Vehicle detection on Singapore Roads to assist in road maintenance

Quek Zhi Qiang

Thursday, 29 Apr 2021

Background

- ~ 1 million vehicles on our roads
- > 9000 lane kilometers
    - 12% of Singapore’s area
- Crucial infrastructure for economic development

Problem Statement

Existing maintenance regime by LTA

- Preventive maintenance
  - Periodic inspection and road survey
- May not be optimal

How could we improve and value add?

- Use traffic condition as a source of information

Object Detection as an approach

Handling the “**what**” and “**where**”

- Vehicle type (**What**)
- Location in the image (**Where**)

Considerations

Time

Model development and data annotation

Framework

Ease of use and documentation

Deployability

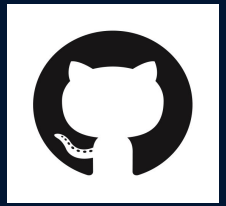
Fast inference by model

Data

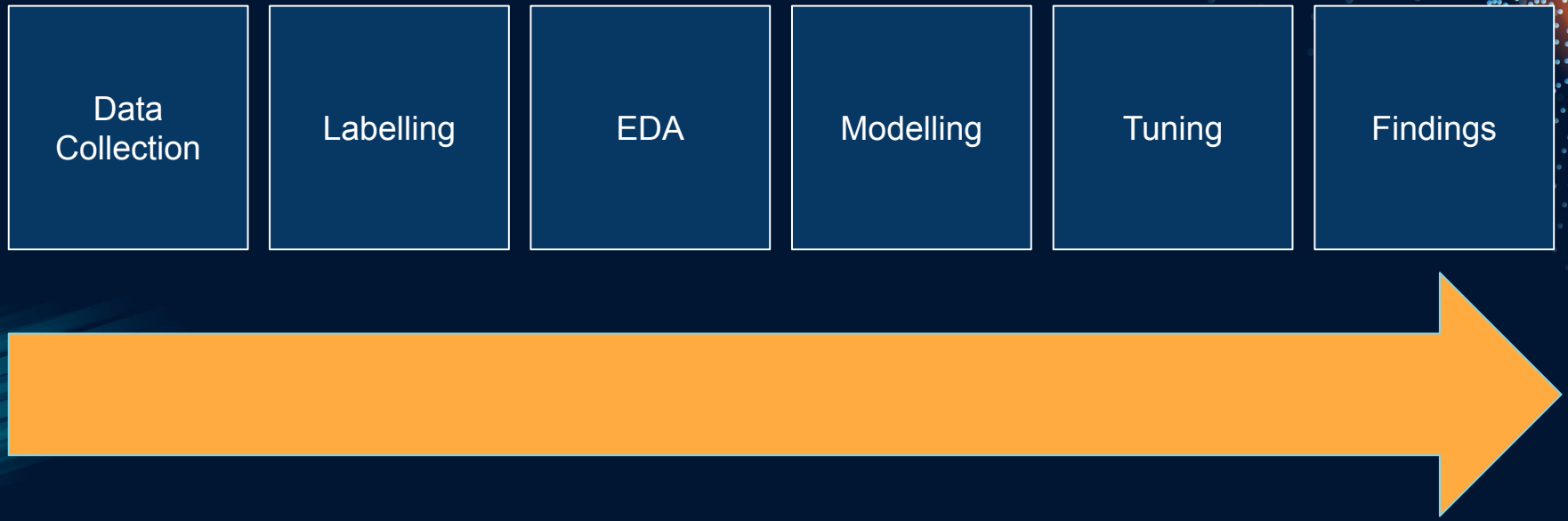
Resolution
Road conditions
Colour

Tools and Frameworks

- **Anaconda environment**
  - **Python language**
- **Github** - Code repository
- **Makesense.ai** - Annotation tool
- Object Detection frameworks
  - Tensorflow (Had issues)
  - PyTorch (Had issues)
  - **Darknet**
    - **Written in C and CUDA**



Process



35,984

Images scraped via data.gov.sg API on 20 Mar 2021

87

CCTV footages

2,700

Images manually labeled

5

Vehicle classes considered

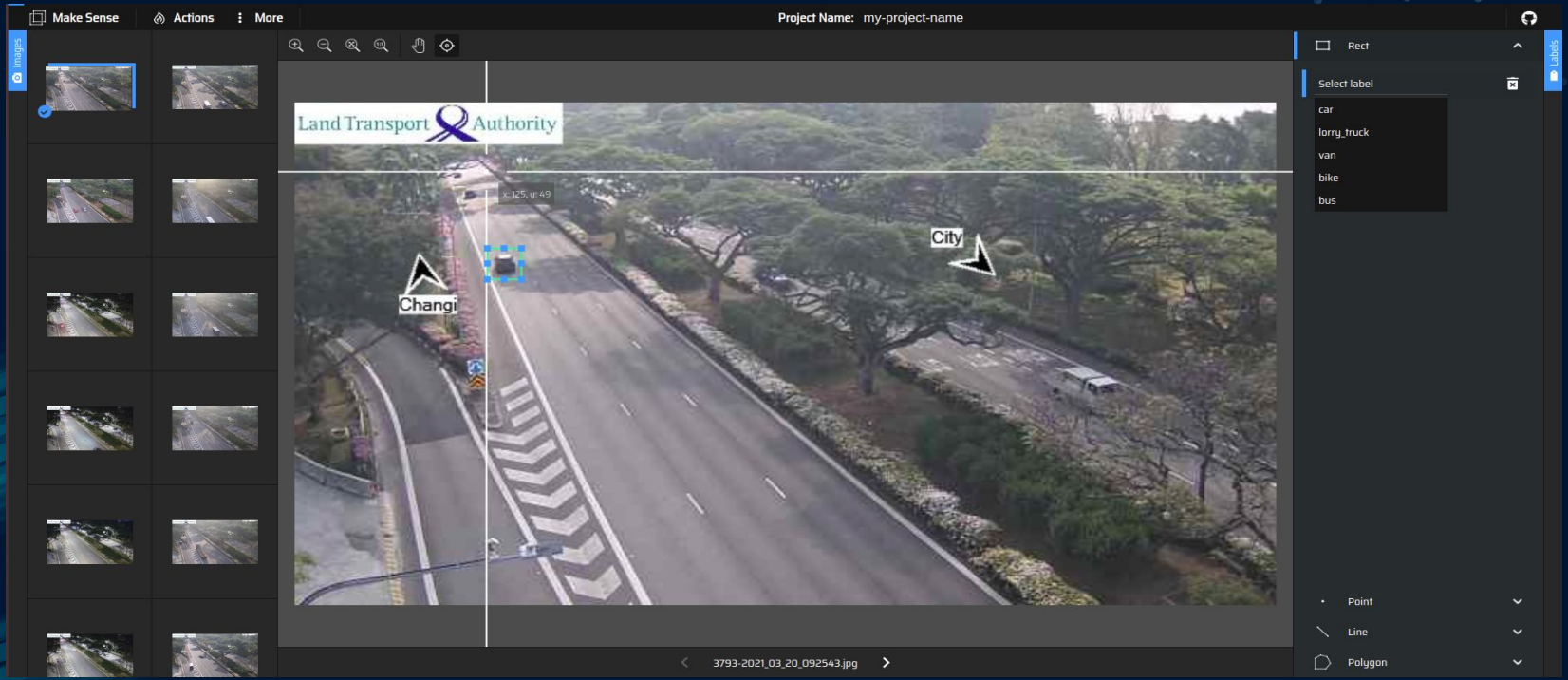
Data resolutions

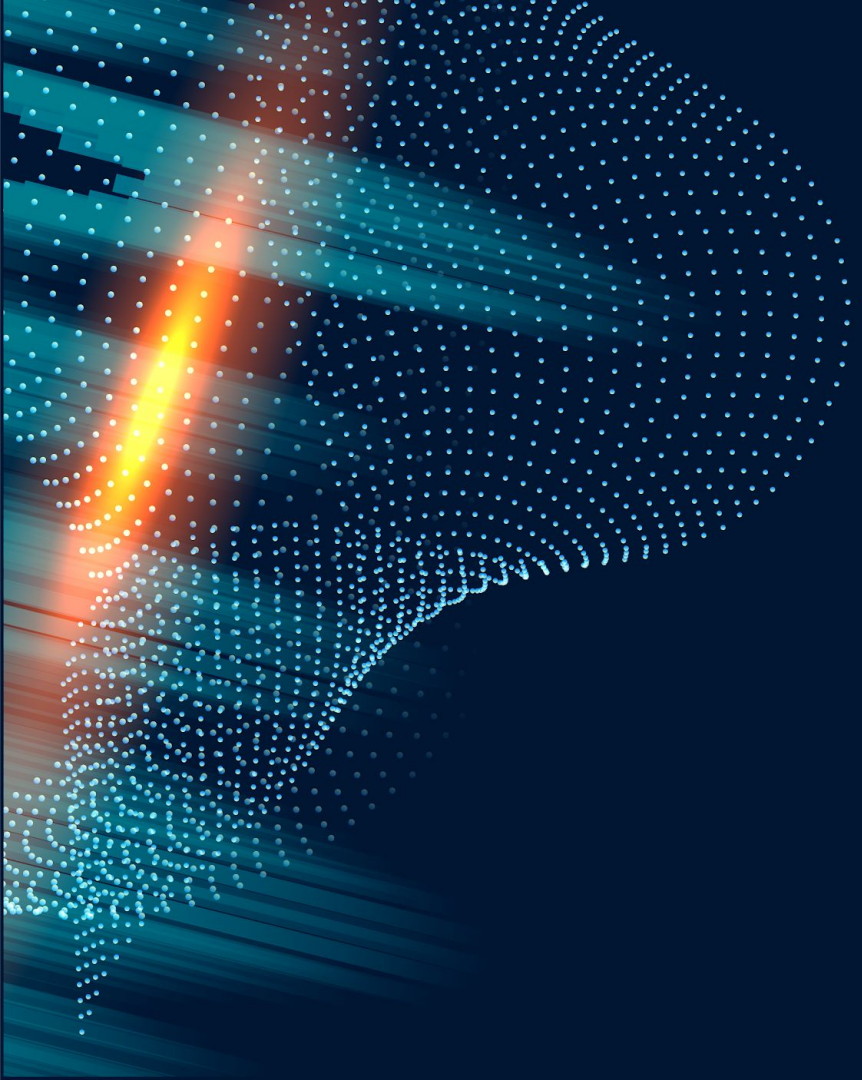
640 X 480

640 X 360

320 X 240

Annotation tool



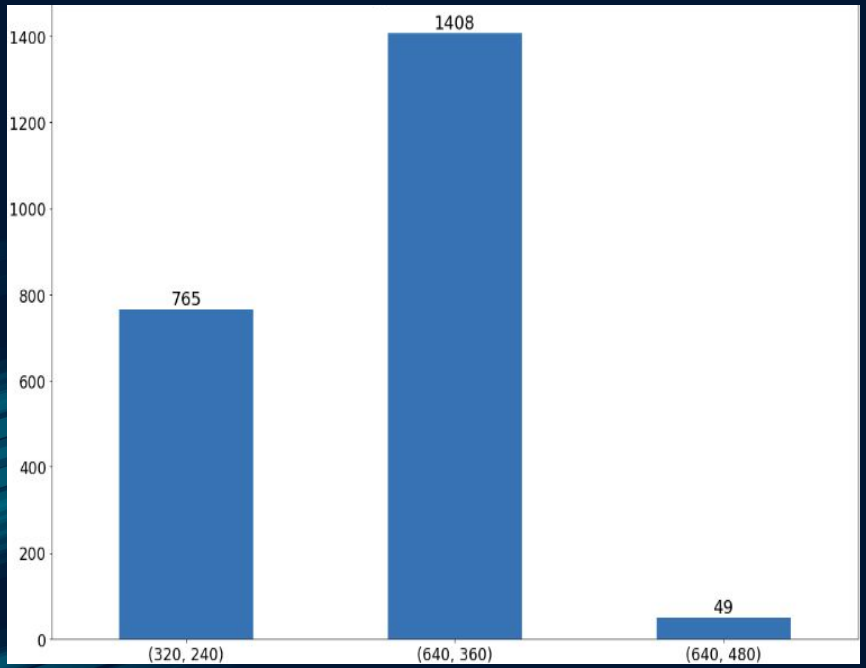


EDA

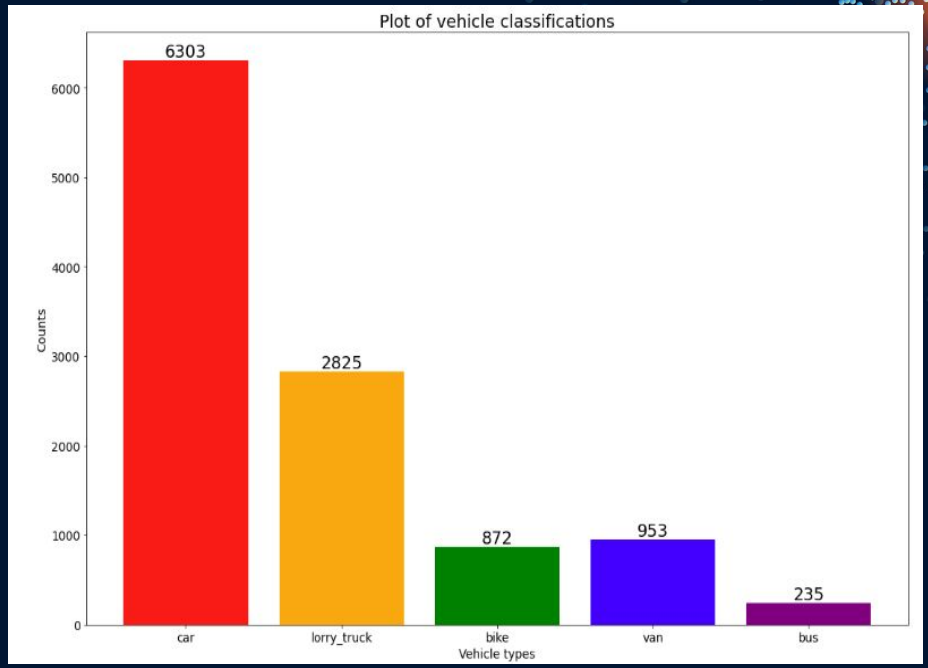
Exploratory Data Analysis

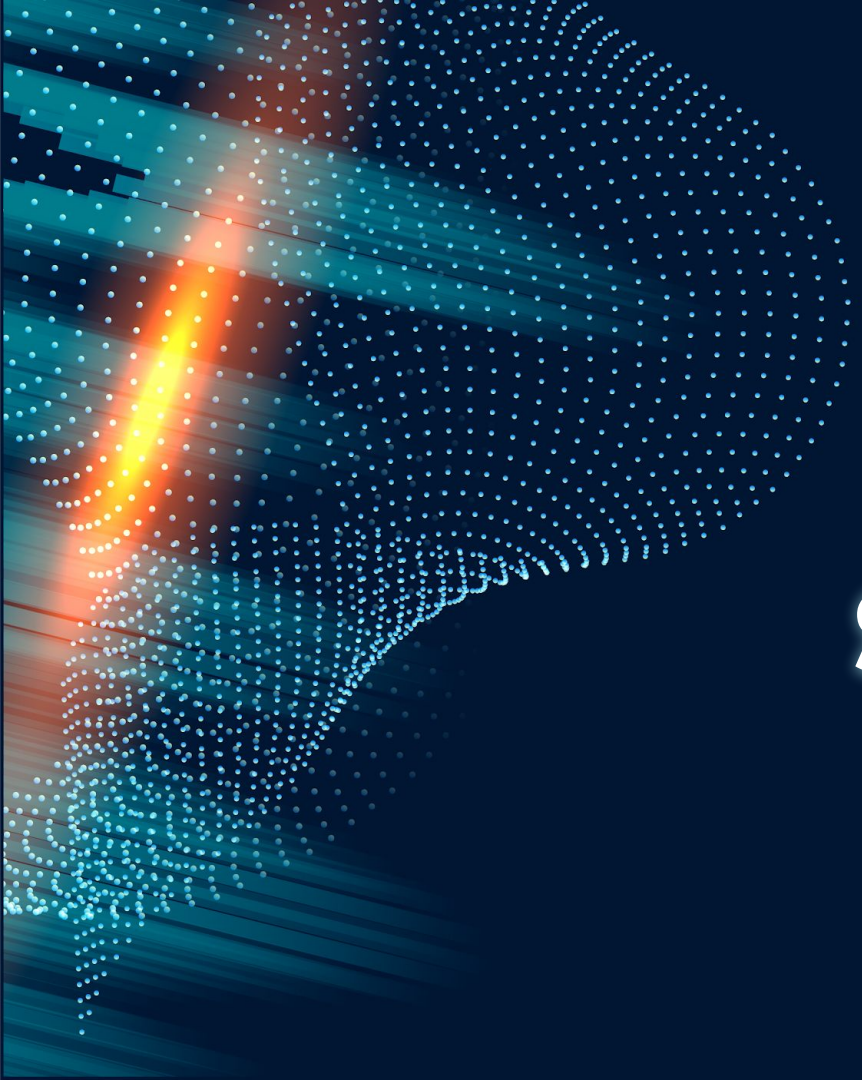
Distributions of images with vehicles

Image Resolutions



Vehicle Classifications





Data splittng

Preparing for model training

Train test split

5 classes

Car , Lorry/Truck, Bike, Van and Bus

27

CCTV sources

100

Images per CCTV source

50

Images each from 0600 to 1000H and 1800H to 2200H per CCTV source

2222

Images with at least 1 vehicle labeled

1741

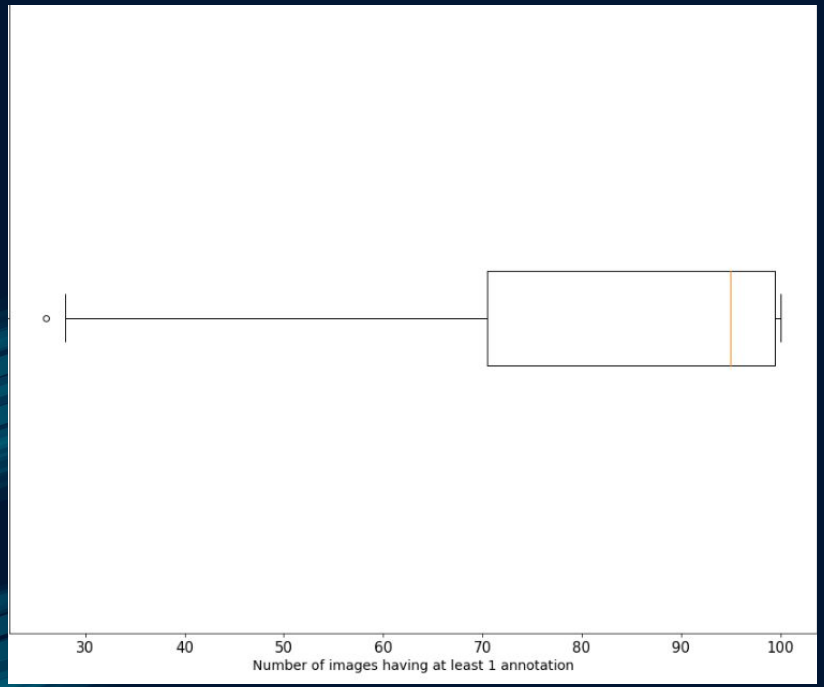
Images from 21 CCTV sources used for training

481

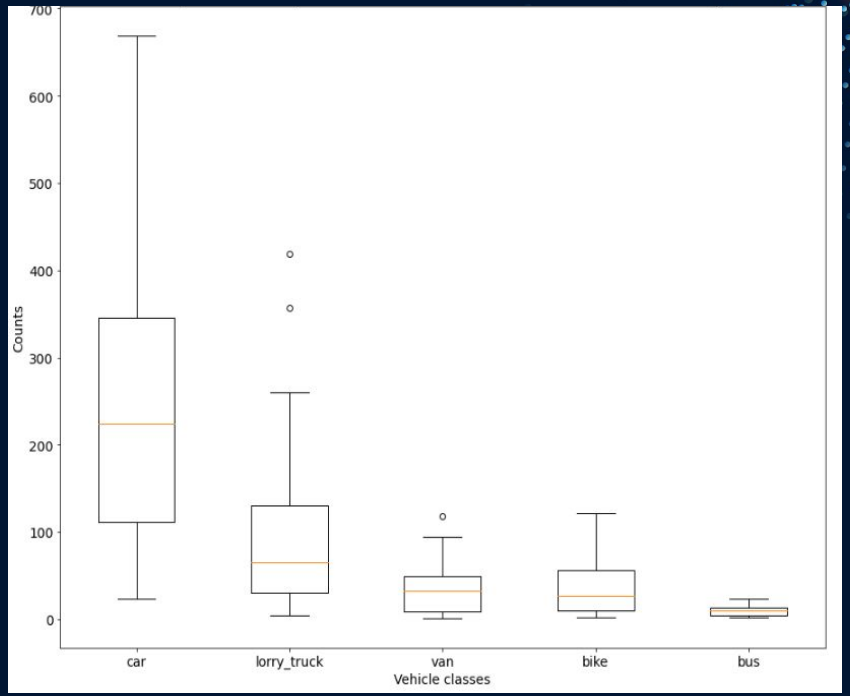
Images 6 CCTV sources used for validation

Distribution of data across CCTV sources

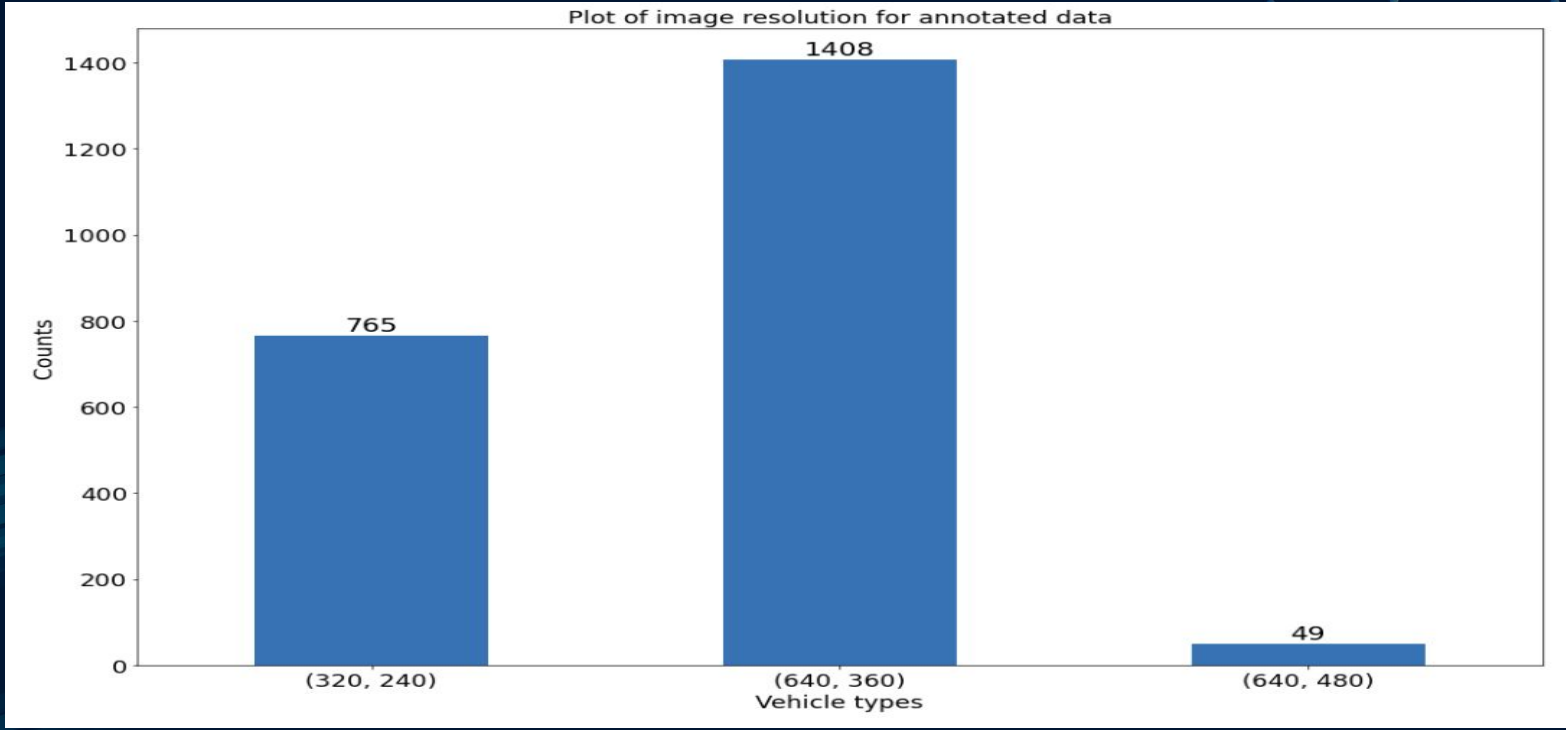
Uneven amount of annotated images per CCTV ID source



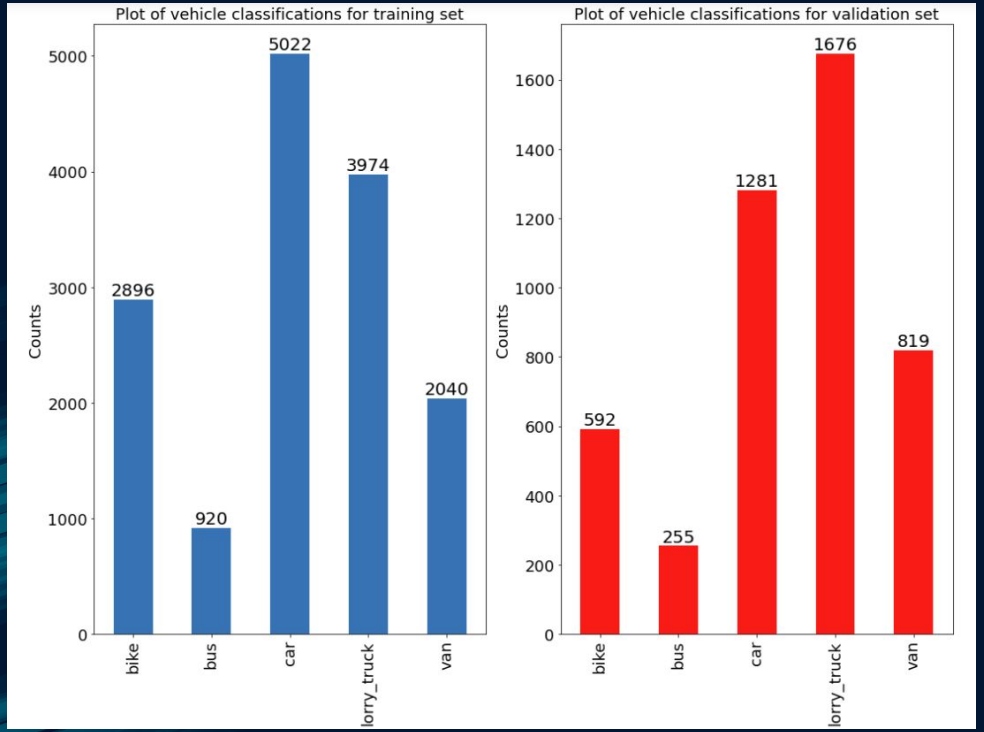
Vehicle types distribution

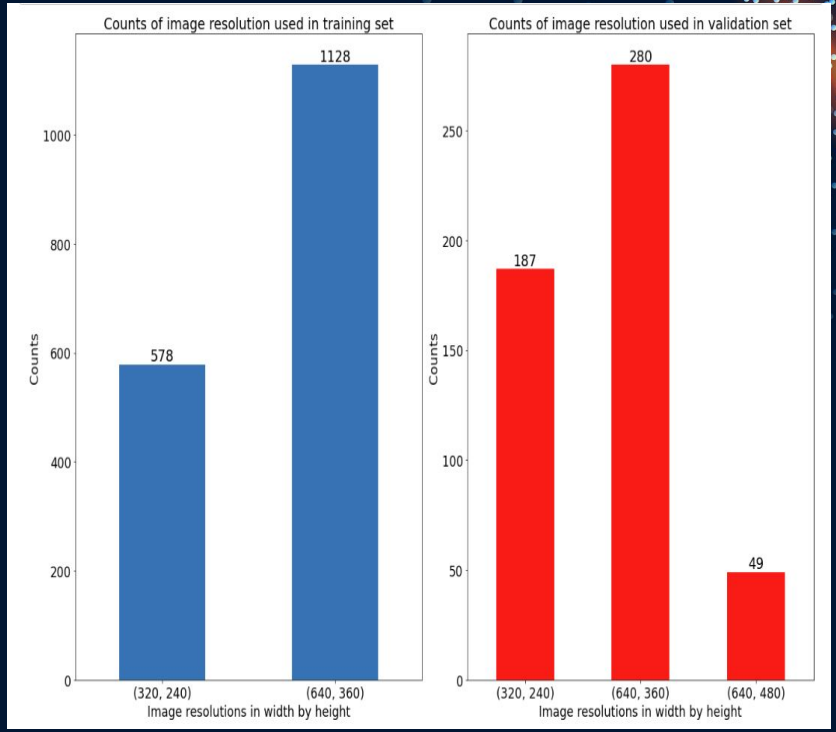


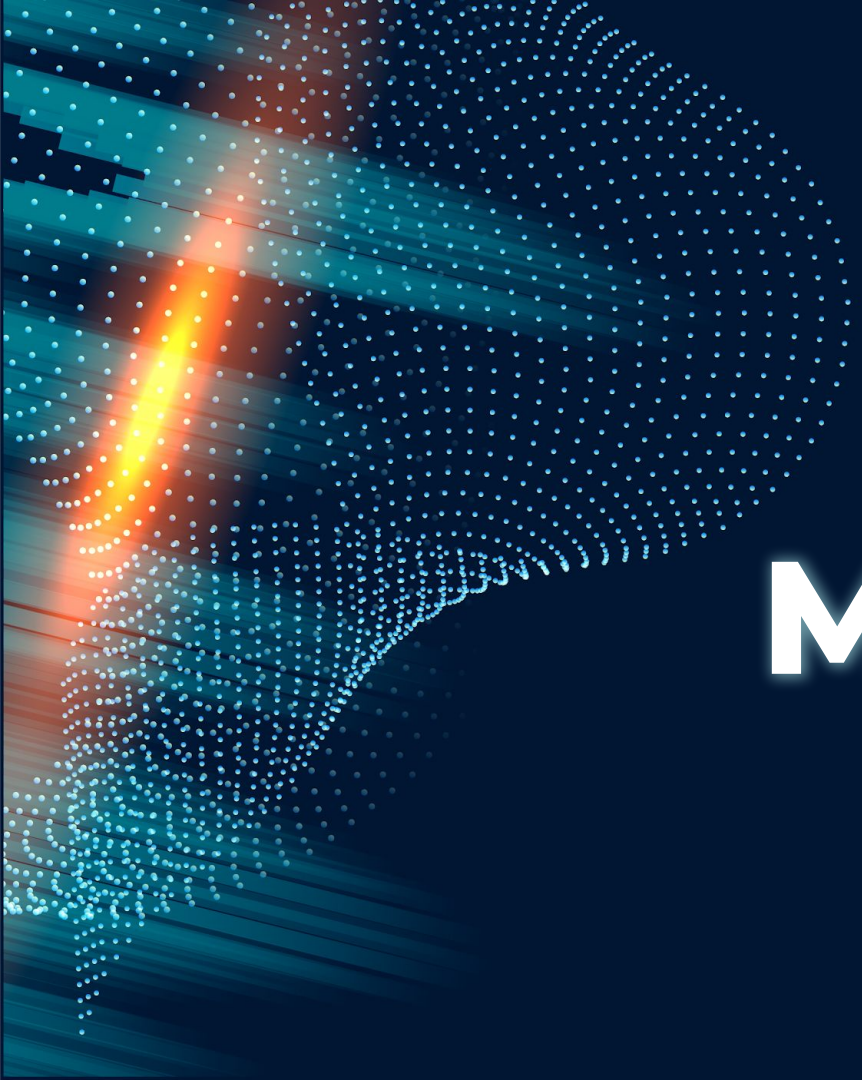
Distribution of image resolution



Train and validation data





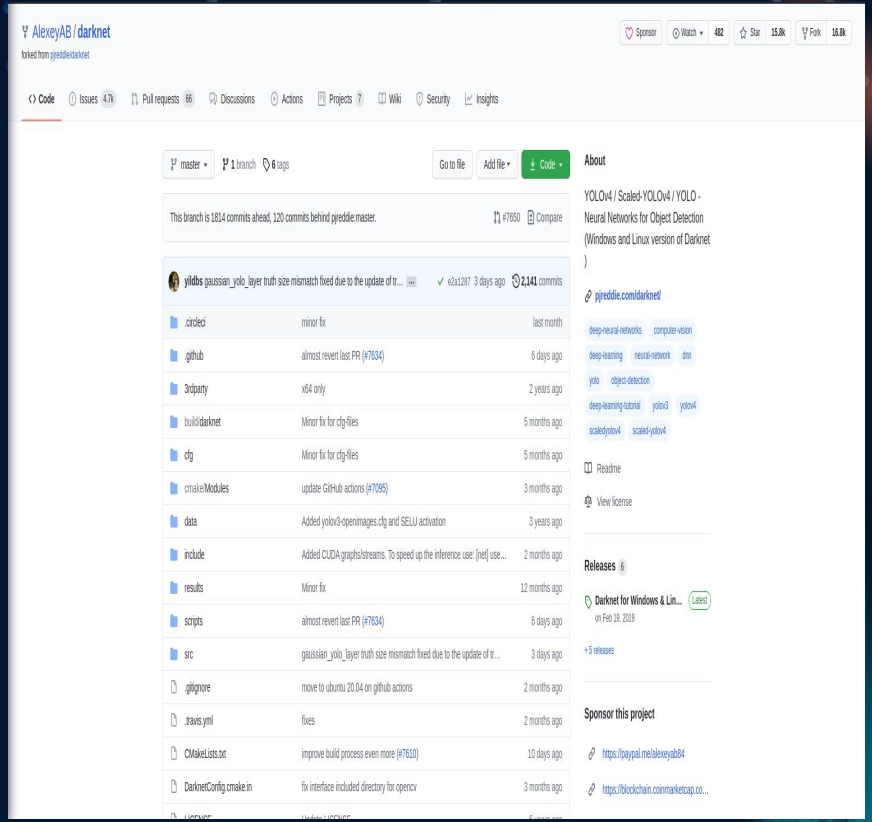


Modelling

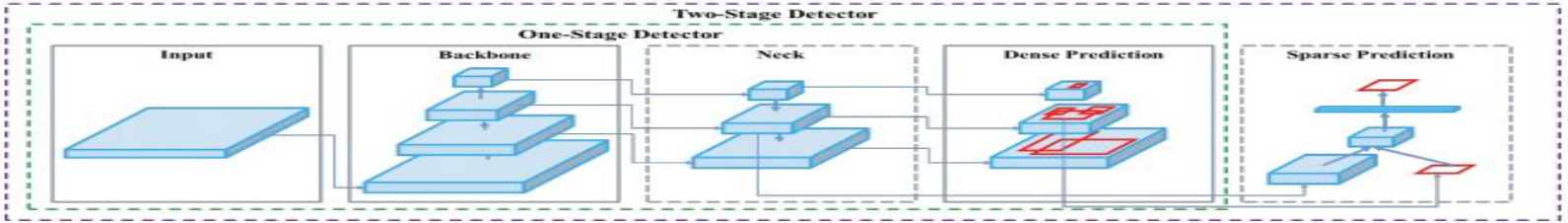
Using Darknet framework

You-Only-Look-Once (YOLO) from Darknet framework

- Repository: https://github.com/AlexeyAB/darknet
- Easy to use
- State-of-the-art model



Object Detection Architecture



Input: { Image, Patches, Image Pyramid, ... }

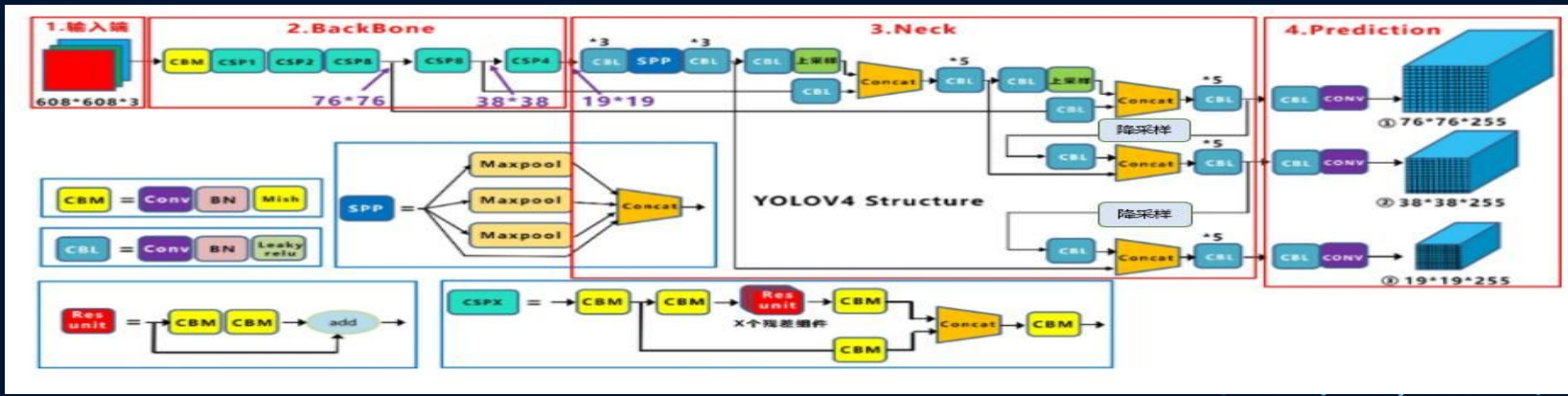
Backbone: { VGG16 [68], ResNet-50 [26], ResNeXt-101 [86], Darknet53 [63], ... }

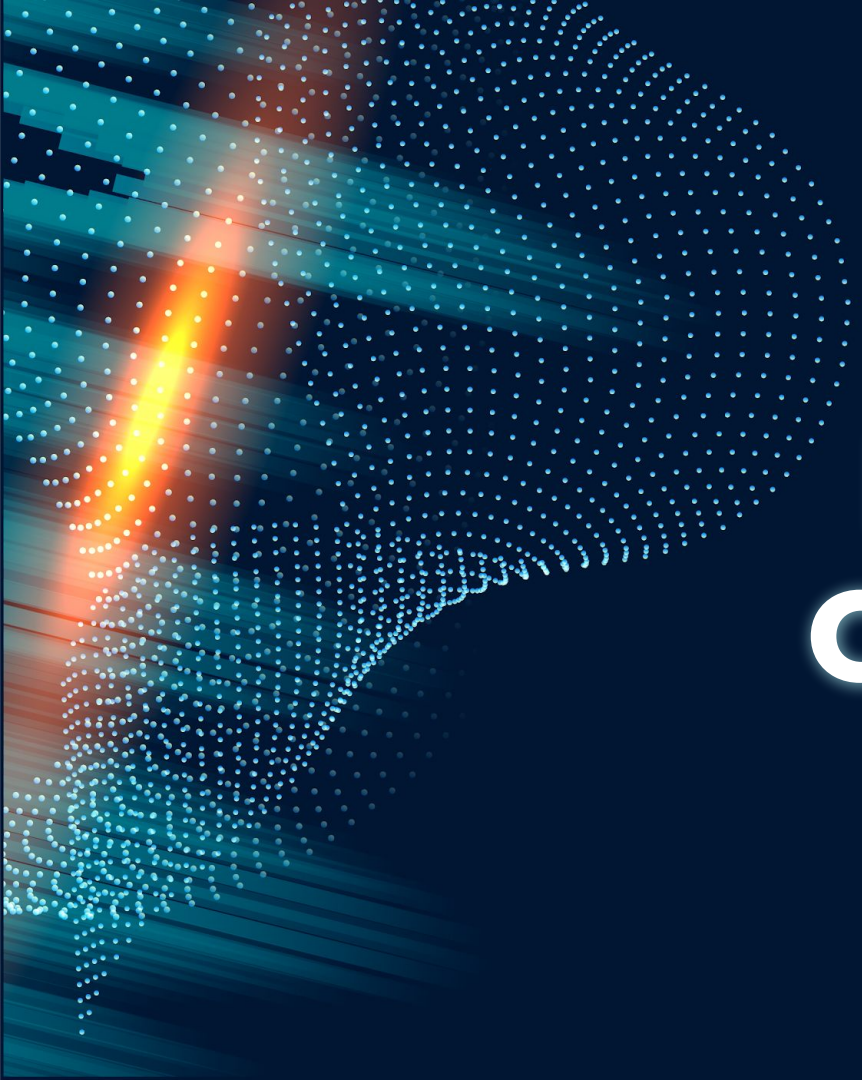
Neck: { FPN [44], PANet [49], Bi-FPN [77], ... }

Head:

Dense Prediction: { RPN [64], YOLO [61, 62, 63], SSD [50], RetinaNet [45], FCOS [78], ... }

Sparse Prediction: { Faster R-CNN [64], R-FCN [9], ... }





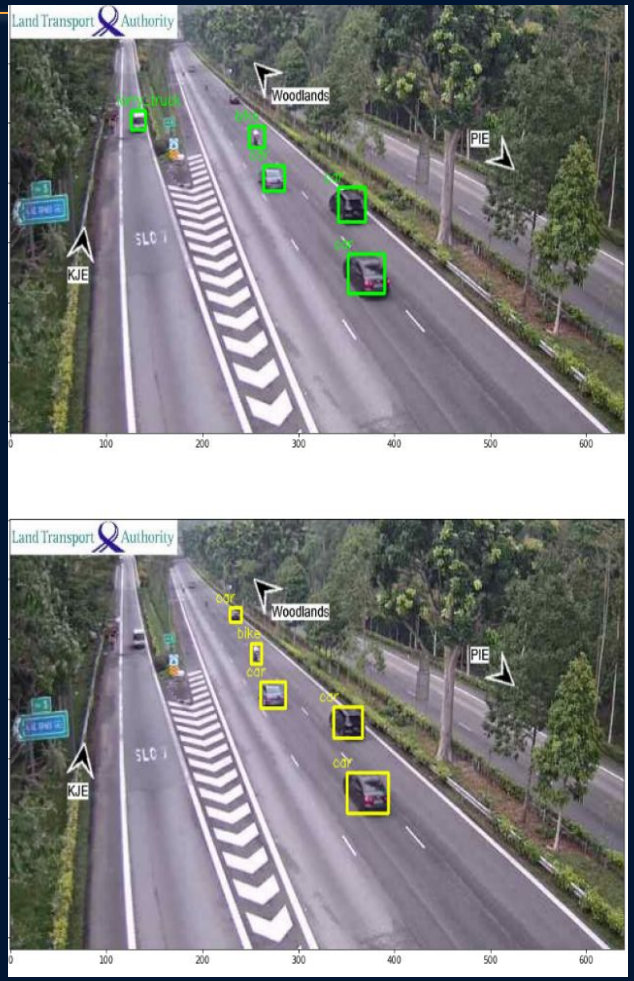
Results & Conclusion

Using Darknet framework

Visual results

How do we measure performance?

- Mean Average Precision (MAP)

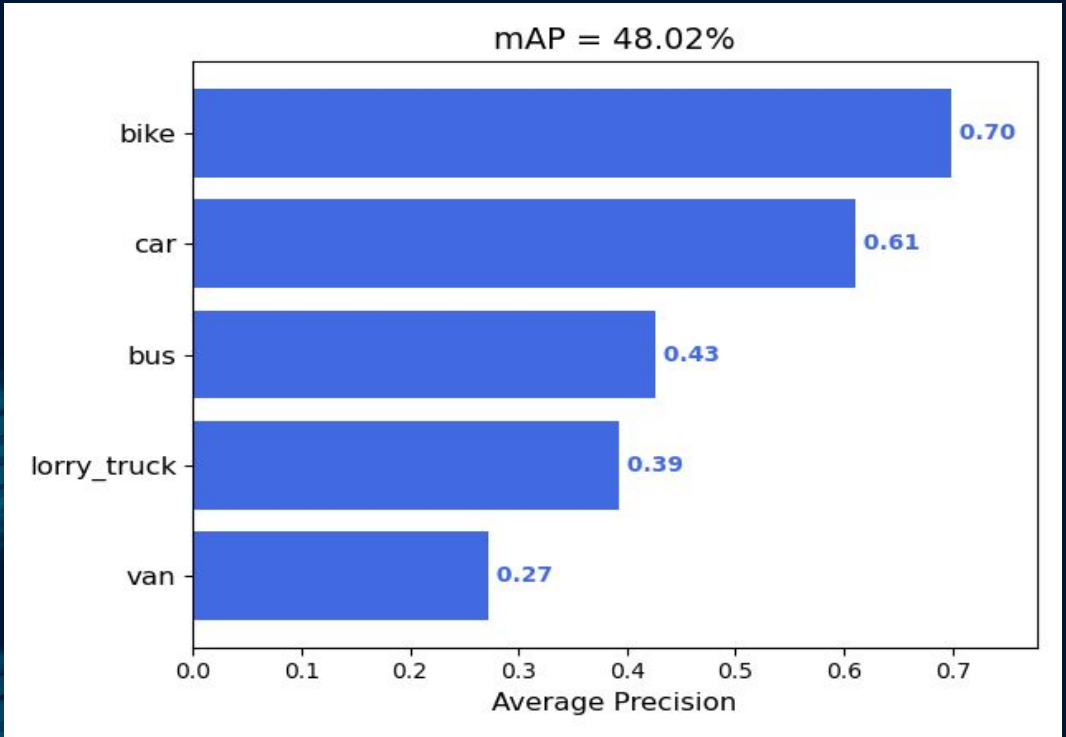


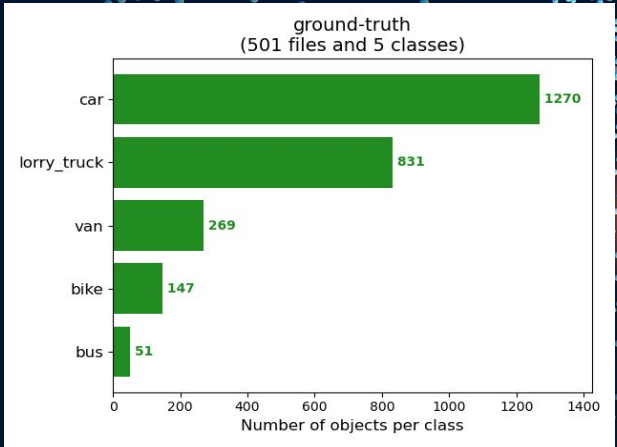
Summary of YOLO models

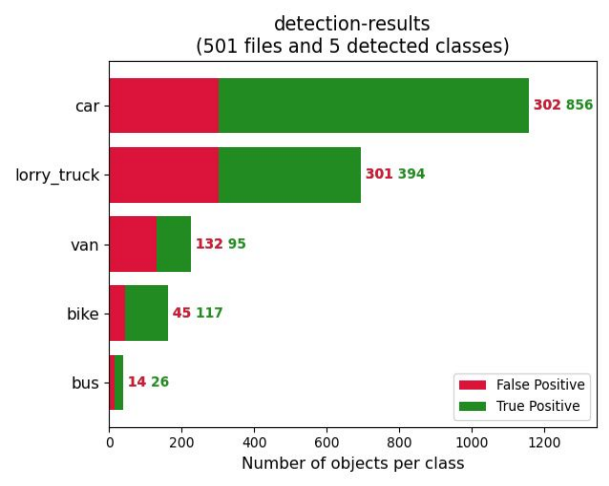
Models	Backbone	Resize dimension	Learning rate	Pre-trained-weights	Validation images detected by model out of 516 images	Mean Average Precision MAP (%)
v4	CSPDarknet53	416 x 416	0.001	No	504 (97.7%)	42.07
v4	CSPDarknet53	416 x 416	0.01	No	481 (93.2%)	43.07
v4	CSPDarknet53	416 x 416	0.001	Yes	501 (97.1%)	48.02
v4	CSPDarknet53	480 x 480	0.001	Yes	489 (94.8%)	48.14
V3 (5 layers)	Darknet53	416 x 416	0.001	Yes	446 (86.4%)	36.11
Scaled YOLOv4	Darknet-CSPDarknet combination	512 x 512	0.001	Yes	494 (95.7%)	43.09

Config: Batches of 64 images , 10000 iterations (375 epochs), 3 channels, mozaic augmentation for YOLOv4

MAP/ True positives and False positive







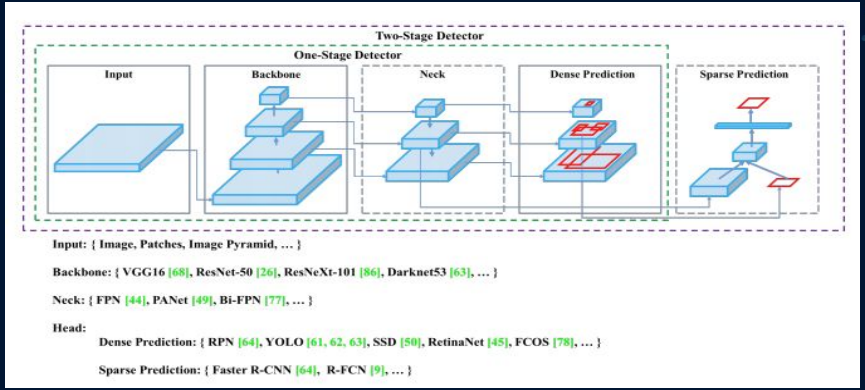
Further Improvements

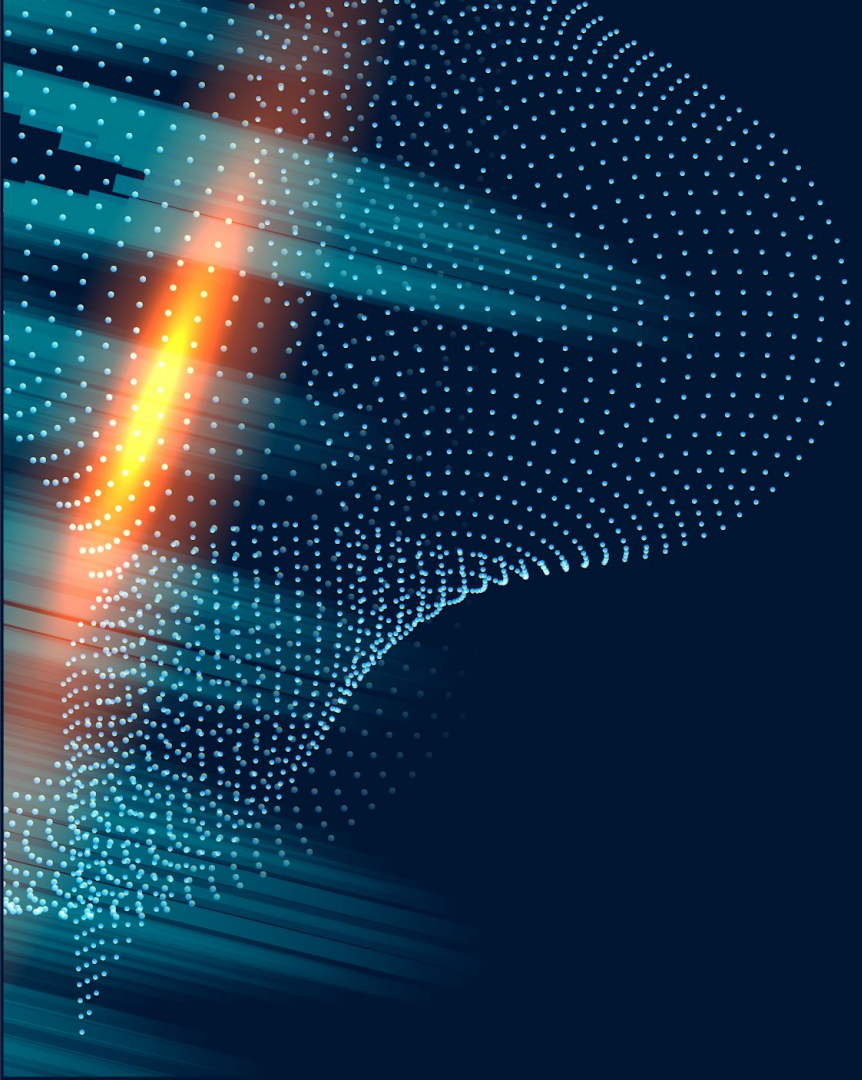
Process wise:

- Annotate more data
- Reduce class or resolution imbalances
- Get traffic data with adverse weather conditions

Model wise:

- Further tuning of parameters
- Backbone replacement



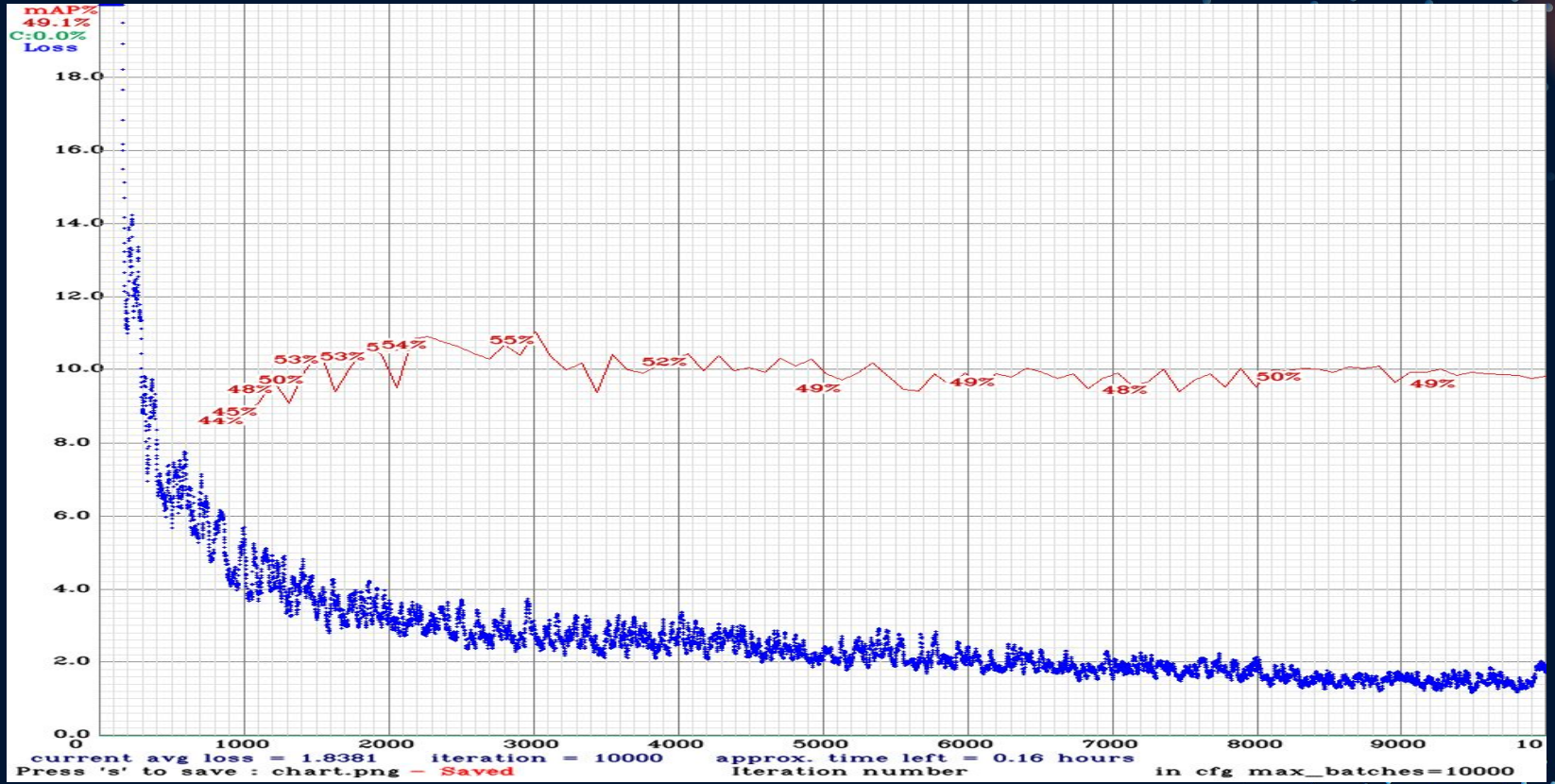


DEMO

Inference with YOLO on a video

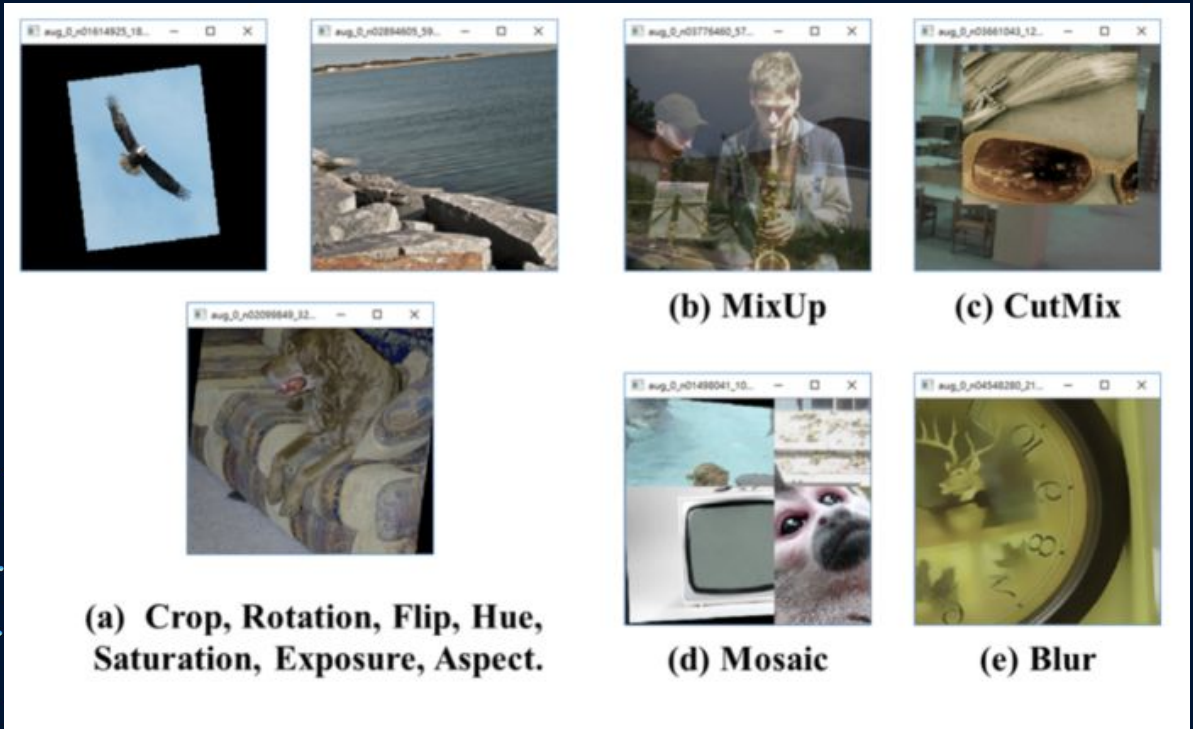
End

Training process

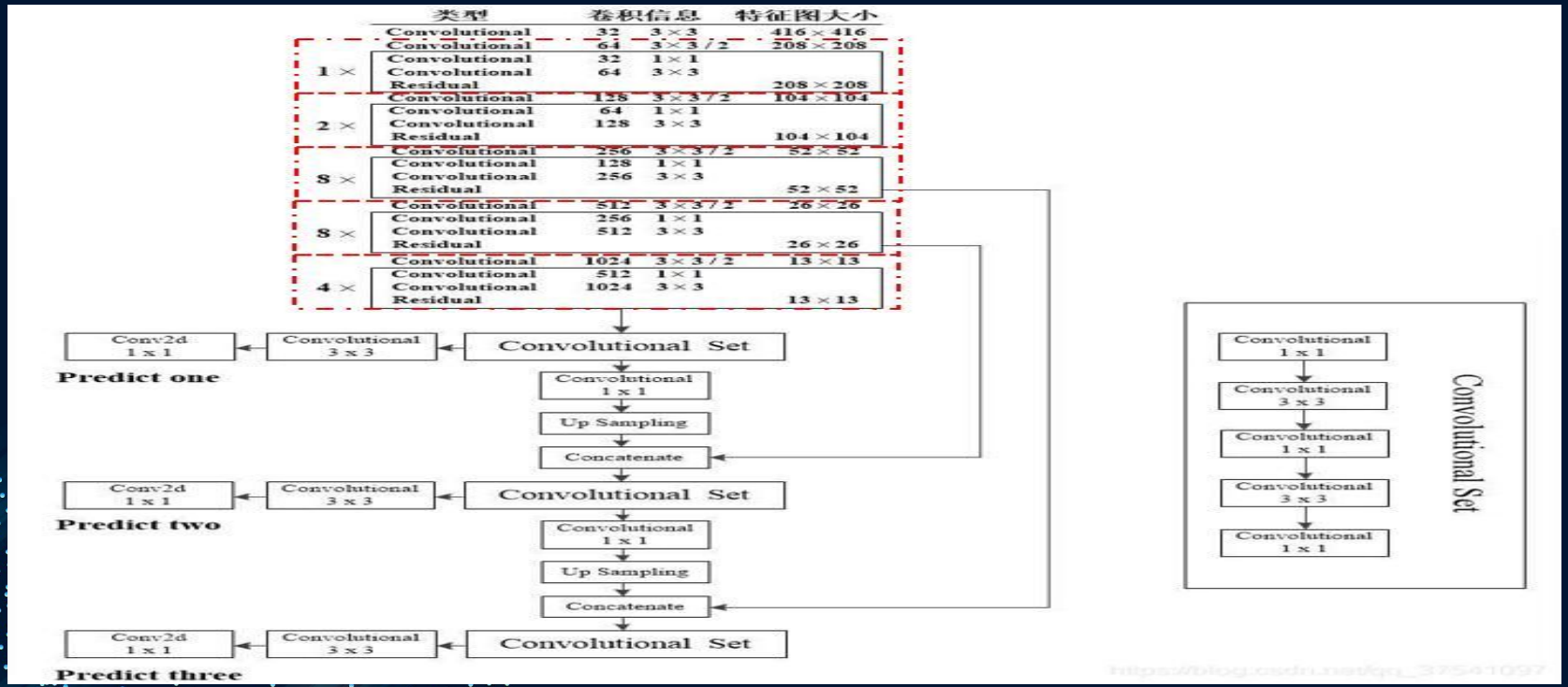


Appendix

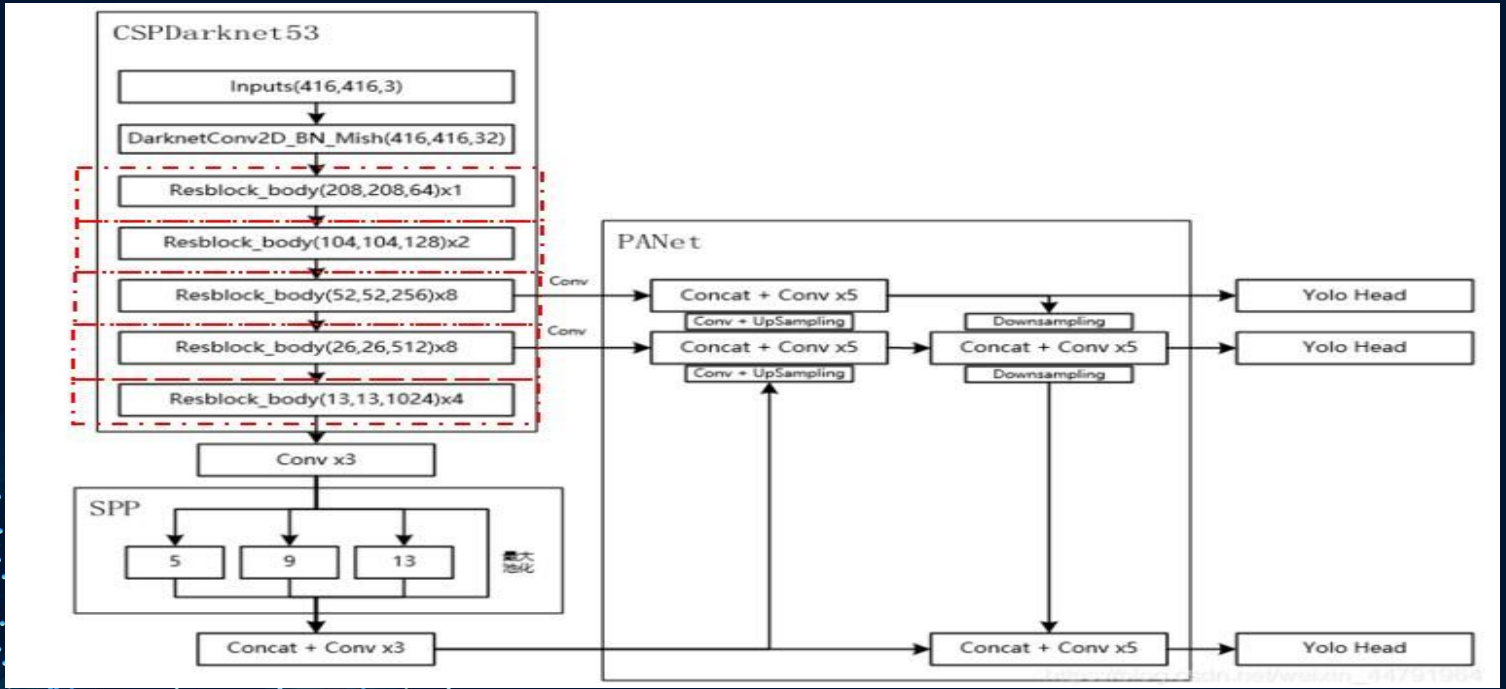
Data augmentation



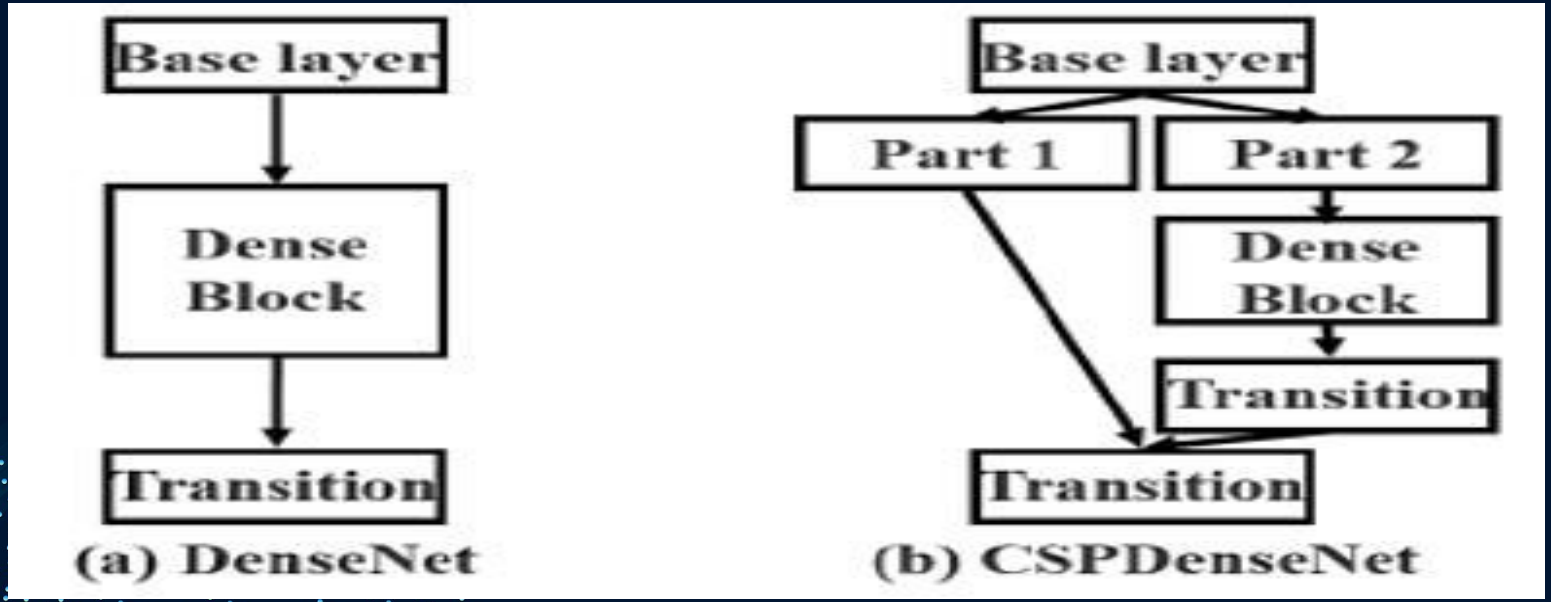
YOLOv3 in detail



YOLOv4 in detail



CSPDensenet vs Densenet



Scaled YOLO

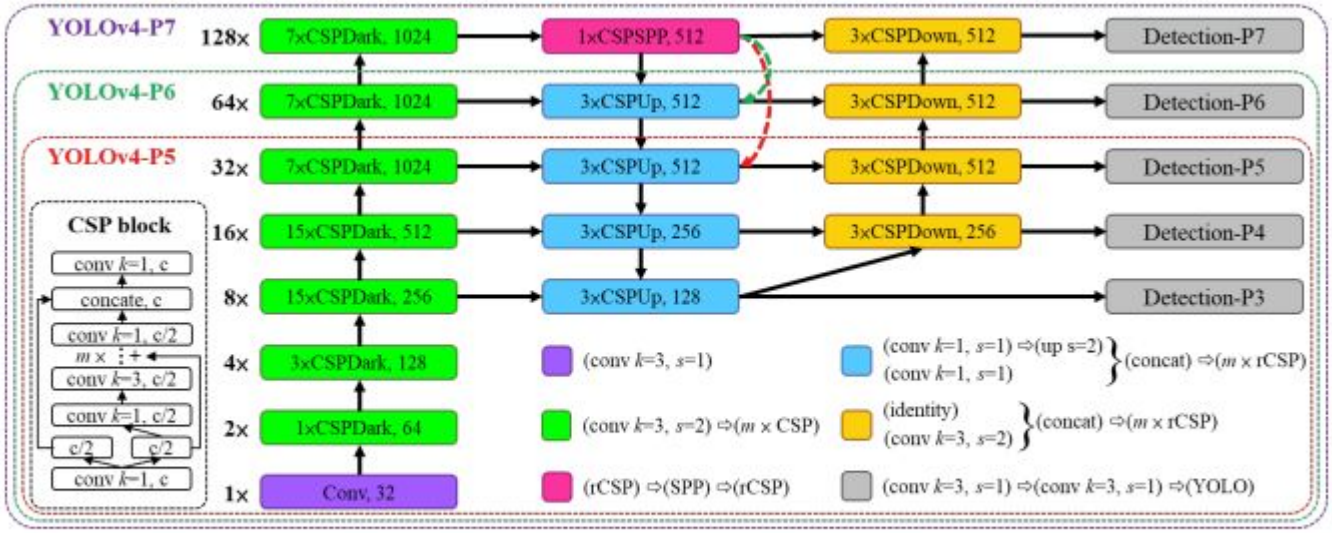


Figure 4: Architecture of YOLOv4-large, including YOLOv4-P5, YOLOv4-P6, and YOLOv4-P7. The dashed arrow means replace the corresponding CSPUp block by CSPSPP block.