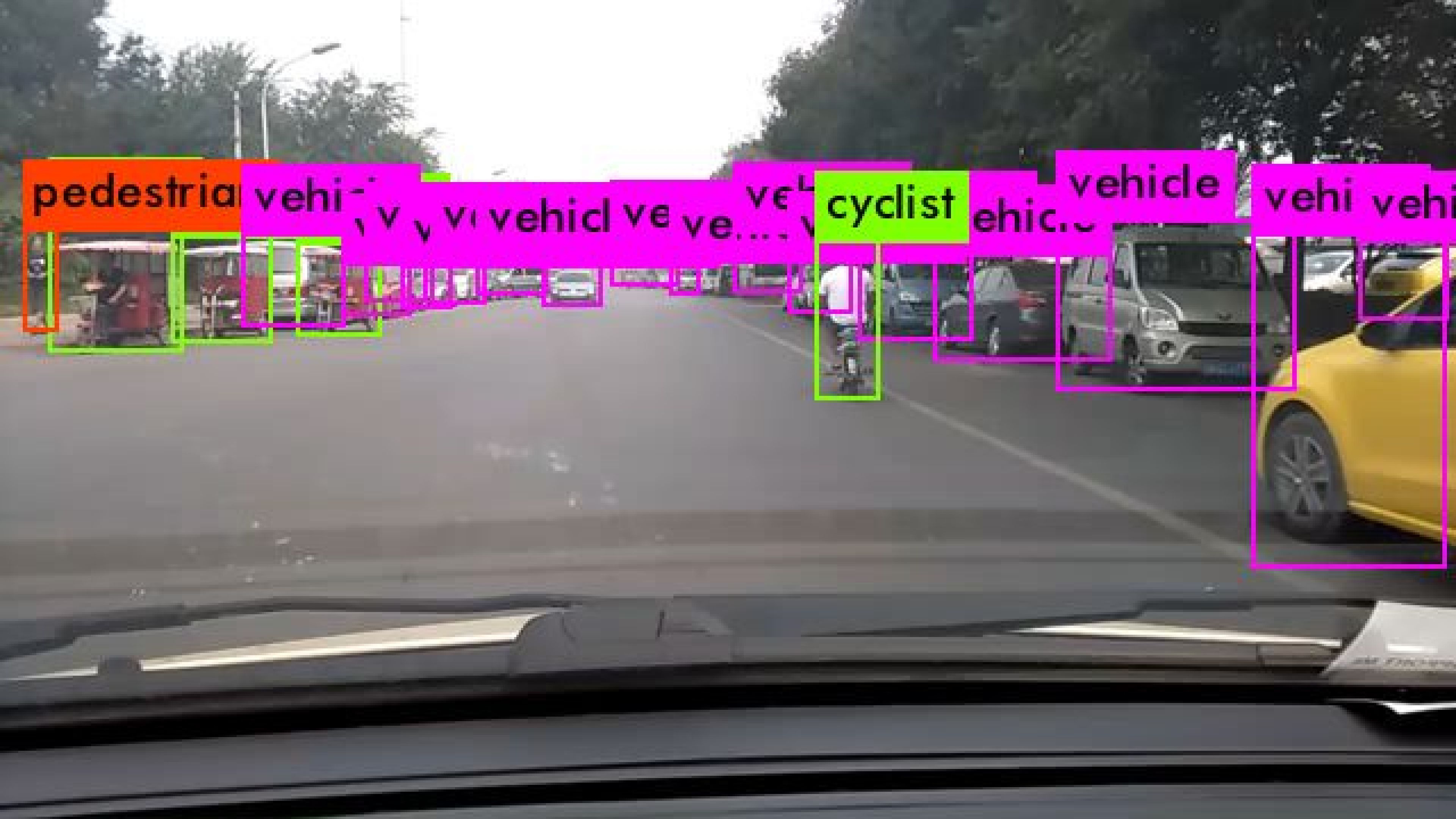
Object Detection on Self-Driving Cars in China

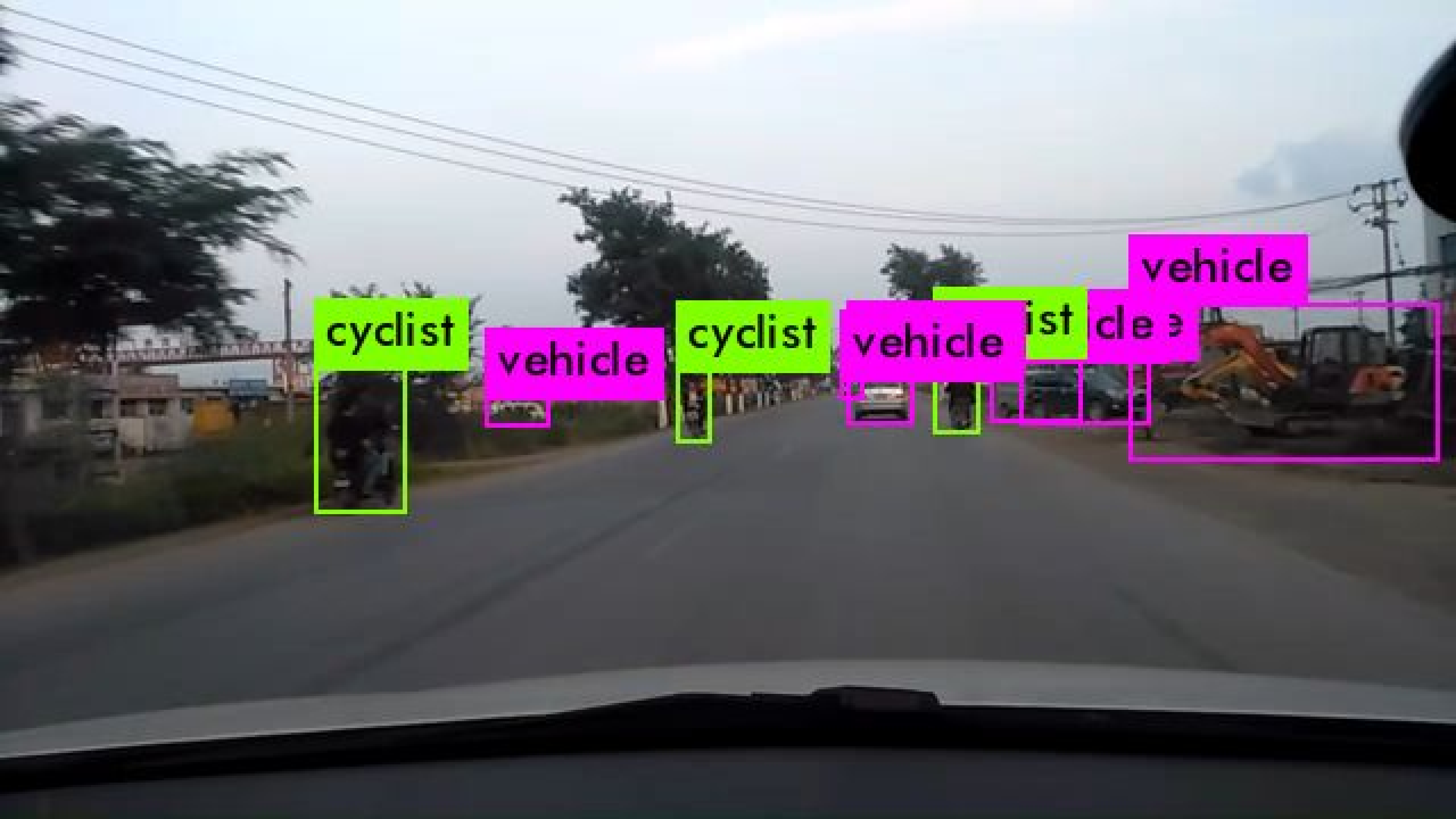
Lingyun Li

Introduction

- Motivation: Perception is the key of self-driving cars
- Data set:
  - 10000 images with annotation
  - 2000 images without annotation (not used)
  - 640 * 360 pixels
- Complex road conditions in China
- Annotation: Object category and Bounding box
- 4 Categories: Vehicle, Pedestrian, Cyclist, Traffic_lights
- Task: Predict bounding box, category, and confidence
- Randomly select 2000 images from 10000, as test/validation set.


vehicle

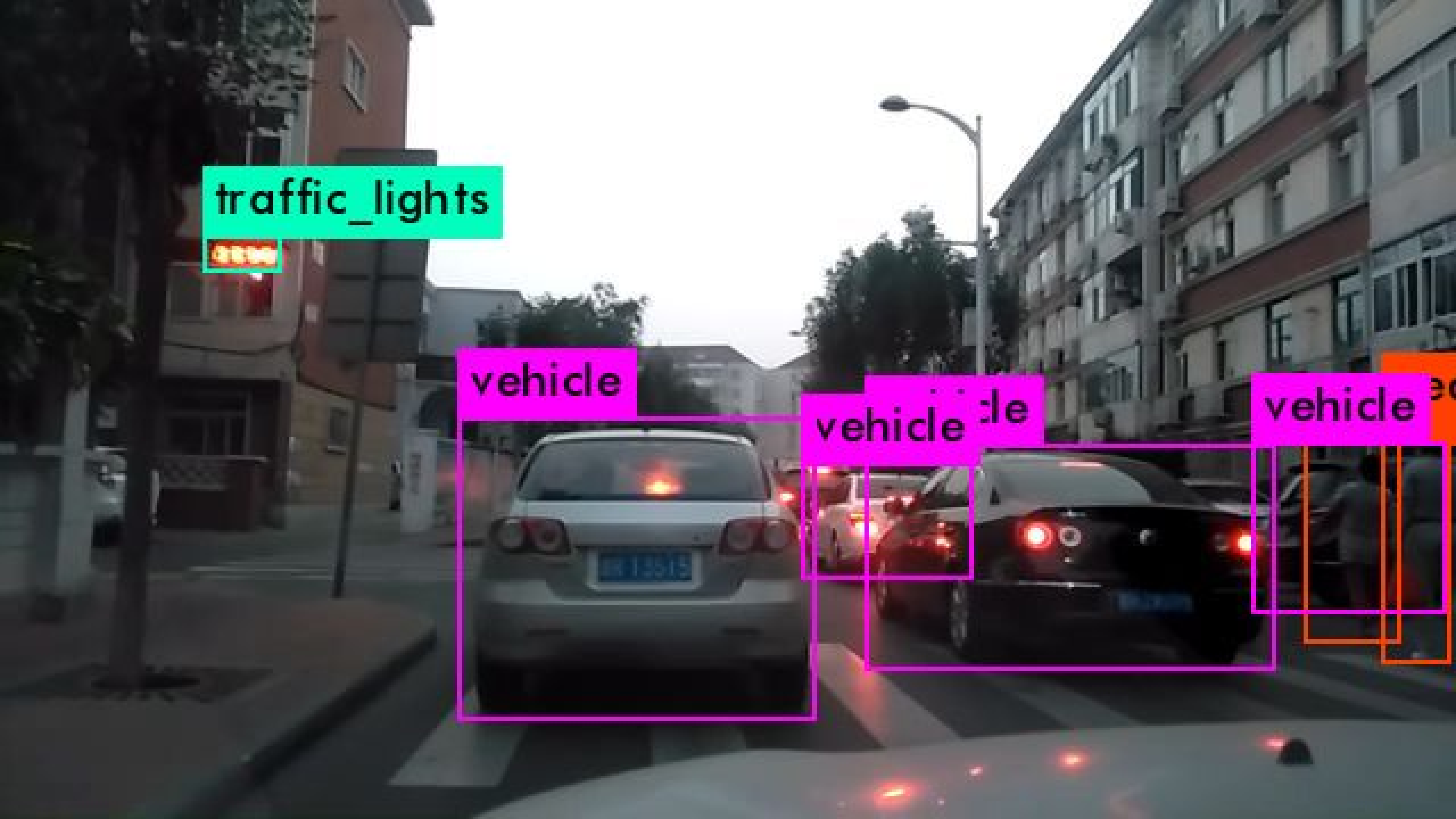

pedestria
vehi
vehicl
cyclist
vehicle
vehi


cyclist
vehicle
cyclist
vehicle
vehicle


vehicle
pedestrian


vehicle
vehicle
hicle


vehicle
vehicle


traffic_lights
vehicle
vehicle
vehicle


pedestrian

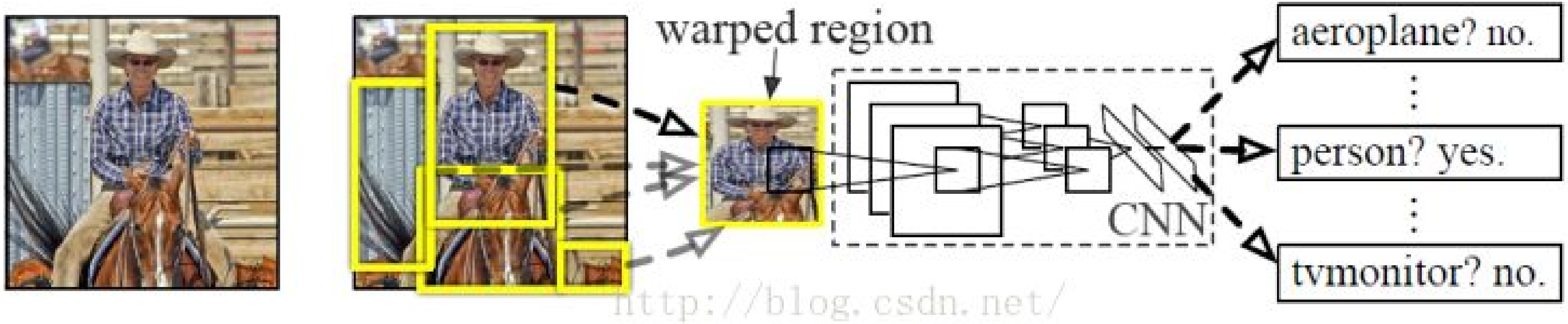
Data

- Small objects
- Objects overlapped and cropped
- Poor image quality
- Poor annotations

Why is Object Detection Difficult?

- Image classification:
  - Shift of an object inside an image is indiscriminate
  - Favours translation-invariance (CNN)
- Object detection
  - Describing how good the candidate box overlaps the object
  - Need both translation-invariance and translation-variance
  - Deep CNNs are less sensitive to translation

R-CNN: Region Proposal + CNN (2014)



	localization	feature extraction	classification
this paper:	selective search	deep learning CNN	binary linear SVM
alternatives:	objectness, constrained parametric min-cuts, sliding window …	HOG, SIFT, LBP, BoW, DPM …	SVM, Neural networks, Logistic regression …

SPPnet: Spatial Pyramid Pooling (2014)

- Fully-connected layers take fixed sized input
- CNN can take input of any size
- Pooling to fixed size after CNN
- Improvement:
  - Can take image of any size
  - Only run CNN once for input image

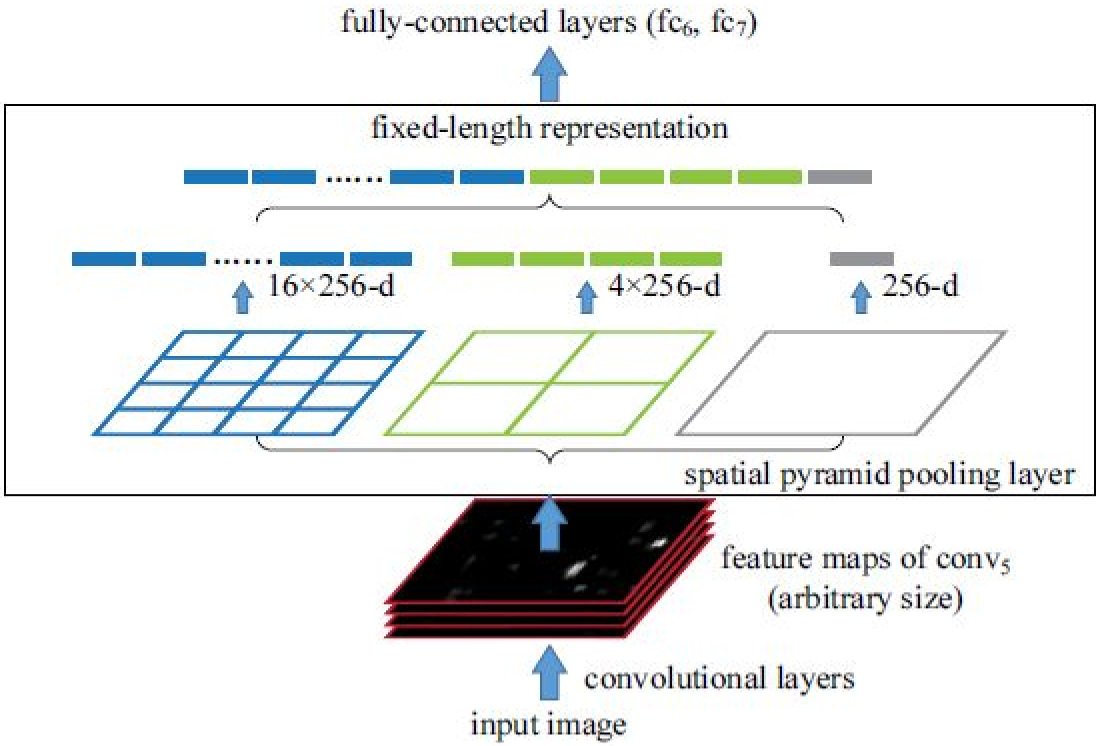
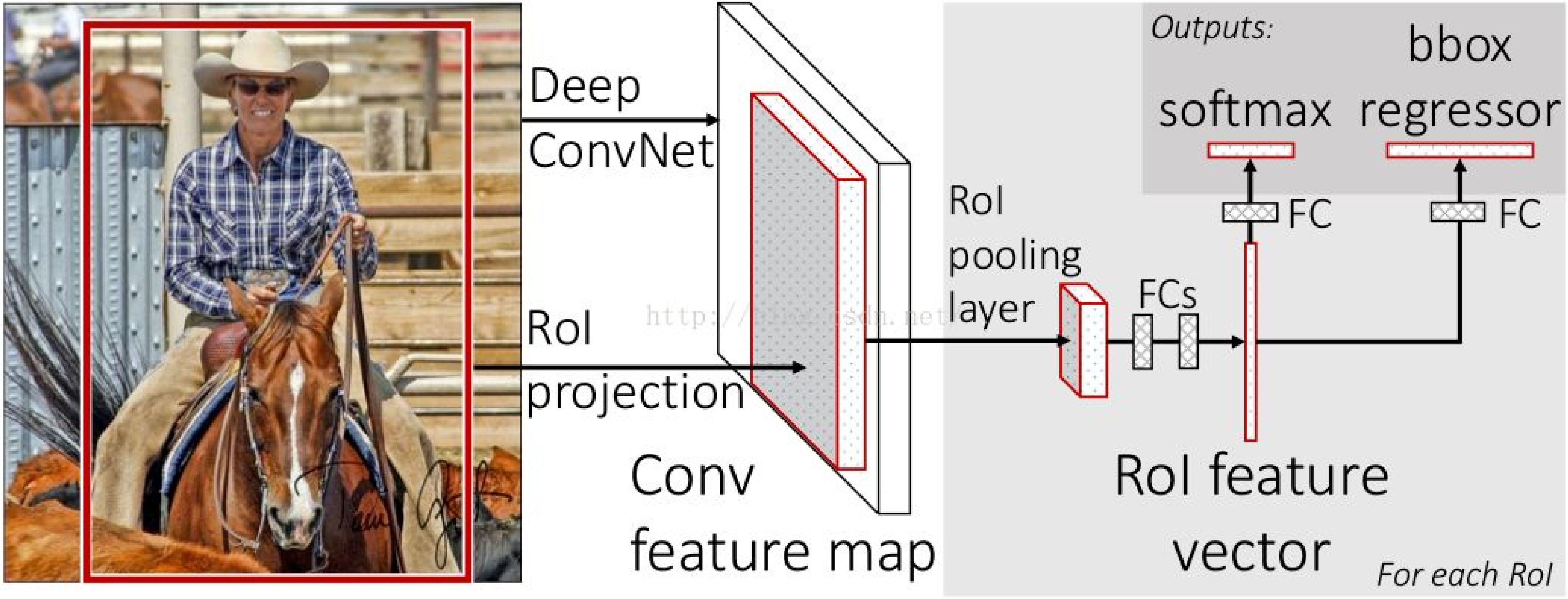


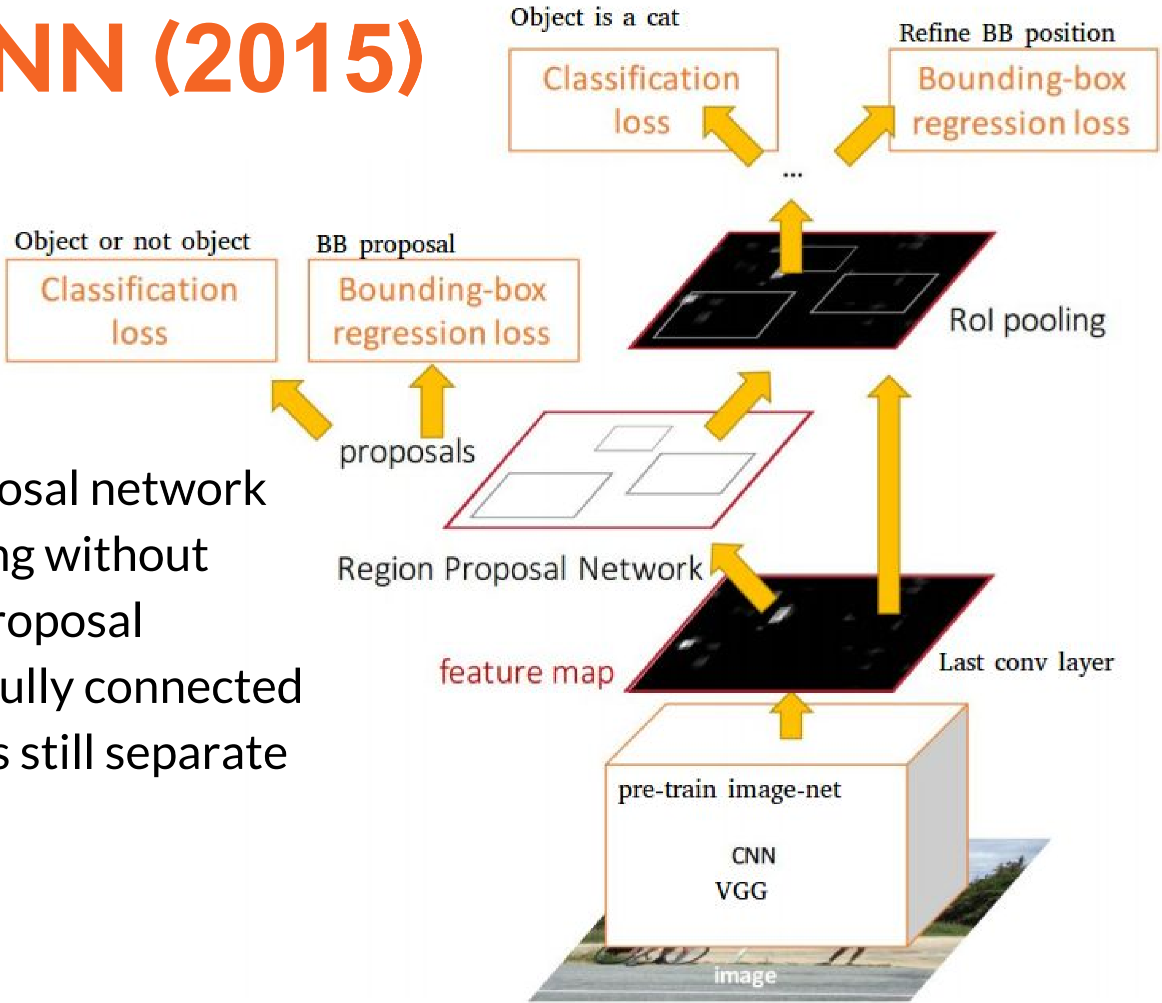
Figure 3: A network structure with a **spatial pyramid pooling layer**. Here 256 is the filter number of the $conv_5$ layer, and $conv_5$ is the last convolutional layer.

Fast R-CNN (2015)

- RoI (Region of Interest) pooling layer: a special type of SPP after CNN
  - Run for each region proposal to get fixed size output
- Multi-task loss: Train category classifier and bounding box regression together



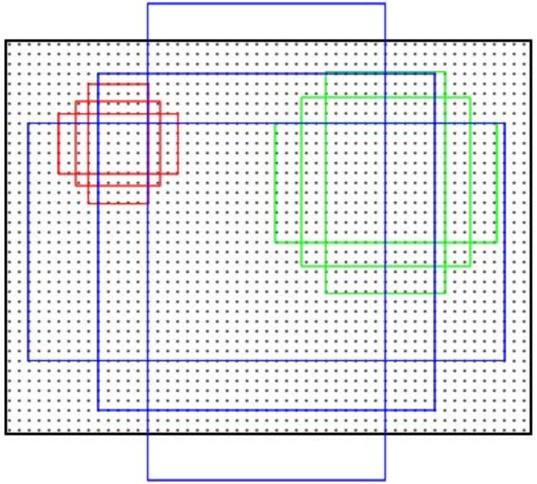
Faster R-CNN (2015)

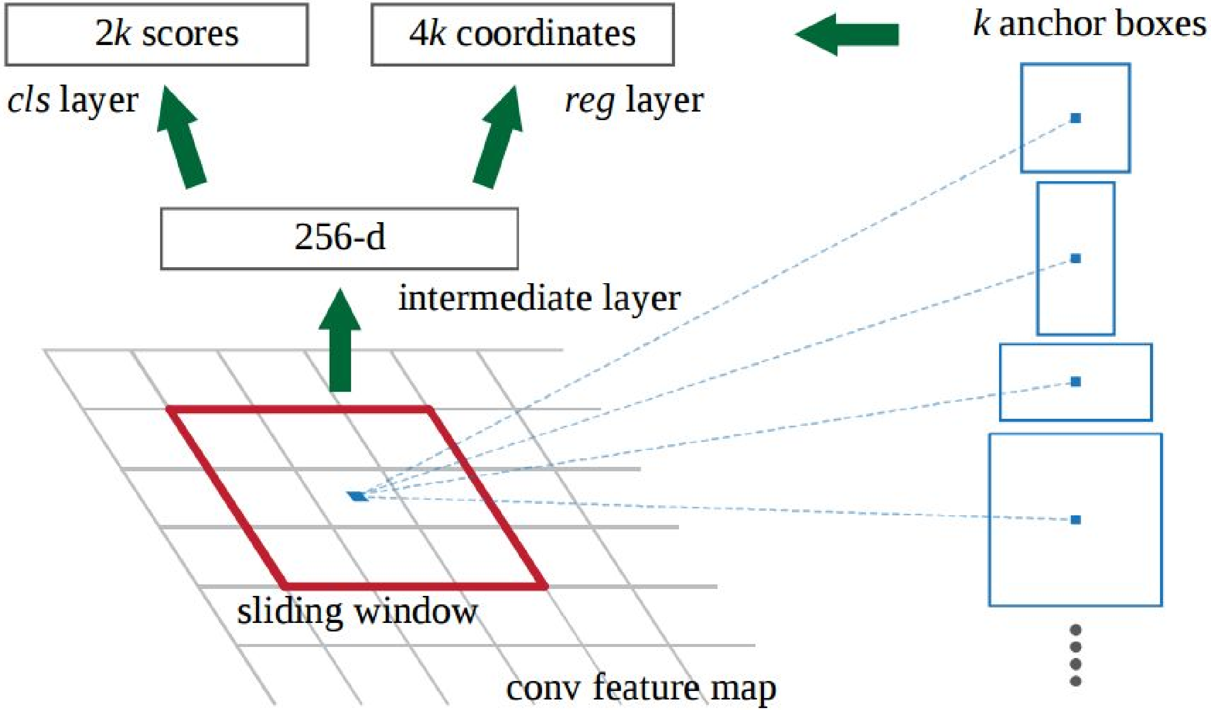


- RPN: Region proposal network
- End-to-end training without separate region proposal
- Computation for fully connected layers after RPN is still separate for each RoI

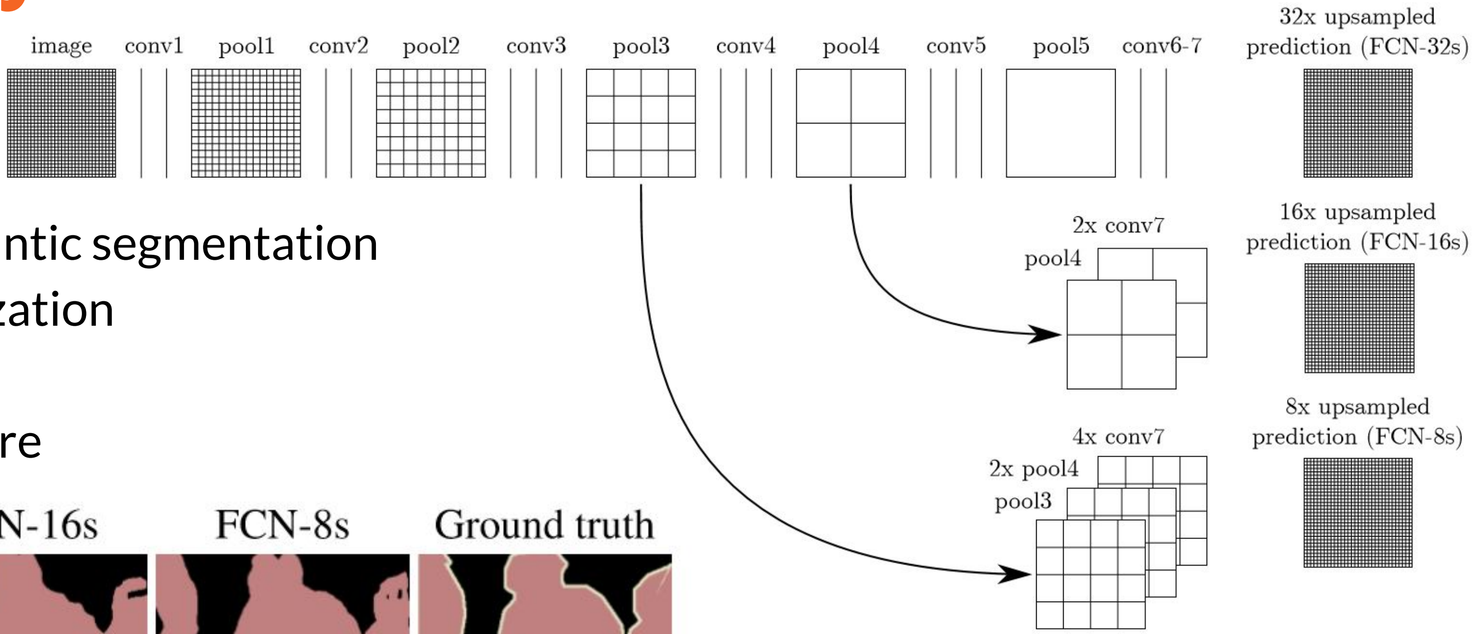
Faster R-CNN (2015)

- RPN: Region Proposal Network
- Generate k different anchor boxes (RoI) for each 3*3 region on feature map
- Center of sliding window on feature map maps to center of Anchor box on original image

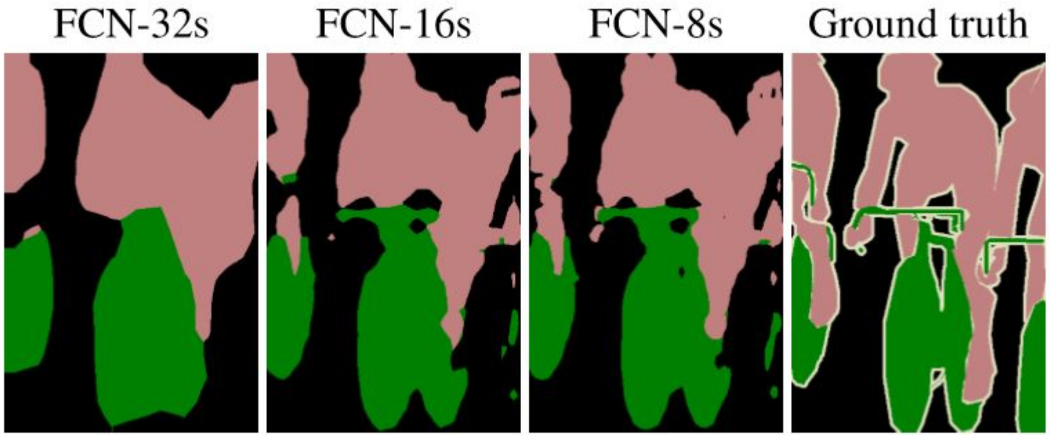




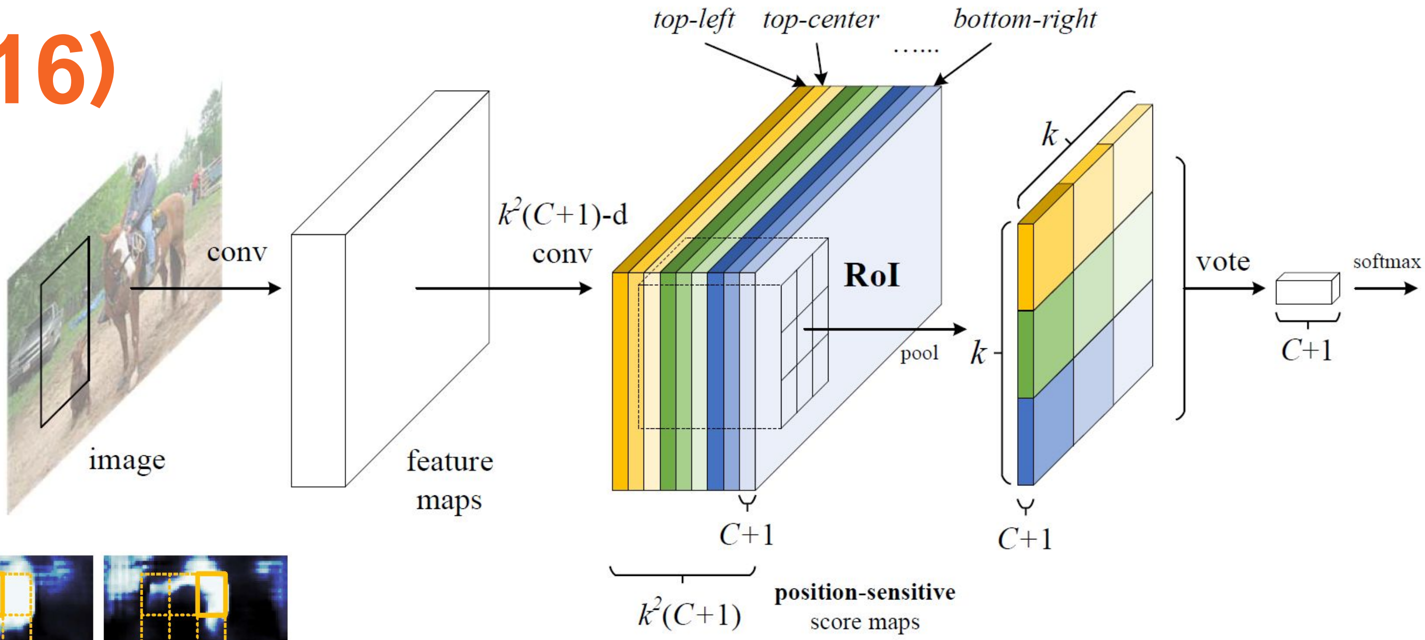
FCN: Fully Convolutional Networks (2016)

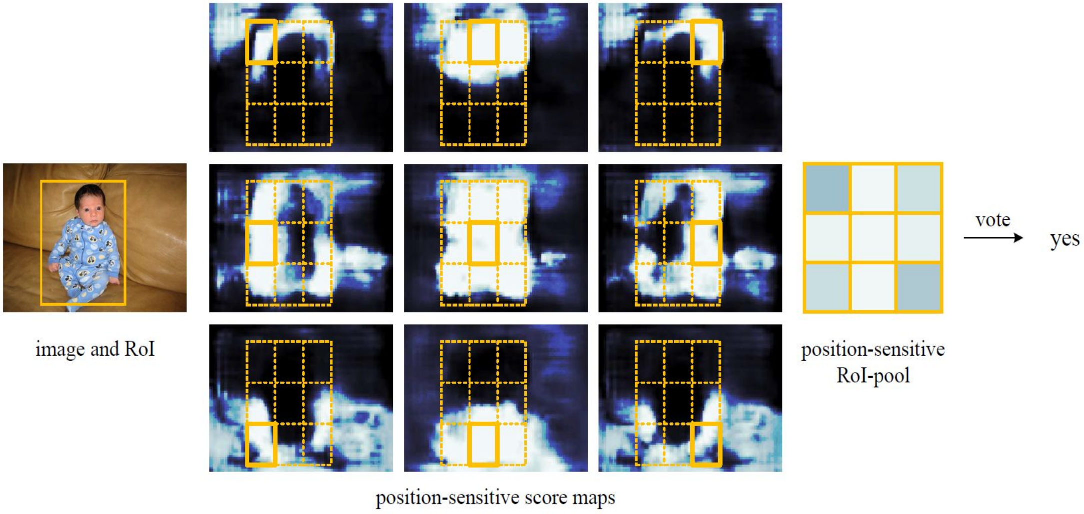


- For image semantic segmentation
- Convolutionalization
- Upsampling
- Skip Architecture



R-FCN (2016)

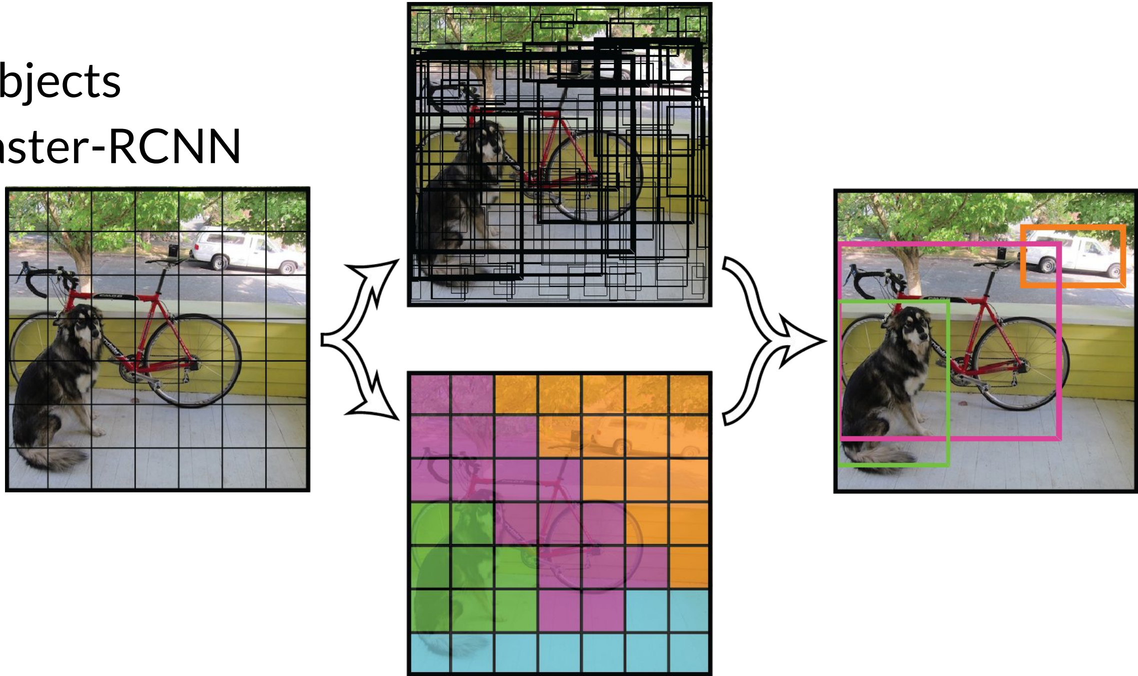


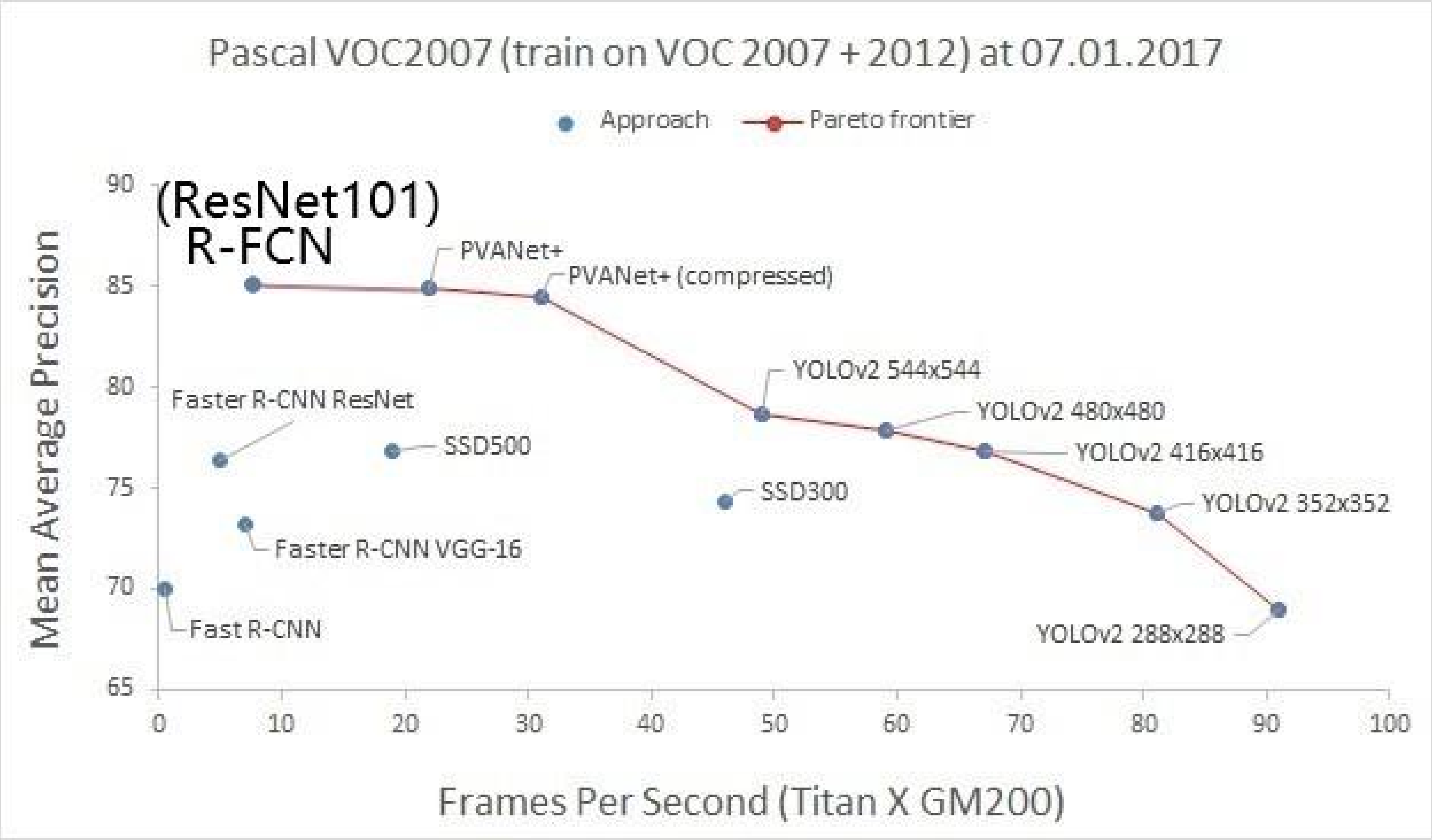


- Divide RoI into k^2 grids
- $k^2*(C+1)$ score maps generated from Fully Convolutional Network
- RoI pooling generates $k^2*(C+1)$ scores

YOLO(2015) & YOLOv2 (2016)

- No Region Proposal Network
- Divide image into k*k grids
- Each grid responsible for object centered in that grid
- Fast
- Bad for small and overlapped objects
- YOLOv2 integrates YOLO & Faster-RCNN



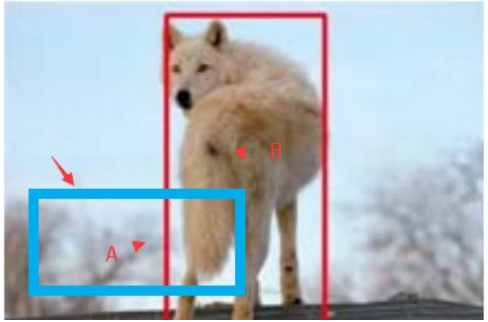


Pascal VOC2007 (train on VOC 2007 + 2012) at 07.01.2017
Approach
Pareto frontier
Mean Average Precision
Frames Per Second (Titan X GM200)
(ResNet101)
R-FCN
PVANet+
PVANet+ (compressed)
YOLOv2 544x544
YOLOv2 480x480
YOLOv2 416x416
YOLOv2 352x352
YOLOv2 288x288
Faster R-CNN ResNet
SSD500
SSD300
Faster R-CNN VGG-16
Fast R-CNN
90
85
80
75
70
65
0
10
20
30
40
50
60
70
80
90
100

Analysis & Evaluation

$$AP = \frac{1}{11} \sum_{r \in \{0,0.1,\ldots,1\}} p_{interp}(r)$$

- For each category:
  - Intersection over Union (IoU) threshold: 50%
  - Average precision: 11-point average precision/recall
  - Same as The PASCAL Visual Object Classes (VOC) Challenge
- Proportion of bounding boxes:
  - Vehicles: 87%
  - Pedestrian: 7%
  - Cyclist: 6%
  - Traffic_lights: 3%
- Evaluation: Weighted average precision



Low IoU

Baseline

- Models trained on VOC 2007+2012
- Classifier: ResNet101 > DarkNet19 > ZF

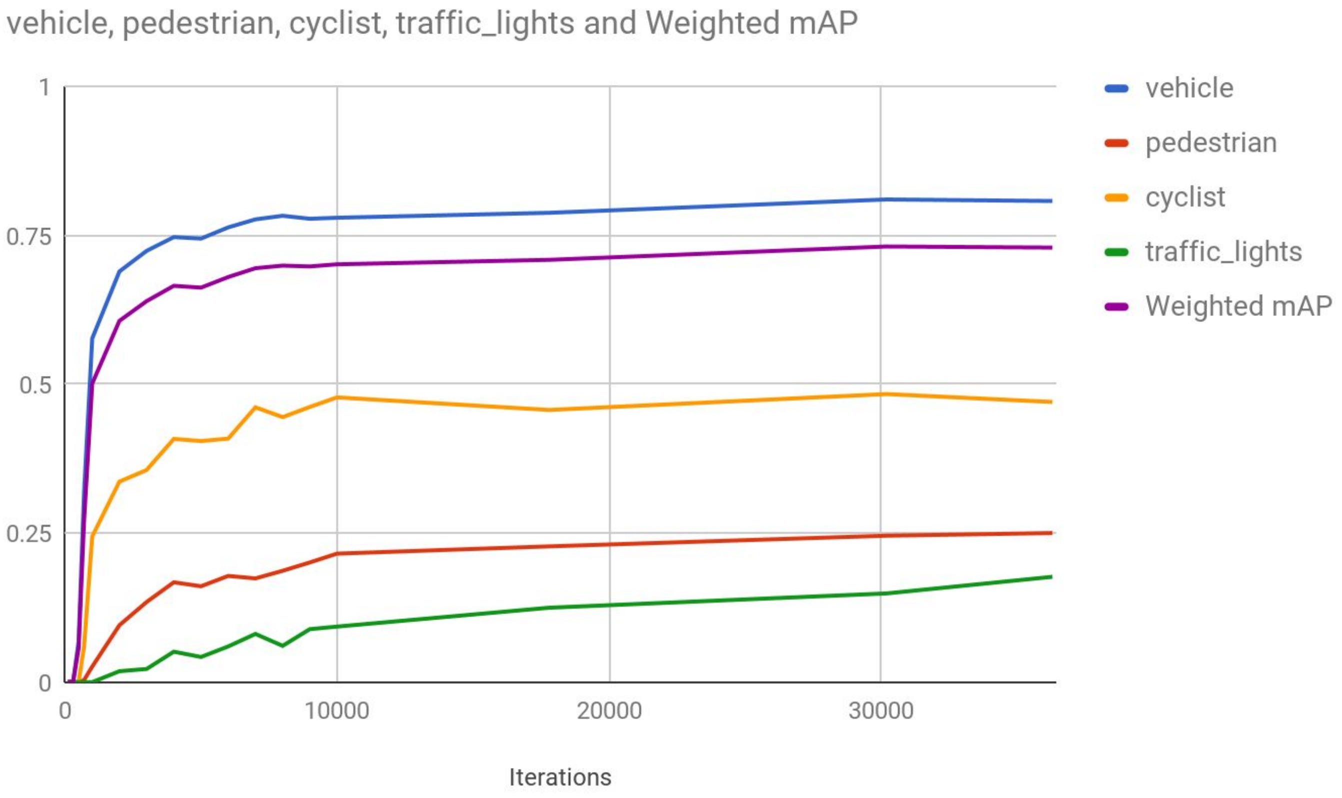
Model	Classifier	FPS	Vehicle (car + bus)	Pedestrian (person)	Cyclist (bicycle + motocycle)	Traffic_lights (N/A)	Weighted mAP
Faster-RCNN	ZF	7.87	0.4246	0.0695	0.0609	0	0.3779
R-FCN	ResNet101	4.58	0.6144	0.1412	0.2033	0	0.55661
YOLOv2	DarkNet19	37.04	0.4466	0.0499	0.0543	0	0.395293

Train

- Setup: Caffe + AWS g3.16xlarge
  - 4*Tesla M60, 64 vCPUs, 488G RAM
- Modify network and data pipeline to fit our data

Model	Classifier	Iterations	Detection FPS	Vehicle	Pedestrian	Cyclist	Traffic_lights	Weighted mAP
R-FCN	ResNet101	60000	4.57	0.8002	0.3184	0.5783	0.1215	0.7329
YOLOv2	DarkNet19	30000	27.8	0.8045	0.2335	0.4739	0.146	0.7249

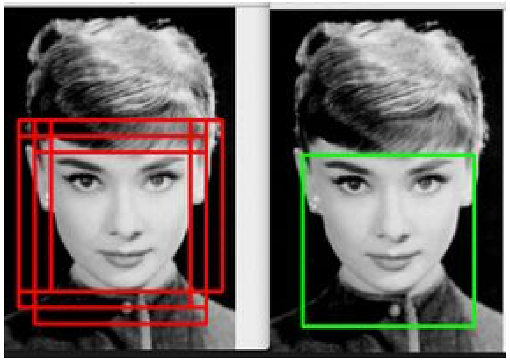
Sample W-mAP vs Iterations



NMS (Non-Maximum Suppression)

- Remove duplicate boxes for same box

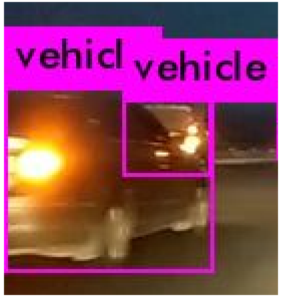
Set detected_boxes = all bounding boxes detected;
Set valid_boxes = empty;
while *detected_boxes is not empty* **do**
    Box valid_box = box in detected_boxes with max confidence;
    foreach *Box box in detected_boxes* **do**
        if *IoU(valid_box, box) > nms_threshold* **then**
            detected_boxes.remove(box);
        end
    end
    valid_boxes.add(valid_box);
    detected_boxes.remove(valid_box);
end



IoU threshold	0.4	0.45	0.5
R-FCN	0.7273	0.7329	0.7316
YOLOv2	0.723	0.7249	0.7228

Soft-NMS (2017)

```
Set detected_boxes = all bounding boxes detected;
Set valid_boxes = empty;
while detected_boxes is not empty do
    Box valid_box = box in detected_boxes with max confidence;
    foreach Box box in detected_boxes do
        box.confidence = box.confidence * (1 - IoU(valid_box, box));
    end
    valid_boxes.add(valid_box);
    detected_boxes.remove(valid_box);
end
```



W-mAP	Threshold =0.45	Soft-NMS
R-FCN	0.7329	0.7408
YOLOv2	0.7249	0.7231

Multi-Scale Training (YOLOv2 Only)

- Randomly scale input image to 704*352, 640*320, 576*288 or 512*256
- R-FCN already has multi-scale anchors in Region Proposal Network

Model	Multi-Scale	Iterations	Vehicle	Pedestrian	Cyclist	Traffic_lights	Weighted mAP
YOLOv2	Yes	30000	0.81	0.2459	0.4837	0.149	0.7314
YOLOv2	No	30000	0.8045	0.2335	0.4739	0.146	0.7249

Modify RPN Anchors (R-FCN Only)

- Original anchors (scale is based on input size of 1000*563):
  - scale: [8, 16, 32] * 16 pixels
  - Ratio: [0.5, 1, 2]
  - RPN_MIN_SIZE = 16 pixels
  - 9 anchors per sliding window
- Observations:
  - A lot of small objects
  - Objects with large ratio: pedestrian & cyclist
- Modified anchors:
  - scale: [2, 4, 8, 16, 32] * 16
  - Ratio: [0.3, 0.5, 1, 2, 3] pixels
  - RPN_MIN_SIZE = 4 pixels
  - 25 anchors per sliding window

R-FCN	Weighted mAP
Before	0.7408
After	0.7895

Data Augmentation

- Crop
  - Most objects appear in the bottom 75%
  - Crop left bottom and right bottom (480*270)
  - Discard bounding boxes that are cropped more than 75%
- Flip
- Results in 48000 training data
- Also tried to Stretch image, but failed to improve

W-mAP	Before	After
R-FCN	0.7895	0.7941
YOLOv2	0.7314	0.7388

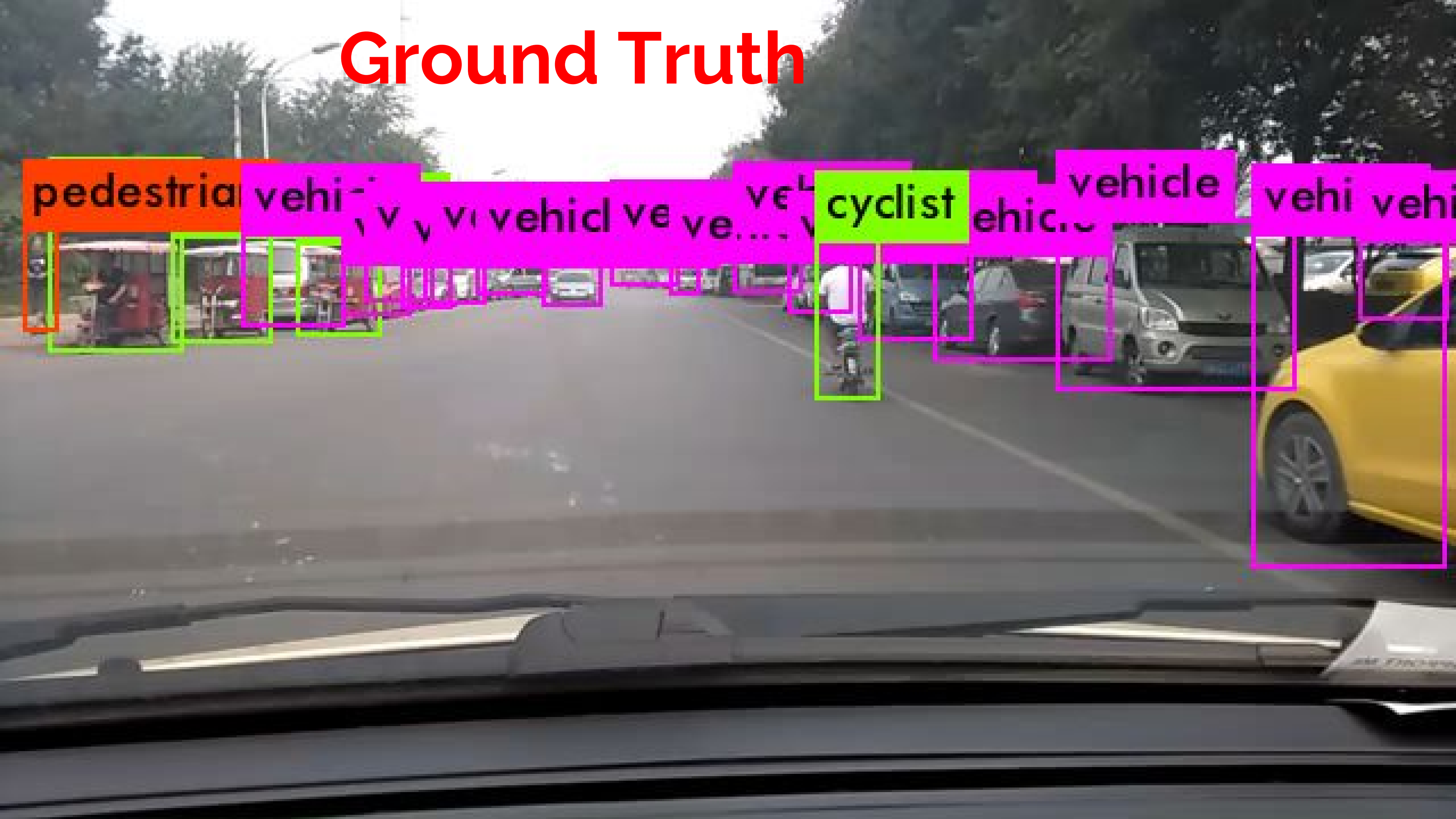
Finally, Model Integration

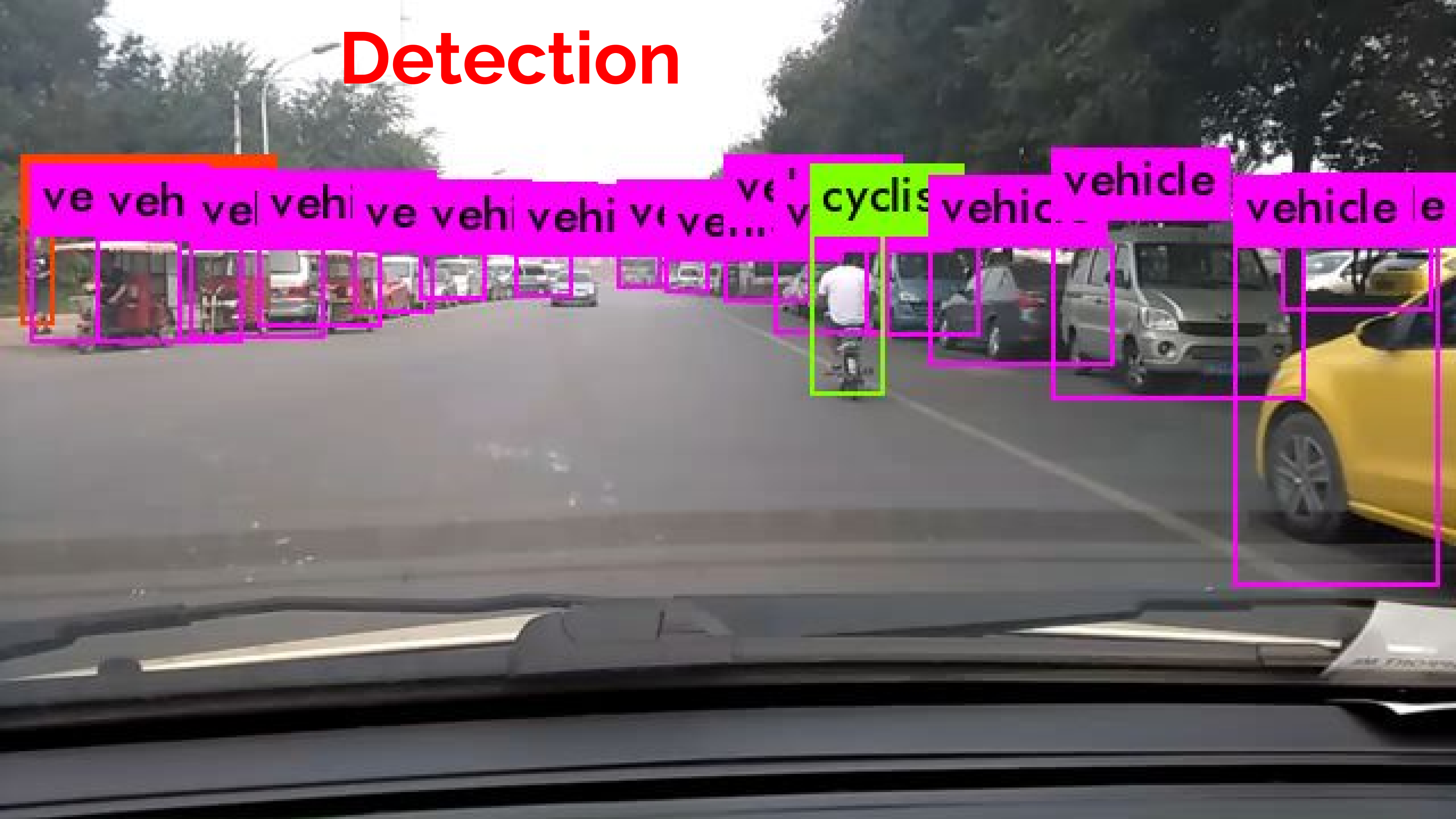
- Detect with both YOLOv2 and R-FCN
- Remove overlapping box using Soft-NMS

W-mAP	Before
R-FCN	0.7895
YOLOv2	0.7314
Integration	0.7912

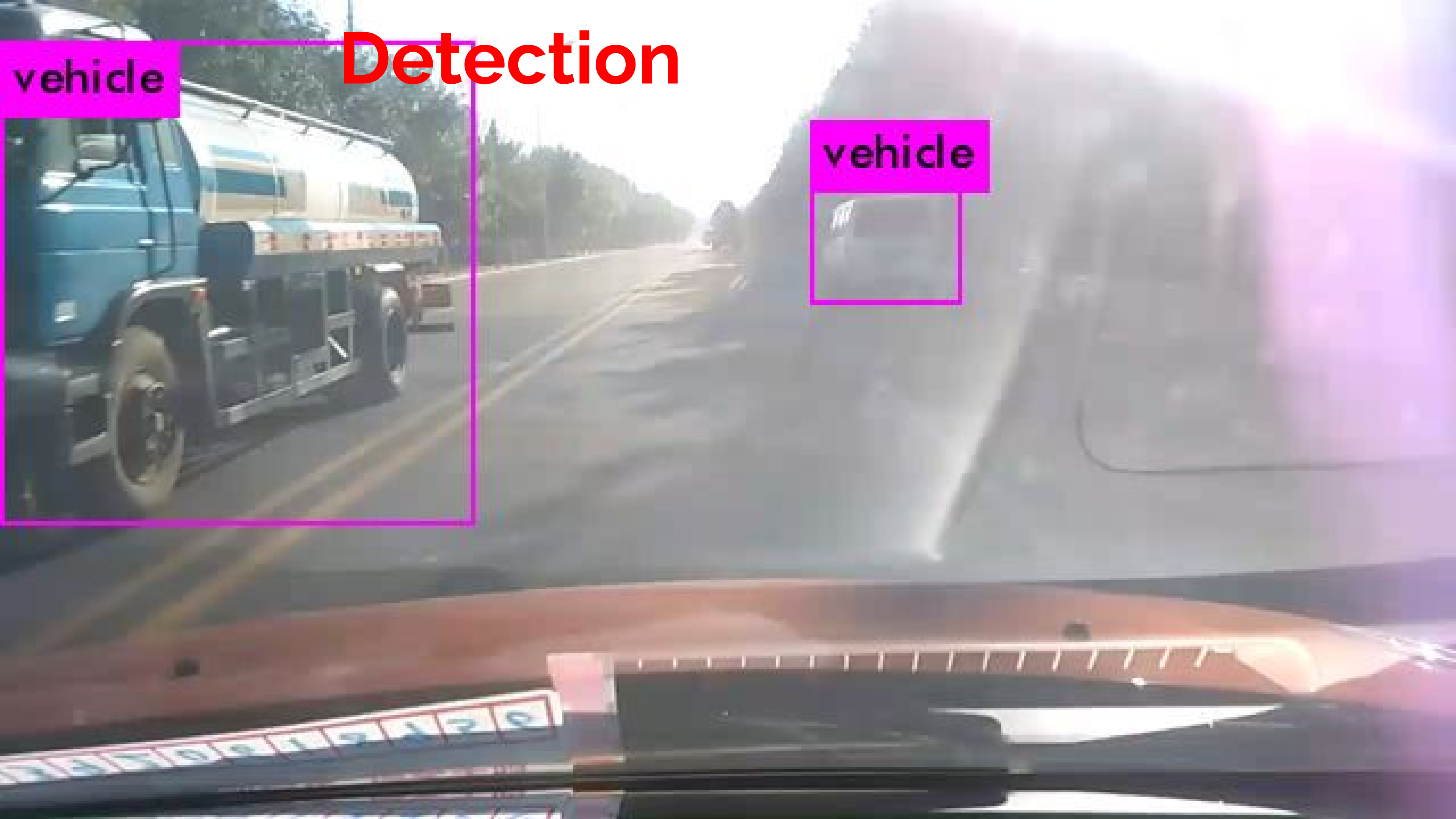

Ground Truth
vehicle


Detection
vehicle


Ground Truth
cyclist
vehicle


Detection
vehicle
vehicle

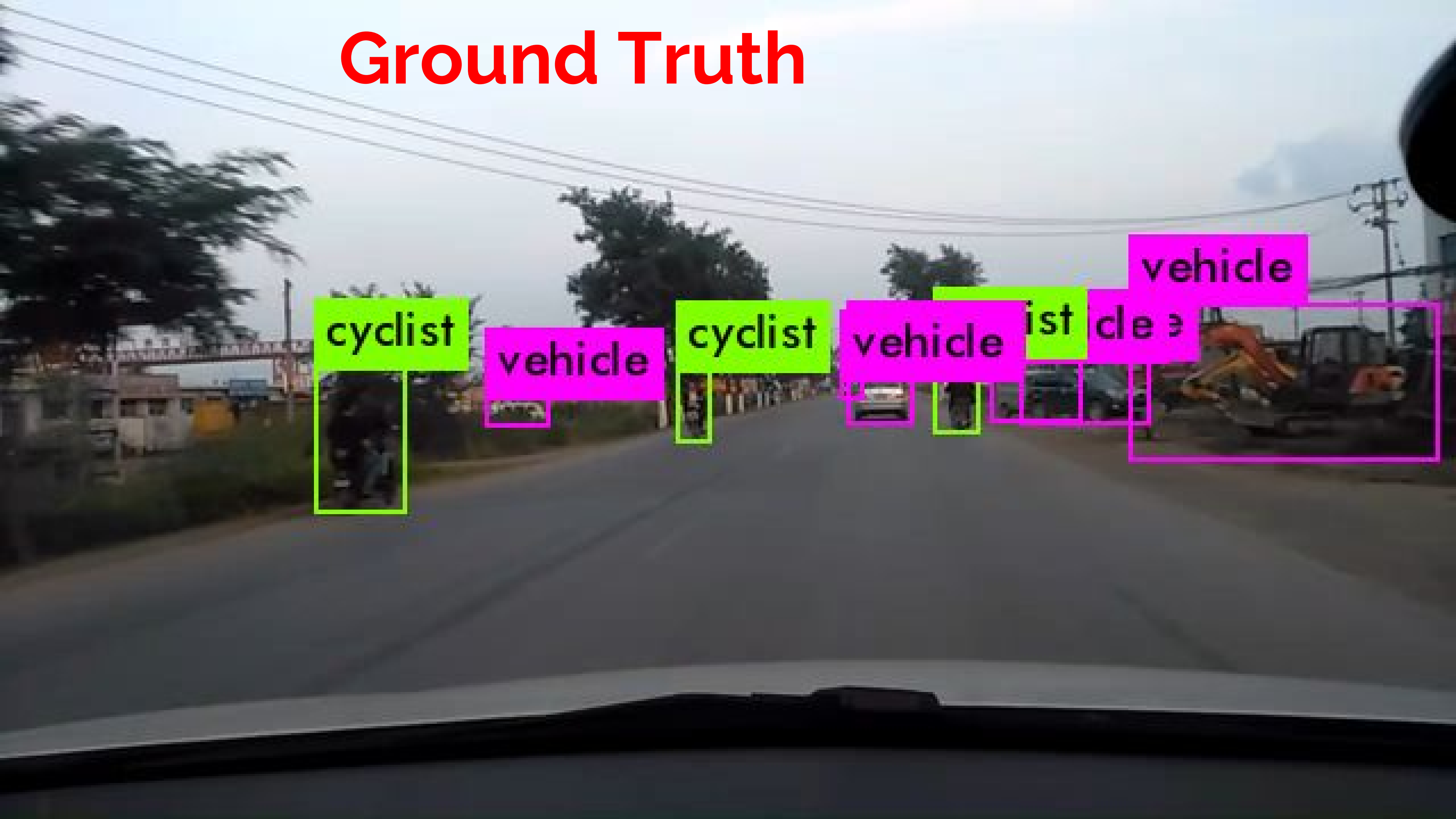

Ground Truth
vehicle
vehicle
hicle

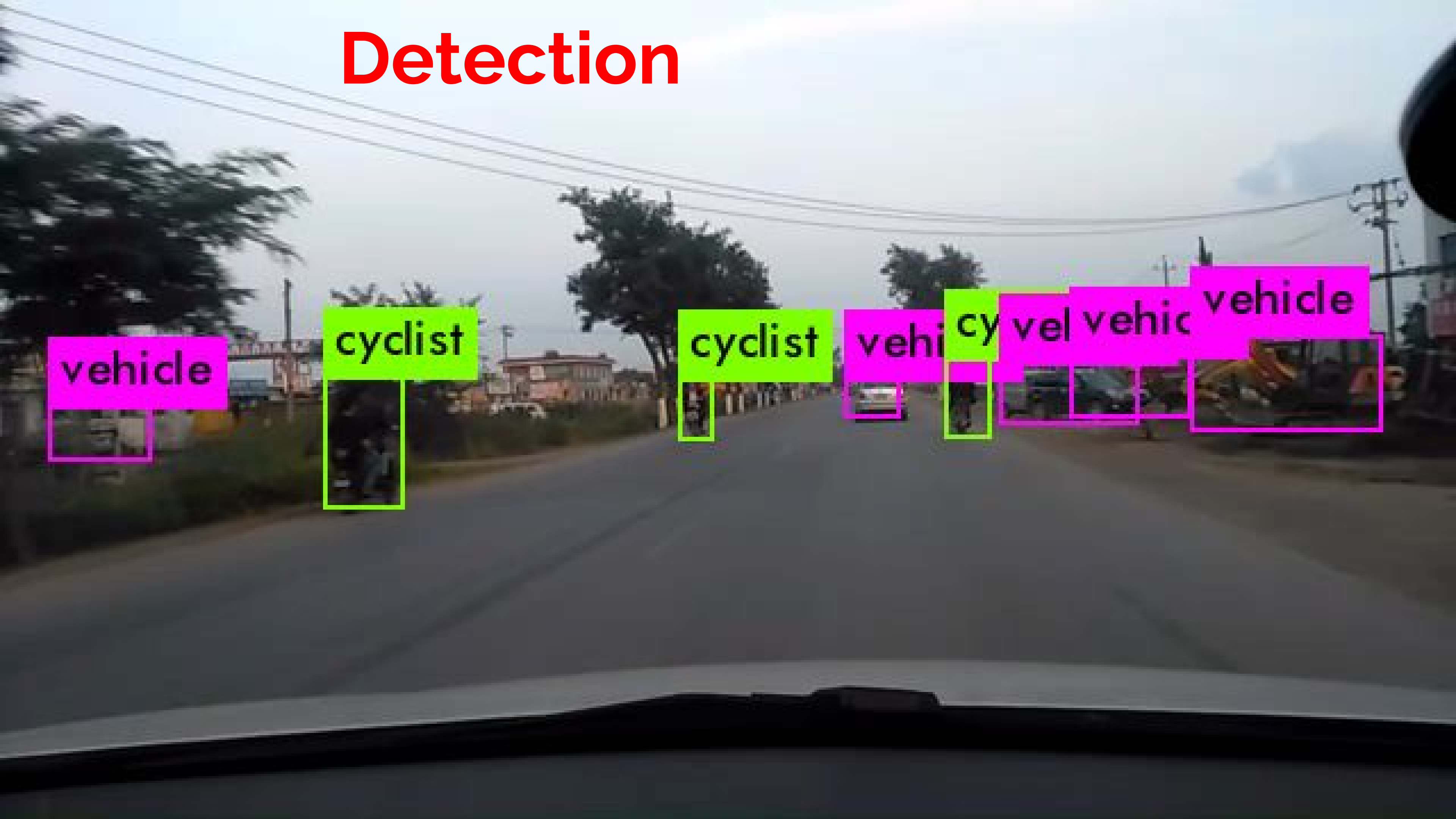

Detection
vehicle
vehicle

Ground Truth

Detection

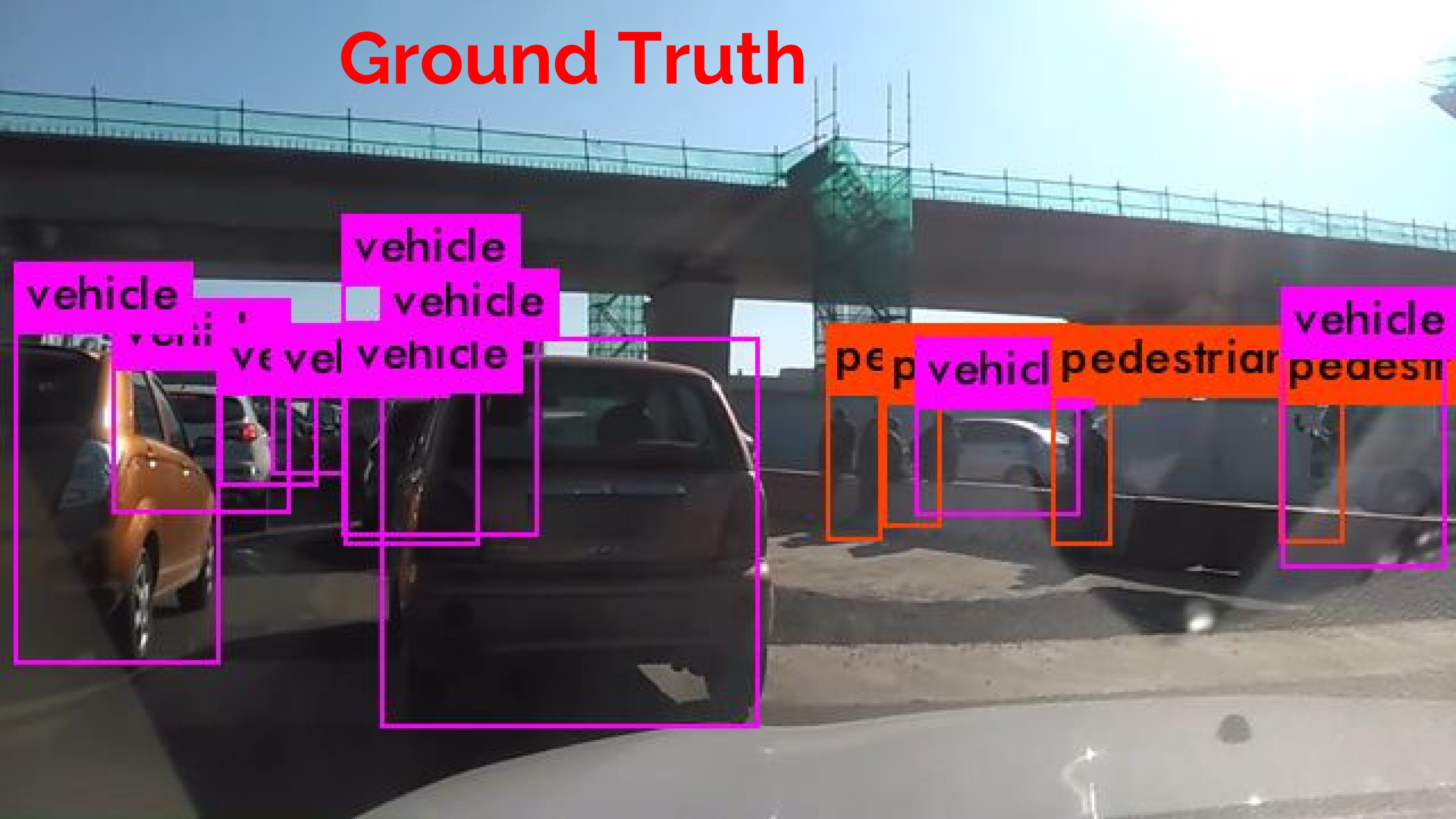
Ground Truth

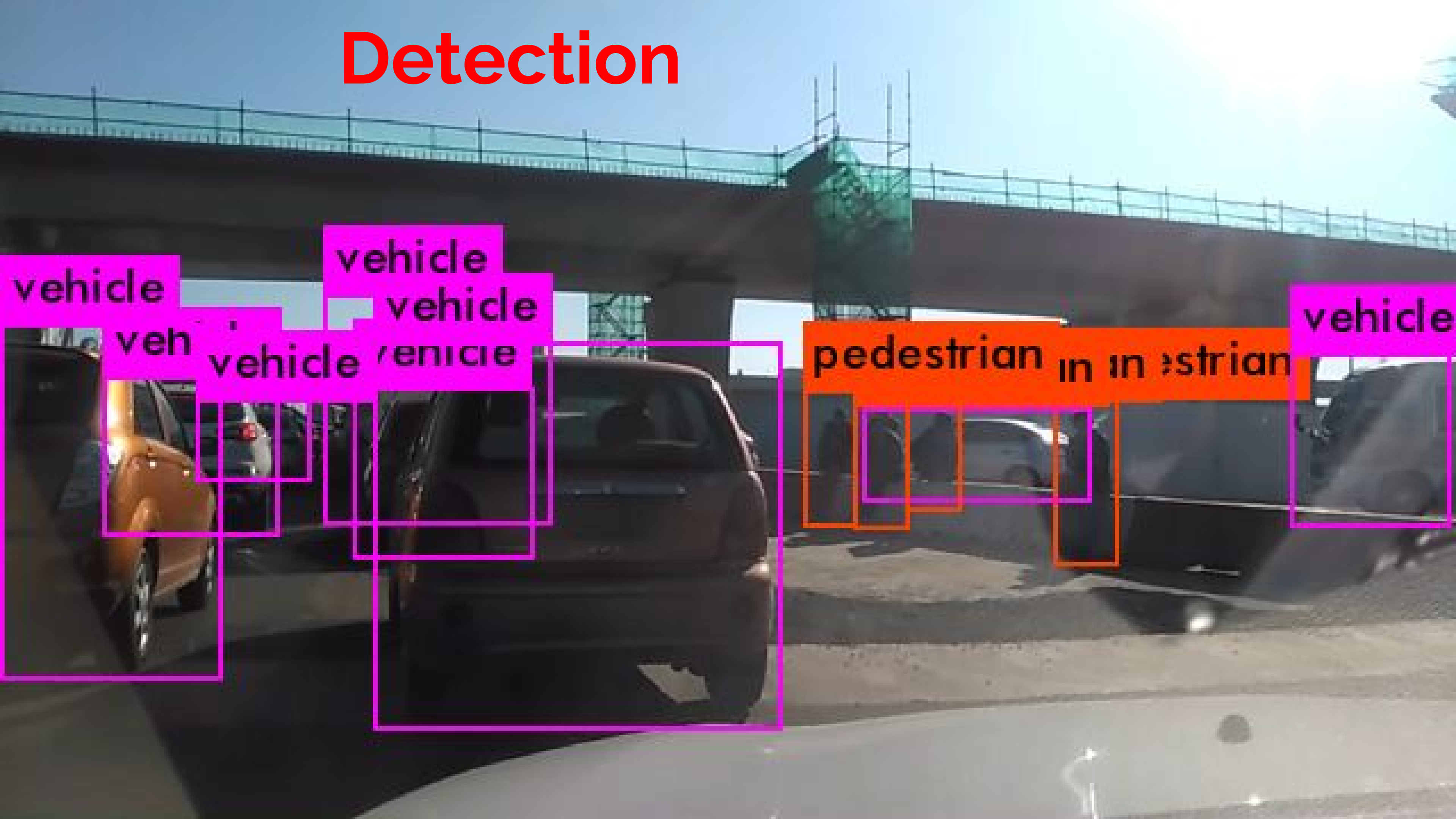



Detection
vehicle
cyclist
cyclist
vehicle


Ground Truth


Detection
vehicle
vehicle


Ground Truth
vehicle
vehicle
vehicle
vehicle
vehicle

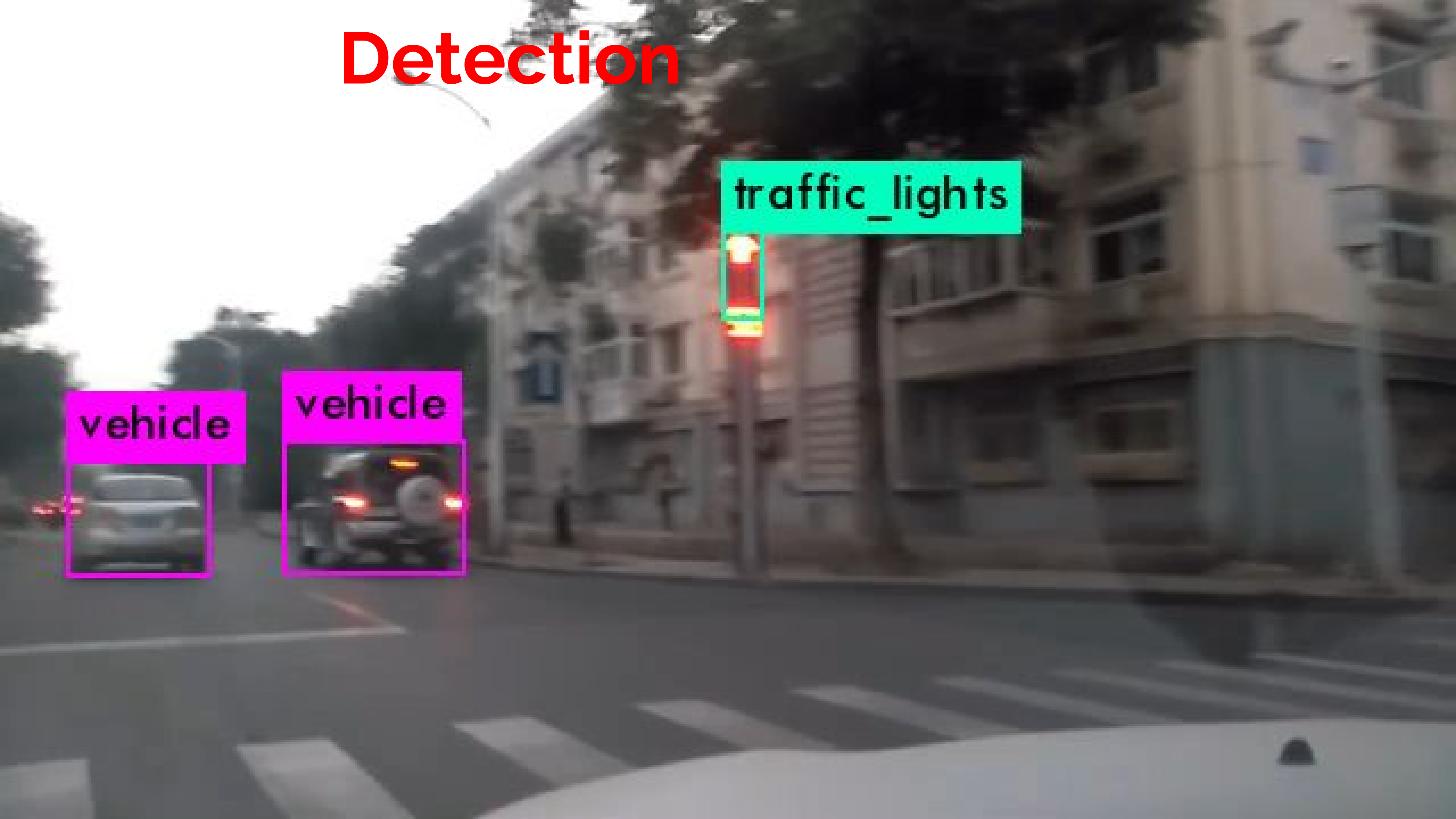

Detection
vehicle
vehicle
vehicle
vehicle
vehicle
pedestrian

Ground Truth


Detection
vehicle


Ground Truth
traffic_lights
vehicle
vehicle

Detection



Future Work

- Clean data
- Fine-tuning for pedestrian, cyclist, and traffic_lights, will lose generalization
- Deformable-R-FCN (2017)
- OHEM: Online Hard Example Mining (2016)
- Stratified-OHEM (2017)

Q&A