Supplementary Materials: InfoFocus: 3D Object Detection for Autonomous Driving with Dynamic Information Modeling

Jun Wang[1⋆], Shiyi Lan[1⋆], Mingfei Gao[2], and Larry S. Davis[1]

[1]University of Maryland, College Park MD 20742, USA
[2]Salesforce Research, Palo Alto, CA 94301, USA
{junwang,lsd}@umiacs.umd.edu, sylan@cs.umd.edu
mingfei.gao@salesforce.com

1 More Implementation Details

As mentioned in the Section 4.2, here we further introduce our implementation details to encourage reproductivity.

1.1 Deep Feature Extractor

Pillar Feature Network. Closely following the codebase[1], we focus on the input LiDAR point cloud range with [-50, -50], [-50, 50] and [-5, 3] meters in x, y, z axis respectively. The voxel size is [0.25, 0.25, 8] and thus the pillars' size is 400×400 after the voxelization. The generated pseudo 2D image[5] is $400 \times 400 \times 64$ from Pillar Feature Network with 64-d output channels. Specifically, the width and length of the pseudo image, W and L, are set to be 400 and 400, while the channel size C is 64.

DCNN. The structure of DCNN includes three blocks of fully convolutional layers, where each of block consists of down-sampling convolutional layers to produce top-down features and de-convolutional layers to upsample and concatenate the feature maps from different strides [5]. Each convolution/deconvolutional uses the kernel with size 3×3 and is followed by a BatchNorm layer and a ReLU layer. In details, the input/output size of the first convolutional block is $400 \times 400 \times 64$ and $200 \times 200 \times 64$, respectively. After the first deconvolutional layer, the dimension of first block feature map becomes $100 \times 100 \times 128$. Similarly, the input/output size of the second convolutional block is $200 \times 200 \times 64$ and $100 \times 100 \times 128$. After the second deconvolutional layer, the dimension of second block feature map is $100 \times 100 \times 128$. For the last deconvolutional layer, the input/output size is $100 \times 100 \times 128$ and $50 \times 50 \times 256$. After the last deconvolutional layer, the dimension of third block feature map becomes $100 \times 100 \times 128$. The final concatenated feature map from the output of three deconvolutional layers has a size of $100 \times 100 \times 384$.

⋆ Equal contribution.
[1] https://github.com/traveller59/second.pytorch.

Table 1. Object detection results (%) on KITTI val set with IoU threshold of 0.7 for Car class

Method	Easy	Moderate	Hard
VoxelNet [12]	81.97	65.46	62.85
Second [9]	87.43	76.48	69.10
PointRCNN [7]	88.88	78.63	77.38
Fast Point R-CNN [2]	89.12	79.00	77.48
STD[11]	89.70	79.80	79.30
PointPainting[8]	87.15	76.66	74.75
3DSSD[10]	89.71	79.45	78.67
Ours	89.21	78.94	78.06

1.2 RPN

Unlike the original PointPillars [5] that adopts a Single Shot Detector (SSD) [6] as detection head, we utilize an improved implementation with a dual-head for the RPN. Specifically, an 1×1 convolutional layer is used in each of three branches following [1]. The dataset is empirically divided into two groups based on the object size, e.g., {car, bus, construction_vehicle, trailer, truck} and {barrier, bicycle, motorcycle, pedestrian, traffic_cone}. Specifically, the small-scale head takes the feature map from the first convolutional block with size of $200 \times 200 \times 64$, while the large-scale head takes the concatenated feature map with size of $100 \times 100 \times 384$, both from the deep feature extractor. The output of RPN is the candidate proposals with classification, bounding box and direction predictions.

1.3 Data Preparation

Similar with [5, 4, 1, 9], we adopt the temporal aggregation from multiple LiDAR sweeps to form a richer point cloud as input. Specifically, we aggregate the current single frame with 10 previous different frames to form the final input.

2 Results on KITTI

KITTI dataset [3] is a widely used dataset for 3D object detection task. We also conduct experiments on KITTI benchmarks to illustrate the effectiveness and robustness of our approaches as shown in Table. 1. We observe that our method achieves comparable performance to the state-of-the-art (STD) [11] on the *Car* class of the KITTI validation set.

3 Additional Qualitatively Visualization on nuScenes

We report additional visualization results of our framework[2] on the nuScenes validation set in Fig. 1.

[2] Here, we use the model that is trained with the default setting of training epochs.

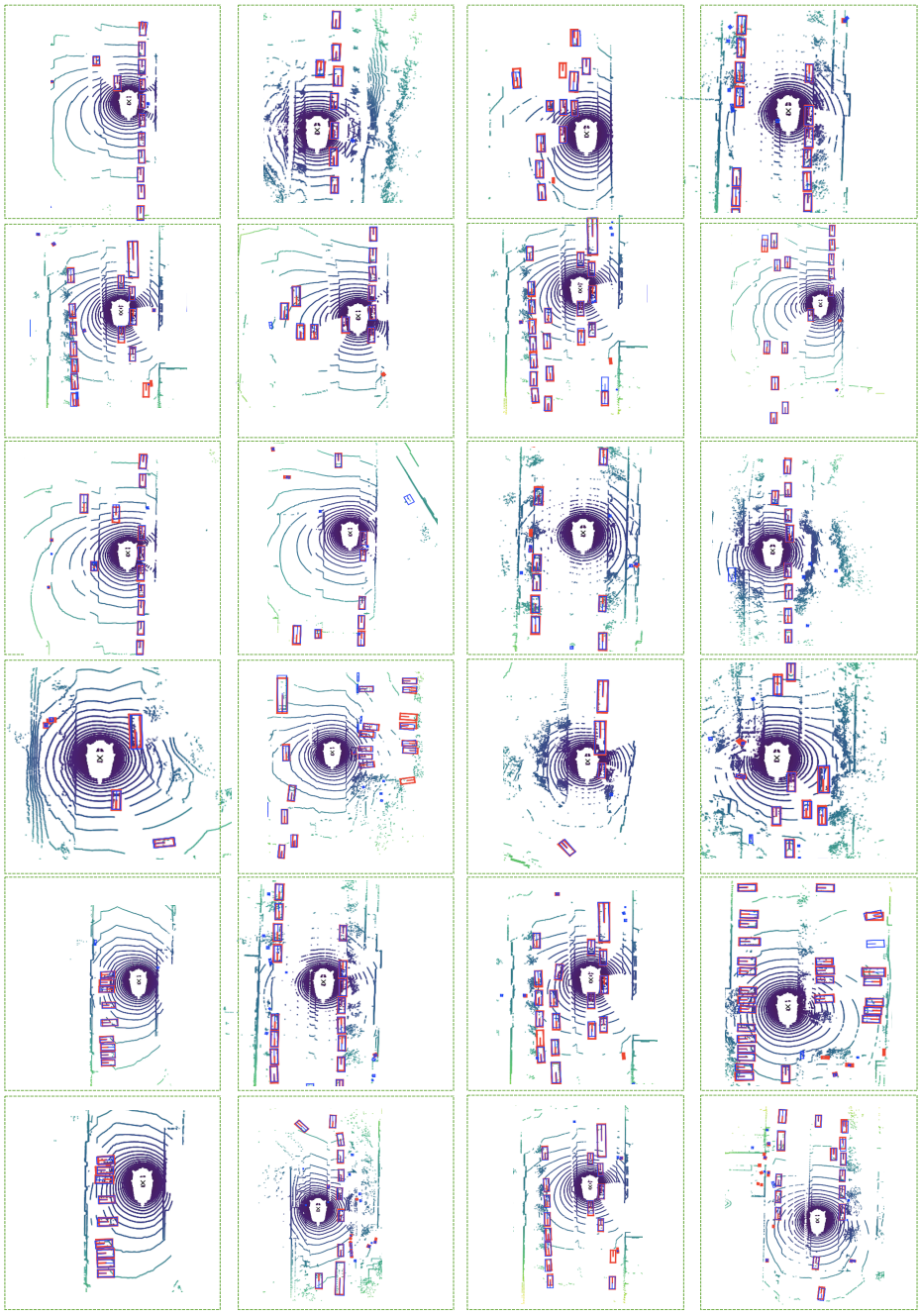

Fig. 1. Visualization results of 3D BEV images with ground truth (red) and detection (blue) box on the nuScenes validation set

References

1. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. arXiv preprint arXiv:1903.11027 (2019)
2. Chen, Y., Liu, S., Shen, X., Jia, J.: Fast point r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9775–9784 (2019)
3. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. The International Journal of Robotics Research **32**(11), 1231–1237 (2013)
4. Hu, P., Ziglar, J., Held, D., Ramanan, D.: What you see is what you get: Exploiting visibility for 3d object detection. arXiv preprint arXiv:1912.04986 (2019)
5. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12697–12705 (2019)
6. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016)
7. Shi, S., Wang, X., Li, H.: Pointrcnn: 3d object proposal generation and detection from point cloud. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–779 (2019)
8. Vora, S., Lang, A.H., Helou, B., Beijbom, O.: Pointpainting: Sequential fusion for 3d object detection. arXiv preprint arXiv:1911.10150 (2019)
9. Yan, Y., Mao, Y., Li, B.: Second: Sparsely embedded convolutional detection. Sensors **18**(10), 3337 (2018)
10. Yang, Z., Sun, Y., Liu, S., Jia, J.: 3dssd: Point-based 3d single stage object detector. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11040–11048 (2020)
11. Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J.: Std: Sparse-to-dense 3d object detector for point cloud. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1951–1960 (2019)
12. Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4490–4499 (2018)