

Baseline for 2D Object Detection and Tracking

Cascade R-CNN X152 and SORT

Yuan Xu, Erdene-Ochir Tuguldur
DAI-Labor, Technische Universität Berlin
Ernst-Reuter-Platz 7, 10587 Berlin, Germany

{yuan.xu, tuguldur.erdene-ochir}@dai-labor.de

Abstract

This paper explores efficient and reliable baseline for 2D tracking in autonomous driving: implementation of a tracking-by-detection framework only using combination of existing techniques: Cascade R-CNN X152 and Simple Online and Realtime Tracking. Our detector and tracker has been trained and tuned on Waymo open dataset, and finished in 3rd place of Waymo open dataset challenge 2D tracking track. The code is available at https://github.com/xuyuan/waymo_2d_tracking.

1. Introduction

Autonomous navigation of robots and cars requires appropriate models of their static and dynamic environment. The task of 2D tracking is producing a set of 2D boxes and the correspondences between boxes across given a temporal sequence of camera images.

This work focuses on efficient and reliable implementation of a tracking-by-detection framework only using combination of existing techniques. Rather than aiming to solve the problem in end-to-end fashion, we instead use state-of-art detector to find objects of a given category and a multi-hypothesis tracker to handle data association in crowded scenes. An important advantage of knowing the object category is that one can resort to category-specific, physically meaningful dynamic models for tracking.

2. Detection: Cascade RCNN X152

Since detection quality is identified as a key factor influencing tracking performance, we have chosen state of art Cascade R-CNN[3] to produce high quality detection boxes. We also want to use big backbone, such as senet[5] for better accuracy. Nevertheless, we use transfer learning to reduce enormous resources required to train such deep models on such large dataset.

Our *Cascade RCNN X152* model was COCO[6] pre-trained from detectron2[8] model zoo. The full description of architecture can be found in ¹. The mask header has been removed, and trained on Waymo open dataset by customized pipeline with bag of tricks:

- The input images were random cropped to fixed resolution of 1280×886 for fitting 1 image per GPU during training.
- Due to the deep model used, heavy data augmentation was used to combat with overfitting; we have applied various well known methods, including: re-size, color, cutout[4], noise, deform and policies from AutoAug[9]. However, we didn't have computation resources for ablation study on Waymo open dataset.
- Because of big amount of data, it is prohibitive for us to train on every frame. Progressive dataset sampling was applied: the model was trained on 1/10th of full dataset for first 30 epochs, and then tuned on 1/3rd of full dataset for 3 additional epochs.
- In each phase of training, cosine annealing learning rate with warm restarts[7] was used. The base learning rate was $2e-3$ with 1 epoch warm up.

Rest of training settings are from detectron2's common settings for COCO models. The whole training took 180 hours in a server with 8 NVIDIA TITAN RTX GPUs. The software in use were PyTorch 1.5, CUDA 10.2, cuDNN 7.6.5, detectron2 0.1.3. Tensorboard training logs are provided in code repository for reference.

Last but not least, test augmentation with horizontal flip, auto contrast and multi-scale images has been used to generate total 13 individual inferences with 3 model checkpoints;

¹https://github.com/facebookresearch/detectron2/blob/master/configs/Misc/cascade_mask_rcnn_X_152_32x8d_FPN_IN5k_gn_conv.yaml

data	TTA	Vehicle		Pedestrian		Cyclist		All_NS	
		L1	L2	L1	L2	L1	L2	L1	L2
validation	no	0.6675	0.5599	0.7122	0.6621	0.5643	0.4761	0.6480	0.5660
	yes	0.7000	0.6000	0.7703	0.7266	0.6189	0.5304	0.6964	0.6190
testing	yes	0.7483	0.6583	0.7986	0.7694	0.6321	0.5664	0.7264	0.6647

Table 1. Detection results on Waymo open dataset.

data	Type	MOTA		MOTP		FP		Mismatch		Miss	
		L1	L2	L1	L2	L1	L2	L1	L2	L1	L2
validation	Vehicle	0.5064	0.4100	0.1184	0.1184	0.0698	0.0565	0.0240	0.0194	0.3998	0.5140
	Pedestrian	0.5081	0.4675	0.1868	0.1868	0.0802	0.0738	0.0433	0.0399	0.3684	0.4188
	Cyclist	0.4304	0.3566	0.1363	0.1363	0.0661	0.0548	0.0140	0.0116	0.4894	0.5771
	All_NS	0.4816	0.4114	0.1472	0.1472	0.0720	0.0617	0.0271	0.0236	0.4192	0.5033
testing	Vehicle	0.5704	0.4742	0.1142	0.1142	0.0625	0.0519	0.0222	0.0184	0.3450	0.4554
	Pedestrian	0.5123	0.4853	0.1962	0.1962	0.1042	0.0987	0.0435	0.0412	0.3400	0.3747
	Cyclist	0.4238	0.3651	0.1350	0.1350	0.0670	0.0577	0.0157	0.0135	0.4936	0.5637
	All_NS	0.5022	0.4415	0.1485	0.1485	0.0779	0.0694	0.0271	0.0244	0.3928	0.4646

Table 2. Tracking results on Waymo open dataset.

the final submission is ensemble of them by linear Soft-NMS[2]. Finally, our submission ranks 9th in 2D detection leaderboard, see Table 1 for evaluation metrics in detail.

3. Simple Online and Realtime Tracking

Simple Online and Realtime Tracking (SORT)[1] uses a linear constant velocity model to approximate the inter-frame displacements of each object which is independent of other objects and camera motion. Despite only using combination of familiar techniques such as the Kalman Filter and Hungarian algorithm for the tracking components, this approach achieves an accuracy comparable to state-of-the-art online trackers.

However there are number of parameters which has impact on tracking performance, for example:

score threshold minimal score of detection to be tracked, it reduces number of trackers and false positive detections.

IOU threshold minimum IOU is imposed to reject assignments;

min hits probationary period where the target needs to be associated with detection to accumulate enough evidence in order to prevent tracking of false positives;

max age maximal number of frames without detection which prevents an unbounded growth in the number of trackers and localization errors caused by predictions overlong duration without corrections from the detector.

	Vehicle	Pedestrian	Cyclist
score threshold	0.95	0.60	0.90
IOU threshold	0.01	0.01	0.00
min hits	0	0	0
max age	2	2	2

Table 3. Category-specific optimized parameters.

We used grid search to determine optimal parameters for each category on validation dataset. With parameters in Table 3, SORT achieves an accuracy comparable to state-of-the-art trackers in the Waymo open dataset challenge. Evaluation metrics of best validation and final testing submission are detailed in Table 2.

4. Conclusion

This paper explores implementation of a tracking-by-detection framework only using combination of existing techniques: Cascade R-CNN X152 and SORT. Our detector and tracker has been trained and tuned on Waymo open dataset, and finished in 3rd place of Waymo open dataset challenge 2D tracking track. The result shows that the tracking quality is highly dependent on detection performance and by capitalizing on recent developments in detection, state-of-the-art tracking quality can be achieved with classical tracking methods. Code is open sourced to help establish a baseline for research experimentation.

References

- [1] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Uppcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, 2016.
- [2] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S. Davis. Improving object detection with one line of code. *CoRR*, abs/1704.04503, 2017.
- [3] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: delving into high quality object detection. *CoRR*, abs/1712.00726, 2017.
- [4] Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017.
- [5] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CoRR*, abs/1709.01507, 2017.
- [6] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.
- [7] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with restarts. *CoRR*, abs/1608.03983, 2016.
- [8] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019.
- [9] Barret Zoph, Ekin D. Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V. Le. Learning data augmentation strategies for object detection. *CoRR*, abs/1906.11172, 2019.