Focal Sparse Convolutional Networks for 3D Object Detection

Yukang Chen¹, Yanwei Li¹, Xiangyu Zhang², Jian Sun², Jiaya Jia¹,³
¹The Chinese University of Hong Kong ²MEGVII Technology ³SmartMore

Abstract

Non-uniformed 3D sparse data, e.g., point clouds or voxels in different spatial positions, make contribution to the task of 3D object detection in different ways. Existing basic components in sparse convolutional networks (Sparse CNNs) process all sparse data, regardless of regular or submanifold sparse convolution. In this paper, we introduce two new modules to enhance the capability of Sparse CNNs, both are based on making feature sparsity learnable with position-wise importance prediction. They are focal sparse convolution (Focals Conv) and its multi-modal variant of focal sparse convolution with fusion, or Focals Conv-F for short. The new modules can readily substitute their plain counterparts in existing Sparse CNNs and be jointly trained in an end-to-end fashion. For the first time, we show that spatially learnable sparsity in sparse convolution is essential for sophisticated 3D object detection. Extensive experiments on the KITTI, nuScenes and Waymo benchmarks validate the effectiveness of our approach. Without bells and whistles, our results outperform all existing single-model entries on the nuScenes test benchmark. Code and models are at github.com/dvlab-research/FocalsConv.

1. Introduction

A key challenge in 3D object detection is to learn effective representations from the unstructured and sparse 3D geometric data such as point clouds. In general, there are two ways for this job. The first is to process point clouds [37,51] directly, based on PointNet++ [33] networks. However, the neighbour sampling and grouping operations are time-consuming. This makes it improper for large-scale autonomous driving scenes that require real-time efficiency. The second is to convert point clouds into voxelizations and apply 3D sparse convolutional neural networks (Sparse CNNs) for feature extraction [11,36]. 3D Sparse CNNs resemble 2D CNNs in structures, including several feature stages and down-sampling operations. They typically consist of regular and submanifold sparse convolutions [15].

Although regular and submanifold sparse convolutions have been widely used, they have respective limitations. Regular sparse convolution dilates all sparse features. It is not optimal to handle non-uniform data with uniform treatment. In terms of sparsity, upon the distance to LiDAR sensors, objects present large sparsity variance. In terms of importance, the contribution of features varies with different locations for 3D object detection, e.g., foreground or background. Although 3D object detection is achieved [11,36,37,53], state-of-the-art methods still rely on RoI (region-of-interest) feature extraction. It corresponds to the idea that we should shoot arrows at the target in the feature extraction of 3D detectors.

In this paper, we propose a general format of sparse convolution by relaxing the conceptual difference between regular and submanifold ones. We introduce two new modules that improve the representation capacity of Sparse CNNs for 3D object detection. The first is focal sparse convolution (Focals Conv). It predicts cubic importance maps for the output pattern of convolutions. Features predicted as important ones are dilated into a deformable output shape,
as shown in Fig 1. The importance is learned via an additional convolutional layer, dynamically conditioned on the input features. This module increases the ratio of valuable information among all features. The second is its multimodal improved version of Focal sparse Convolution with Fusion (named as Focals Con-F). Upon the LIDAR-only Focals Conv, we enhance importance prediction with RGB features fused, as image features typically contain rich appearance information and large receptive fields.

The proposed modules are novel in two aspects. First, Focals Conv presents a dynamic mechanism for learning spatial sparsity of features. It makes the learning process concentrated on the more valuable foreground data. With the down-sampling operations, valuable information increases in stages. Meanwhile, the large amount of background voxels are removed. Fig. 2 illustrates the learnable feature sparsity, including the common, crowded, and remote objects, where Focals Conv enriches the learned voxel features on the foreground without redundant voxels added in other areas. Second, both modules are lightweight. The importance prediction involves small overhead parameters and computation, as measured in Tab. 1. The RGB feature extraction of Focals Con-F involves only several layers, instead of heavy 2D detection or segmentation models [43].

The proposed modules of Focals Conv and Focals Con-F can readily replace their original counterparts in sparse CNNs. To demonstrate the effectiveness, we build the backbone networks on existing 3D object detection frameworks [11,36,53]. Our method enables non-trivial enhancement with small model complexity overhead on both the KITTI [14] and nuScenes [2] benchmarks. These results manifest that learnable sparsity with focal points is essential. Without bells and whistles, our approach outperforms state-of-the-art ones on the nuScenes test split [2].

Convolutional dynamic mechanism adapts the operations conditioned on input data, e.g., deformable convolutions [10, 64] and dynamic convolutions [7, 49]. The key difference is that our approach makes use of the intrinsic sparsity of data. It promotes feature learning to be concentrated on more valuable information. We deem the non-uniform property as a great benefit. We discuss the relations and differences to previous literature in Sec. 2.

2. Related Work

2.1. Convolutional Dynamic Mechanism

Dynamic mechanisms have been widely studied in CNNs, due to their advantages of high accuracy and easy adaption in scenarios. We discuss two kinds of related methods, i.e., kernel shape adaption [10, 41, 64], and input attention mask [34, 42, 45].

Kernel shape adaption. Kernel shape adaption methods [8, 10, 13, 64] adapt the effective receptive fields of networks. Deformable convolution [10] predicts offsets for feature sampling. Its variant [64] introduces an additional attention mask to modulate features. For 3D feature learning, KPConv [41] learns local offsets for kernel points. MinkowskiNet [8] generalizes sparse convolution to arbitrary kernel shape. Overall, these methods modify the input feature sampling process.

Deformable PV-RCNN [1] applies offset prediction for feature sampling in 3D object detection. In contrast, focal sparse convolution improves the output feature spatial sparsity and makes it learned, helpful for 3D object detection.

Attention mask on input. Methods of [39, 42, 45, 50] seek spatial-wise sparsity for efficient inference. These methods receive dense images and prune unimportant pixels based on attention masks. These methods aim to sparsify dense data while we make use of intrinsic data sparsity. Although SBNNet [34] also utilizes the sparse property, it limits application to 2D BEV (bird-eye-views) images, and shares the
2.2. 3D Object Detection

**LIDAR-only detectors.** 3D object detection frameworks usually resemble 2D detectors, e.g., the R-CNN family [11, 28, 36, 37] and the SSD family [17, 51, 60, 61]. The main difference on 2D detectors lies in input encoders. VoxelNet [62] encodes voxel features using PointNet [32] and applies a RPN (region proposal network) [35]. SECOND [48] uses accelerated sparse convolutions and improves efficiency from VoxelNet [62]. VoTr [29] applies transformer architectures to voxels. Various detectors [11, 36, 53] have been presented based on feature encoders. We validate the proposed approach on backbones of frameworks of [11, 36, 53] on multiple datasets [2, 14, 40].

**Completion-based detectors.** Completion-based methods [16, 23, 46, 58] form another line of efforts in enriching foreground information. We focus on feature learning instead of point completion. PC-RGNN [58] has a point completion module by a graph neural network. SIENet [23] builds upon PCN [56] for point completion in a two-stage framework. The completion process relies on the prior generated proposals. GSDN [16] expands all features first through transposed convolutions and then by pruning. SPG [46] designs a semantic point generation module for domain adaption 3D object detection. It is applied during data preprocessing, complicating the detection pipelines.

**Multi modal fusion.** Multi-modal fusion methods [19, 25, 55] use more information than LIDAR-only ones. The KITTI [14] benchmark had been dominated by LIDAR-only methods until PointPainting [43] was proposed. It decorates raw point clouds with the corresponding image segmentation scores. PointAugmenting [44] further replaces the segmentation model with an 2D object detection one [12]. They are both decoration-based methods, which require image feature extraction on off-the-shelf 2D networks, before feeding into 3D detectors. Although promising results are achieved by these methods, the overall inference pipelines are complicated. Our multi-modal focal sparse convolution differs from the above methods in two aspects. First, we only require several jointly trained layers for image feature extraction, rather than the heavy segmentation or detection models. Second, we only strengthen the predicted important features, instead of the uniform decoration [43, 44] for all LIDAR features.

3. Focal Sparse Convolutional Networks

In this section, we first review the formulation of sparse convolution in Sec. 3.1. Then, the proposed focal sparse convolution and its multi-modal extension will be elaborated in Sec. 3.2 and Sec. 3.3. We finally introduce the resulting focal sparse convolutional networks in Sec. 3.4.

3.1. Review of Sparse Convolution

Given an input feature $x_p$ with number of $c_{in}$ channels at position $p$ in the $d$ dimensional spatial space, we process this feature by a convolution with kernel weights $w \in \mathbb{R}^{K^d \times c_{in} \times c_{out}}$. For example, in the 3D coordinate space, $w$ contains $c_{in} \times c_{out}$ spatial kernels with size 3 and $|K^d| = 3^d$. The convolution process is represented as

$$y_p = \sum_{k \in K^d} w_k \cdot x_{\bar{p}_k}$$

where $k$ enumerates all discrete locations in the kernel space $K^d$, $\bar{p}_k = p + k$ is the corresponding location around center $p$, where $k$ is an offset distance from $p$.

This formulation accommodates most types of convolutions with simple modifications. When $p \in \mathbb{Z}$, the common convolution for dense input data is yielded. When $\bar{p}_k$ is added with a learned offset $\Delta \bar{p}_k$, it includes the kernel shape adaption methods, e.g., deformable convolutions [10, 64]. Further, if $W$ equals to a weighted sum $\sum \alpha_i W_i$, it generalizes to weight attention, e.g., dynamic convolution [7, 49]. Finally, when attention masks are multiplied to the input feature map $x$, this formulation makes input attention mask methods [34, 45].

For sparse input data, the feature position $p$ does not belong to the dense discrete space $\mathbb{Z}$. The input and output feature spatial space is relaxed to $P_{in}$ and $P_{out}$, respectively.
Figure 3. Framework of focal sparse convolution and its multi-modal extension. An additional branch predicts a cubic importance map for each input sparse feature, which determines the output feature positions. In the multi-modal version, the additional branch takes fusion of LIDAR and RGB features for better prediction. Output sparse features predicted as important are also fused with the RGB features.

The formulation is converted to

$$y_{p \in P_{\text{out}}} = \sum_{k \in K^d(p, P_{\text{in}})} w_k \cdot x_{\tilde{j}_k},$$

where $K^d(p, P_{\text{in}})$ is a subset of $K^d$, leaving out the empty position. It is conditioned on the position $p$ and input feature space $P_{\text{in}}$ as

$$K^d(p, P_{\text{in}}) = \{k | p + k \in P_{\text{in}}, k \in K^d\}.$$  \hspace{1cm} (3)

If $P_{\text{out}}$ includes a union of all dilated positions around $P_{\text{in}}$ within $K^d$ neighbours, this process is formulated as

$$P_{\text{out}} = \bigcup_{p \in P_{\text{in}}} P(p, K^d),$$

where

$$P(p, K^d) = \{p + k | k \in K^d\}.$$  \hspace{1cm} (5)

On this condition, the formulation becomes regular sparse convolution. It acts at all positions where any voxels exist in its kernel space. It does not skip any information gathering in the total spatial space.

This strategy involves two drawbacks. (i) It introduces considerable computation cost. The number of sparse features is doubled or even tripled, increasing burden for following layers. (ii) We empirically find that continuously increasing the number of sparse features may harm 3D object detection (Tab. 2). Crowded and unpromising candidate features may blur the valuable information. It degrades foreground features and further declines the feature discrimination capacity of 3D object detectors.

When $P_{\text{in}} = P_{\text{out}}$, submanifold sparse convolution [15] is yielded. It happens only when the kernel centers locate at the input, restricting the active positions to input sets. This setting avoids the computation burden, but abandons necessary information flow between disconnected features. Note that the flow is common in the irregular point cloud data. Thus, effective receptive field sizes are constrained by the feature disconnection, which degrade the model capability.

3.2. Focal Sparse Convolution

Regardless of regular or submanifold sparse convolution, output positions $P_{\text{out}}$ are static across all $p \in P_{\text{in}}$, which is undesirable. In contrast, we perform adaptive determination of sparsity or receptive field sizes in a fine-grained manner. We relax output positions $P_{\text{out}}$ to be dynamically determined by the sparse features. We illustrate this proposed process in Fig. 3 (via solid lines).

In our formulation, output positions $P_{\text{out}}$ generalize to a union of all important positions with their dilated area and other unimportant positions. The dilated areas are deformable and dynamic to input positions. Eq. (5) becomes

$$P_{\text{out}} = \left( \bigcup_{p \in P_{\text{im}}} P(p, K^d_{\text{im}}(p)) \right) \cup P_{\text{in}/\text{im}}.$$  \hspace{1cm} (6)

We factorize this process into three steps: (i) cubic importance prediction, (ii) important input selection, and (iii) dynamic output shape generation.

**Cubic importance prediction.** A cubic importance map $I^p$ involves importance for candidate output features around the input feature at position $p$. Each cubic importance map shares the same shape $K^d$ with the main processing convolution kernel weight, e.g., $k^3 = 3 \times 3 \times 3$ with the kernel size $3$. It is predicted by an additional submanifold sparse convolution with a sigmoid function. The latter steps depend on the predicted cubic importance maps.

**Important input selection.** In Eq. (5), $P_{\text{im}}$ is a subset of $P_{\text{in}}$. It contains the positions of relatively important input features. We select $P_{\text{im}}$ as

$$P_{\text{im}} = \{p | I^p_0 \geq \tau, p \in P_{\text{in}}\},$$  \hspace{1cm} (7)
where $I_0^p$ is the center of the cubic importance map at position $p$. And $\tau$ is a pre-defined threshold (Tab. 3 and 6). Our formulation becomes the regular or submanifold sparse convolution when $\tau$ is 0 or 1 respectively.

**Dynamic output shape generation.** Features in $P_{im}$ are dilated to a dynamic shape. The output around $p$ is determined by the dynamic output shape $K_{im}^d(p)$. Note that our deformable output shapes are pruned inside the original dilation without offsets. It is computed similarly to Eq. (7) as

$$K_{im}^d(p) = \{ k | p + k \in P_{im}, I_k^p \geq \tau, k \in K^d \}. \quad (8)$$

We analyze the dynamic output shape in Tab. 2. For the remaining unimportant features, their output positions are fixed as input, i.e., submanifold. We found that directly removing them or using a fully dynamic manner without preserving them makes the training process unstable.

**Supervision manners.** In 3D object detection, we have a prior knowledge that foreground objects are more valuable information. Based on this prior, we apply focal loss [26] as an objective loss function to supervise the importance prediction. We construct the objective targets for the centers of feature voxels inside 3D ground-truth boxes. We keep its loss weight as 1 for the generality of our modules.

Additional supervision comes from multiplying the predicted cubic importance maps to output features as attention. It makes the importance prediction branch differentiable naturally. It shares motivation with the kernel weight sparsification methods [27] in the area of model compression. We empirically show that this attention manner benefits the performance for minor classes, e.g., Pedestrian and Cyclist on KITTI (investigated in Tab. 4).

### 3.3. Fusion Focal Sparse Convolution

We provide a multi-modal version of focal sparse convolution, as illustrated in Fig. 3 (via dashed lines). This extension is conceptually simple but effective. We extract RGB features from images and align LIDAR features to them. The extracted features are fused to input and important output sparse features in focal sparse convolution.

**Feature extraction.** The fusion module is lightweight. It contains a conv-bn-relu layer and a max-pooling layer. It down-samples the input image to 1/4 resolutions. It is followed by 3 conv-bn-relu layers with residual connection [18]. The channel number is then reduced to be consistent with that of sparse features, with an MLP layer. This facilitates a simple summation of multi-modal features.

**Feature alignment.** A common issue during fusion is misalignment in the 3D-to-2D projection. Point cloud data is commonly processed by transformation and augmentation. Transformations include flip, re-scale, rotation, translation. The typical augmentation is ground-truth sampling [48], copying paste objects from other scenes. For these invertible transformations, we reverse the coordinates of sparse features with the recorded transformation parameters [44, 57]. For ground-truth sampling, we copy the corresponding 2D objects onto images. Rather than using an additional segmentation model or mask annotations [57], we directly crop objects in bounding boxes for simplification.

**Fusion manners.** The aligned RGB features are directly fused to sparse features in summation, as they share the same channel numbers. Although other fusion methods, e.g., concatenation or cross-attention, can be used, we choose the most concise summation for efficiency. The aligned RGB features are fused with sparse features twice in this module. It is first fused to input features for cubic importance prediction. Then we fuse RGB features only to important output sparse features, i.e., the first part in Eq. (5), instead of all of them (investigated in Tab. 10).

Overall, the multi-modal layers are lightweight in terms of parameters and fusion strategies. They are jointly trained with detectors. It provides an efficient and economical solution for the fusion module in 3D object detection.

### 3.4. Focal Sparse Convolutional Networks

Both focal sparse convolution and its multi-modal extension can readily replace their counterparts in the backbone networks of 3D detectors. During training, we do not use any special initialization or learning rate settings for the introduced modules. The importance prediction branch is trained via back-propagation through the attention multiplication and objective loss function as introduced in Sec. 3.2.

The backbone networks in 3D object detectors [11, 36, 53] typically consist of one stem layer and 4 stages. Each stage, except the first one, includes a regular sparse convolution with down-sampling and two submanifold blocks. In the first stage, there are one [11, 36] or two [53] sparse convolutional layers. By default, each sparse convolution is followed by batch normalization [20] and ReLU activation.

We validate focal sparse convolution on the backbone networks of existing 3D detectors [11, 36, 53]. We directly apply focal sparse convolution at the last layer of certain stages. We analyze the stages for using our focal sparse convolution in experiments (ablated in Tab. 5 and 10).

### 4. Experiments

We conduct ablations and comparisons for Focal Conv and its multi-modal variant. More experiments, such as results on Waymo [40], are in the supplementary material.

#### 4.1. Setup and Implementation

**KITTI.** The KITTI dataset [14] consists of 7,481 samples and 7,518 testing samples. The training samples are split into a train set with 3,717 samples and a val set with 3,769 samples. The KITTI dataset consists of 7,481 samples and 7,518 testing samples. The training samples are split into a train set with 3,717 samples and a val set with 3,769 samples.

5422
samples. Models are commonly evaluated in terms of the mean Average Precision (mAP) metric. mAP is calculated with recall 40 positions (R40). We perform ablation studies with AP3D (R40) on the val split. We conduct main comparisons with AP3D (R40) on test split and AP3D (R11) on the val split. For the optional multi-modal settings, RGB features are extracted from single front-view for fusion.

**nuScenes**. The nuScenes [2] is a large-scale 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. In evaluation, the main metrics are mAP and nuScenes detection score (NDS). In terms of multi-modal experiments, we use images of 6 views for fusion. For ablation study, models are trained on $\frac{1}{4}$ training data and evaluated on the entire validation set, i.e., nuScenes $\frac{1}{4}$ split.

**Implementation details**. In experiments, we validate our modules on state-of-the-art frameworks of PV-RCNN [36], Voxel R-CNN [11] on KITTI [14], and CenterPoint [53] on nuScenes [2]. In LIDAR-only experiments, we apply Focals Conv in the first three stages of backbone networks. In multi-modal cases, we apply Focals Conv-F only in the first stage of the backbone network, for affordable memory and inference cost. We set the importance threshold $\tau$ to 0.5. We keep other settings intact. More experimental details are provided in the supplementary material.

### 4.2. Ablation Studies

**Improvements on KITTI**. We first evaluate our methods over PV-RCNN [36] in Tab. 1, as it is a high-performance, multi-class, and open-sourced framework. In Tab. 1, the 1st and 2nd lines show the reported results [36] and results tested from the released model. We take the latter as the baseline. Focal S-Conv and Focals Conv-F achieve non-trivial improvement over this strong baseline.

**Dynamic output shape**. In Focals Conv, the output shape from every single voxel is dynamically determined by the predicted importance maps. We ablate this by fixing output shapes as regular dilation, without any other change. Tab. 2 shows that dilating all sparse features is harmful. It dramatically increases the number of unpromising voxel features.

**Importance sampling**. Focals Conv selects sparse features that need dilation with predicted importance. To ablate this module, we replace the importance selection (the important input selection step) with a random sample in Tab. 3 without other changes. It shows that large performance drop occurs without the guidance of importance. This validates that the importance prediction is necessary.

**Supervision setting**. The additional branch in Focals Conv is supervised by both attention multiplication and the objective loss. We ablate them in Tab. 4. Only using objective loss supervision is enough to ensure performance on Car. However, its performance on minor classes, Ped. and Cyc., is not optimal. Attention multiplication is beneficial.

### Table 1. Improvements on PV-RCNN in AP3D (R40) on KITTI val.

| Method          | #Params | Runtime | Easy | Mod. | Hard |
|-----------------|---------|---------|------|------|------|
| PV-RCNN [36]    | –       | –       | 92.57| 84.83| 82.69|
| PV-RCNN o       | 13.16M  | 103ms   | 92.10| 84.36| 82.48|
| Focals Conv-F   | 13.44M  | 112ms   | 92.32| 85.19| 82.62|
| Focals Conv     | 13.70M  | 125ms   | 92.26| **85.32**| 82.95|

o These results are evaluated on the official released model.

### Table 2. Ablations on dynamic shape in AP3D (R40) on KITTI val.

| Method          | Dynamic shape | Car Easy | Mod. | Hard | Ped. Mod. | Cyc. Mod. |
|-----------------|---------------|---------|------|------|-----------|-----------|
| Baseline        | –             | 92.10   | 84.36| 82.48| 54.49     | 70.38     |
| Focals Conv     | ✓             | 91.10   | 84.02| 82.22| 57.62     | 69.82     |

### Table 3. Ablations on input selection in AP3D (R40) on KITTI val.

| Method          | Important selection | Car Easy | Mod. | Hard | Ped. Mod. | Cyc. Mod. |
|-----------------|---------------------|---------|------|------|-----------|-----------|
| Baseline        | –                   | 92.10   | 84.36| 82.48| 54.49     | 70.38     |
| Focals Conv     | ✓                   | 91.36   | 82.77| 82.12| 57.86     | 71.77     |

### Table 4. Ablations on supervisions in AP3D (R40) on KITTI val.

| Method          | Supervision | Car Easy | Mod. | Hard | Ped. Mod. | Cyc. Mod. |
|-----------------|-------------|---------|------|------|-----------|-----------|
| Baseline        | –           | 92.10   | 84.36| 82.48| 54.49     | 70.38     |
| Focals Conv     | Attention   | 91.81   | 84.49| 82.31| 60.64     | 72.93     |
|                 | Obj. loss   | 92.39   | 85.05| 82.62| 59.27     | 71.46     |
|                 | Both        | 92.32   | **85.19**| 82.62| **61.61**| **72.76**|

### Table 5. Ablations on use stages in AP3D (R40) on KITTI val.

| Method          | Stages | Car Easy | Mod. | Hard | Ped. Mod. | Cyc. Mod. |
|-----------------|--------|---------|------|------|-----------|-----------|
| Baseline        | –      | 92.10   | 84.36| 82.48| 54.49     | 70.38     |
| Focals Conv     | (1)    | 92.19   | 84.83| 82.43| 60.56     | 72.29     |
|                 | (1, 2) | 91.95   | 84.95| 82.67| 60.17     | 72.74     |
|                 | (1, 2, 3) | 92.32 | **85.19**| 82.62| **61.61**| **72.76**|
|                 | (1, 2, 3, 4) | 91.96 | 84.42| 82.31| 60.33     | 72.53     |

### Table 6. Ablations on the importance threshold $\tau$ on KITTI val.

| Importance Threshold $\tau$ | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
|-----------------------------|-----|-----|-----|-----|-----|
| AP3D (R40) - Car            | 84.97| 85.09| 85.19| 84.96| 84.68|
to Ped. and Cyc. We assume that minor classes cannot get balanced supervision from the objective loss like the long-tailed distribution. In contrast, attention multiplication is object-agnostic, relaxing the imbalance to some degree.

**Stages for using focal sparse convolution.** Tab. 5 shows results of using Focals Conv in different numbers of stages. (1) Applying Focals Conv in the first stage, which already obtains clear improvement. The performance enhances as the used stage increases until all stages are involved. Since Focals Conv adjusts output sparsity, it is reasonable to be used in early stages that make effects on subsequent feature learning. The spatial feature space in the last stage is down-sampled to a very limited size, which might not be large enough for sparsity adaptation. Empirically, usage in the last layer of the first three stages is the best choice. It is thus used as the default setting in our experiments.

**Importance threshold.** We ablate the importance threshold \( \tau \) used in Focals Conv in Tab. 6. We run experiments with this value ranging from 0.1 to 0.9 and interval 0.2, without other change of settings. The accuracy \( \text{AP}_{3D} \) (R40) on Car serves as the metric in this ablation. The performance is stable as the threshold value \( \tau \) varies.

**Improvements over multi-modal baseline on nuScenes.** We evaluate our multi-modal Focals Conv on the nuScenes [2] \( \frac{1}{4} \) dataset. More improvement is presented in Tab. 9. We build a multi-modal CenterPoint baseline by fusing image features to the same fusion layer used in our methods, with the same fusion and feature extraction layers. This multi-modal CenterPoint enhances the LIDAR-only baseline from 56.1% to 59.0% mAP. Focals Conv-F improves to 61.7% mAP on this strong baseline.

**Use stages and fusion scope for Focals Conv-F.** We ablate the usage stages and fusion scope for Focals Conv-F in Tab. 10. Fusion scope is the scope of sparse features to fuse with RGB features at the output of Focals Conv-F. It shows that fusion in the early stages is beneficial, and becomes adverse in the last two stages. **Imp.** means only fusing onto important output features (judged by importance maps). When fusing in the first stage, it is better to fuse on important features, instead of all of them, making representation discriminative.

**Model complexity and runtime.** We report the model complexity and runtime comparisons in Tab. 1 and 9. The runtime is evaluated on an NVIDIA 2080Ti GPU. Focals Conv and its multi-modal variant only add a small overhead to model parameters and computation, on KITTI [14]. This
| Method               | Fusion | mAP  | NDS  | Car     | Truck   | Bus     | Trailer | C.V.  | Ped   | Mot   | Byc   | C.T.  | Bar   |
|---------------------|--------|------|------|---------|---------|---------|---------|-------|-------|-------|-------|-------|-------|
| PointPillars [22]   | ✗      | 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 [51]          | ✗      | 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 [63]           | ✗      | 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 [5]      | ✗      | 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 [4]          | ✗      | 58.2 | 66.6 | 82.6    | 49.5    | 59.4    | 51.1    | 16.2  | 83.0  | 61.8  | 38.8  | 69.7  | 69.7  |
| PointPainting       | ✓      | 46.4 | 58.1 | 77.9    | 35.8    | 36.2    | 37.3    | 15.8  | 73.3  | 41.5  | 24.1  | 62.4  | 60.2  |
| 3DCVF [55]          | ✓      | 52.7 | 62.3 | 83.0    | 45.0    | 48.8    | 49.6    | 15.9  | 74.2  | 51.2  | 30.4  | 62.9  | 65.9  |
| FusionPainting [47] | ✓      | 66.3 | 70.4 | 86.3    | 58.5    | 66.8    | 59.4    | 27.7  | 87.5  | 71.2  | 51.7  | 84.2  | 70.2  |
| MVF [54]            | ✓      | 66.4 | 70.5 | 86.8    | 58.5    | 67.4    | 57.3    | 26.1  | 89.1  | 70.0  | 49.3  | 85.0  | 74.8  |
| PointAugmenting     | ✓      | 66.8 | 71.0 | 87.5    | 57.3    | 65.2    | 60.7    | 28.0  | 87.9  | 74.3  | 50.9  | 83.6  | 72.6  |
| PointPainting       | ✗      | 58.0 | 65.5 | 84.6    | 51.0    | 60.2    | 53.2    | 17.5  | 83.4  | 53.7  | 28.7  | 76.7  | 70.9  |
| CenterPoint [53]    | ✗      | 60.3 | 67.3 | 85.2    | 53.5    | 63.6    | 56.0    | 20.0  | 84.6  | 59.5  | 30.7  | 78.4  | 71.1  |
| CenterPoint v2       | ✓      | 67.1 | 71.4 | 87.0    | 57.3    | 69.3    | 60.4    | 28.8  | 90.4  | 71.3  | 49.0  | 86.8  | 71.0  |
| Focals Conv         | ✗      | 63.8 | 70.0 | 86.7    | 56.3    | 67.7    | 59.5    | 23.8  | 87.5  | 64.5  | 36.3  | 81.4  | 74.1  |
| Focals Conv-F       | ✓      | 67.8 | 71.8 | 86.5    | 57.5    | 68.7    | 60.6    | 31.2  | 87.3  | 76.4  | 52.5  | 84.6  | 72.3  |
| Focals Conv-F ⃦      | ✓      | 68.9 | 72.8 | 86.9    | 59.3    | 68.7    | 62.5    | 32.8  | 87.8  | 78.5  | 53.9  | 85.5  | 72.8  |
| Focals Conv-F ‡     | ✓      | 70.1 | 73.6 | 87.5    | 60.0    | 69.9    | 64.0    | 32.6  | 89.0  | 81.1  | 59.2  | 85.5  | 71.8  |

† Flip testing. ‡ Flip and rotation testing. ⃦ CenterPoint v2 includes PointPainting with Cascade R-CNN [3] and model-ensembling.

indicates that the performance improvement comes from the model capacity of sparsity learning, instead of increasing model sizes. On nuScenes [2], the overall runtime rises from 93 ms to 159 ms. But parameters are still limited. It is a common limitation in multi-view fusion methods. The multi-modal baseline also requires 145 ms. The reason is that there are 6-view images to process per frame.

### 4.3. Main Results

**KITTI.** We compare our Focals Conv modules upon Voxel R-CNN [11] with previous state-of-the-art methods on both the KITTI test and val split. In Tab. 7, we compare with both LIDAR-only and multi-modal methods. The original Voxel R-CNN [11] is comparable to PV-RCNN [36] and is inferior to Pyramid-PV [28] and VoTr-TSD [29]. Focals Conv improves it to surpass these two new methods. Using Focals Conv-F, the multi-modal Voxel R-CNN achieves 82.28% AP$_{3D}$ on the KITTI test split. Tab. 8 shows comparisons on KITTI val split in AP$_{3D}$ in recall 11 positions. Focals Conv and Focals Conv-F enhance this leading result to 84.93% and 85.22% respectively in Car class.

**nuScenes.** On the nuScenes dataset, we evaluate our models on the test server and compare them with both LIDAR-only and multi-modal methods, as in Tab. 11. Focals Conv improves CenterPoint [53] by a large margin to 63.8% mAP. Multi-modal methods present much better performance than LIDAR-only methods on the nuScenes dataset. CenterPoint v2* includes PointPainting [43], Cascade R-CNN [3] instance segmentation models pre-trained on nuImages, and five-model ensembling. As the testing augmentations are not unified or stated in previous methods, we provide two results of our final model. Focals Conv-F achieves 67.8% mAP and 71.8% mAP without any ensembling or testing augmentation. Focals Conv-F ‡ further achieves 70.1% mAP and 73.6% NDS with test-time augmentations [53]. Both results are better than previously published ones.

### 5. Conclusion and Discussion

This paper presents a focal sparse convolution and a multi-modal extension, which are simple and effective. They are end-to-end solutions for LIDAR-only and multi-modal 3D object detection. For the first time, we show that the learned sparsity with focal points is essential for 3D object detectors. Notably, focal and fusion sparse CNNs achieve leading performance on the large-scale nuScenes.

**Limitations.** In the multi-modal 3D detection that requires multiple views, e.g., 6 high-resolution images per frame in nuScenes [2], computation cost increases, although the image branch is already largely simplified.

**Boarder Impacts.** The proposed method relies on the sparsity of data distribution. It might reflect biases in data collection, including the ones of negative societal impacts.

**Acknowledgements.** This work is in part supported by The National Key Research and Development Program of China (No. 2017YFA0700800) and Beijing Academy of Artificial Intelligence (BAAI).
References

[1] Prarthana Bhattacharyya and Krzysztof Czarnecki. Deformable PV-RCNN: improving 3d object detection with learned deformations. In *ECCV Workshop*, 2020. 2, 7

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Lleon, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, pages 11618–11628, 2020. 2, 3, 6, 7, 8

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: delving into high quality object detection. In *CVPR*, pages 6154–6162, 2018. 8

[4] 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. In *NeurIPS*, 2020. 8

[5] 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 *ECCV*, volume 12366, pages 68–84, 2020. 8

[6] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, pages 6526–6534, 2017. 7

[7] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *CVPR*, pages 11027–11036, 2020. 2, 3

[8] Christopher B. Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, pages 3075–3084, 2019. 2

[9] Ruihang Chu, Yukang Chen, Tao Kong, Lu Qi, and Lei Li. Icm-3d: Instantiated category modeling for 3d instance segmentation. *IEEE RAL*, 7(1):57–64, 2021. 3

[10] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, pages 764–773, 2017. 2, 3

[11] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel R-CNN: towards high performance voxel-based 3d object detection. In *AAAI*, pages 1201–1209, 2021. 1, 2, 3, 5, 6, 7, 8

[12] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection in *ICCV*, pages 6568–6577, 2019. 3

[13] Hang Gao, Xizhou Zhu, Stephen Lin, and Jifeng Dai. Deformable kernels: Adapting effective receptive fields for object deformation. In *ICLR*, 2020. 2

[14] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *Int. J. Robotics Res.*, 32(11):1231–1237, 2013. 2, 3, 5, 6, 7

[15] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, pages 9224–9232, 2018. 1, 4

[16] JunYoung Gwak, Christopher B. Choy, and Silvio Savarese. Generative sparse detection networks for 3d single-shot object detection. In *ECCV*, volume 12349, pages 297–313, 2020. 3

[17] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *CVPR*, pages 11870–11879, 2020. 3, 7

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5

[19] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. Epnet: Enhancing point features with image semantics for 3d object detection. In *ECCV*, volume 12360, pages 35–52, 2020. 3, 7

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37, pages 448–456, 2015. 5

[21] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L. Waslander. Joint 3d proposal generation and object detection from view aggregation. In *IROS*, pages 1–8, 2018. 7

[22] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, pages 12697–12705, 2019. 7, 8

[23] Ziyu Li, Yuncong Yao, Zhibin Quan, Wankou Yang, and Jin Xie. Sienet: Spatial information enhancement network for 3d object detection from point cloud. *CoRR*, abs/2103.15396, 2021. 3

[24] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *CVPR*, pages 7345–7353, 2019. 7

[25] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, volume 11220, pages 663–678, 2018. 3

[26] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *T-PAMI*, 42(2):318–327, 2020. 5

[27] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall F. Tapen, and Marianna Pensky. Sparse convolutional neural networks. In *CVPR*, pages 806–814, 2015. 5

[28] 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 *ICCV*, 2021. 3, 7, 8

[29] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashii Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In *ICCV*, 2021. 3, 7, 8

[30] Su Pang, Daniel D. Morris, and Hayden Radha. Clocs: Camera-lidar object candidates fusion for 3d object detection. In *IROS*, pages 10386–10393, 2020. 7

[31] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from RGB-D data. In *CVPR*, pages 918–927, 2018. 7

[32] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 77–85, 2017. 3

5426
