First Place Solution to the 3D Object Detection of the SSLAD2022 Challenge

Tengteng Huang[1], Zhuyu Yao[1], Lei Liu[1], Bin Wang[1], Tianyuan Jiang[2], Jianjian Sun[1], Xuefeng Wang[1], Zeming Li[3], and Haotian Yao[1]

MEGVII Technology
{huangtengteng, yaozhuyu, liulei02, wangbin04, sunjianjian, wangxuefeng, yaohaotian}@megvii.com[1] tianyuan@whu.edu.cn[2] zengarden2009@gmail.com[3]

Abstract. In this report, we present our winning solution to the 3D object detection of the SSLAD2022 Challenge at ECCV 2022. We propose a simple yet effective one-stage siamese network to fully leverage the complementary information between multiple temporally adjacent frames. Besides, we integrate channel-wise and spatial-wise attention into our backbone network efficiently to obtain discriminative feature representation. Moreover, we add an IoU prediction head to alleviate the issue of the inconsistency between classification and localization confidences. The IoU loss is also employed and optimized jointly with common classification and regression losses for more precise detection boxes. Our approach achieves 85.13% mAP and wins first place in the 3D object detection track.

Keywords: 3D Object Detection, One-Stage, Multiple Frames

1 Introduction

We brief the Self-supervised Learning for Next-Generation Industry-level Autonomous Driving Challenge at ECCV 2022. It introduces ONCE dataset [1] which consists of 1 million LiDAR scenes and 7 million camera images. To simulate various real scenarios, the data is collected from different periods, areas, and weather conditions. Moreover, the dataset provide three unlabeled split with different amounts of unlabeled data, namely raw small, raw medium, and raw large, which facilitates many researches on semi- and self-supervised learning for 3D object detection.

2 Methods

We design a one-stage and anchor-free [2–4] 3D point cloud detector. Fig 1 illustrates the overall pipeline of our framework, which consists of three core parts, namely siamese 3D backbone, BEV feature extractor, and anchor-free detector heads.

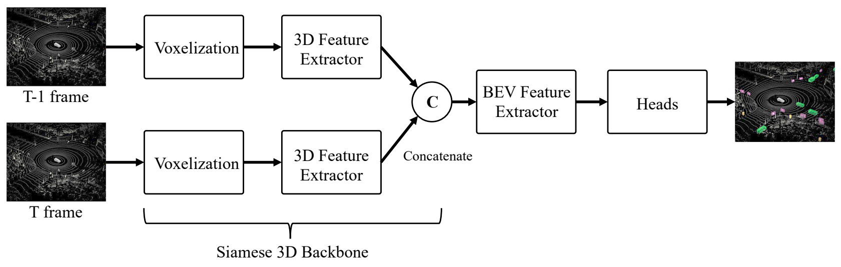

Fig. 1. Illustration of the overall architecture of our method.

2.1 Siamese 3D Backbone

The siamese 3D backbone consists of two branches which share the same architecture but their parameters are not shared. Each branch includes a point cloud voxelization module and a 3D feature extractor. The point clouds of the current frame and its previous frame are fed into the two branches respectively, and then their features are concatenated to get the fusion feature. In this way, the complementary information between temporally adjacent frames are effectively combined. We empirically find this simple multi-frame fusion mechanism is valueable for small objects like pedestrians.

In the following, we present more details about the 3D feature extractor. We employ a ResNet-like [5] 3D backbone model which consists of five stages, as is illustrated in Figure 2 (a). The first four stages are stacked with a single (submanifold) sparse convolution layer [6] (SpConv for short) and {2, 2, 2, 2} residual blocks equipped with SE operation [7], respectively. Each residual block involves two groups of SpConv-BatchNorm-ReLU layers. Besides, we append a SE layer after the second group, which adaptively enhances informative features while suppress those less related to the detection task by conducting channel-wise attention operation. The stride for the first SpConv layer inside stage 1 (*resp.*, stage 2-4) is set to 1 (*resp.*, 2) along X, Y, and Z axis. For the last stage, we use only one SpConv layer with stride 1 (*resp.*, stride 2) for X and Y axis (*resp.*, Z axis). In this way, the height information is compressed for efficiency with minor losses of height information. The 3D feature extractor downsamples the X and Y axis by a factor of 8 while 16 for the Z axis. Finally, we flatten the 3D feature map into 2D BEV representation.

2.2 BEV Feature Extractor

We adopt an hourglass [8] architecture for the BEV feature extractor to make full use of the context information from different scales. Concretely, our network consists of two hourglass submodules. Each hourglass submodule is composed of an encoder and a decoder. The encoder consists of a single convolutional layer for resolution reduction and 5 SC-Conv modules [9] with bottleneck architecture (SCBottleNeck for short). The SCBottleNeck module effectively enlarges

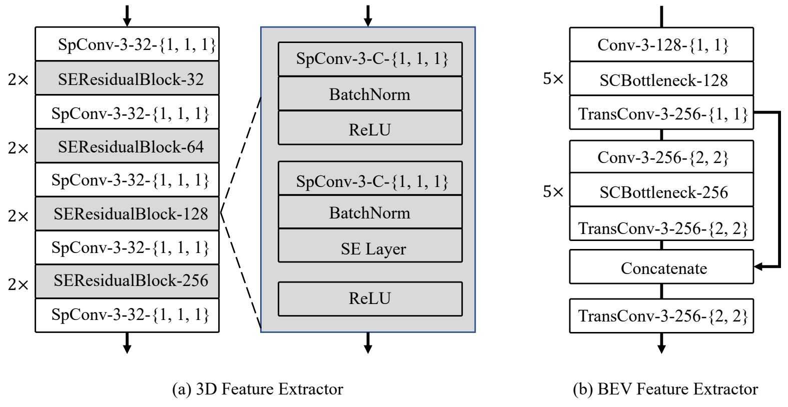

Fig. 2. Illustration of the architecture of our 3D feature extractor and BEV feature extractor. For each layer, we specify the setting of each layer in the format of *kernel size-channels-stride*. For SEResidualBlock and SCBottleNeck, we only present its channels, with the default kernel size and stride setting to 3 and 1, respectively. **n**× on the left of blocks denotes that n identical blocks are sequentially stacked.

the receptive field and performs efficient attention operation along both the channel and spatial dimension. The decoder contains only a single transposed convolutional layer to recover the resolution. The strides for encoders of these two submodules are set to 1 and 2, respectively. In other words, all the layers from the first hourglass submodule undergo the same resolution. For the second hourglass submodule, its encoder first downsamples the input by a factor of 2 and then the decoder upsamples the input to recover the resolution. We combine the decoder outputs from these two hourglass submodules in a concatenation manner. Finally, we use an extra transposed convolutional layer for producing a denser feature map. In this way, the input BEV feature map with stride 8 is upsampled to stride 4, which keeps more spatial information and plays a key role for more precise prediction, especially for small objects, such as a pedestrian.

2.3 Anchor-free Detection Head

There are five annotated categories in the ONCE dataset, namely Car, Bus, Truck, Pedestrian, and Cyclist. In our design, we merge the first three categories into a single category called Vehicle. We define two subtasks, the first subtask is responsible for the prediction of Vehicle and Pedestrian, while the second one predicts the Cyclist. We investigate multiple different ways of subtask design and show their inherent complementarity in Section 3.3.

Each subtask consists of a shared convolutional layer and three separate detection heads, namely classification head, regression head, and IoU prediction head. The classification head predicts the category of a certain position. The regression head regresses the offset to the center of the ground truth box, the box size, and the heading orientation. The IoU prediction head infers the IoU between the prediction box and corresponding ground truth, which can be viewed as a metric for localization confidence. In the NMS process, we combine the classification confidence and localization confidence in a weighted sum manner to provide a more reliable measurement for the quality of predicted boxes. In addition, We use the smooth L1 loss for the IoU prediction head. The IoU score predicted by our model is just used while doing NMS. We resort bounding boxes by its IoU-aware category score, which is defined as:

$$score_{nms} = (score_{category})^{1-\alpha} \times (score_{iou})^{\alpha} \tag{1}$$

The $score_{category}$ is the original category score, $score_{iou}$ is the IoU score predicted by the IoU-aware head. In our model α is set to 0.65.

2.4 Label Assignment

Multiple positive assignment (one-to-many assignment) is more efficient than one-to-one assignment, as mentioned in [10–12]. We adopt a simple label assignment strategy similar to FCOS in our method. That is, 9 anchor points closest to the center of each ground truth are assigned as positives. When an anchor point is assigned to multiple ground truths, the ground truth whose center is closest to it is taken as the final target. The 9 anchor points are a positive bag for each ground truth, while those which are not assigned to any ground truth are taken as negatives.

To further improve our model's performance, we propose a method to re-weight the classification loss of anchor points. We generate a Gaussian distribution $\mathcal{G}(\mu, \sigma^2)$ for each ground truth. μ equals the distance between the centers of anchor points and the ground truth. σ equals 1. The focal loss of negative anchors is re-weighted by (1 - $\mathcal{G}$), aiming at decreasing the loss of anchors that are close to the center of ground truths. On the other hand, we also re-weight the loss of positive anchors. First, we compute the loss of positive anchors (including classification and localization loss). Second, the loss weight of positive anchors and the focal loss [13] of our method can be defined as follows (we set β_1 to 1 and β_2 to 0.5).

$$\mathcal{L}_{pos} = \beta_1 \mathcal{L}_{cls\_pos} + \beta_2 \mathcal{L}_{reg\_pos} \tag{2}$$

$$\mathcal{W}_{pos} = 2 \times (1 - Sigmoid(\mathcal{L}_{pos})) \tag{3}$$

$$\mathcal{W}_{neg} = 1 - \mathcal{G}(\mu, \sigma^2) \tag{4}$$

$$\mathcal{L}_{heat} = \begin{cases} \mathcal{L}_{focal\_pos} \times \mathcal{W}_{pos} & \text{if positive} \\ \mathcal{L}_{focal\_neg} \times \mathcal{W}_{neg}^2 & \text{else} \end{cases} \tag{5}$$

2.5 Loss

For the classification head, we adopt the focal loss [13] for heatmap prediction to alleviate the issue of foreground/background imbalance. For the regression head, L1 loss is used for prediction of center offsets $(\Delta x, \Delta y, \Delta z)$, box sizes (l, w, h), and heading orientation θ. Besides, we employ an IoU loss to jointly optimize center offsets and box sizes through IoU Metric, which is beneficial to generate a more precise prediction. The overall optimization goal is

$$\mathcal{L} = \lambda_1 \mathcal{L}_{heat} + \lambda_2 \mathcal{L}_{reg} + \lambda_3 \mathcal{L}_{IoU\_h} \tag{6}$$

$$\mathcal{L}_{reg} = \mathcal{L}_{offset} + \mathcal{L}_{size} + \mathcal{L}_{IoU\_m} \tag{7}$$

where $\mathcal{L}_{IoU\_h}$ denotes the smooth L1 loss for IoU prediction head while $\mathcal{L}_{IoU\_m}$ represents the IoU loss for regression.

It is worth noting we adopt an auto-weighted module [14] to dynamically assign weights (λ_1, λ_2 and λ_3) to different losses. In such a way, we get rid of time-consuming grid search for the weighting factors. The auto-weighted module has a learnable parameter Θ in the shape of n which indicates the assigned weights for different losses. In our case, n is 3. Note that Θ is jointly optimized with other model parameters through backpropagation. Besides, we add an extra regularization term to avoid the too large value of Θ. Hence, the overall optimization goal can be reformulated as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{cls} + \lambda_2 \mathcal{L}_{reg} + \lambda_3 \mathcal{L}_{IoU\_h} + \mathcal{L}_{\Theta} \tag{8}$$

$$\mathcal{L}_{\theta} = \log(1 + \Theta^2) \tag{9}$$

$$\lambda_i = \frac{1}{2\Theta_i^2} \tag{10}$$

3 Experiments

3.1 Implementation Details

Data Augmentations. We follow the data augmentation strategies [15, 16] used in 3D object detection. GT sampling [17] is the most effective augmentation to improve the model's performance. We select 14, 5, 4, 5, 13 ground truth samples for car, bus, truck, pedestrian, and cyclist. They are directly put into the current frame without extra transforms. We do randomly flipping along the x-axis and y-axis, global rotation following $\mathcal{U}\left(-\frac{\pi}{4}, \frac{\pi}{4}\right)$, global scaling following $\mathcal{U}\,(0.95, 1.05)$, and global translation along x, y, z-axis following $\mathcal{U}\,(-0.2m, 0.2m)$.
Training Details. We implement our methods based on the official repository based on ONCE-Benchmark [1]. The point cloud range is limited to [(-75.2, 75.2), (75.2, 75.2), (-5.0, 3.0)] respect to x, y, z-axis during training and testing process. The voxel size along x, y, z-axis is set to [0.1m, 0.1m, 0.2m], and the max number of voxels is 60000. As for focal loss, the alpha is 0.25 and gamma is 2. AdamW [18] with one-cycle policy [19] is used as optimizer. We set the max

learning rate to 3×10^{-3}, division factor to 10, momentum ranges from 0.95 to 0.85, fixed weight decay to 0.01, pct stat to 0.4, grad norm clip to 35. Our model is trained with 80 epochs on the training set.

Inference Details. We apply the classification score threshold 0.1 and NMS with an IoU threshold of 0.1 to filter the predict bounding boxes. The max predict size before NMS is 1500 and the max size after NMS is 100. We raise the number of the max size before NMS because our method applies one-to-many assignment like FCOS [2] instead of one-to-one assignment as CenterPoint [20] does. Instead of sorting bounding boxes by original category score, we combine the IoU score predicted by our model and the original category score to get a new score for NMS ranking. IoU-aware score weight α at test time is 0.65 for all categories.

Test Time Augmentation (TTA). We adopt multiple test-time augmentations, such as point cloud rotation, global scaling, and flip. The results of different augmentations are merged by Weighted Boxes Fusion (WBF) [21]. We perform [0, ±22.5, ±45, ±135, ±157.5, 180] for yaw rotation. Besides original yaw rotation, we add an extra augmentation for each angle of rotation. It includes flip along the x-axis, y-axis, and along both x and y-axis, scale in [0.95, 0.975, 1.025, 1.05] for global scaling. However, yaw rotation contributes most to boosting the model's performance.

Ensemble Method. We apply Weighted Box Fusion (WBF) as our ensemble method for two purposes, one is to merge the results of TTA, the other is to merge the results of different models. Especially, we ensemble 10 models to get our final results, including different ways of subtask designs and their two-frame versions. We set the box filtering threshold to [0.05, 0.05, 0.25] and the IoU threshold to [0.7, 0.3, 0.5] for the WBF process. Besides, boxes with scores less than 0.03 are also discarded.

Baseline	IoU Loss	Multi-Positive	SE Backbone	Multi-Frame	mAP
✓					67.4
✓	✓				68.5
✓	✓	✓			70.8
✓	✓	✓	✓		71.6
✓	✓	✓	✓	✓	**72.4**

Table 1. Ablation studies on the ONCE validation dataset. All the model are trained with the only the ONCE training dataset and evaluated without any test time augmentation.

3.2 Ablation Study

In Table 1, we ablates the effect of different components of our model. As is shown, IoU loss leads to an improvement of +1.1% mAP. Adopting the multiple

Subtask Design	Car	Bus	Truck	Pedestrian	Cyclist	mAP	mAP †
Default	H1	H1	H1	H1	H2	83.34	85.79
Alternative1	H1	H1	H1	H2	H3	**84.09**	**86.40**
Alternative2	H1	H1	H1	H2	H1	83.65	85.98
Alternative3	H1	H1	H1	H2	H2	82.50	85.32
Alternative4	H1	H1	H1	H1	H1	82.57	85.39
Alternative5	H1	H2	H3	H4	H5	81.67	-

Table 2. Results of our approach with different subtask designs. Models are trained on the ONCE trainval split and evaluated on the validation set. H1, H2, H3, H4, H5 denotes different detection heads, while the same letter means they belong to the same head. † denotes the results with post-procedure.

positive label assignment strategy further yields an improvement of +2.3% and boosts the mAP to 70.8%. Besides, using residual blocks equipped with SE Layer brings +0.8% improvement. Multiple frame strategy futher yileds +0.8% improvement and leads to a mAP of 72.4%. We also investigate the effect of different ways of subtask design, which is shown in Table 2.

3.3 Post Processing Procedure

We employ TTA and WBF as the post processing procedure to further enhance the performance of our model. As is shown in Table 2, TTA+WBF consistently yields a remarkable improvement of around +2.5% mAP for multiple different models. Besides, we observe that ensembling the results of different models through TTA+WBF leads to even more significant improvement. Ensembling all models but the last one in Table 2 and their 2-frame versions results in a mAP of 88.2% on the validation split. We notice that when the gap between the prediction and the ground truth of orientations exceeds a certain range, the evaluation metric will ignore the wrong heading predictions, which leads to a much better metric. Ignoring wrong heading predictions by adding 4π to the orientation prediction may result in an improvement of 3%-4% mAP. Besides, we enlarge the predicted pedestrian boxes by 5% and achieve slightly better performance for pedestrian, since we empirically find that these boxes tend to be smaller than their ground truth.

3.4 Attempts on Semi-supervised Learning

We pay much effort to investigate the semi-supervised learning framework for 3D object detection. Our method can be categorized into a popular stream of the pseudo-label-based framework. Concretely, our approach follows a two-stage pipeline: 1) first training on the labeled data; 2) then fine-tuning with both labeled and unlabeled data. Our framework consists of a student and a teacher network, both of which are initialized with the same well-trained weights on the labeled data. The student is optimized with ground truth of labeled data and pseudo labels produced by the teacher for the unlabeled data through the same optimization goal as stated in Sec. 2.5. While the teacher is updated by exponential moving average (EMA) to absorb up-to-date weights from the student.

However, limited by the performance of the teacher model, many foreground objects may be misclassified as background when producing the pseudo labels. Simply taking all the positions on the heatmap with no foreground objects predicted as negative will amplify the wrong prediction and bias the fine-tuning process. To alleviate this issue, we propose a soft-weighted strategy to dynamically assign weights to negative samples with the guidance of the classification confidence S_{cls} provided by the teacher. Concretely, we use $1 - S_{cls}$ to weigh each negative sample for the unlabeled data. In this way, positions with higher classification confidence will be weakly optimized to be negative samples and vice versa.

In the fine-tuning phase, we set the ratio of labeled and unlabeled data to 1:4 for each minibatch. We finetune our model for 25 epochs on the raw small unlabeled data split. Other training details are mostly the same with the supervised setting as stated in Sec. 3.1 except that we use a smaller learning rate of 1e-4. Boxes with a classification score larger than 0.4 are taken as foreground objects and used as pseudo labels. Our semi-supervised framework yields an improvement of around +1.7% mAP when finetuned on models trained on the train split. However, we also observe that the gain is marginal when finetuning on models trained with the trainval split. Hence, the semi-supervised method is not adopted in our final submission.

Besides the online generation of pseudo labels as mentioned above, we also try a offline pseudo label generation method. In more detail, we save the predicted boxes of unlabeld data on the disk offline, using a model well trained on the train set. We adopt the test time augmentation (TTA) to obtain more precise pseudo labels. We treat the pseudo labels as the ground truth of unlabeled data and then finetune our model with both the labeled data and unlabeled data, leading to slightly better performance than the counterpart of online pseudo label generation.

4 Conclusion

In this report, we propose a simple one-stage detector with a siamese 3D backbone network to make full use of the complementary information between adjacent frames. Besides, we design a semi-supervised learning framework to leverage the large-scale unlabeled data. We hope our work could inspire more researches on the large-scale ONCE Dataset in the future.

References

1. Jiageng Mao, Minzhe Niu, Chenhan Jiang, Hanxue Liang, Xiaodan Liang, Yamin Li, Chaoqiang Ye, Wei Zhang, Zhenguo Li, Jie Yu, Hang Xu, and Chunjing Xu. One million scenes for autonomous driving: ONCE dataset. *CoRR*, 2021.
2. Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection, 2019.
3. Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. *ECCV*, 2018.
4. Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection, 2019.
5. 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.
6. Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. *CVPR*, 2018.
7. Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CVPR*, 2018.
8. Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016.
9. Jiang-Jiang Liu, Qibin Hou, Ming-Ming Cheng, Changhu Wang, and Jiashi Feng. Improving convolutional networks with self-calibrated convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10096–10105, 2020.
10. Jianfeng Wang, Lin Song, Zeming Li, Hongbin Sun, Jian Sun, and Nanning Zheng. End-to-end object detection with fully convolutional network, 2021.
11. Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. Ota: Optimal transport assignment for object detection, 2021.
12. Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021, 2021.
13. Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. *ICCV*, 2017.
14. Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
15. Zhuangzhuang Ding, Yihan Hu, Runzhou Ge, Li Huang, Sijia Chen, Yu Wang, and Jie Liao. 1st place solution for waymo open dataset challenge – 3d detection and domain adaptation, 2020.
16. Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv:1908.09492*, 2019.
17. Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 2018.
18. Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *ICLR*, 2019.
19. Sylvain Gugger. The 1cycle policy. https://sgugger.github.io/the-1cycle-policy.html, 2018.
20. Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking, 2021.

21. Roman Solovyev, Weimin Wang, and Tatiana Gabruseva. Weighted boxes fusion: Ensembling boxes from different object detection models. *Image and Vision Computing*, 2021.