Supplementary Materials for "Fully Sparse 3D Object Detection"

Anonymous Author(s)
Affiliation
Address
email

A Qualitative Analysis of Center Feature Missing

A.1 Center Heatmap

We first visualize the learned centerness heatmap of SST_{center} in Fig. 1. For vehicle with regular sizes, there are usually sharp and accurate heatmap peaks. However, the score distribution in large vehicle is usually smooth and ambiguous, or even broken (multiple peaks in an object).

A.2 Comparison Instance Point Grouping with Center Assignment

We also make a qualitative comparison between our strategy and the center assignment (SST_{center}) in Fig. 2. We list some analysis as follows.

1. FSD does not heavily rely on the quality of center voting, because FSD only use center voting as a step for instance point grouping (i.e., instance segmentation). Even the center positions are not accurate, we can still segment the instances.
2. For the large vehicles, the voted centers could also exhibit a non-sharp distribution (e.g., the first row in Fig. 2). However, FSD could group them via connected components labeling to obtain the instance segmentation.
3. For center assignment, the predicted box centers are around the heatmap peaks. That said, if the peaks are ambiguous or even broken due to center feature missing, the predicted boxes would be of low quality.

A.3 Prediction Visualization

Finally, we show the visualization of predictions in Fig. 3 to further illustrate that our method avoids the adverse effects of center feature missing (CFM). Due to CFM, the predictions from center features in large vehicles usually have less confident scores. So these less confident predictions are very likely to be suppressed by some smaller inaccurate bounding boxes with higher scores in NMS.

B Implementation Details

We provide the detailed configuration file in our attached `fsd_config.py` file, which is the config we used to immplement FSD in MMDetection3D codebase [1]. For a better understanding, we leave detailed code comments in the attached file. Here we select and list some common hyper-parameters. If readers are interested in more details, please refer to the attached `fsd_config.py` file. The full code will be soon released after refactoring.

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

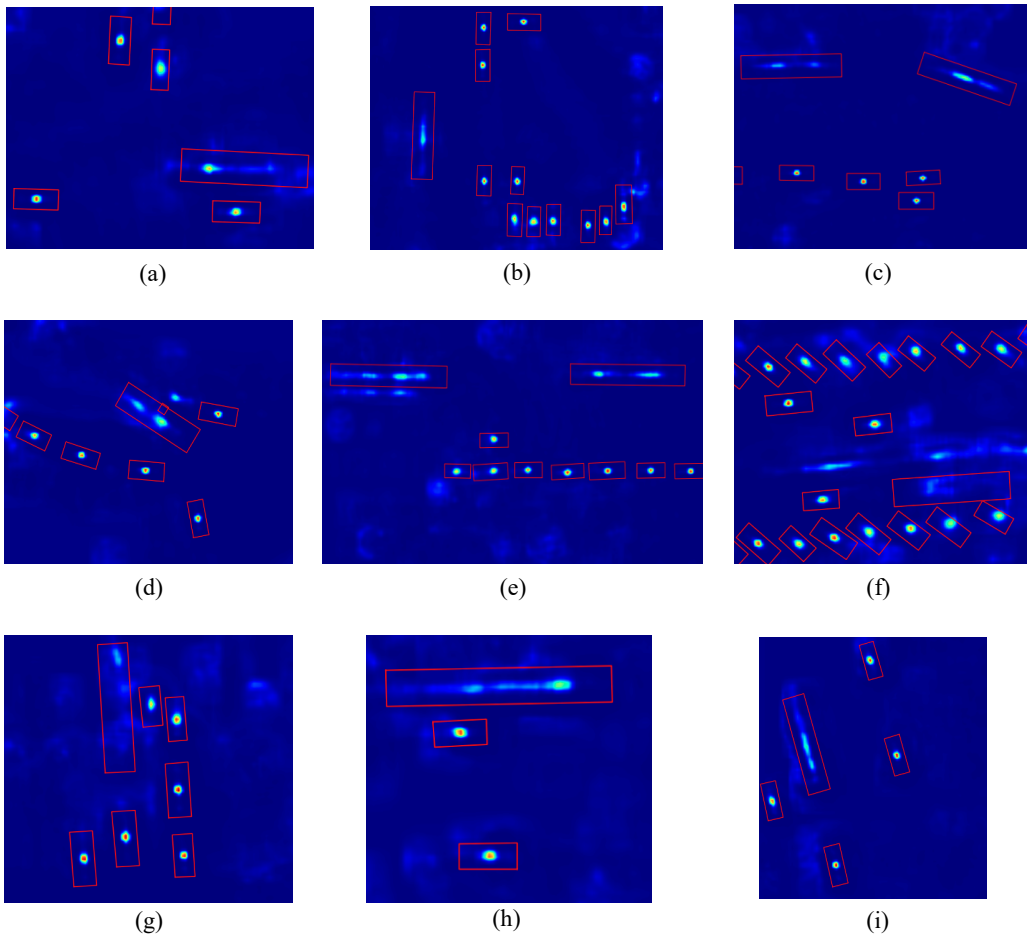

Figure 1: Center heatmap visualization of SST$_{center}$. Red boxes are ground-truth bounding boxes. For vehicle with regular sizes, there are sharp and accurate heatmap peaks. However, the score distribution in large vehicle is usually smooth and ambiguous, or even broken. For example, in (a) and (h), the position of the center peak in the large vehicle is inaccurate (closer to vehicle heads). In (d) and (e), there are two or more ambiguous center peaks in the large vehicles.

Hardware For experiments on Waymo Open Dataset, we use 8 2080Ti GPUs. For experiments on Argoverse 2 (AV2), we use 8 3090 GPUs because the long-range experiments of CenterPoint on AV2 require GPU memory larger than 11GB.

Optimization Following SST [2], the batch size we adopt is 1 for each GPU and the synchronized-BN is enabled. We use AdamW as the optimizer with 0.05 weight decay. The maximum learning rate is 1e-3 scheduled by cosine schedule strategy.

Loss functions For the semantic classification in Instance Point Grouping, we apply the Focal Loss to each point, which we denoted as L_{sem} in the main paper. L_{sem} is normalized by the number of all points. For voting loss (L_{vote}) in Instance Point Grouping, it is normalized by the number of points inside the ground-truth bounding boxes. For the classification loss in SIR/SIR2 (L_{cls} and L_{iou}), they are normalized by the number of all groups in a batch. L_{reg} and L_{res} are normalized by the number of *positive* groups. Moreover, the loss weight of L_{cls} is 2.0. The loss weights of others are all 1.0.

Instance Point Grouping In the Instance Point Grouping, the foreground score thresholds for vehicle/pedestrian/cyclist are 0.5/0.25/0.25, respectively. Only points with scores higher than the thresholds contribute to the voting. The distance thresholds used in Connected Components Labeling (CCL) are 0.6m/0.1m/0.2m for vehicle/pedestrian/cyclist, respectively. In our implementation, we

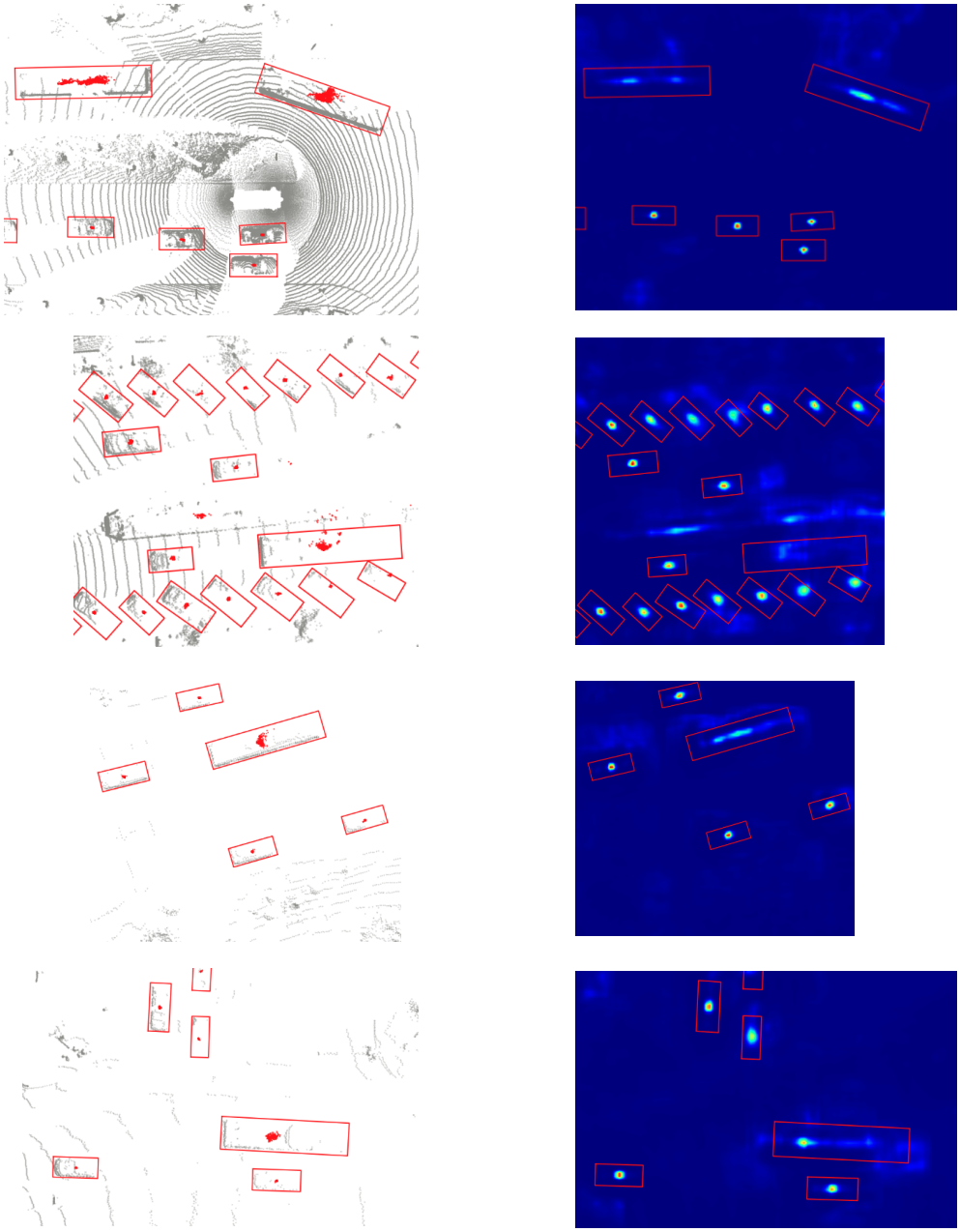

Figure 2: Qualitative comparison between Instance Point Grouping in FSD and center heatmap in SST_{center}. **The left column** shows the results of Instance Point Grouping, and the **the right column** show the learned center heatmaps. Red points are the voted centers. Red boxes are ground truth bounding boxes. We provide analysis in Sec. A.2.

use voxelization to accelerate the CCL. Specifically, we first voxelize the predicted centers and then apply CCL on the voxel centers. Thus, each voxel has a group ID after CCL. All the points inside a voxel use the group ID of the voxel. In practice, the voxelization size is $0.2m \times 0.2m \times 6.0m$, which is pillar voxelization. Note that the predicted centers are usually closed to each other, so there are only hundreds of pillars after voxelization, making the CCL highly efficient.

Network Architecture For voxelization, we follow SST adopting the voxel size of $0.32m \times 0.32m \times 6m$, which is the pillar representation. The numbers of hidden channels in SIR/SIR2 are all

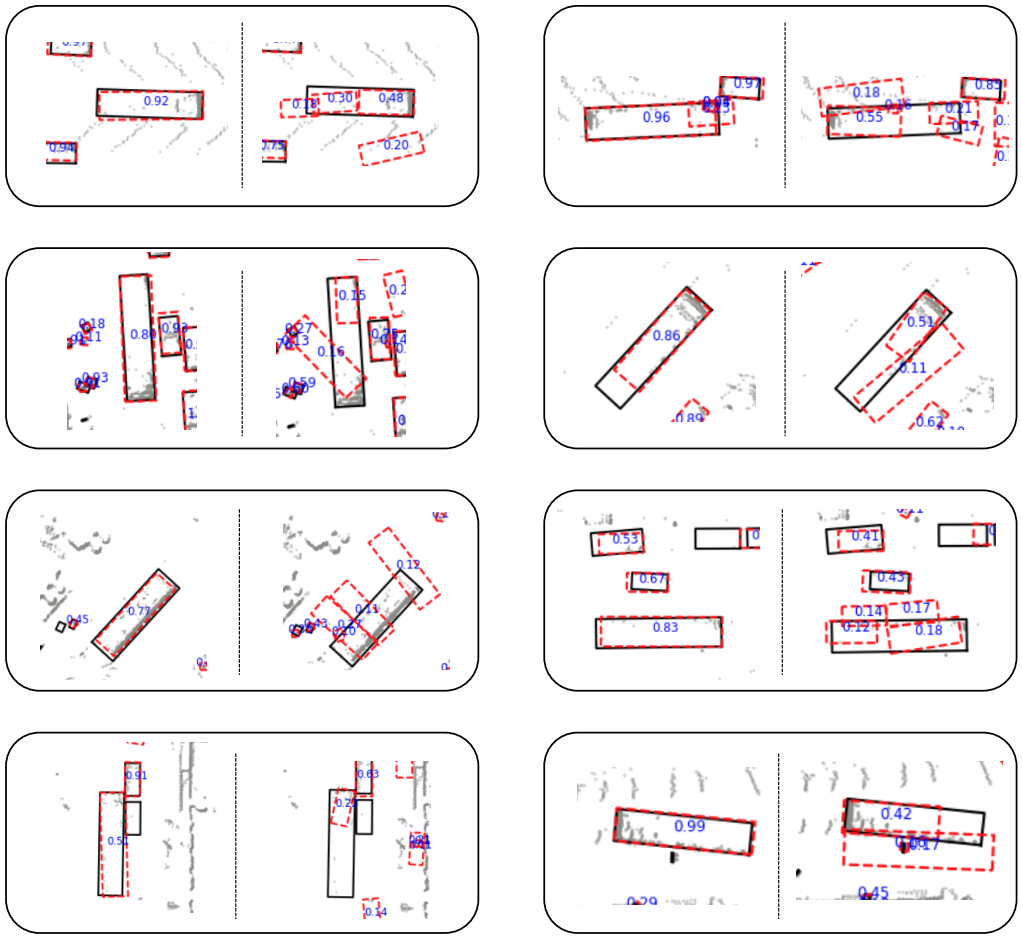

Figure 3: Qualitative comparison between the predictions from FSD and SST$_{center}$. In each subfigure, the left patch shows the predictions of FSD, and the right one shows the predictions of SST. We label the confident scores of predictions with blue numbers. Red boxes are the predictions and black boxes are ground-truth boxes. Due to center feature missing of SST, the predictions from center features in large vehicles usually have less confident scores. So these less confident predictions are very likely to be suppressed by some smaller inaccurate bounding boxes with higher scores in NMS. Best viewed in color and with zooming in.

128. GeLU is adopted as the activation function, and LayerNorm is adopted as the normalization function.

Group Correction As we discussed in the main paper, the prerequisite of dynamic broadcast/pooling is that groups do not overlap with each other. Since SIR predicts a single box proposal for a group, the proposals do not overlap with each other usually. However, in a few cases, there are a small number of points falling into multiple proposals. In practice, we simply copy these points for each group. In this way, each point has a unique group ID, and then dynamic broadcast/pooling can be adopted to realize SIR2.

Inference The prediction with a score higher than 0.1 will be sent to Non-Maximum Suppression (NMS). The IoU threshold in NMS is 0.25 for all classes.

C A Rare Failure Case

Fig. 4 show a very rare failure case and our analysis.

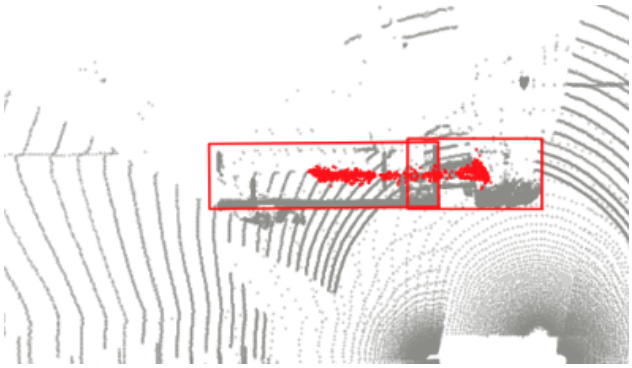

Figure 4: A failure case of point grouping due to improper overlapped annotation. Red points are the voted centers. The red ground-truth bounding boxes of the truck box and truck head have an improper overlap. The simple CCL is likely to recognize them as a single instance. Fortunately, this problem could be avoid if we have more accurate annotations or adopt other grouping methods (e.g., ball query in SSG).

D Details of Argoverse 2

Argoverse 2 (AV2) dataset contains 30 classes, where 26 classes are officially defined as valid classes. Following the CenterPoint model provided by the authors of AV2, we divide the 26 classes into 6 groups. In the attached `argoverse_data_config.py` file, we show the specific grouping strategy and related hyper-parameters.

E Sparse Convolution Based Voxel Encoder

In the main paper, we report the results of FSD$_{spconv}$, where we adopt sparse convolution based UNet [5] (SC-UNet) as the sparse voxel encoder. Specifically, we utilize the implementation of SC-UNet in MMDetection3D [1] (`https://github.com/open-mmlab/mmdetection3d/blob/master/mmdet3d/models/middle_encoders/sparse_unet.py`). The input voxel size of SC-UNet is $0.2m \times 0.2m \times 0.2m$. We also provide a `fsd_spconv_config.py` in our attached files, where all the architecture parameters needed in SC-UNet.

F Results on Test Server

Table 1 shows the results on test split. We also list several top-performing detectors. For a fair comparison, all the detectors we list use single frame point cloud, do not use ensemble strategies, and do not adopt test-time augmentation.

Table 1: Performances on the Waymo Open Dataset test split. We report the top-performing non-ensemble methods taking single frame point clouds and single modality as input.

Methods	mAP/mAPH L2	*Vehicle* 3D AP/APH L1	*Vehicle* 3D AP/APH L2	*Pedestrian* 3D AP/APH L1	*Pedestrian* 3D AP/APH L2	*Cyclist* 3D AP/APH L1	*Cyclist* 3D AP/APH L2
CenterPoint-Voxel [9]	-/69.0	-/-	-/71.9	-/-	-/67.0	-/-	-/68.2
PV-RCNN [4]	71.3/68.8	80.6/80.1	72.8/72.4	78.2/72.0	71.8/66.0	71.8/70.4	69.1/67.8
AFDetV2-lite [3]	72.2/70.0	80.5/80.0	73.0/72.6	79.8/74.3	73.7/68.6	72.4/71.2	69.8/69.7
PV-RCNN++ [6]	72.4/70.2	81.6/81.2	73.9/73.5	80.4/75.0	74.1/69.0	71.9/70.8	69.3/68.2
FSD (ours)	73.0/71.0	81.5/81.2	73.0/72.7	81.7/76.8	74.6/70.1	74.2/73.0	71.4/70.2

G Issues Related to Checklist

Codebase We use MMDetection3D [1] for all of our experiments. MMDetection3D offers solid implementation of a wide variety of 3D detection algorithms. MMDetection3D is licensed under Apache License, Version 2.0.

Dataset We use Waymo Open Dataset [7] and Argoverse 2 [8] dataset as the benchmark for our experiments. They are all public benchmark. See https://waymo.com/open/terms/ and https://www.argoverse.org/about.html#terms-of-use for the detailed terms of use.

Error Bar WOD and AV2 are all large-scale dataset, so the performances are very robust and the run-to-run error is less than 0.2 mAP.

References

[1] MMDetection3D Contributors. MMDetection3D: OpenMMLab Next-generation Platform for General 3D Object Detection. https://github.com/open-mmlab/mmdetection3d, 2020.

[2] Lue Fan, Ziqi Pang, Tianyuan Zhang, Yu-Xiong Wang, Hang Zhao, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Embracing Single Stride 3D Object Detector with Sparse Transformer. In *CVPR*, 2022.

[3] Yihan Hu, Zhuangzhuang Ding, Runzhou Ge, Wenxin Shao, Li Huang, Kun Li, and Qiang Liu. AFDetV2: Rethinking the Necessity of the Second Stage for Object Detection from Point Clouds. *arXiv preprint arXiv:2112.09205*, 2021.

[4] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection. In *CVPR*, 2020.

[5] 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*, 2020.

[6] 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.

[7] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in Perception for Autonomous Driving: Waymo Open Dataset. In *CVPR*, 2020.

[8] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next Generation Datasets for Self-Driving Perception and Forecasting. In *NeurIPS Datasets and Benchmarks 2021*, 2021.

[9] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3D Object Detection and Tracking. *arXiv preprint arXiv:2006.11275*, 2020.