Improve Object Detections in Traffic Scenes

Adwin Jahn

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2017

Committee:

Jenq-Neng Hwang, Chair

Linda Shapiro

Program Authorized to Offer Degree:
Electrical Engineering

University of Washington

Abstract

Improve Object Detections in Traffic Scenes

Adwin Jahn

Chair of the Supervisory Committee:
Professor Jenq-Neng Hwang
Electrical Engineering

Transportation is one of the largest area that can benefit from actionable insights derived from traffic image data. Object detections in traffic scenes aim to extract accurate localization and classification information. Since recognition and tracking often rely on the results from detection, a fast and accurate object detection framework can make transportation systems safer and smarter. However, there are several major road blocks for object detection in traffic scenes when training model in real world traffic dataset: (1) state-of-the-art deep neural network relies on large and well labeled supervised data (2) far away and crowded objects are hard to precisely located (3) quality of crowd-sourcing annotation is no guaranteed (4) some categories only have few training instances for training. To address the issues, a multi-window method merging with pre-trained model is proposed. Experiments on NVIDIA AI City Dataset shows that the proposed method has 12 to 57 % of average precision improvement for categories with few training instances and 3% of mean average precision improvement. It is more clear to understand the improvement through the qualitative results in section 3.3.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

By the rapid growth of the amount of visual data in recent years, computer vision researchers and developers are able to apply data driven and supervised algorithm to address real world problem.

This chapter will introduce the background of object detections in traffic scenes and the Nvidia AI city dataset which contains over 1.4 million objects in 150,000 key-frames with labels and the challenge and related work for object detections in traffic scenes.

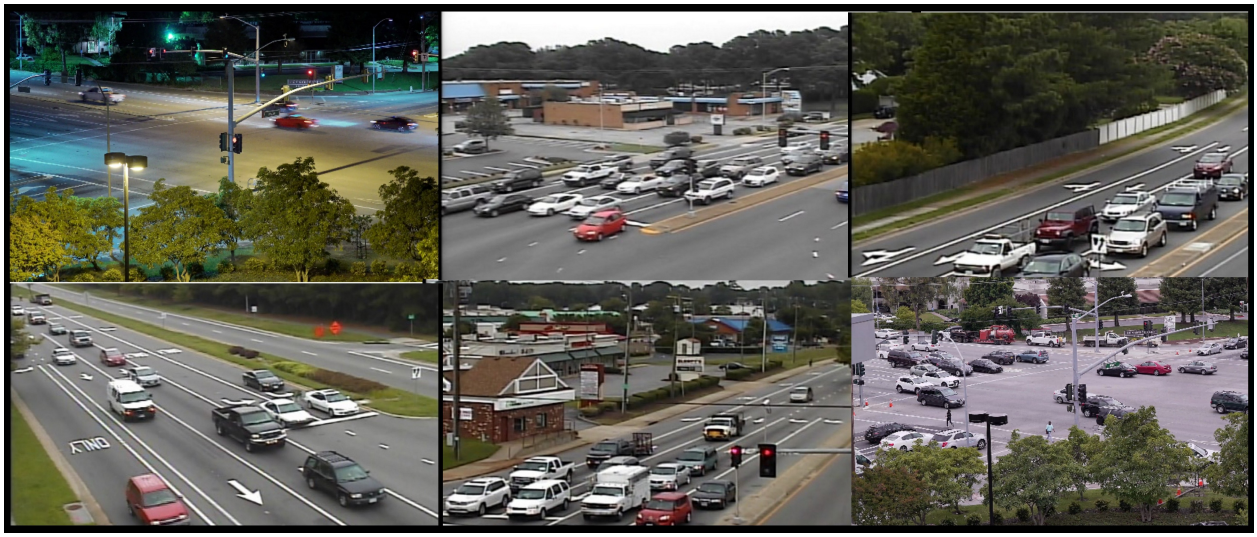

Figure 1.1: Nvidia AI city dataset

1.1 Background

The need of building an AI city drives both academia and industry to solve the problem together. Since 2017 June, IEEE and Nvidia Corporation cooperated and initialed a huge

crowd-sourcing traffic scene visual dataset called IEEE NVIDIA AI City Data Set with more than 150,000 key-frames extracted from over 80 hours of traffic video captured at various intersections in Santa Clara, and San Jose, California, Lincoln, Nebraska, and Virginia Beach, Virginia.

There are 14 categories, which are common objects in traffic scenes, annotated in the AI city dataset: Car, SUV, SmallTruck, MediumTruck, LargeTruck, Pedestrian, Bus, Van, GroupOfPeople, Bicycle, Motorcycle, TrafficSignal-Green, TrafficSignal-Yellow and TrafficSignal-Red.

The AI city dataset integrates frames from 480 and 1080 pixels resolution sources. The 1080 pixels resolution source is duplicated and scale down as a new source of 540 pixels resolution. The design tries to prevent over fitting algorithm and demand an more sophisticated algorithm. To be consistent, 480 pixels resolution sources will be expressed as aic480 and the same rules apply to 540 and 1080 pixels.

This thesis adopts the AI city dataset to develop object detection framework because of its great amount of annotations and images and focus on how to improve the accuracy.

1.2 Challenge

Object detections in traffic scenes aim to optimize the performance evaluated by mean average precision. However, there are several challenges even for the latest dataset of traffic scene images.

First, the quality of annotation is not guaranteed. In 1.2, the two images are very close frames with ground truth bounding boxes. However, the annotator for right frames skipped many objects he should annotate. If an far away object takes small number of pixels and lies around the boundary of recognizable and not recognizable, it is a hard task for annotators. But, more often, even for easily recognizable objects, annotators tend to skip them when there are too many objects in a frame.

Second, annotators have different standard for labeling. In 1.3, both frames are from ground truth image served as training data. However, the annotator tends to draw larger

bounding box than needed in left frame while smaller bounding box in the right one. These kinds of bad annotations will harm the localization performance of framework that targets to learn from data.

For the above two properties in dataset, it will be called lazy annotator effect in the following chapters and this work will present the method to address.

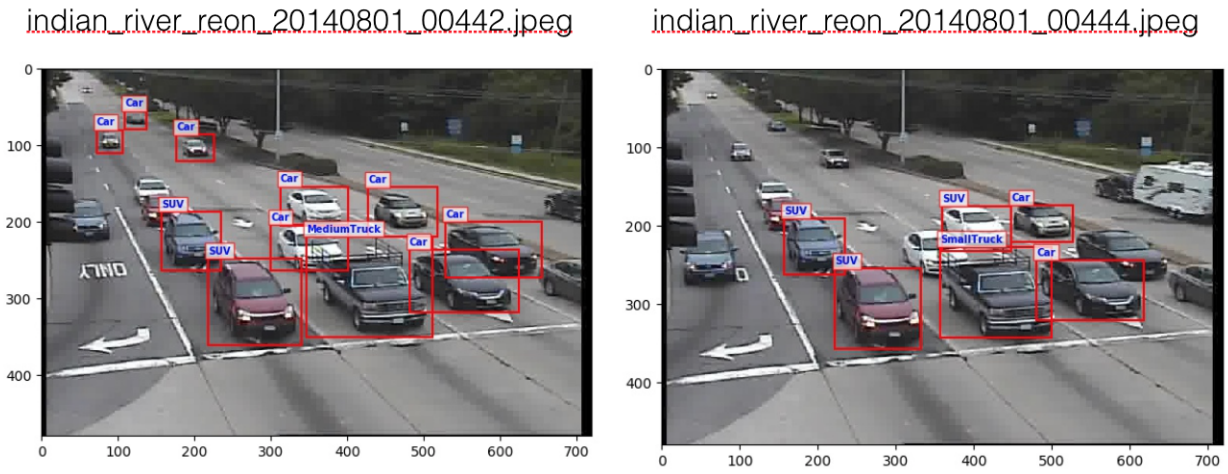

Figure 1.2: skip-annotations

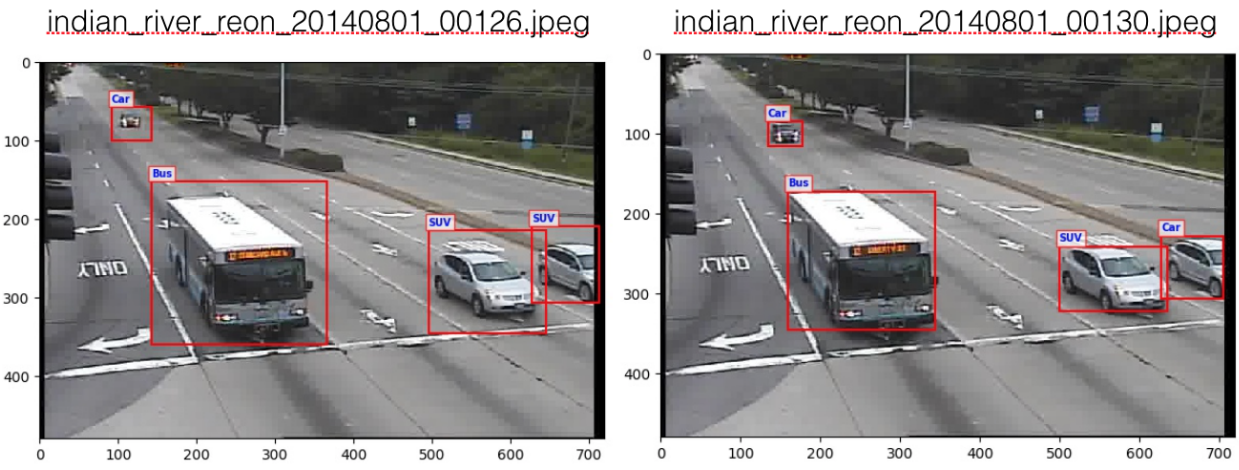

Figure 1.3: different-standard-for-annotation

Third, one third of categories in AI city dataset have less than 2 percent of instances in the training set. For example, there are only 11 instances of bicycle and 1 instance of pedestrian in aic480 training set. Categories with less than 2 % of training instances are annotated in red color in 1.4.

	AIC480 TRAIN	AIC480 TRAIN %	AIC480 VAL	AIC540 TRAIN	AIC540 TRAIN %	AIC540 VAL
CAR	45321	53.36	12961	156903	27.19	51112
SUV	23265	27.39	9584	77395	13.41	24560
VAN	5402	6.36	1073	11253	1.95	3830
BICYCLE	11	0.01	3	2237	0.38	389
BUS	662	0.77	64	982	0.17	270
MOTORCYCLE	139	0.16	8	2934	0.50	1687
PEDESTRIAN	1	0.00	22	77395	13.41	1062
GROUPOFPEOPLE	-	-	-	1062	0.18	229
SIGNAL-R	-	-	-	167099	28.96	58143
SIGNAL-G	93	0.10	-	53854	9.33	20027
SIGNAL-Y	-	-	-	4447	0.77	2021
S-TRUCK	6769	7.97	2177	16394	2.84	7289
M-TRUCK	2408	2.83	360	3701	0.64	1687
L-TRUCK	852	1.00	230	1291	0.22	908
TOTAL	84923		26482	576947		173214

Figure 1.4: category table of AI city

1.3 AI city dataset versus PASCAL VOC

Comparing with the PASCAL Visual Object Class (PASCAL VOC), a standardised image data sets for object class recognition since 2005, AI city dataset shares some different properties. First, each category in PASCAL VOC contains minimum 500 training objects, which guarantees more than 2 % of training objects with 23,374 objects total in training set. However, AI city is less balanced than PASCAL VOC with more than one third of categories having less than 2 % of objects.

Though the unbalanced properties will increase the difficulty of training object detection, the property reveals the real world problem for collecting objects from real world videos. It is hard to control the frequency of appearance between different categories. On the other side,

PASCAL VOC dataset scrapes images from websites and balances the amount of instances on purpose.

In 1.5, we can observe that the camera position and angle are dynamic in PASCAL VOC. But the camera is static in AI city and always far away to objects. The difference brings a good news that objects in AI city mostly is far away to camera enough that objects will not take more than 25% pixels of image. This property of AI city will be used for the design of multi-window and sliding window methods.

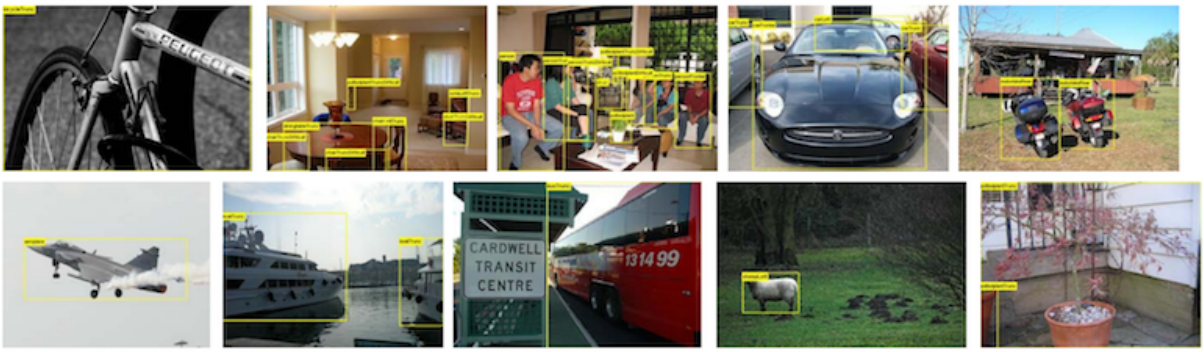

Figure 1.5: VOC dataset

1.4 Related work

single shot model is a type of light-weight convolution neural network model that takes object detection as a regression task. It features in speed that it can achieve higher than 30 frame per second but still with competitive accuracy. For frameworks in R-CNN family such as R-CNN and Fast R-CNN, they slow down the detection process by separating object detection as multiple stages problem including region proposals and classification for proposed image. 1.7 is a comparison over speed and accuracy for different object detection framework including You Only Look Once(YOLO), Single Shot MultiBox Detector(SSD), YOLO9000(YOLOv2) and R-CNN family.

YOLO is one of the first single shot model for object detection. It divides an input image into a S x S grids. Each grid predicts B bounding boxes for localization and a confidence vector presenting the probability over all categories. Each box consists of 5 values including center location, width, height and the confidence that there is an object inside.

The final regression output is visualized in the picture 1.6. For bounding boxes, there will be S x S x B x 4 values storing for coordinates (orange in 1.6) and S x S x B values for confidence (green in 1.6). Since each grid cell predicts a per-class scores that indicate the presence of a class instance in each of those boxes, the output holds S x S x the number of all categories floating point values.

In YOLO, the final probability of the bounding box detection is formulated by category confidence times the confidence of the bounding box.

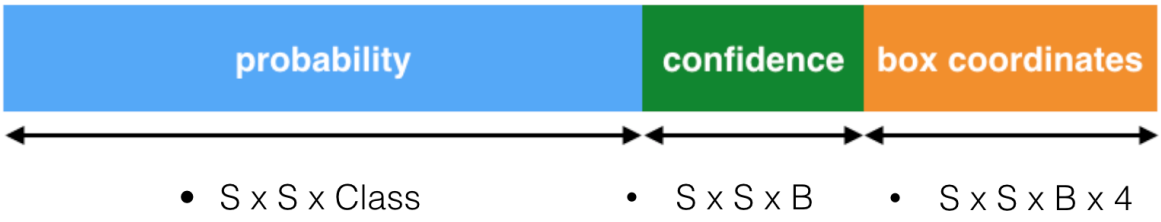

Figure 1.6: YOLO output structure

SSD is an improved single shot model framework that collects outputs from different convolution layers to be scale-invariant. Bounding box prediction in SSD is expressed as the offsets relative to the default anchors not the global axis, thus the localization values will lie in smaller region and easier to converge.

YOLO9000 is a second version of YOLO that converts YOLO to fully convolution model that consists of 9 convolution layers and 5 pooling layers. There works found out that the accuracy of single shot model can be improved greatly by having lager size of input images. Comparing input size of 288 pixels, YOLO v2 with 544 pixels as input size can improve mAP from 69.0 to 78.6 in VOC dataset.

This paper is organized as follows: Section 2 defines our proposed method. Experiments and results are presented in Section 3, and Section 4 concludes this paper.

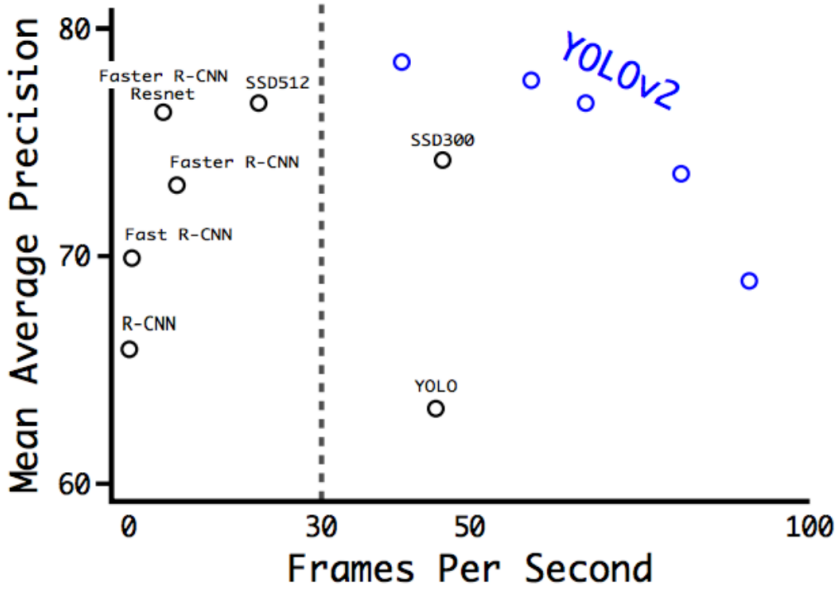

Figure 4: Accuracy and speed on VOC 2007.

Figure 1.7: Accuracy and speed

Chapter 2

METHOD

In this section, we propose an object detection pipeline for traffic scenes. With the goal to build real world and real time application, we choose single shot model as backbone of our object detection pipeline.

2.1 Object Detection Pipeline

With single image as input, the detection pipeline has single shot multibox detector (SSD) as a basic detection algorithm to locate and classify objects in the input image. To improve the pipeline, a parallel detection branch, trained in Common Object in Context(COCO) dataset with YOLO9000 architecture, processes the input image and predicts objects for categories which has few training instances. The final output result is merged from both branches with a merge policy.

2.2 Why we need a model trained in COCO

For categories having few training instances such as bicycle and bus, a single trained SSD model fails to have good performance for location and classification. If a model trained on other dataset that shares identical categories as AI city, it might provide information that the trained SSD lacks of. We import a YOLO9000 model trained on general purpose COCO dataset to compensate the results from SSD.

Figure 2.2 visualized the intersection between AI city dataset and COCO with 80 classes. Covered by COCO dataset, the blue region in Figure 2.2 are the categories that AI city dataset doesn't have sufficient amount of instances. For those categories, YOLO9000 trained in COCO should have better prediction for both localization and classification because of its

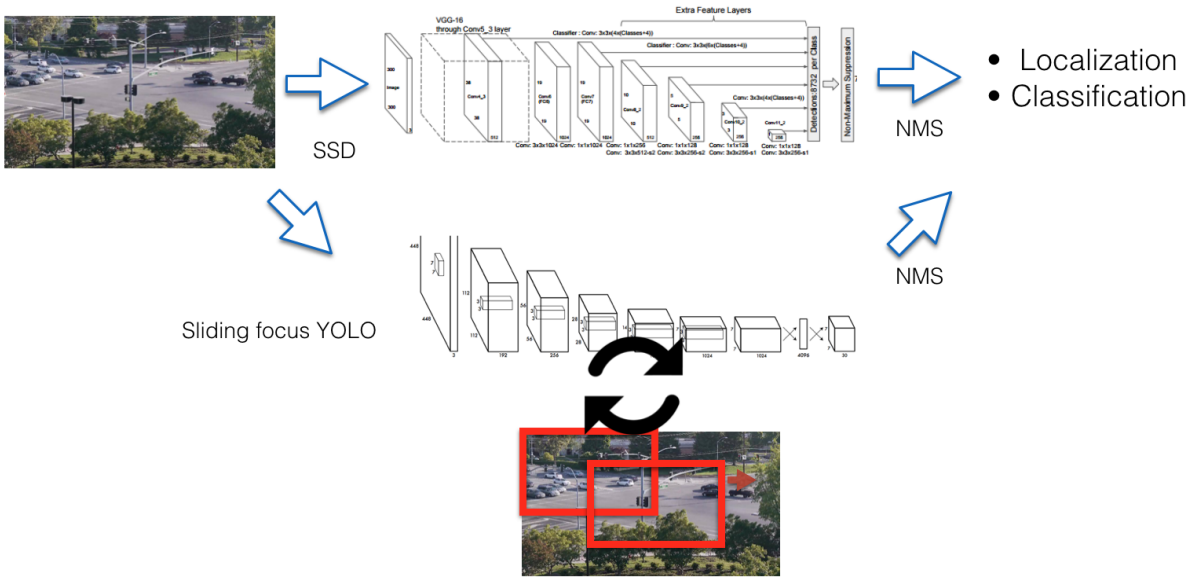

Figure 2.1: Object Detection Pipeline

abundance of training instances. Green region, which is car, is also a category that covered by both sources. However, since car is a category that AI city already has great amount of training instances, the performance of SSD should have good prediction sorely.

The traffic light categories, which are in red region in 2.2, suffer from lazy annotator effect and tend to predict larger size of bounding box than the actual size of traffic light. Though model trained in COCO can only predict traffic light without color information, it can help the pipeline to reshape the traffic light.

2.3 Multi-window method

Single shot model has advantage for fast prediction, but not sensitive to small objects and crowded scenario. If the input image is smaller, each grid in single shot model handles less pixels and the bounding can have more precise prediction. Plus, the amount of bounding box that each grid can predict is fixed, single shot model's ability to process crowded scenario is limited. Thus, this work presents a multi-window method for YOLO9000. Mentioned

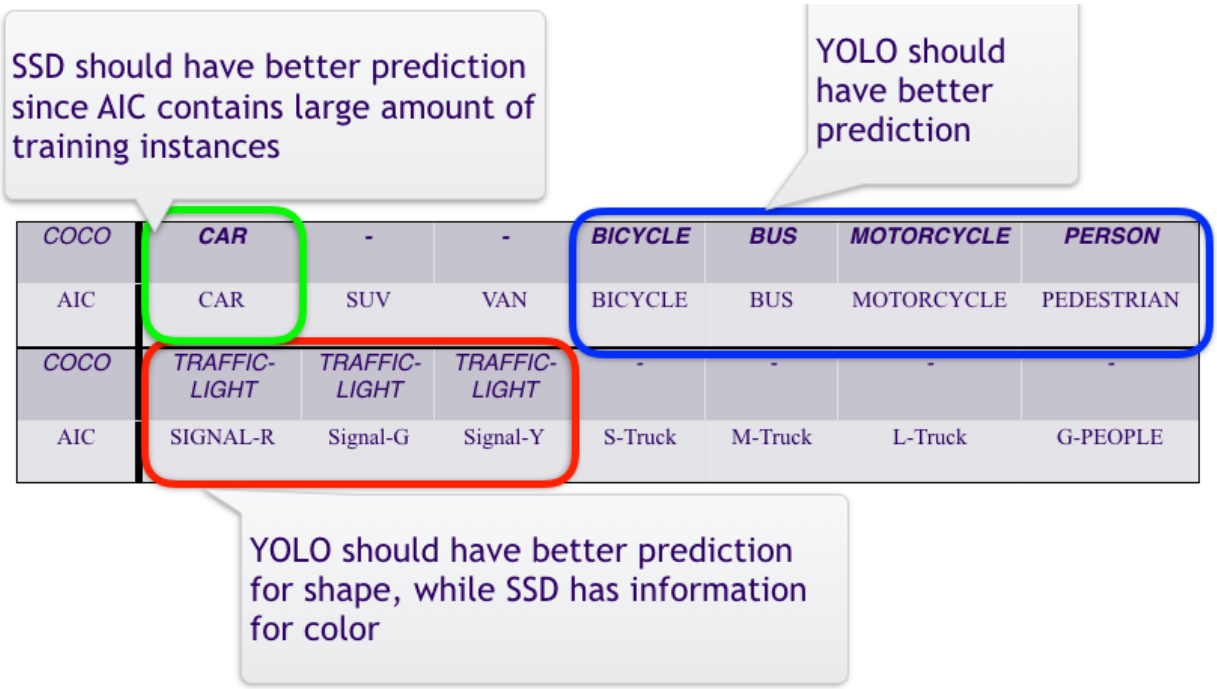

Figure 2.2: Reference from a model trained in COCO dataset

in section 1.3, the nature of traffic scene video is that the camera is static and far away from the objects and mostly objects will not take more than 25 % of pixels of the whole image. Multi-window uses a window half sized of both width and height of origin image and the window will be taken as input for YOLO9000. The window will be slided through the whole image with half sized overlapped and there will be 3 times for each row and 3 times for each column. Notice that the multi-window method doesn't need to be processed in sequence but in parallel. This method can be extended with smaller sized of windows and more overlapped region for sliding if the user have better hardware. After collecting all the predicted results from multi-window YOLO9000, an object is likely detected more than once by different windows. A non-maximum suppression clean up is needed to search bounding boxes that overlap with each other more than a threshold (empirically set as 30%) and filter out all the boxes except for the one with highest confidence score.

Figure 2.3: Multi-window method

2.4 Merge policy

When merging detection results from both SSD and multi-window YOLO9000, different category is applied with different rule. The merge policy is only activated when bounding boxes proposed by both sources intersect with each other more than an IOU threshold.

As discussed in section 2.2, multi-window YOLO9000 is expected better prediction for categories that has few training instances in AI city dataset such as bicycle, bus, motorcycle and person. For those categories, if the confidence of proposed detection from YOLO is higher than a threshold, we replace SSD proposed detection by YOLO proposed detection. If the proposed bounding box has high confidence value but not intersect with any bounding box from SSD, we append the bounding box to the final result. The pseudo-code is listed in Figure 2.4.

To optimize the merge policy for traffic light, if proposed bounding boxes from both

sources intersect with each other more than an IOU threshold and having confidence higher than the threshold, the merge policy will replace YOLO's box to SSD's box when box from YOLO has aspect ratio closer to 3. Another case will trigger the replacement when box from YOLO predicts a smaller region. Without the merge policy for traffic light, the results from SSD suffer from lazy annotator effect and tend to predict wrong aspect ratio and larger size of bounding boxes than its actual size. The qualitative section will showcase the difference between results with or without merge policy for traffic scenes. The pseudo-code is listed in Figure 2.5.

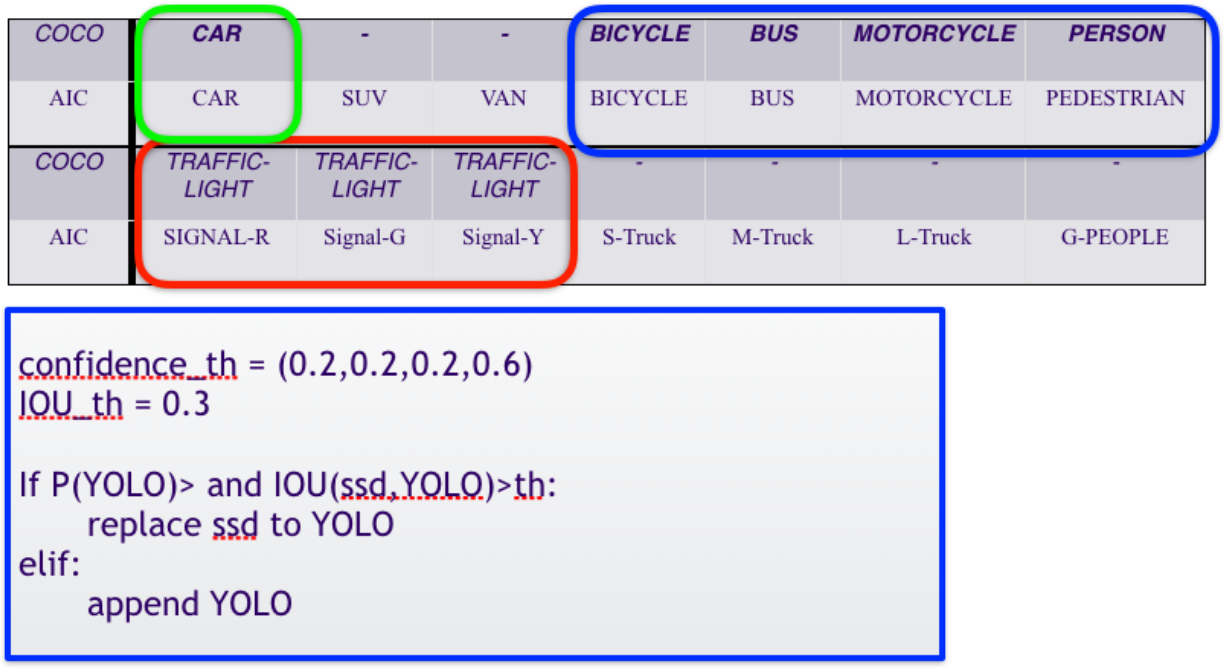

Figure 2.4: merge policy for categories have rare training instances

COCO	***CAR***	*-*	*-*	***BICYCLE***	***BUS***	***MOTORCYCLE***	***PERSON***
AIC	CAR	SUV	VAN	BICYCLE	BUS	MOTORCYCLE	PEDESTRIAN
COCO	*TRAFFIC-LIGHT*	*TRAFFIC-LIGHT*	*TRAFFIC-LIGHT*	*-*	*-*	*-*	*-*
AIC	SIGNAL-R	Signal-G	Signal-Y	S-Truck	M-Truck	L-Truck	G-PEOPLE

rule1 = function(return True if YOLOr box is more like 3:1 ratio else False)
rule2 = function(return True if YOLO's box has smaller region else False)

If P(YOLO)> (0.3,0.4,...) and IOU(ssd,YOLO)>th and (rule1 or rule2):
replace ssd to YOLO

Figure 2.5: merge policy for traffic light

Chapter 3

EXPERIMENT AND RESULT

3.1 Experiment Setup and Implementation

For training SSD in AI city dataset, we adopt the VGG-based SSD network to train on the joint datasets of aic480 and aic540. The network is based on a model trained in ImageNet. We set the number of iterations as 200,000 with a batch size of 16. We choose 512 pixels input size instead of 300 to enhance detection of small objects like traffic lights. The YOLO9000 model is trained in COCO dataset with 612 pixels input size.

3.2 Quantitative results on AI city dataset

We compare the effectiveness of merging policy and multi-window design on the aic1080 validation dataset. As an ablation study, we present the result for solely SSD for detection, the result for SSD merged with YOLO9000 branch and the result for SSD merged with YOLO9000 and multi-window design. Figure 3.1 shows the average precision for categories having few training instances and mean average precision. For bus category, merging with YOLO9000 gains 42% of improvement and multi-window design gains extra 10 % of improvement. The mAP gains 2% of improvement from merging and extra 1% improvement from multi-window.

Figure 3.1 shows the result of testing dataset. Testing dataset contains aic480, aic540 and aic1080 dataset. The result is calculated by Nvidia testing server and the objects smaller than size threshold 30 by 30 pixels are filtered out for computing average precision and mean average precision. The prediction result of aic540 and aic 1080 are the same but scale, boxes in aic540 could be filtered out because they are smaller than size threshold but will be kept in aic1080 because their sizes are doubled. This can explain the average precision of traffic

TABLE II
COMPARISON OF DETECTION OF RARE-INSTANCE CLASSES ON *AIC1080*

Methods	Bus	Bicycle	Motorcycle	mAP
SSD	0.214	0.034	0.149	0.267
SSD+YOLO9000	0.304	0.035	0.164	0.274
SSD+YOLO9000 w/ MST	0.336	0.040	0.168	0.277

Figure 3.1: Method comparison

light signals are all zeros in aic540 but not in aic1080.

TABLE I
EXPERIMENTAL RESULTS OF TRACK 1

Categories	*aic480*	*aic540*	*aic1080*
Car	0.75	0.61	0.59
SUV	0.52	0.48	0.45
Van	0.22	0.24	0.22
Bicycle	0.38	0.05	0.03
Bus	0.35	0.46	0.45
Motorcycle	0.14	0.29	0.22
Pedestrian	0.00	0.03	0.10
GroupOfPeople	--	0.12	0.09
Signal-R	--	0.00	0.38
Signal-G	--	0.00	0.30
Signal-Y	--	0.00	0.06
S-Truck	0.45	0.48	0.45
M-Truck	0.19	0.27	0.27
L-Truck	0.02	0.14	0.14
mAP	0.34	0.25	0.28

Figure 3.2: Result table

3.3 Qualitative results on AI city dataset

3.3.1 Visualization of ablation study

In this section, we will present the comparison over predictions and ground truth annotation. The comparison will have prediction by sorely SSD, SSD merged with YOLO, SSD merged with multi-window YOLO and the ground truth.

The left top image is the result of sorely SSD, right top is the result of SSD merged with YOLO9000 and the left bottom shows the result of SSD merged with multi-window YOLO9000. The ground truth is listed in the right bottom.

There will be several comparisons for category bus, followed by bicycle, pedestrian and motorcycle. Since traffic lights almost exist in every traffic images, we can observe the effectiveness of multi-window design for every set of comparison.

Figure 3.3- 3.7 are for category bus. Figure 3.8- 3.10 are for category bicycle. Figure 3.11- 3.13 are for category pedestrian. Figure 3.14- 3.16 are for category motorcycle.

3.3.2 Case study

This section shows some comparisons between predictions and ground truth annotations based on aic540 validation set image. In Figure 3.17, the model predicts truck, SUV and bicycle for the top image, but there is no any annotation in the ground truth image. While the model predicts two pedestrians for the bottom image, the ground truth labels one pedestrian only. For correct prediction, human errors of annotations in validation or testing set could incorrectly lead to poor performance in benchmark. In this way, testing or validation mean average precision should not be the only benchmark for evaluation. Figure 3.18 and Figure 3.19 shows correct prediction of model but not annotated in ground truth. Even for human with attention, it is hard to notice the car in the left top region occluded by bushes. Unfortunately, cases that model has good predictions but it is not annotated by annotator (mostly is by human) will penalize the precision result in benchmark. Sampling and visualizing the difference between prediction and ground truth is important for object

Figure 3.3: Comparison for bus

detection researches.

3.3.3 Fail case

This section is a collection for fail case of object detections in traffic scenes. In Figure 3.20, first and fifth images are false positive cases predicting a car and a pedestrian while they are signs in traffic scenes. Third image is also a false positive case predicting a car while there is no object. Second and fourth images are predicting bad shape for bicycle detection. The definition of bicycle in COCO dataset points to bicycle itself while the image collected in AI city dataset means a bigger box containing bicycle rider and bicycle. When merging the result from models trained by different dataset, there might be a gap between the definition of bounding boxes even for the same name. In the last case, the model just detects a pedestrian

Figure 3.4: Comparison for bus

while there are two.

Figure 3.5: Comparison for bus

Figure 3.6: Comparison for bus

Figure 3.7: Comparison for bus

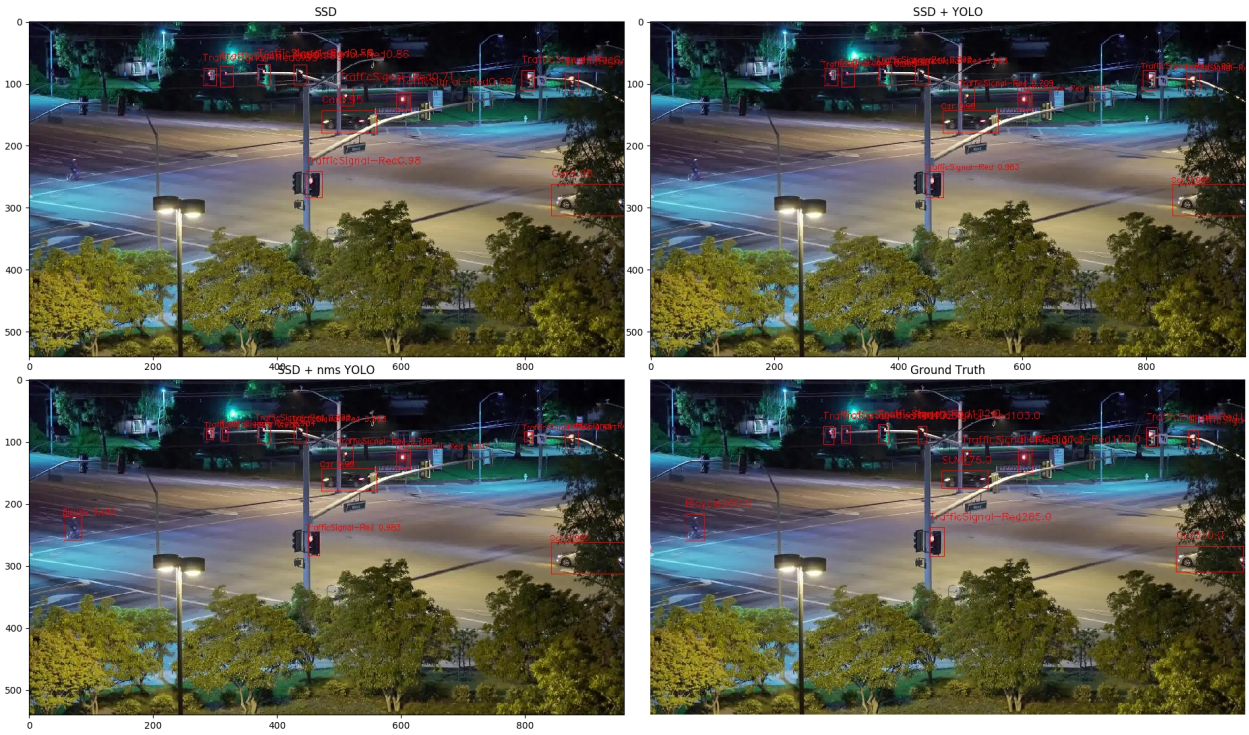

Figure 3.8: Comparison for bike

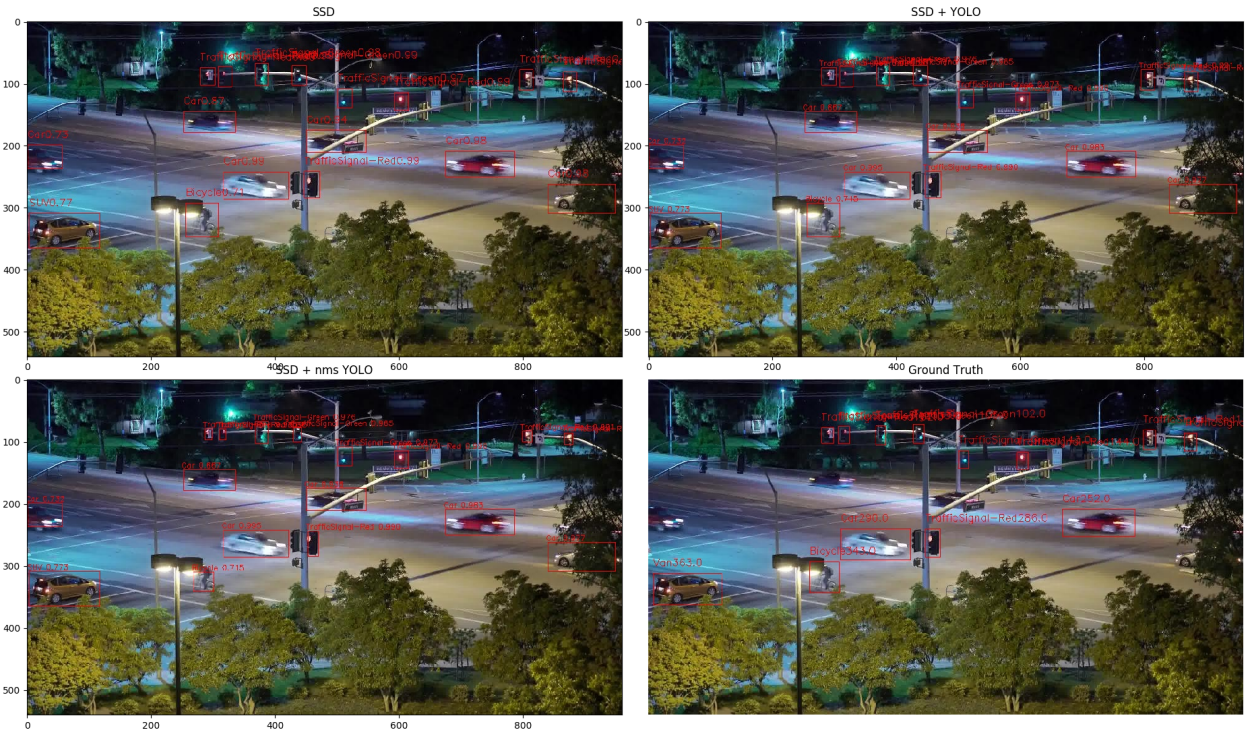

Figure 3.9: Comparison for bike

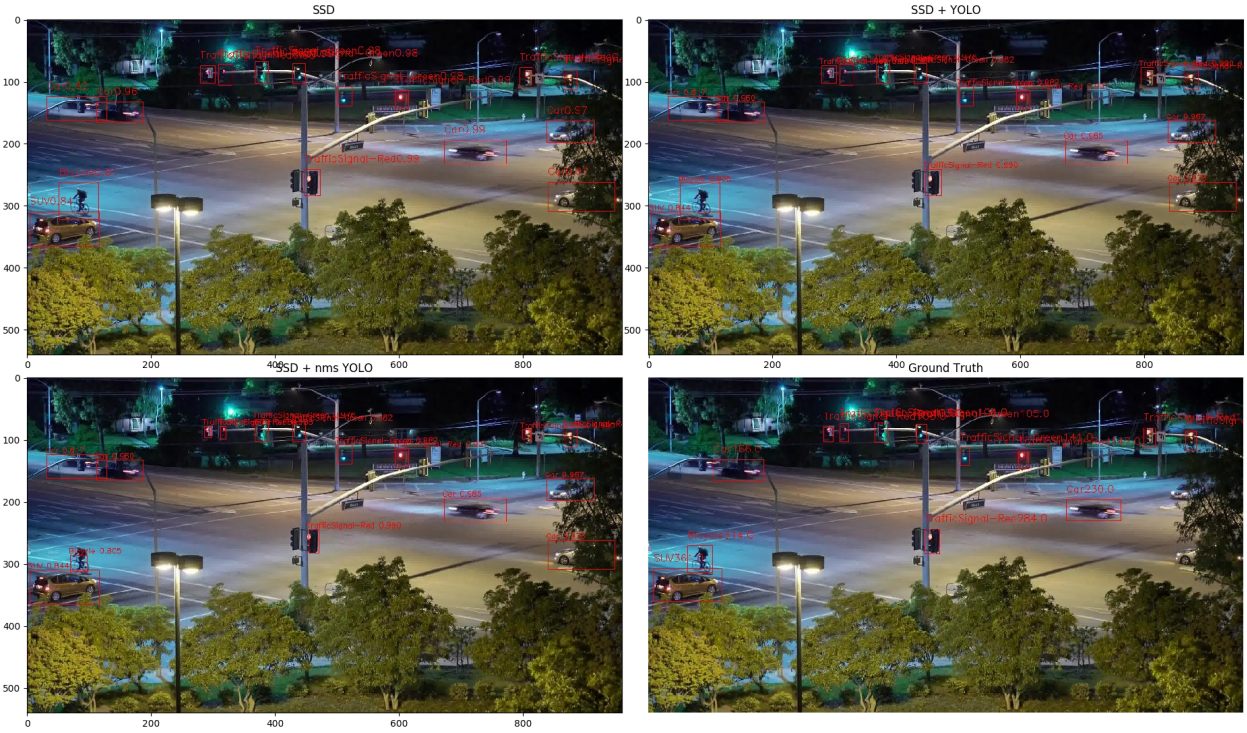

Figure 3.10: Comparison for bike

Figure 3.11: Comparison for pedestrian

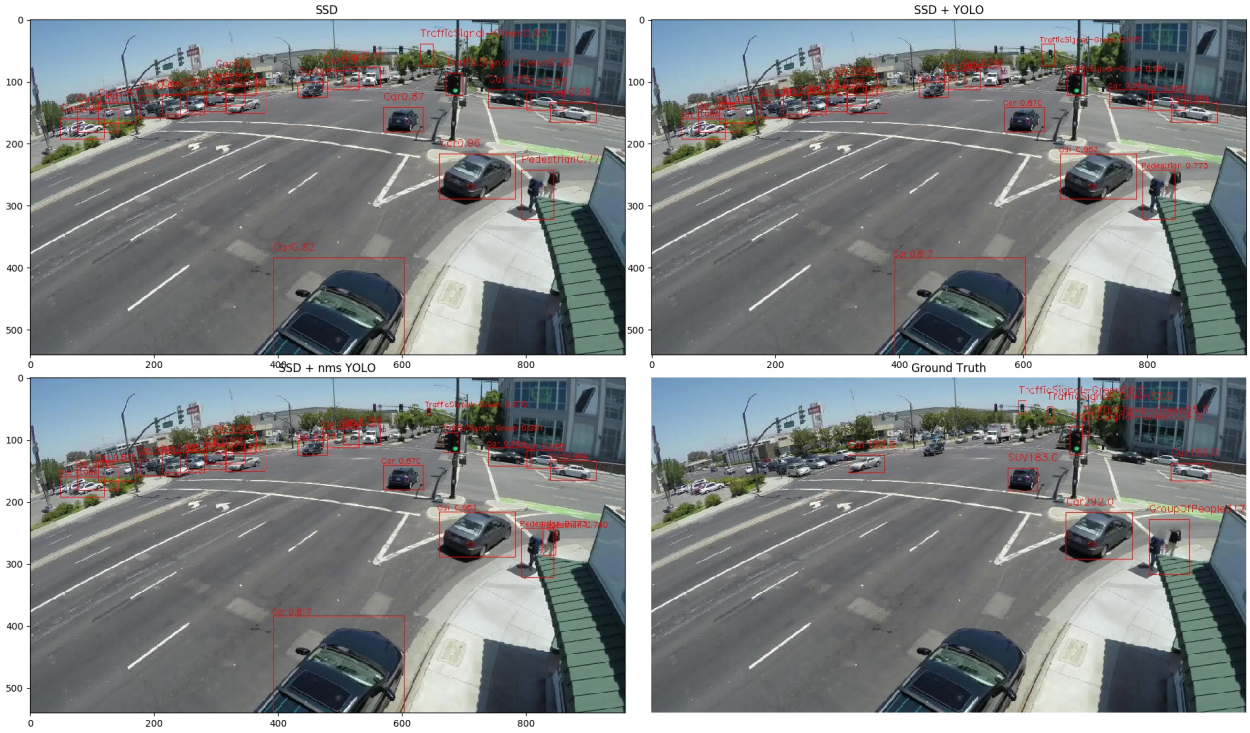

Figure 3.12: Comparison for pedestrian

Figure 3.13: Comparison for pedestrian

Figure 3.14: Comparison for motorbike

Figure 3.15: Comparison for motorbike

Figure 3.16: Comparison for motorbike

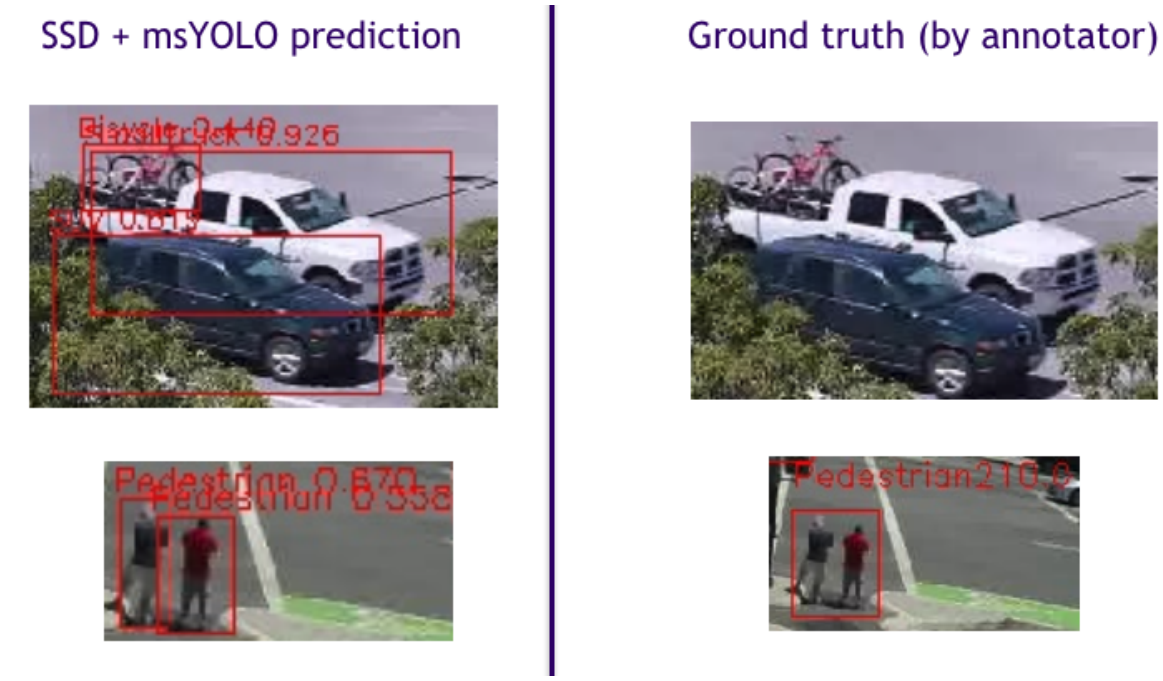

Figure 3.17: Case study 1

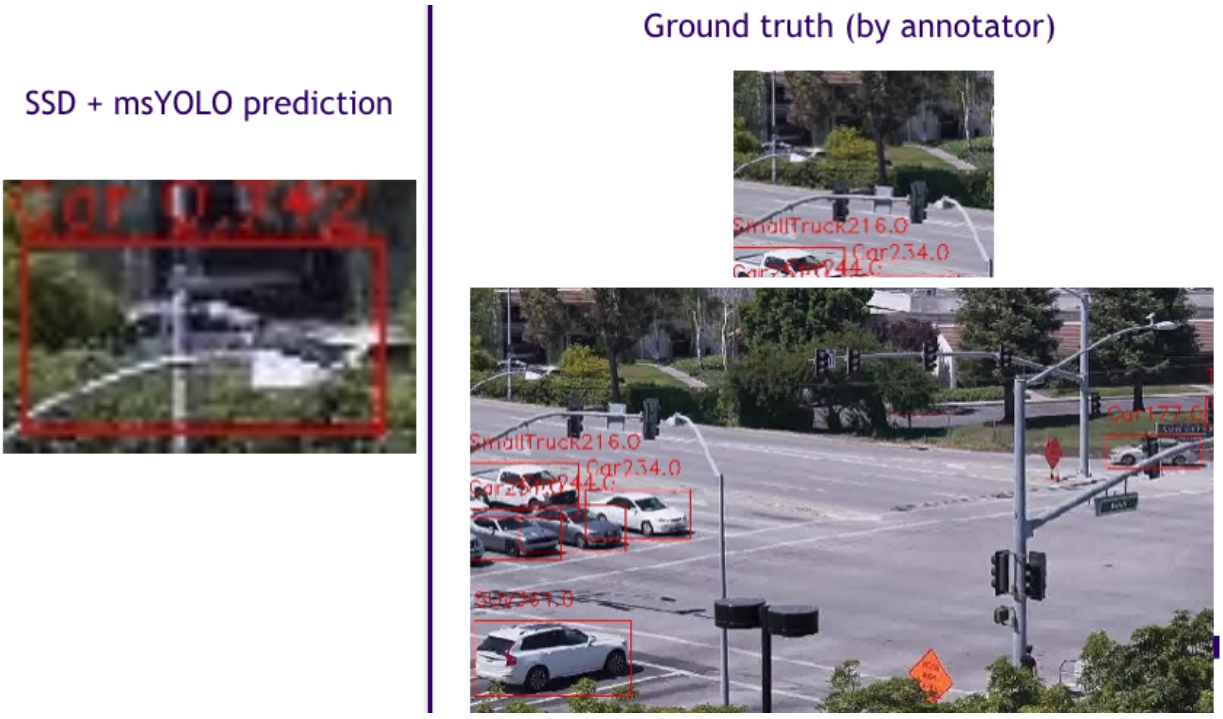

Figure 3.18: Case study 2

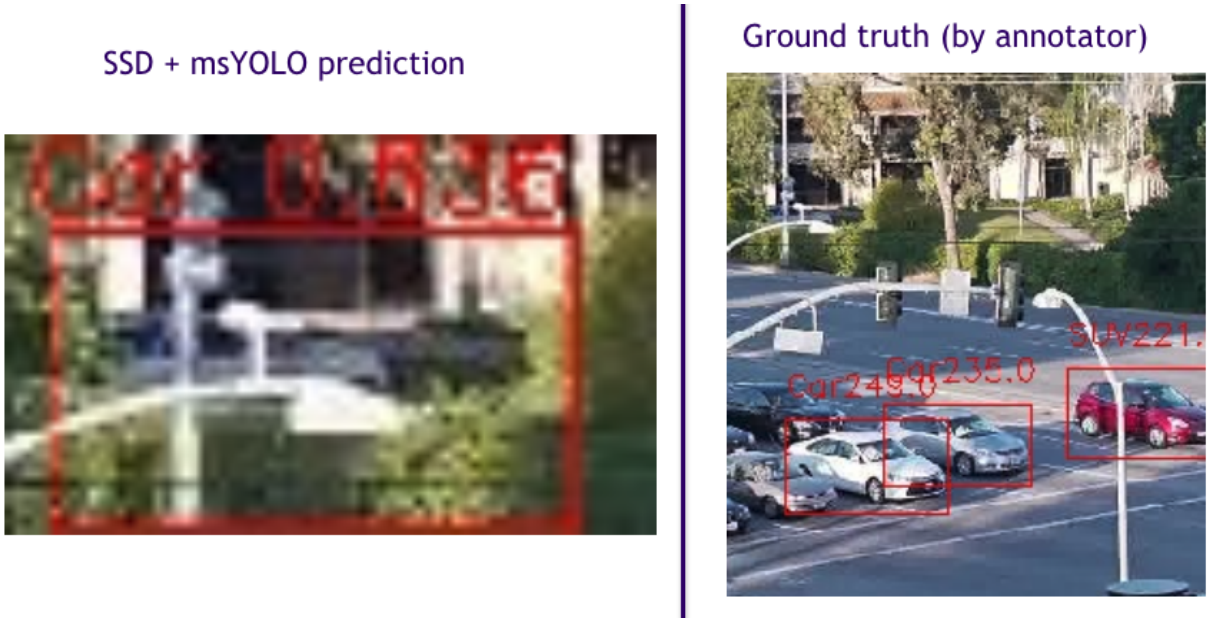

Figure 3.19: Case study 3

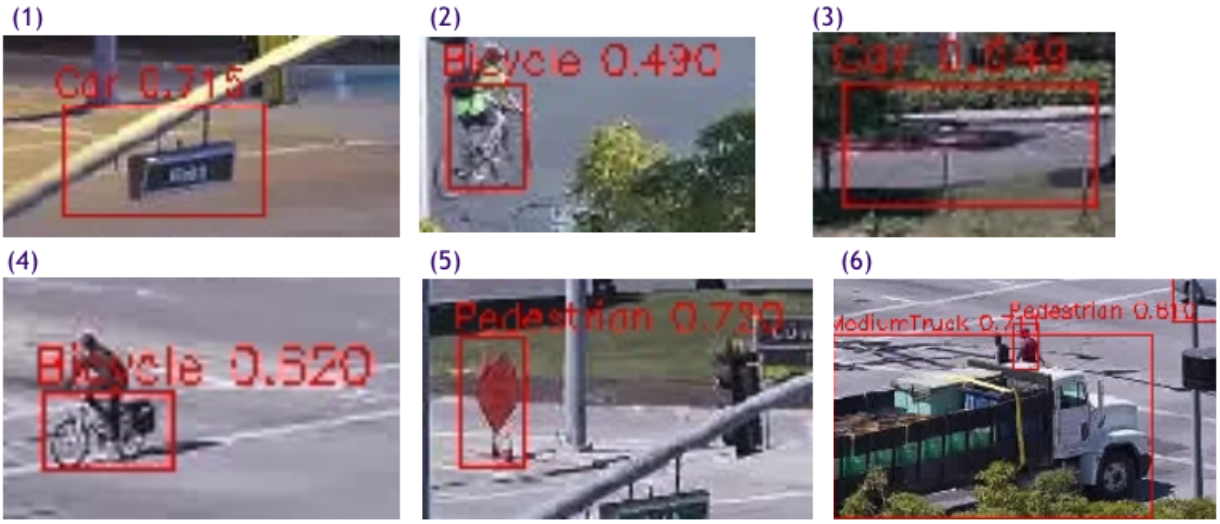

Figure 3.20: Fail case

Chapter 4

CONCLUSION

The work presented multi-window method that can effectively improve detection accuracy of single shot model without any fine-tune. Merging result with a model trained on general dataset can solve the challenge that some categories have few training instances. Adopting model trained on general dataset also helps to address lazy annotator effect and predicts better shape of detection result. Though the ground truth is influenced by lazy annotator effect, mAP is still improved by 3% with the proposed pipeline. The qualitative results clearly demonstrate the improvement and contribution.

BIBLIOGRAPHY

[1] Jonathan Huang. *Speed/accuracy trade-offs for modern convolutional object detectors.* Addison-Wesley, 2017.

[2] Kaiming He Jian Sun Jifeng Dai, Yi Li. R-FCN: Object detection via region-based fully convolutional networks. *arXiv preprint arXiv:1605.06409*, 2016.

[3] Ali Farhadi Joseph Redmon. Yolo9000: Better, faster, stronger. *arXiv:1612.08242*, 2016.

[4] Ross Girshick Ali Farhadi Joseph Redmon, Santosh Divvala. You only look once: Unified, real-time object detection. *arXiv:1506.02640*, 2015.

[5] Christopher K.I. Williams John Winn Andrew Zisserman Mark Everingham, Luc Van Gool. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision manuscript*, 2012.

[6] Serge Belongie Lubomir Bourdev Ross Girshick James Hays Pietro Perona Deva Ramanan C. Lawrence Zitnick Piotr Dollr Tsung-Yi Lin, Michael Maire. Microsoft coco: Common objects in context. *arXiv:1405.0312*, 2014.

[7] Dumitru Erhan Christian Szegedy Scott Reed Cheng-Yang Fu Alexander C. Berg Wei Liu, Dragomir Anguelov. Ssd: Single shot multibox detector. *arXiv:1512.02325*, 2015.