**	36.6	73.0	71.6
CVCNET [6]	58.2	66.6	82.6	49.5	59.4	51.1	16.2	83.0	61.8	**38.8**	69.7	69.7
CenterPoint [36]	58.0	65.5	**84.6**	51.0	60.2	**53.2**	17.5	83.4	53.7	28.7	76.7	**70.9**
SP-CenterPoint (Ours)	**59.3**	**66.9**	84.2	**52.4**	**61.5**	52.6	20.1	**84.6**	58.6	31.5	**77.3**	70.2

Table 1: Comparison with other methods on nuScenes *test* split.

Method	mAP	NDS	GFLOPs	Car	Truck	Bus	Trailer	C.V.	Ped	Mot	Byc	T.C.	Bar
CenterPoint [36]	58.6	66.2	62.9	85.0	58.2	69.5	35.7	15.5	85.3	58.8	40.9	70.0	67.1
SP-CenterPoint (Ours)	58.5	66.1	**34.3**	84.9	58.1	70.2	34.7	17.2	85.3	58.7	38.6	69.7	67.0

Table 2: Comparison with base method on nuScenes *val* split. GFLOPs only include the floating-point operations in sparse CNNs.

4.4 Spatial Pruned Convolution Network

Our method is model-agnostic thus it can be easily plugged into any existing sparse CNNs. As shown in Fig 5, a regular frame of sparse CNN is composed of a stem layer and four stages, each of which contains a down-sampling layer and two submanifold sparse convolution blocks. To demonstrate the effectiveness and generalization ability, we replace all regular sparse convolutions and submanifold sparse convolutions in sparse CNN, except those in the stem layer, with our proposed SPRS-Conv and SPSS-Conv, respectively. Detailed experimental results and analysis are presented in Sec. 5.

5 Experiments

5.1 Experimental Setting

3D Object Detection Datasets. We evaluate our method on two challenging benchmarks KITTI[13] and nuScenes[4]. KITTI dataset consists of 7,481 samples and 7,518 testing samples, where the training samples are generally divided into the *train* split (3, 712 samples) and the *val* split (3, 769 samples). nuScenes dataset is a large-scale autonomous driving dataset, which contains 1,000 driving sequences in total. It is split into 700 scenes for training, 150 scenes for validation, and 150 scenes for testing. It is collected using a 32-beam synced LIDAR and 6 cameras with the complete 360° environment coverage.

Implementation Detail. We conduct experiments on different detection frameworks, *i.e.,* VoxelR-CNN [10], PV-RCNN [23] and SECOND [30] for KITTI, CenterPoint [36] for nuScenes. We replace the submanifold convolution and regular convolution in sparse CNNs with SPSS-Conv and SPRS-

Method	Car		
	Easy	Moderate	Hard
PointPillars [18]	86.62	76.06	68.91
Point R-CNN [22]	88.88	78.63	77.38
Part-A^2 [24]	89.47	79.47	78.54
STD [1]	89.70	79.80	79.30
SA-SSD [31]	90.15	79.91	78.78
PV-RCNN [23]	89.35	83.69	78.70
VoTr-TSD [20]	89.04	84.04	78.68
Pyramid-PV [19]	89.37	84.38	78.84
Voxel R-CNN [10]	89.41	84.52	78.93
SP-Voxel R-CNN (Ours)	89.69	84.57	78.89

Table 3: Comparison with other methods on KITTI *val* split in AP_{3D} (R11) for Car.

Method	GFLOPs	Car		
		Easy	Moderate	Hard
Second [30]	7.6	88.36	78.75	77.57
SP-Second (Ours)	3.6	88.57	78.64	77.42
PV-RCNN [23]	7.6	89.35	83.69	78.70
SP-PV-RCNN (Ours)	3.6	89.27	83.22	78.87
Voxel R-CNN [10]	7.6	89.41	84.52	78.93
SP-Voxel R-CNN (Ours)	3.6	89.69	84.57	78.89

Table 4: Comparison with base methods on KITTI *val* split in AP_{3D} (R11) for Car. GFLOPs only include the floating-point operations on sparse CNNs.

Method	mAP	NDS
magnitude + skip	58.19	66.08
magnitude + avg pooling	58.03	65.99
conv3x3 + skip	57.29	65.37
conv3x3 + avg pooling	57.57	65.61
linear + avg pooling	57.67	65.54

Table 5: Ablations on sampling and interpolation strategy in SPSS-Conv on nuScenes *val* split.

Method	Car		
	Easy	Moderate	Hard
SPSS-Conv	89.22	84.36	78.83
SPSS-Conv inverse	89.15	79.13	78.47

Table 6: Ablations on the importance of position with high magnitude in SPSS-Conv on KITTI *val* split in AP_{3D} (R11) for Car.

Conv, respectively, other experimental hyperparameters flowing the default settings of the baseline methods [10, 23, 30, 36, 30]. For KITTI, we set pruning ratio in SPSS-Conv and SPRS-Conv as 0.5 and 0.5 respectively, as for nuscenes they are set as 0.3 and 0.5. More implementation details are given in the supplementary file.

5.2 Main Results

nuScenes For the sake of fairness, we choose LIDAR-only methods for comparison, model ensembling and additional augmentation are not included during inference. Our method achieves SOTA performance on both mAP and NDS on nuScenes $test$ split, in Tab. 1. Besides, We apply the method to the base model for a more detailed comparison, as shown in Tab. 2, our method can help the model to preserve the original performance while skipping redundant computation, specifically, SP-CenterPoint's GFLOPs are only 54.5% of the base model. It is worth noting that although our method removes and ignores redundant locations in 3D space, the mAP does not decrease significantly on small object categories, which indicates that the network can learn the importance of spatial locations and distinguish them accordingly.

KITTI. We make comparisons with recent state-of-the-art methods on KITTI val split. As shown in Tab. 3, Our Spatial Pruned Voxel R-CNN achieves competitive results with other methods. To better demonstrate the effectiveness of our method, we validate our method on popular voxel-based detectors, $i.e.$, Second, PVRCNN, and Voxel-RCNN. Tab. 4 shows that with our method, the GFLOPs of the sparse CNN are reduced by more than 50%, and the matching performance can still be obtained on KITTI val split in AP_{3D} in recall 11 positions.

5.3 Ablation Study

SPSS-Conv pruning ratio. The pruning ratio of SPSS-Conv is used to control the proportion of unimportant positions selected in each stage, the higher the proportion, the fewer positions are involved in the calculation. We evaluate on nuScenes val split upon CenterPoint. As shown in Tab. 7 with the pruning ratio increases, there is a relatively obvious and proportional decrease in GFLOPs, in contrast, the performance drop is not so severe, except when the pruning ratio increases to 0.9. This also reveals that 3D Scenes contain spatial redundancy, and when we selectively skip these redundant positions, it will not affect the performance.

Method	Pruning ratio	GFLOPs	mAP	NDS
CenterPoint	-	62.9	58.59	66.20
SPSS-Conv	0.1	56.6	58.58	66.24
	0.3	45.2	58.48	66.11
	0.5	33.5	58.19	66.08
	0.7	21.6	57.80	65.69
	0.9	9.7	55.21	64.23

Table 7: Ablations on pruning ratio in SPSS-Conv on nuScenes *val* split.

Method	ratio per stage	GFLOPs	mAP	NDS
CenterPoint	-	62.9	58.59	66.20
SPRS-Conv	(0.1, 0.1, 0.1)	59.1	58.57	66.01
	(0.3, 0.3, 0.3)	55.4	58.61	66.07
	(0.5, 0.5, 0.5)	47.2	58.59	66.23
	(0.7, 0.7, 0.7)	32.3	58.18	65.93
	(0.9, 0.9, 0.9)	9.9	52.09	61.89

Table 8: Ablations on pruning ratio (per stage) in SPRS-Conv on nuScenes *val* split.

SPRS-Conv pruning ratio. Due to the expansion of the convolution, some non-redundant positions in the 3d sparse CNNs are expanded, increasing the computational burden. Therefore, we explore the influence of the pruning ratio in our designed SPRS-Conv on this problem. We tried different combinations of pruning ratios when down-sampling between different stages. Notably, we found that the performance of the network is not very sensitive to the pruning rate. Tab. 8 shows that even if 70% of the position dilation is suppressed during down-sampling, the impact on performance is not fatal.

Sampling and interpolation strategy in SPSS-Conv. In order to verify the effectiveness of our method, we conduct a combination of several sampling and interpolation methods, where the pruning ratio is limited to 0.5. Results are shown in Tab. 5, when we compare the sampling strategy, it is not difficult to find that the learnable methods consume more computation than the magnitude method with similar or worse accuracy. It makes us think that magnitude is another manifestation of network attention, and regions with high magnitude represent high attention, so we select important regions according to this logic, which conforms to the inherent performance of the network. In addition, for the interpolation operation, compared to the average pooling operation, a simple skip connection can play a good role. A possible explanation is that these predicted unimportant regions are not task-essential, so even using the original values would not have much impact on the results

Importance of position with high magnitude. SPSS-Conv dynamically selects important positions in the space to perform convolution operations based on magnitude. To verify that the location chosen by this mechanism is task-friendly, we exchanged the calculation method of positions with high magnitude and other positions, that is, convolution for low magnitude positions, and skip connection for low magnitude positions. As shown in Tab. 6, a dramatic performance drop occurs when we only take positions with low activation into consideration. This further proves that the value of feature magnitude is positively correlated with importance.

6 Concluding Remarks

In this paper, we investigate the spatial redundancy in sparse CNNs and propose a new simple and efficient convolution operator SPSS-Conv with two variants, i.e., SPSS-ConV and SPRS-Conv. Extensive experiments have proved the following conclusions: 1) Magnitude can serve as a good indicator for sparse CNNs to dynamically identify the informative points in the diverse scenes. 2) The overwhelming background region in the 3D scene results in spatial redundancy, which can be pruned by our method without adversely affecting the performance. 3) Selectively inflating the elements near the region of interest during the down-sampling process not only saves computation but also keeps the necessary information intact. 4) The proposed SPSS-Conv can be effortlessly applied to generic sparse CNNs without specific structural constraints. We hope our work can provide new thoughts for inspiring future research in the community.

References

[1] *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, 2019. IEEE. ISBN 978-1-7281-4803-8. URL https://ieeexplore.ieee.org/xpl/conhome/8972782/proceeding.

[2] Sheng Ao, Qingyong Hu, Bo Yang, Andrew Markham, and Yulan Guo. Spinnet: Learning a general surface descriptor for 3d point cloud registration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11753–11762, 2021.

[3] Prarthana Bhattacharyya and Krzysztof Czarnecki. Deformable PV-RCNN: improving 3d object detection with learned deformations. *CoRR*, abs/2008.08766, 2020. URL https://arxiv.org/abs/2008.08766.

[4] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[5] Shijie Cao, Lingxiao Ma, Wencong Xiao, Chen Zhang, Yunxin Liu, Lintao Zhang, Lanshun Nie, and Zhi Yang. Seernet: Predicting convolutional neural network feature-map sparsity through low-bit quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11216–11225, 2019.

[6] Qi Chen, Lin Sun, Ernest Cheung, and Alan L Yuille. Every view counts: Cross-view consistency in 3d object detection with hybrid-cylindrical-spherical voxelization. *Advances in Neural Information Processing Systems*, 33:21224–21235, 2020.

[7] Qi Chen, Lin Sun, Zhixin Wang, Kui Jia, and Alan L. Yuille. Object as hotspots: An anchor-free 3d object detection approach via firing of hotspots. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXI*, volume 12366 of *Lecture Notes in Computer Science*, pages 68–84. Springer, 2020. doi: 10.1007/978-3-030-58589-1_5. URL https://doi.org/10.1007/978-3-030-58589-1_5.

[8] Yukang Chen, Yanwei Li, Xiangyu Zhang, Jian Sun, and Jiaya Jia. Focal sparse convolutional networks for 3d object detection, 2022. URL https://arxiv.org/abs/2204.12463.

[9] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 764–773. IEEE Computer Society, 2017. doi: 10.1109/ICCV.2017.89. URL https://doi.org/10.1109/ICCV.2017.89.

[10] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. *arXiv preprint arXiv:2012.15712*, 1(2):4, 2020.

[11] Xuanyi Dong, Junshi Huang, Yi Yang, and Shuicheng Yan. More is less: A more complicated network with less inference complexity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[12] Liang Du, Xiaoqing Ye, Xiao Tan, Jianfeng Feng, Zhenbo Xu, Errui Ding, and Shilei Wen. Associate-3ddet: Perceptual-to-conceptual association for 3d point cloud object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13329–13338, 2020.

[13] 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.

[14] Benjamin Graham. Spatially-sparse convolutional neural networks. *CoRR*, abs/1409.6070, 2014. URL http://arxiv.org/abs/1409.6070.

[15] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017.

[16] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 11870–11879. Computer Vision Foundation / IEEE, 2020. doi: 10.1109/CVPR42600.2020.01189. URL `https://openaccess.thecvf.com/content_CVPR_2020/html/He_Structure_Aware_Single-Stage_3D_Object_Detection_From_Point_Cloud_CVPR_2020_paper.html`.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019.

[19] Jiageng Mao, Minzhe Niu, Haoyue Bai, Xiaodan Liang, Hang Xu, and Chunjing Xu. Pyramid R-CNN: towards better performance and adaptability for 3d object detection. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 2703–2712. IEEE, 2021. doi: 10.1109/ICCV48922.2021.00272. URL `https://doi.org/10.1109/ICCV48922.2021.00272`.

[20] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 3144–3153. IEEE, 2021. doi: 10.1109/ICCV48922.2021.00315. URL `https://doi.org/10.1109/ICCV48922.2021.00315`.

[21] Mengye Ren, Andrei Pokrovsky, Bin Yang, and Raquel Urtasun. Sbnet: Sparse blocks network for fast inference. *CoRR*, abs/1801.02108, 2018. URL `http://arxiv.org/abs/1801.02108`.

[22] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779, 2019.

[23] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020.

[24] 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. *IEEE transactions on pattern analysis and machine intelligence*, 43(8):2647–2664, 2020.

[25] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection. *arXiv preprint arXiv:2102.00463*, 2021.

[26] 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 *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 6410–6419. IEEE, 2019. doi: 10.1109/ICCV.2019.00651. URL `https://doi.org/10.1109/ICCV.2019.00651`.

[27] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 2317–2326. Computer Vision Foundation / IEEE, 2020. doi: 10.1109/CVPR42600.2020.00239. URL `https://openaccess.thecvf.com/content_CVPR_2020/html/Verelst_Dynamic_Convolutions_Exploiting_Spatial_Sparsity_for_Faster_Inference_CVPR_2020_paper.html`.

[28] Zhenda Xie, Zheng Zhang, Xizhou Zhu, Gao Huang, and Stephen Lin. Spatially adaptive inference with stochastic feature sampling and interpolation. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I*, volume 12346 of *Lecture Notes in Computer Science*, pages 531–548. Springer, 2020. doi: 10.1007/978-3-030-58452-8_31. URL `https://doi.org/10.1007/978-3-030-58452-8_31`.

[29] Shaoqing Xu, Dingfu Zhou, Jin Fang, Junbo Yin, Bin Zhou, and Liangjun Zhang. Fusion-painting: Multimodal fusion with adaptive attention for 3d object detection. In *24th IEEE International Intelligent Transportation Systems Conference, ITSC 2021, Indianapolis, IN, USA, September 19-22, 2021*, pages 3047–3054. IEEE, 2021. doi: 10.1109/ITSC48978.2021.9564951. URL `https://doi.org/10.1109/ITSC48978.2021.9564951`.

[30] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018.

[31] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048, 2020.

[32] Zhendong Yang, Zhe Li, Xiaohu Jiang, Yuan Gong, Zehuan Yuan, Danpei Zhao, and Chun Yuan. Focal and global knowledge distillation for detectors. *arXiv preprint arXiv:2111.11837*, 2021.

[33] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. Hvnet: Hybrid voxel network for lidar based 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1631–1640, 2020.

[34] Hongwei Yi, Shaoshuai Shi, Mingyu Ding, Jiankai Sun, Kui Xu, Hui Zhou, Zhe Wang, Sheng Li, and Guoping Wang. Segvoxelnet: Exploring semantic context and depth-aware features for 3d vehicle detection from point cloud. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2274–2280. IEEE, 2020.

[35] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Multimodal virtual point 3d detection. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 16494–16507, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/895daa408f494ad58006c47a30f51c1f-Abstract.html`.

[36] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793, 2021.

[37] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *CoRR*, abs/1612.03928, 2016. URL `http://arxiv.org/abs/1612.03928`.

[38] Na Zhao, Tat-Seng Chua, and Gim Hee Lee. Sess: Self-ensembling semi-supervised 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11079–11087, 2020.

[39] Wu Zheng, Weiliang Tang, Sijin Chen, Li Jiang, and Chi-Wing Fu. Cia-ssd: Confident iou-aware single-stage object detector from point cloud. *arXiv preprint arXiv:2012.03015*, 2020.

[40] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018.

[41] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv preprint arXiv:1908.09492*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code will be released upon acceptance.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]