# 1st Place Solution for Waymo Open Dataset Challenge 2021 Real-time 2D Detection

Qiankun Xie, Fenfen Wang, Lindong Li, Yaonong Wang
Han Xu, Yu Wang, Xiubo Ye, Hongtao Zhou
Leapmotor Technology Co.,Ltd
{xie_qiankun, wang_fenfen, li_lindong, wang_yaonong}@leapmotor.com

## Abstract

*In this technical report, we present a both state-of-the-art and fast 2D object detection algorithm which reached the 1st place in the Real-time 2D Detection of the Waymo Open Dataset Challenges. We adopted two-stage detector as our basic framework, integrating with some new one-stage detector algorithms. Our team LeapMotor_Det achieved 70.41 L2 mAP with 61.6 ms latency in the Real-time 2D detection challenge by the methods we used.*

## 1. Datasets

The Waymo Open Dataset (WOD)[8] is comprised of high resolution sensor data, including a wide variety of environments, objects and weather conditions. The dataset contains images captured from 5 cameras associated with 5 different directions, the images have 2 resolution, 1280x1920 and 886x1920 respectively. In summarize, there are 0.85M images and 9.9M 2D bounding boxes for vehicle, pedestrian and cyclist after removing images with no usable annotations. We extracted 1/10 size dataset by intervals of 10 frames as sub-train and sub-validation dataset, which would be used in initial experimental stage.

## 2. Evaluation

In the real-time 2D Detection, the detection result was reported by Mean Average Precision(mAP) at Level_2 among ALL_NS(vehicles, pedestrians, cyclists). The positive IoU thresholds for vehicles, pedestrians and cyclists were set to 0.7, 0.5 and 0.5 respectively.

Additionally, there is a latency requirement that the model must run faster than 70ms/frame on Nvidia Tesla V100 GPU.

## 3. Methods

In this section, our method with bells and whistles used in this challenge will be presented. Our experiments consisted of two parts, namely, one-stage and two-stage, where the two-stage detector was based on the former. All the experiments were conducted in Detectron2[1]. We converted the dataset to COCO format thanks to the codes provided by the author of the UniverseNet[2].

### 3.1. one-stage methods

Considering the time limit, one-stage algorithms would be the first choice.

**Baseline model**

We applied the powerful one-stage detector ATSS[9] as the baseline model.

For quick verification, in initial experimental stage, resnet18[1] was adopted as our backbone. For the neck part, we adopted the Feature Pyramid Network (FPN)[6] from [7], whereas the channel was changed to 128. The feature maps P3$\sim$P7 ($l$ indicates pyramid level, where P$l$ has resolution $2^l$ lower than the input) were computed from FPN, as shown in Figure1. The height and width of input images were resized to 1280 and 1920 respectively for training and testing. Random horizontal flipping was the only form of data augmentation we adopted in training phase. We adopted synchronized Batch Normalization in the backbone, FPN neck and heads. By default, models are trained on 4 GPUs with 4 images per GPU for 100K iterations (24 epochs). In training, we employed stochastic gradient descent (SGD) optimizer with 0.9 momentum, 0.0001 weight decay, a linear warmup of 2000 iterations. The initial learning rate was set to 0.02, and was then divided by 10 at 70k and 90k iterations, respectively.

**Add P2 to detector**. According to Huang et.al. [2], relative size of objects in WOD are much smaller (3 times) than those in COCO and most of objects in WOD could be

[1] https://github.com/facebookresearch/detectron2
[2] https://github.com/shinya7y/WaymoCOCO

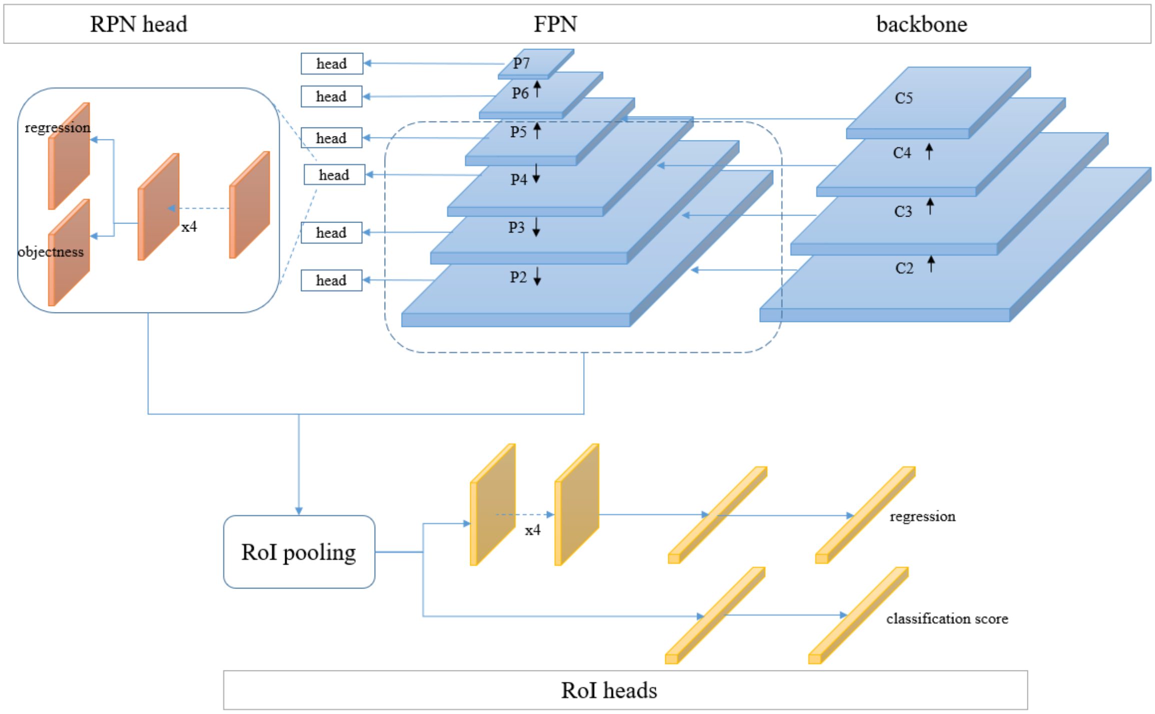

Figure 1. network architecture of our 2-stage detector

Table 1. experiments on baseline model. WAP means Waymo official mAP and CAP means COCO mAP. Initial LR was set to 0.2 only when higher LR is used, otherwise, 0.02.

| methods | WAP | CAP | CAPs | CAPm | CAPl |
|---|---|---|---|---|---|
| baseline | 52.47 | 36.67 | 9.45 | 38.25 | 67.27 |
| + P2 | 53.03 | 36.63 | 10.73 | 37.89 | 65.82 |
| + multi-scale training | 56.77 | 39.40 | 12.95 | 42.22 | 66.97 |
| + higher LR and cosine LR | 57.15 | 39.83 | 13.39 | 43.49 | 66.82 |

regarded as small objects(area smaller than 32x32). Thus, we added feature map from P2 to handle these small objects. For fair comparison, FPN channel was decreased to 64 so that the model FLOPs (the number of multiply-accumulates) would be as same as the baseline. As shown in Table 1, CAPs improves obviously.

**Multi-scale training**. We adopted multi-scale training, which does not increase inference time. The height of input image was randomly sampled from a list which starts from 640, increments by 32 and stops at 1280, and width-to-height ratio was kept unchanged with a limitation that width cannot be larger than 1920.

**Higher LR and cosine LR**. We increased initial learning rate(LR) from 0.02 to 0.2. Cosine LR with warmup was used to decay the learning rate.

**More one-stage algorithms**. Other two one-stage algorithms were compared: GFLv2[4], AutoAssign[11]. For fair comparison, all three algorithms were using almost same network architectures and hyperparameters .

It can be seen from Table 2 that GFLv2 obtains higher validation result than ATSS and AutoAssign.

Table 2. results of ATSS, AutoAssign and GLFv2, GFLv1

| detector | WAP | CAP | CAPs | CAPm | CAPl |
|---|---|---|---|---|---|
| ATSS | 57.15 | 39.83 | 13.39 | 43.49 | 66.82 |
| AutoAssign | 49.90 | 33.47 | 12.45 | 38.81 | 63.66 |
| GFLv2 | 58.18 | 41.00 | 13.82 | 44.36 | 68.68 |
| GFLv1 | 57.71 | 40.80 | 13.57 | 43.81 | 68.97 |

We found that GFLv2 costs 10 milliseconds(ms) more than GFLv1[5], yet only increases a little in WAP. So GFLv1 was adopted.

To train the whole train dataset, we replaced resnet18 with GPU-efficient backbone vovnet-v2-39[3]. FPN channel was set to 128, and GFLv1 was chosen to be the detector. For Multi-scale training, cosine LR and higher LR were adopted. The initial learning rate was set to 0.1, and the batch size was set to 8. The total iterations were 700K. We used automatic mixed precision for training. We submit the result and get AP which is shown in table 3. It is good, but not enough.

Table 3. Results of one-stage detector submitted to Test

| detector | Latency(s) | AP/L1 | AP/L2 |
|---|---|---|---|
| one-stage detector | 0.0537 | 79.89 | 68.39 |

### 3.2. Two stage methods

For one-stage detector, there was still 20 ms left. We did not increase the model size, but took a two-stage algorithm which based on the one-stage model we just created.

We replaced RPN with ATSS in Faster R-CNN, which only provides proposals, we removed the *center-ness* branch and shared the heads for classification and regression. We chose double heads as RoI heads and IoU threshold was set to 0.6 for better result.

Figure1 shows the network architecture of our 2-stage detector.

The training hyper parameters were as same as previous one-stage method except the learning rate decreased to 0.05.

At inference stage, we took some strategies as shown in Table 4.

**Probabilistic two-stage detector**[10]. The final score was determined by object likelihood from ATSS RPN and classification score from ROI heads.

**Per class NMS**. Considering WOD evaluation metrics, we set non-maximum suppression (NMS) thresholds of vehicle, pedestrian, cyclists to 0.7,0.5 and 0.5 respectively.

**Calibration**. Following the implementation in [2], we set temperature T=2 to calibrate the classification score in the 2nd stage.

**Detections per image**. We set the maximum output boxes to 150 instead of 100 by default.

Table 4. Incremental trick validation results of 2-stage detector

| Strategies | WAP |
|---|---|
| baseline | 65.29 |
| + calibration | 65.70 |
| + per class NMS | 65.95 |
| + 150 detections per image | 66.35 |

## 4. Results

For final submission, the union of training and validation sets was applied to be the training data for our model. We did the inference of our model on Pytorch with half precision float point (FP16). Table 5 illustrates the result and we can observe that our solution achieves superior detection results and ranks the 1st place among all the competitors.

Table 5. Leaderboard of WOD challenge Real-time 2D Detection

| Method Name | Latency(s) | AP/L1 | AP/L2 |
|---|---|---|---|
| LeapMotor_Det | 0.0616 | 75.71 | 70.41 |
| YOLOR_TensorRT | 0.0458 | 75.00 | 69.72 |
| YOLOR_P6_TRT | 0.0374 | 74.84 | 69.56 |
| derely_self_ensemble | 0.0687 | 71.73 | 65.65 |
| YOLO_v5 | 0.0381 | 70.25 | 64.14 |

## 5. Conclusions

In this report, we present a state-of-the-art and fast 2D object detection system for autonomous driving scenarios. We used strong one-stage detector to generate better proposals rather than traditional RPN for two-stage detector. We believe that two-stage detector can perform better than one-stage detector and also run faster by careful designing. Little effort has been put into the inference of the model, we surely believe it could perform better with other inference tools like TensorRT.

## References

[1] 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 (CVPR)*, June 2016.

[2] Zehao Huang, Zehui Chen, Qiaofei Li, Hongkai Zhang, and Naiyan Wang. 1st place solutions of waymo open dataset challenge 2020 - 2d object detection track. *CoRR*, abs/2008.01365, 2020.

[3] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[4] Xiang Li, Wenhai Wang, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss V2: learning reliable localization quality estimation for dense object detection. *CoRR*, abs/2011.12885, 2020.

[5] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. *CoRR*, abs/2006.04388, 2020.

[6] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[7] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[8] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference*

*on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[9] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[10] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Probabilistic two-stage detection. *CoRR*, abs/2103.07461, 2021.

[11] Benjin Zhu, Jianfeng Wang, Zhengkai Jiang, Fuhang Zong, Songtao Liu, Zeming Li, and Jian Sun. Autoassign: Differentiable label assignment for dense object detection. *CoRR*, abs/2007.03496, 2020.