1st Place Solutions of Waymo Open Dataset Challenge 2020 2D Object Detection Track

Zehao Huang[1]   Zehui Chen[2*◇]   Qiaofei Li[3*◇]   Hongkai Zhang[4]   Naiyan Wang[1]

[1]TuSimple   [2]Tongji University   [3]Xidian University

[4]Institute of Computing Technology, Chinese Academy of Sciences

{zehaohuang18, lovesnowbest, qjliqiaofei, kevin.hkzhang, winsty}@gmail.com

Abstract

In this technical report, we present our solutions of Waymo Open Dataset (WOD) Challenge 2020 - 2D Object Track. We adopt FPN as our basic framework. Cascade RCNN, stacked PAFPN Neck and Double-Head are used for performance improvements. In order to handle the small object detection problem in WOD, we use very large image scales for both training and testing. Using our methods, our team RW-TSDet achieved the 1st place in the 2D Object Detection Track.

1. Datasets

Waymo Open Dataset (WOD) [20] is a recently public large-scale dataset for autonomous driving research. The dataset provides 1000 scenes for training and validation, and 150 scenes for testing. Each scene contains about 200 frames for each camera and there are 5 high-resolution cameras with resolution 1280×1920 and 886×1920. Overall, the dataset contains about 1.15M images and 9.9M 2D bounding boxes for vehicles, pedestrians and cyclists. Following the official split provided by WOD, we adopt 798 scenes as the training data and 202 scenes for validation in most of cases. All 5 camera images are used for training and evaluation. We do not use other datasets except for ImageNet [6].

2. Methods

In this section, we will introduce our method with bells and whistles used in this challenge. We will first analyze the object scale distribution of WOD, and then describe our baseline model setting followed by our improvements. All the experiments are conducted in SimpleDet [5].
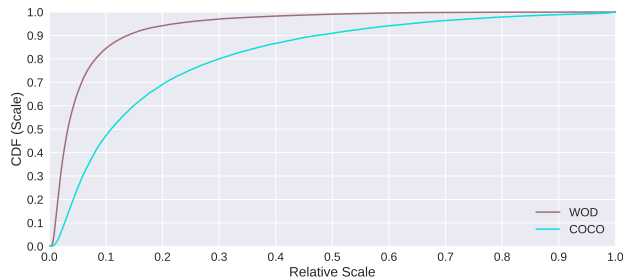

Figure 1. Fraction of bounding boxes in the dataset vs scale of bounding boxes relative to the image.

2.1. Small objects matters in WOD

Following SNIP [19], we compare the scale distribution of WOD with the most common used 2d object detection dataset, COCO [15]. As show in Fig. 1, the relative scale of objects in WOD in about 3 times smaller than COCO. According to the definition of small objects (area smaller than 32×32) in COCO, about 70% of objects[1] in WOD are small objects. Detecting very small objects is a challenging problem in 2d object detection task. Several methods have been proposed to handle this issue, such as image pyramids [19], feature pyramids [14] and so on. In the following, we will describe how we adopt these strategies to improve the detection performances.

2.2. Baseline model

We adopt FPN [14] with ResNet50-v1b [11, 12] as the baseline model. According to the observation in Sec. 2.1, we modify the default setting of anchor scale and RoI assignment strategy in FPN. The area of anchors are defined as $\{12^2, 24^2, 48^2, 96^2, 192^2\}$ pixels on $\{P_2, P_3, P_4, P_5, P_6\}$. As for the RoI assignment strategy in RCNN, we assign an RoI of width w and height h to the level P_k of the

---

[1]Assume the resolution of images in COCO is about 400×600. We define small object as the bounding box whose relative scale is smaller than $\sqrt{32 * 32 / 400 / 600} = 0.065$

feature pyramid by:

$$k = \lfloor k_0 + \log_2(\sqrt{wh}/76) \rfloor. \qquad (1)$$

The input images are resized to a short side of 640 for both training and testing. Random horizontal flip is adopted during training. By default, models are trained in a batch size of 64 on 4 nodes. Each node contains 8 2080Ti GPUs. We adopt SGD with momentum 0.9 and weight decay 1e-4 for training. The learning rate is set to 0.02 for batch size of 16. The linear scaling rule with warmup scheme [9] is adopted for larger batch size. Cosine decay [17] is used to attenuate the learning rate over time. We adopt synchronized Batch Normalization (BN) [18] in the backbone, FPN neck, and heads. Note we only calculate BN statistics across GPUs on the same node. We train the models for 8 epochs[2].

For inference, we use the score 0.03 to filter out background bounding boxes and apply the Non-Maximum Suppression (NMS) with the IoU threshold 0.5 per class to get final predictions. The baseline model achieves an mAP of 57.6 at LEVEL 1 and 50.15 at LEVEL 2 among ALL_NS (the mean over the APs of vehicles, pedestrians and cyclists.)

2.3. Bells and whistles

In this section, we show the step-by-step improvements of different components adopted in our methods. Results on WOD validation set can be found in Table. 3. Note that we adopt score first matcher for fast evaluation.

Multi-scale training. We adopt multi-scale training with random crop. The scale of short side is randomly sampled from {512, 640, 960, 1280} and the scale of long edge is fixed as 2000. After rescaling, images are then cropped to a fixed size of 640×960. For image with annotations, we randomly choose an object and jitter the object center as crop center. As for background image, we do random crop.

Large scale testing. In order to boost the performance of small objects, we adopt a large scale (1280, 1920) for testing. As shown in Table.3, large scale testing achieves 60.37 mAP at LEVEL 2 among ALL_NS, which outperforms the small scale results by 6.7.

Cascade RCNN. WOD adopt different intersection over union (IoU) thresholds to define positives and negatives for different classes, e.g. 0.7 for Vehicles, 0.5 for both Pedestrians and Cyclists. In order to generate high quality detection, we use Cascade RCNN [3]. Following the original implementation, we set the IoU thresholds to 0.5, 0.6 and 0.7 for each RCNN stage respectively. We also try different IoU thresholds, and find that the default setting yields best performance.

Double-Head. We adopt a Double-Head [22] method to replace the default fully connected head (*2-fc*) in FPN. For

Table 1. Step-by-Step improvements on WOD validation set. We report the results of different classes at LEVEL 2.

Methods	Vehicle	Pedestrian	Cyclist	ALL_NS
Baseline	50.74	64.72	34.99	50.15
+ Multi-scale training	51.67	67.19	42.16	53.67
+ Large scale testing	57.33	72.90	50.88	60.37
+ Cascade RCNN	61.10	73.67	52.00	62.26
+ Doubel Head	62.79	74.97	51.96	63.24
+ PAFPN	63.78	75.81	48.62	62.74
+ Training tricks	64.21	75.56	51.51	63.76
+ More scales training	64.02	75.77	54.38	64.72
+ Res2Net50-v1b	64.19	76.16	54.93	65.09
+ Calibration and flip	64.98	77.40	57.98	66.79
+ Scale-aware testing	66.38	78.76	61.66	68.93
+ NMS and Voting	68.99	78.72	61.20	69.64
+ 3 models ensemble	**69.81**	**79.00**	**62.62**	**70.48**

each RCNN stage in Cascade RCNN, a *2-fc* head is used for classification and a convolution head (*conv-head*) with 3 stacked residual blocks is used for bounding boxes regression.

PAFPN. Following PANet [16], we construct PAFPN module by adding bottom-up path aggregation upon FPN. Instead of repeating only once in the original paper, we stack this module 3 times to enhance feature representation.

Training tricks. (1) We do not append ground truth into proposals in the RCNN stage during training. (2) Following the training hyper-parameters described in MMDetetcion 2.0 [4], we change the number of proposals after nms from 2000 to 1000.

Multi-scale training with more scales. We adopt more scales for multi-scale training. The scale of short side is randomly sampled from {640, 960, 1280, 1600, 1920, 2240, 2560} and the scale of long edge is fixed as 3840.

Calibration. We find that our FPN models tend to predict overconfident classification score on WOD. The scores of many predictions are nearly 1.0. Such score distribution is not suitable for AP metric, since the PR curve may not be smooth at high precision interval in WOD evaluation setting. To optimize for AP, inspired by [10] and [13], we calibrate the classification score by adopting a temperature $T = 2$ to produce a softer score distribution over classes.

Scale-aware testing. Multi-scale testing is a widely adopted trick for 2d object detection. We adopt two scales for testing, (1280, 1920) and (2240, 3360). By analyzing results obtained from different scales, we found that testing large scale image yields higher mAP on small objects[3] while the small scale one yields higher mAP on large objects. Based on this observation, we degrade the classification score by multiplying a constant value of 0.6 for small scale predictions when testing with (1280, 1920). As for scale (2240, 3360), we decay the score of large scale ob-

---

[2]we train about 12 epochs for the baseline model, and 8 epochs for the remain experiments.

---

[3]For vehicles, we define a bounding box whose area smaller than 67 × 67 as a small object. For pedestrians and cyclists, we set the area threshold to 57 × 57.

Table 2. The performances of expert models and final ensemble model on the WOD validation set. The results are AP at LEVEL 2 for different classes.

Model	Vehicle	Pedestrian	Cyclist	ALL_NS
4 expert models ensemble	68.61	78.54	62.01	69.72
3 models ensemble	69.81	79.00	62.62	70.48
Final ensemble	**70.13**	**79.07**	**63.31**	**70.90**

Table 3. Submission results on WOD testing set. We report the results of different classes at LEVEL 2. The final submission is an ensemble of models trained on the whole dataset and several expert models trained on a subset of the dataset.

Methods	Vehicle	Pedestrian	Cyclist	ALL_NS
Baseline	56.34	67.63	36.92	53.63
Final submission	**76.63**	**81.72**	**64.94**	**74.43**

jects. This simple strategy brings 1.0 improvement than vanilla multi-scale testing.

NMS and Voting. We set NMS threshold as 0.7,0.5,0.5 for vehicles, pedestrians and cyclists respectively. Bounding box voting [8] is applied after NMS. For simplicity, the threshold of voting is set to the same as NMS threshold. We also try Soft-NMS [2]. Comparing to valina NMS with bounding box voting, Soft-NMS yields similar results but is inefficient with numerous predictions.

Ensemble. We find that the performance between very deep convolutional neural networks, including ResNet101 and ResNet152, and ResNet-50 is not significant. Especially for cyclists, both ResNet101 and ResNet152 yield worse performance than the shallow one. For training efficiency, we adopt three models, Res2Net50-v1b[4] [7], ResNet50-v1d[5] [12] and ResNeXt50[6] [23] for ensemble. We compare the performances of two different ensemble strategies on validation set. The first one is described in PFDet[1] and the other one, named Linear-Reweight, is to assign each model a weight of w_i based on its AP rank k on validation set among n candidates:

$$w_i = \theta_0 + (n - k) \cdot \frac{1 - \theta_0}{n - 1}. \tag{2}$$

Note that θ_0 is set to 0.5 for all classes. Based on our results, Linear-Reweight is slightly better than the method proposed by PFDet. So we adopt Linear-Reweight as our final ensemble strategy.

2.4. Expert model

We train several expert models for cyclists since the number of cyclist objects in WOD is only 81k, which is much less than vehicles and pedestrians (9.0M and 2.7M). We do class-aware sampling during training. For fast experiments, we randomly sample 120K images from the whole training dataset each epoch until the ratio between different classes is 1:1:1:0.3 for cyclists, pedestrian, vehicle and background images. We train expert models for 20 epoch. Comparing to models trained on the whole training set, expert models use much less training samples while provide

competitive performances. The results can be found in Table.2. We combine 4 expert models and 3 models trained on the whole dataset. The final ensemble model yields 70.90 mAP on validation set.

3. Final Results

For final submission, we train multiple models on the union of training and validation sets, including ResNet50-v1b, ResNet50-v1d, Res2Net50-v1b, HRNetv2p-W32 and HRNetv2p-W48[7] [21]. All the models are pretrained on ImageNet. Linear-Reweight strategy is adopted for final ensemble. Results are shown in Table.3. The final model achieves 74.43 at LEVEL 2 among ALL_NS in WOD testing set.

References

[1] Takuya Akiba, Tommi Kerola, Yusuke Niitani, Toru Ogawa, Shotaro Sano, and Shuji Suzuki. PFDet: 2nd place solution to open images challenge 2018 object detection track. *arXiv preprint arXiv:1809.00778*, 2018.

[2] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *ICCV*, 2017.

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *CVPR*, 2018.

[4] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[5] Yuntao Chen, Chenxia Han, Yanghao Li, Zehao Huang, Yi Jiang, Naiyan Wang, and Zhaoxiang Zhang. SimpleDet: A simple and versatile distributed framework for object detection and instance recognition. *JMLR*, 2019.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[7] Shanghua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip HS Torr. Res2Net: A new multi-scale backbone architecture. *TPAMI*, 2019.

[8] Spyros Gidaris and Nikos Komodakis. Object detection via a multi-region and semantic segmentation-aware CNN model. In *ICCV*, 2015.

[9] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch,

---

[4] https://github.com/Res2Net/Res2Net-PretrainedModels

[5] https://gluon-cv.mxnet.io/model_zoo/classification.html

[6] https://github.com/apache/incubator-mxnet/tree/master/example/image-classification

[7] https://github.com/HRNet/HRNet-Image-Classification

Yangqing Jia, and Kaiming He. Accurate, large mini-batch SGD: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[10] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *ICML*, 2017.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[12] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *CVPR*, 2019.

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning Workshop*, 2014.

[14] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

[16] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *CVPR*, 2018.

[17] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[18] Chao Peng, Tete Xiao, Zeming Li, Yuning Jiang, Xiangyu Zhang, Kai Jia, Gang Yu, and Jian Sun. MegDet: A large mini-batch object detector. In *CVPR*, 2018.

[19] Bharat Singh and Larry S Davis. An analysis of scale invariance in object detection snip. In *CVPR*, 2018.

[20] 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. *arXiv*, pages arXiv–1912, 2019.

[21] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *TPAMI*, 2020.

[22] Yue Wu, Yinpeng Chen, Lu Yuan, Zicheng Liu, Lijuan Wang, Hongzhi Li, and Yun Fu. Double-Head RCNN: Rethinking classification and localization for object detection. In *CVPR*, 2020.

[23] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.