EFFECTIVE OBJECT DETECTION FROM TRAFFIC CAMERA VIDEOS

Honghui Shi, Zhichao Liu, Yuchen Fan, Xinchao Wang, Thomas Huang*

Image Formation and Processing (IFP) Group,
University of Illinois at Urbana-Champaign

ABSTRACT

The Nvidia AI City Challenge[1] is an effort to materialize the potential benefits of actionable insights in current traffic systems by utilizing both large amount of richly annotated traffic camera video data captured by growing number of cameras, and advanced technologies developed in recent years in image and video recognition and analysis. In this work, we demonstrate the details of the solution of our winning entry to the 1st Nvidia AI City Challenge detection track. Our code and models are also available at `https://github.com/NVIDIAAICITYCHALLENGE/AICity_Team24`.

***Index Terms*—** Object detection, vehicle detection, pedestrian detection, traffic light detection, traffic cameras, video analysis

1. INTRODUCTION

In modern traffic systems, effective detection of vehicles, pedestrians, traffic lights, etc. have become an essential part in various computer vision applications. For example, both traffic analysis and autonomous driving cars now depend on vehicle and pedestrian detection to get the basic information.

In the past few years, a number of datasets were proposed to help solve some of the object detection problems in traffic systems. For instance,KITTI[2] and UA-DETRAC[3] could be used by researchers to evaluate vehicle detector performances.

Recently, the Nvidia AI City Challenge (AIC)[1] proposed a brand new dataset that further the effort with a deeper and more comprehensive goal. First, the Nvidia AIC dataset consists of traffic camera videos with high quality (1080p and 480p). Second, the AIC dataset has much richer labels: fine-grained vehicle types, traffic lights, pedestrians and bikes, and etc.. Third but not last, the AIC dataset has various lighting conditions, different camera angles to test the robustness and effectiveness of modern high-quality object detection algorithms. In Figure 1, we demonstrate two different example frames of the videos from AIC dataset.

While the accuracy of object detection is at the core of computer vision applications for traffic systems such as autonomous driving due to its direct link to safety, we also keep efficiency in mind when we participate the Nvidia AI City Challenge detection track. We adopted two different algorithms that are built upon on region-based object detection framework, which achieves state-of-the-art performance on object detection tasks from other common object detection datasets such as PASCAL VOC and ImageNet[4].

* visiting student at UIUC, senior at EECS, Peking University

Fig. 1: Examples of Nvidia AI City Challenge dataset

More specifically, we applied two novel region-based deep learning algorithms, in which the object boxes are first generated and then classified in the next stage, the Faster RCNN[5] and Region-based Fully Convolutional Networks (R-FCN)[6]. Both algorithms use the ImageNet pre-trained 101-layer Residual Network[7] as backbones. In both networks, the first region proposal network (RPN) employs a set of convolutional layers to generate candidate boxes. Potential boxes are then sent to the second network in which two different frameworks are leveraged. In Faster RCNN networks for regression and classification are followed, and in R-FCN a voting classification network based on score maps is applied.

Our final models achieved the best performances in the Nvidia AI City Challenge detection track, with a $10 - 40\%$ higher mean average precision(mAP) compared with other solutions.

The rest of this paper is organized as follows. Section 2 introduces the background and related works for object detection in traffic systems. Section 3 discusses the algorithms used in our implementation in detail, Section 4 shows the experiment details and results, and Section 5 concludes the report.

2. RELATED WORK

Vehicle detection, as a special case of object detection for traffic systems, has been studied for decades and has drawn considerable attention due to its potential commercial value for practical applications.

Over the years, methods of vehicle detection have seen a lot of changes. In early years, part-based models succeeded in several datasets. When Deformable Part-based Models (DPM) [8] was introduced for pedestrian and vehicle detection tasks, the performance on generic object detection datasets such as PASCAL VOC[9] was increased as well. DPM detectors successfully detect the target objects by dividing objects into several parts and finding their spatial combinations.

More recently, region-based convolutional networks [10][5] Nevertheless, under this framework, simultaneously detecting and classifying vehicles robustly under different poses, lighting conditions and car types is still a difficult task.

Recently, Convolutional Neural Networks (CNNs) have gained plenty of attention and shown their advantage compared to conventional methods on many computer vision tasks. To leverage the representative power of CNNs to detection tasks, Region-based Convolutional Neural Networks (RCNN) [10] was proposed and had made considerable improvements and achieved state-of-the-art performance on many generic object datasets. In such framework, CNNs are applied for effective feature extraction. To reduce candidate object proposals and further improve efficiency in RCNN, the Faster RCNN[5] was proposed with Region Proposal Network (RPN) and achieves new state-of-the-art performance.

More recently, several object detection frameworks utilize methods based on anchors or grids to generate potential object locations, which balance the speed and accuracy and make real-time end-to-end object detections possible. YOLO[11] uses regression method to get the objects' location for each image. SSD[12] takes advantage of fully convolutional networks and multi-scale feature map. R-FCN[6] improves Faster RCNN by using fully convolutional networks with position-sensitive score maps to tackle the bottleneck in the classification part of object detection systems.

In the following chapters, we applied the Faster RCNN and R-FCN algorithms on Nvidia AI City Challenge dataset and achieved top performance on all subsets of the detection task.

3. METHODS

In this section, we introduce the two different object detection networks we used as illustrated in Figure 2 and Figure 3.

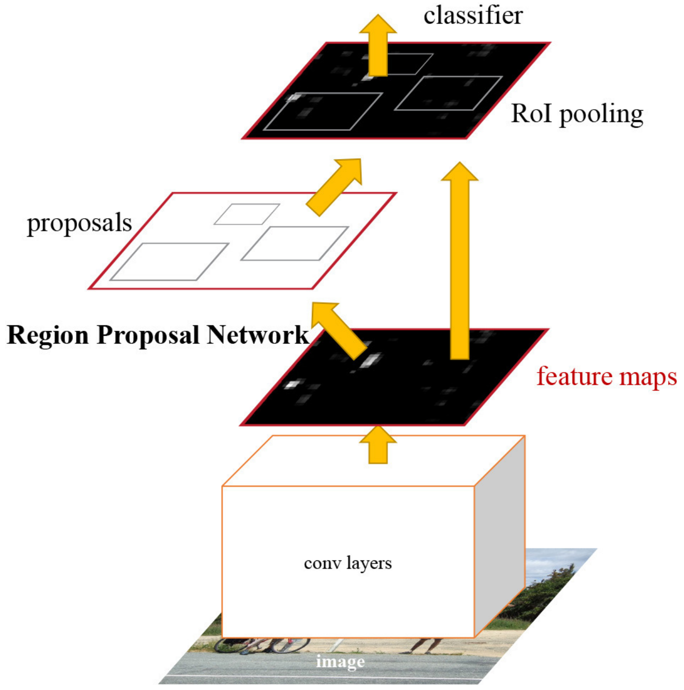

Fig. 2: Faster RCNN[5]

3.1. Method 1: Faster R-CNN

As we can see in Figure 2, the input video frame is first fed into several convolutional layers to generate feature representations. Feature maps are produced by this network. We use the convolutional and max-pooling layers from the ResNet-101 network[7] and the weights pre-trained on the ImageNet dataset[4]. Many state-of-the-art deep learning frameworks use this method to leverage the power from the huge classification dataset.

The output feature maps are then sent to the region proposal network (RPN) where thousands of anchors are generated and the RPN gives the scores and coordinates. Specifically, the image is divided into grids; with each grid as a center, several anchor proposals are generated. For proposal by the anchors, the ROI pooling layer crops out a part of features on the output feature maps and use a regression module to generate the initial bounding boxes and their object scores. Only the proposals with top scores are sent to the third part, which is another regression network for the final scores and coordinates.

As shown in Figure 2, the RPN takes the feature maps from the previous network as input and output the scores and coordinates to the third network.

3.2. Method 2: R-FCN

Since the task of the challenge is to make a classification algorithm among several different car types, increasing the power of the classification part is essential to building an efficient

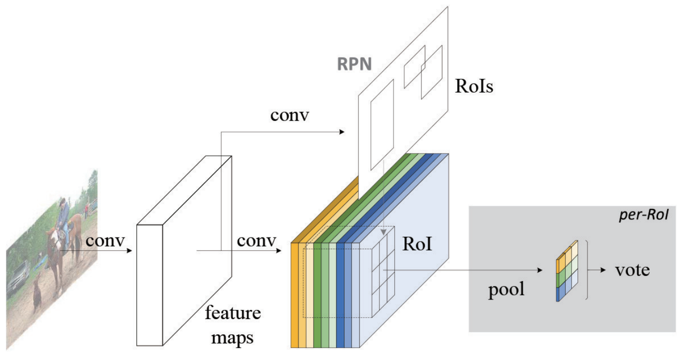

Fig. 3: R-FCN[6]

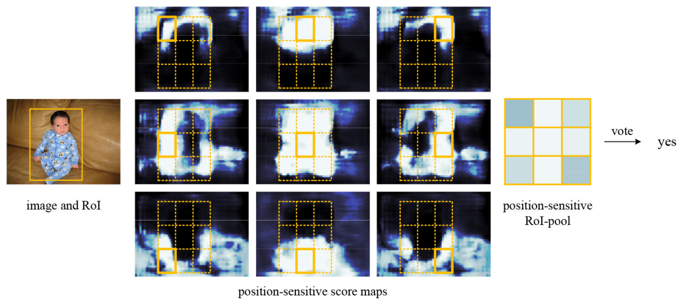

Fig. 4: Visualization of R-FCN score maps for the person category[6]

vehicle detection system for vans, SUVs and different-sized trucks. To achieve this goal, we applied R-FCN as our second model on the challenge to classify the proposed candidate boxes.

Specifically, R-FCN shows improvement in two part of the original region-based models, for example, Faster R-CNN. It first leverages a fully convolutional deep network to output the feature maps. The most important part is that unlike the Faster RCNN, R-FCN doesn't directly use a convolutional network as the regression part for candidate objects. Instead of that, R-FCN uses *kxk* position-sensitive score maps generated by the fully convolutional network mentioned before. The k^2 parts of the score maps encode the cases of 9 different corresponding position of the bounding boxes. In the third part of the framework, k^2 score maps are aggregated to vote for a final score for the proposal in Figure 3.

As shown in Figure 4, the score maps learned by R-FCN show the sensitive parts of the bounding boxes which are considered to be strongly activated when a specific relative position of the object is just on that part. For example, the left-top part of the bounding box in one of nine feature maps, which has only the left-top information particularly, should be strongly activated when and only when the bounding box is just on the right object. Finally, when most of the nine maps are activated, a high score will be generated. The method can reduce the location error brought by the translation invariance of convolutional neural networks and improve the classification power simultaneously.

We find in our experiments that by using the R-FCN framework, we can achieve considerable improvement on the accuracy of this fine-grained classification problem and the voting score maps architecture is superior to directly regressing the proposals, which is done in Method 1.

The details of the implementation of the two methods are introduced as follows.

4. EXPERIMENTS

4.1. Dataset

We trained our model on the provided detection datasets with 89700 labeled frames and 785258 bounding boxes. It contains 78754 images for 1920*1080 frames and 11016 for 720 * 480 ones. We further split 1080p dataset into the training set with 71594 images and the validation set with 7160 images and split 720p dataset into the training set with 10014 images and the validation set with 1002 image. Since 540p images are just the down-sampling version of 1080p dataset, we don't use that in both training and evaluating. The challenge still provides DETRAC dataset[3], but we didn't use that as a part of training set since its labels only contain 4 types of cars.

The dataset is challenging due to its variation; the cameras are mounted on traffic poles in Silicon Valley and the videos are taken in different light condition including sunny, cloudy, and night with different camera angles. Figure 5 shows some statistics information of the training set. The dataset is unbalanced in both labels and resolution.

4.2. Experiment

To evaluate the effectiveness of our models, we focus on two key points on the implementation: (1) how different resolution affect the detection performance, and (2) how different model affect the detection performance. We show mean average precisions (mAP) on the total validation set as well as the two subsets.

4.3. Implementation Details

For a fair comparison, every experiment is reported on the validation set with Resnet-101[7] pretrained on ImageNet Classification Dataset as the backbone feature network. Faster R-CNN framework is implemented on Tensorflow, and the models are trained on Nvidia Tesla P100 GPU provided by the organizers for 560000 iterations on every sub-dataset. R-FCN framework is implemented on Caffe[13], and the models are trained on Nvidia Tesla P100 GPU for 560000 iterations

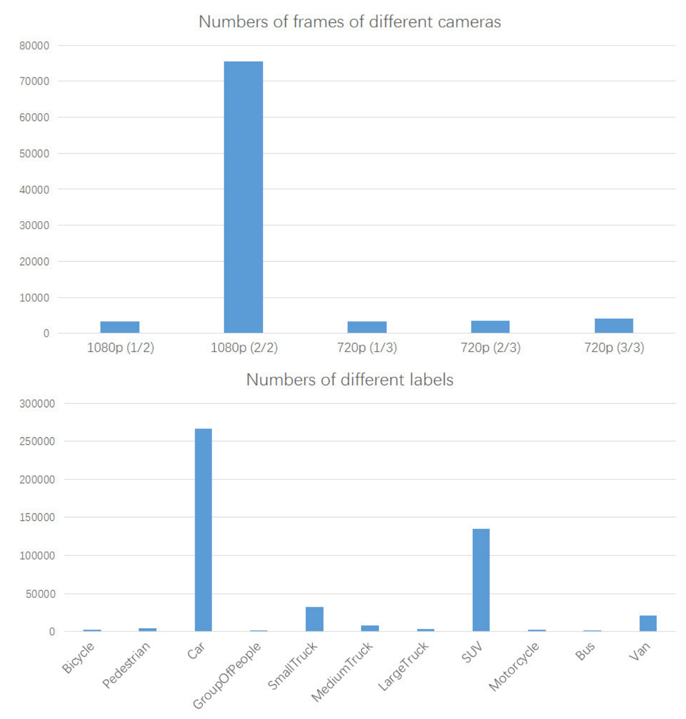

Fig. 5: Statistics information on the training set

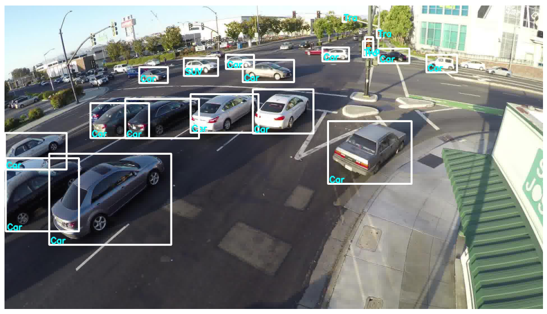

Fig. 6: Successful example from test dataset

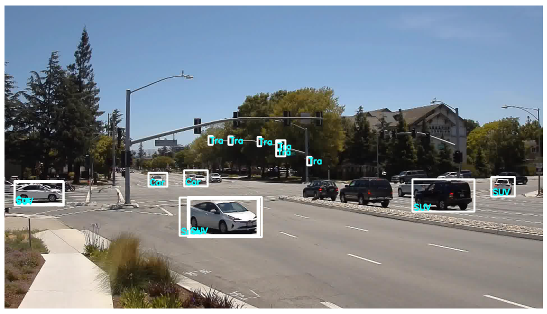

Fig. 7: Partially unsuccessful example from test dataset

on every sub-dataset. Online hard example mining and basic data augmentation are used in training.

As we mentioned before we use 1080p dataset, 480p dataset and their combination to train two different models.The evaluation of detection performances includes the overall mAP and mAPs under different classes.

Quantitatively speaking, R-FCN model with all dataset trained together performs the best among different training settings. As the Faster RCNN does not apply the score map after the RPN net, its performance drops significantly for almost 10% mAP compared with our best model. This accords with our speculation in the above sections.

We consider this model to be useful for vehicle detection, as many types of vehicles usually only show some difference on small details of the whole object.

4.4. More on Networks Training

In the beginning, our networks are initialized by the pre-trained ImageNet model ResNet-101 for the backbone feature network, and the other networks are randomly initialized from a Xavier initialization[14] with standard deviation factor of 0.01. We set the initial learning rate to 10^{-4} in method 1 and 0.001 in method 2. We totally train 500k iterations. Method 1 is trained using ADAM optimizer[15] while model 2 is trained using stochastic gradient descent. We assign positive proposals that overlap the ground truth bounding boxes for more than 50% on intersection over union. We also apply online hard example mining[16] during training the two models. Considering the relatively fixed position of the traffic signals and their different feature compared with cars and motorcycles, we train models on the traffic light data separately from other categories.

4.5. Results

Table 1 demonstrate the comparison of our two models trained on three different sub-datasets. Precision-recall curves and mAPs are reported. We achieve a significant overall improvement of 10% mAP by R-FCN over Faster RCNN. Notably, all our models perform the best when trained on the whole dataset.

Figure 6 and Figure 7 demonstrate some demo images of our approach on the test set; successful and partially unsuccessful results are shown. In Figure 7, one car is recognized as an SUV, which suggests that the bottleneck of this detection algorithm might still be the classification performance.

We successfully detected most of the vehicles with different appearances, especially the little objects such as motorcycles and bicycles. However, there are also plenty of failure cases where the detection framework fails to identify different kinds of vehicles such as van and SUV. Generally speaking, the detection results are robust in different resolution and light conditions but still need more work on the fine-grained vehicle classification.

Class	Car	SUV	S-truck	M-Truck	L-Truck	Man	Bus	Van	GoP	Bike	Motor	Mean
Faster RCNN, 1080p	0.785	0.707	0.722	0.526	0.465	0.222	0.522	0.507	0.406	0.389	0.625	0.534
R-FCN, 1080p	0.815	0.747	0.759	0.560	0.569	0.398	0.575	0.561	0.497	0.628	0.845	**0.632**
Faster RCNN, 480p	0.657	0.644	0.660	0.446	0.468	0.318	0.750	0.444	N/A	0.424	0.790	0.418
R-FCN, 480p	0.707	0.695	0.694	0.511	0.489	0.260	0.742	0.461	N/A	0.212	0.777	**0.504**

Table 1: Results of the two methods

5. CONCLUSION AND FUTURE WORK

We have successfully applied two region-based models on the Nvidia AI Challenge dataset, evaluated their performance under different training situations and achieved state-of-the-art performance on the dataset. We hope that our experiments and code release can support follow-up works on the dataset, which will further improve the performance on the dataset. The frameworks we use, while achieving a competitive result, still have much room for improvement. Our future work may include a coarse-to-fine detection framework to improve the recall rate and reduce classification mistakes.

6. ACKNOWLEDGEMENT

This work is supported in part by IBM-ILLINOIS Center for Cognitive Computing Systems Research (C3SR) - a research collaboration as part of the IBM Cognitive Horizons Network.

7. REFERENCES

[1] IEEE Smart World Nvidia AI City Challenge, “`http://smart-city-conference.com/AICityChallenge/`,” 2017.

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[3] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu, “UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking,” *arXiv CoRR*, vol. abs/1511.04136, 2015.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, “Imagenet: A large-scale hierarchical image database,” *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[5] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1137–1149, 2015.

[6] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun, “R-fcn: Object detection via region-based fully convolutional networks,” in *NIPS*, 2016.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[8] David A. Forsyth, “Object detection with discriminatively trained part-based models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1627–1645, 2010.

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, June 2010.

[10] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, 2014.

[11] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi, “You only look once: Unified, real-time object detection,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, 2016.

[12] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg, “Ssd: Single shot multibox detector,” in *ECCV*, 2016.

[13] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, “Caffe: Convolutional architecture for fast feature embedding,” *arXiv preprint arXiv:1408.5093*, 2014.

[14] Xavier Glorot and Yoshua Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *AISTATS*, 2010.

[15] Diederik P. Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2014.

[16] Abhinav Shrivastava, Abhinav Gupta, and Ross B. Girshick, “Training region-based object detectors with online hard example mining,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 761–769, 2016.