CEAI
Center of Excellence in Artificial Intelligence

AGH University of Krakow

AGH

From imaging algorithms to quantum methods Seminar, 12.01.2026

# Multi-Object Tracking and Label Fusion in Automotive Sensor Data

Piotr Kalaczyński, Tomasz Rybotycki, Piotr Gawron

We gratefully acknowledge the funding support by program “Excellence initiative—research university” for the AGH University in Krakow as well as the ARTIQ project: UMO-2021/01/2/ST6/00004 and ARTIQ/0004/2021.

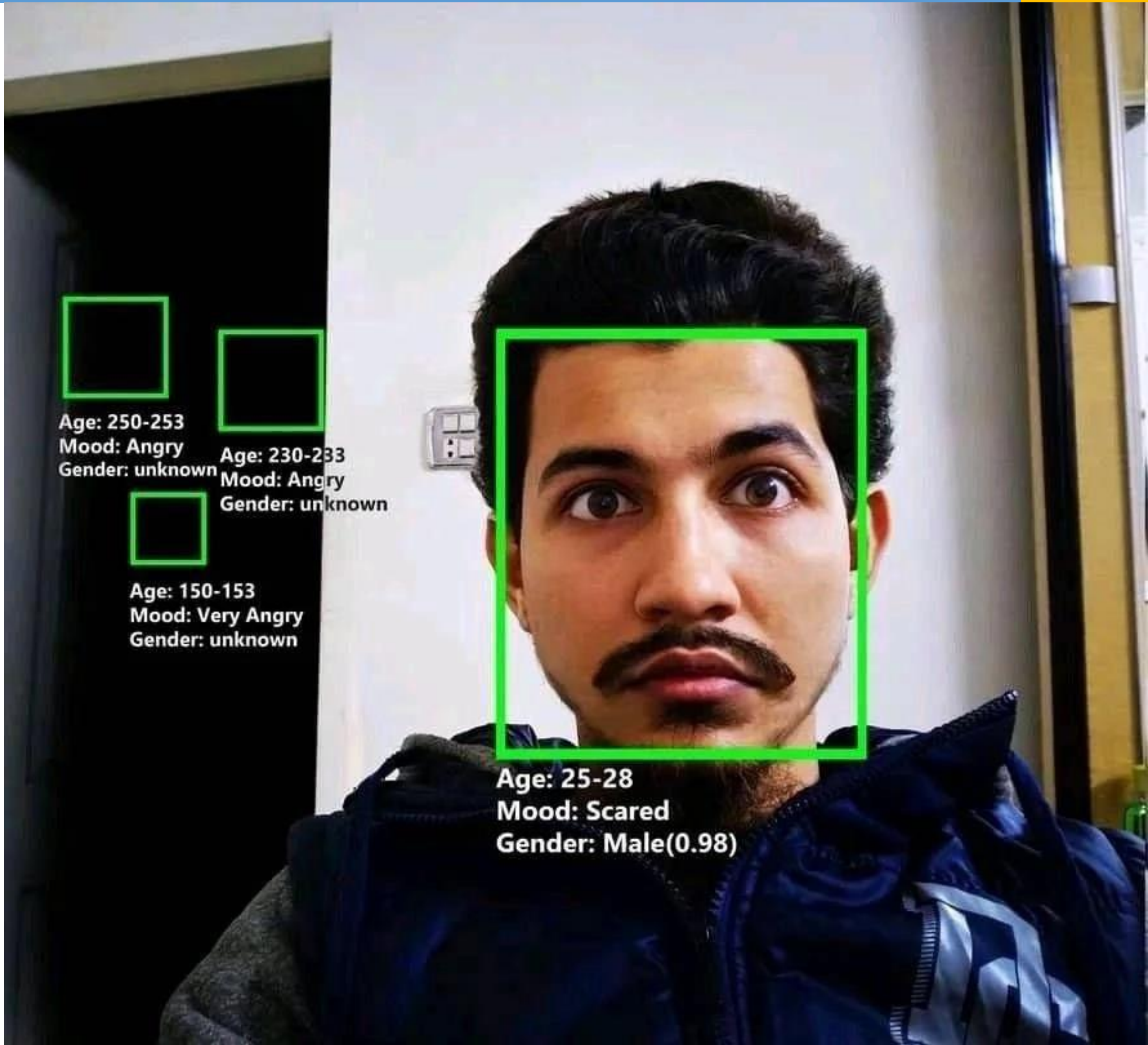
# Outline



Introduction
Data used
Object detection & tracking
Label fusion
Summary

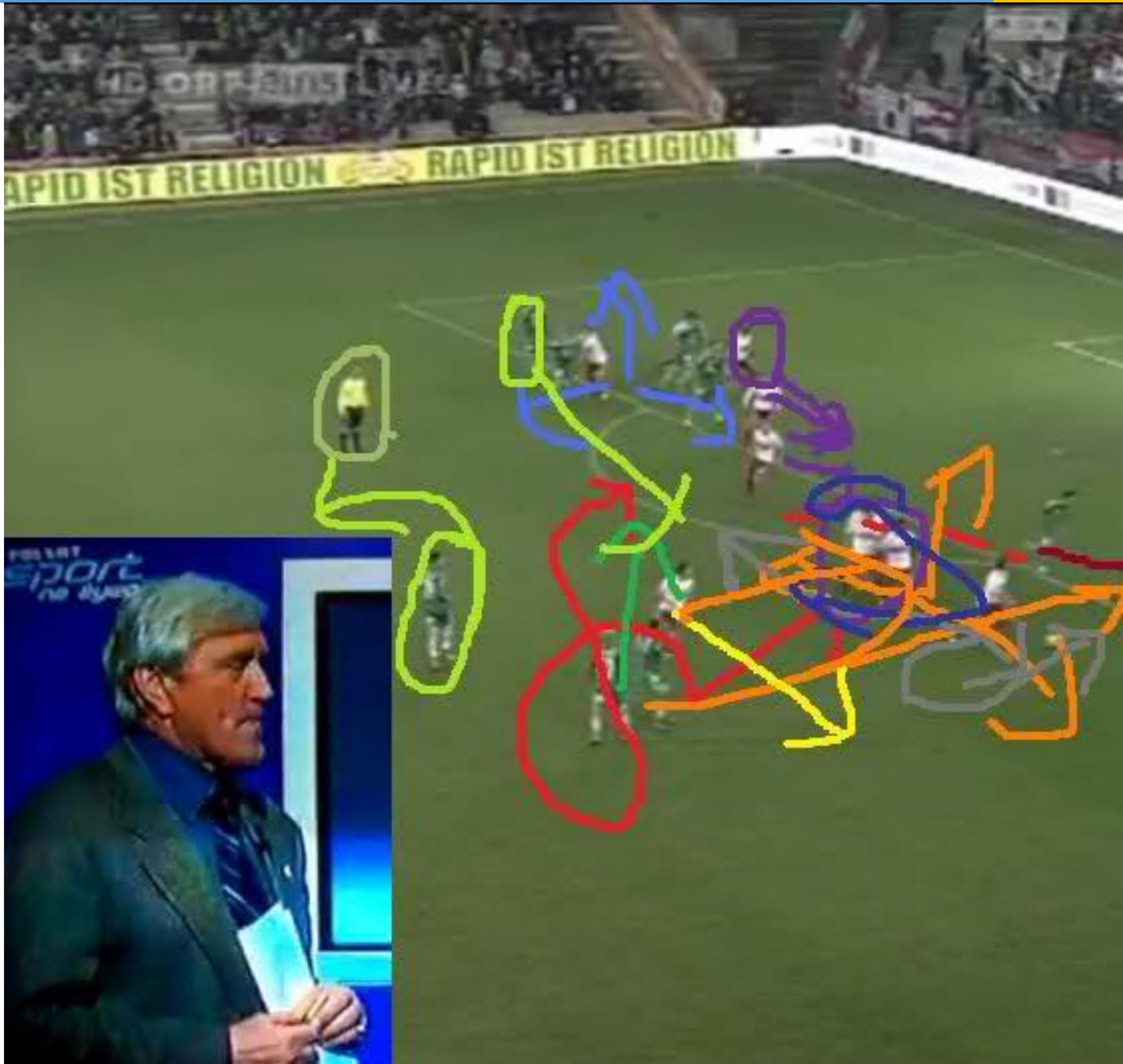
Multi-object tracking (MOT):

- identifying objects in video frames



Multi-object tracking (MOT):

- ❖ identifying objects in video frames
- ❖ maintaining a unique ID for each detected object across video frames.

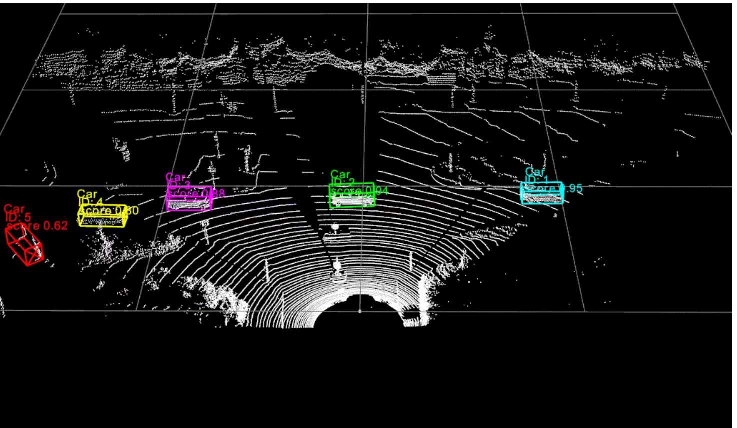


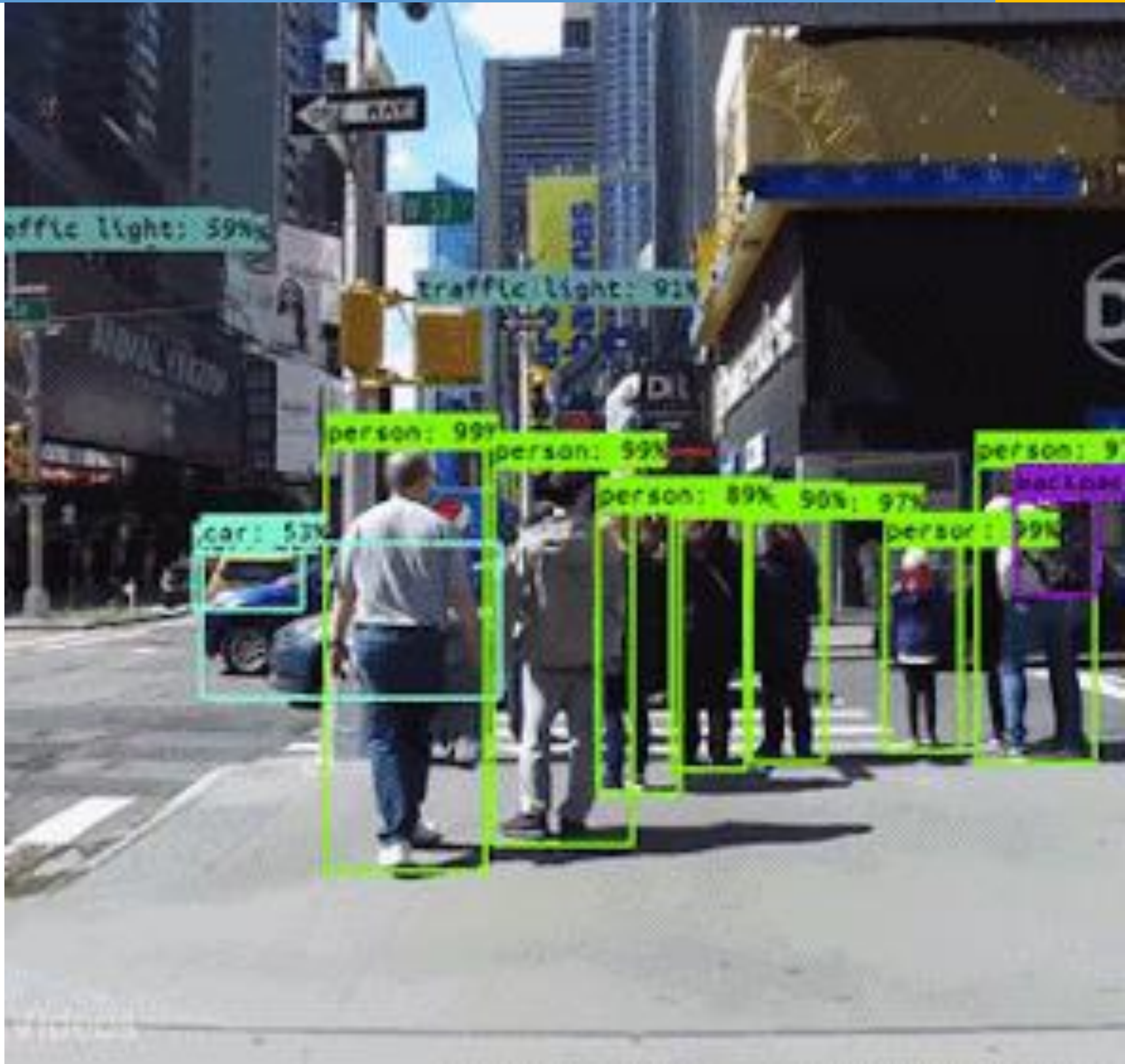
## Multi-object tracking (MOT):

- ❖ identifying objects in video frames
- ❖ maintaining a unique ID for each detected object across video frames.

## Applications:

- ❖ video surveilance
- ❖ sports analytics
- ❖ robotics
- ❖ retail analytics
- ❖ autonomous driving ← this talk





# Outline



Introduction

Data used

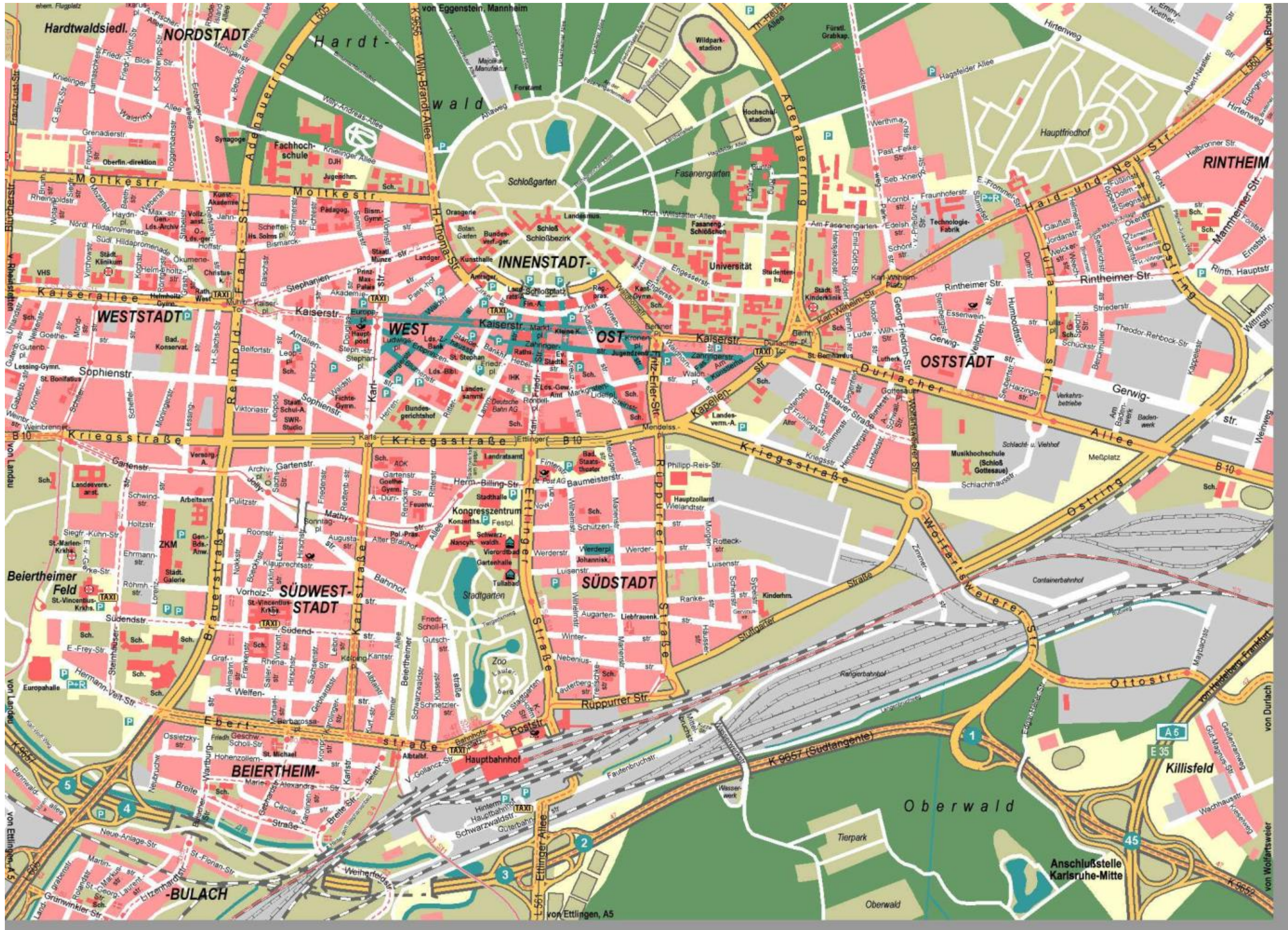
Object detection & tracking

Label fusion

Summary

# The KITTI Vision Benchmark Suite:

- ❖ Project of Karlsruhe Institute of Technology (KIT) & Toyota Technological Institute at Chicago (TTIC)
- ❖ Annotated automotive datasets recorded in and around Karlsruhe, Germany
- ❖ Well-established benchmarks for:
    - Stereo
    - Scene flow
    - Odometry
    - Image depth completion and prediction
    - Object detection: 2D and 3D
    - Multi-object tracking
    - Road/Lane Detection
    - Semantic segmentation
- ❖ Widely used by the computer vision community

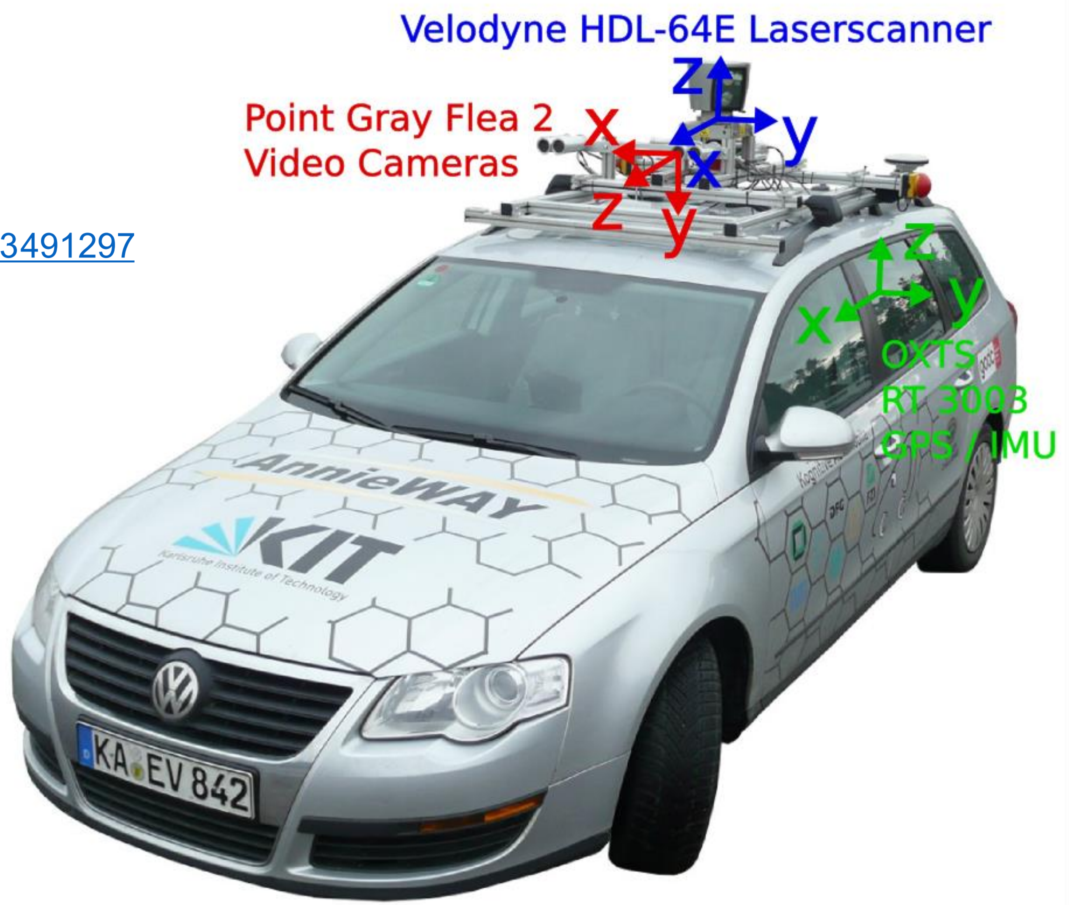



Karlsruhe Institute of Technology

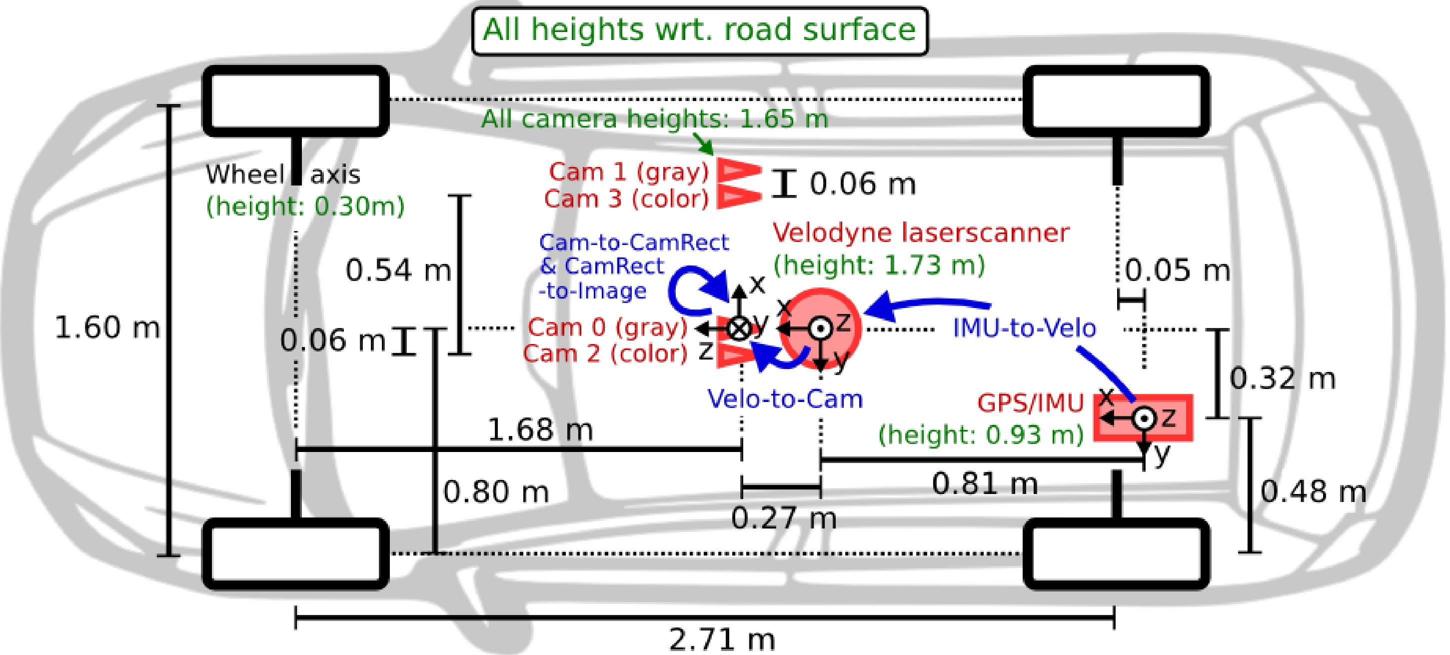
TOYOTA TECHNOLOGICAL INSTITUTE AT CHICAGO

The KITTI dataset has data from:

- 2 grayscale cameras: Point Grey Flea 2 (FL2-14S3M-C), 1.4Mpix each
- 2 color cameras: Point Grey Flea 2 (FL2-14S3C-C), 1.4Mpix each
- 1 lidar: Velodyne HDL-64E (laser scanner)
- 1 GPS/IMU: OXTS RT 3003 (used indirectly, for calibration)




A. Geiger et al., *Vision meets robotics: The KITTI dataset*, https://doi.org/10.1177/0278364913491297



# Outline

- Introduction
- Data used
- Object detection & tracking
- Label fusion
- Summary

# Outline

Introduction

Data used

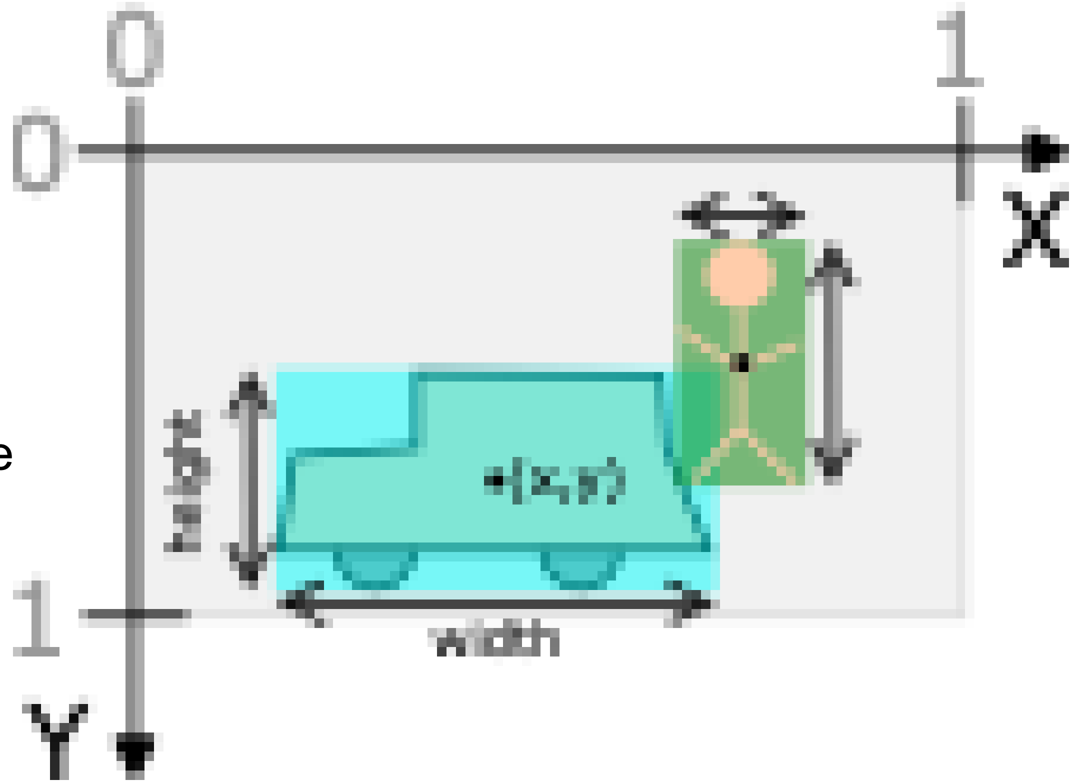
Object detection & tracking

| detection | tracking |
|---|---|

Label fusion

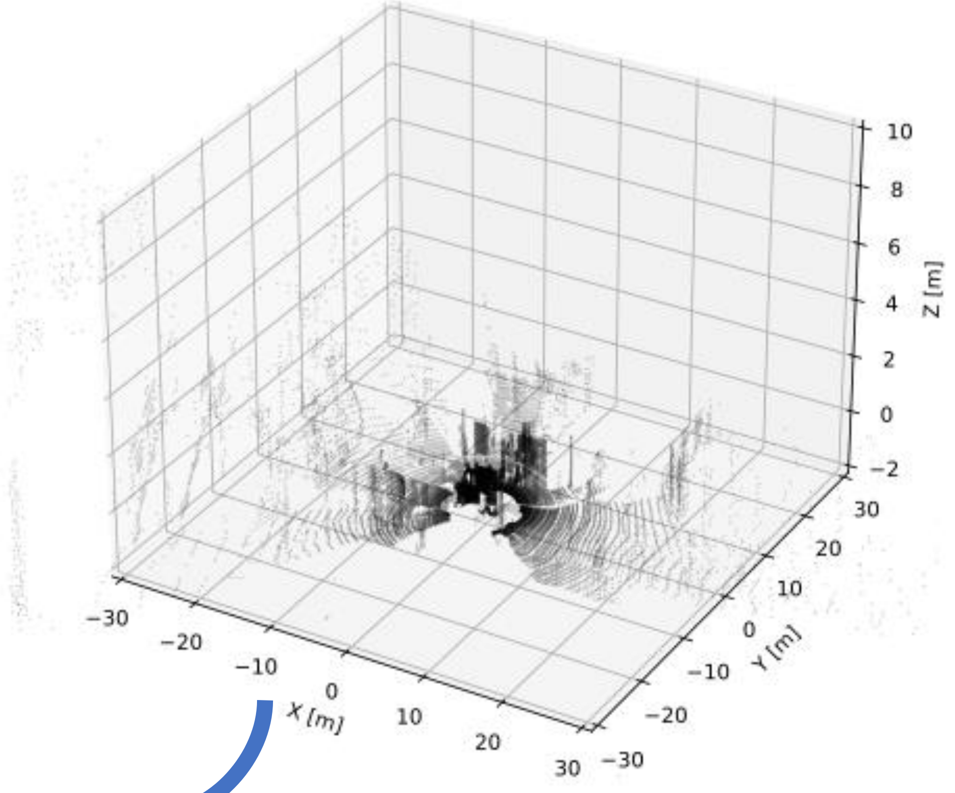
Summary

Our code:
github.com/AGH-CEAI/automotive-tracking/



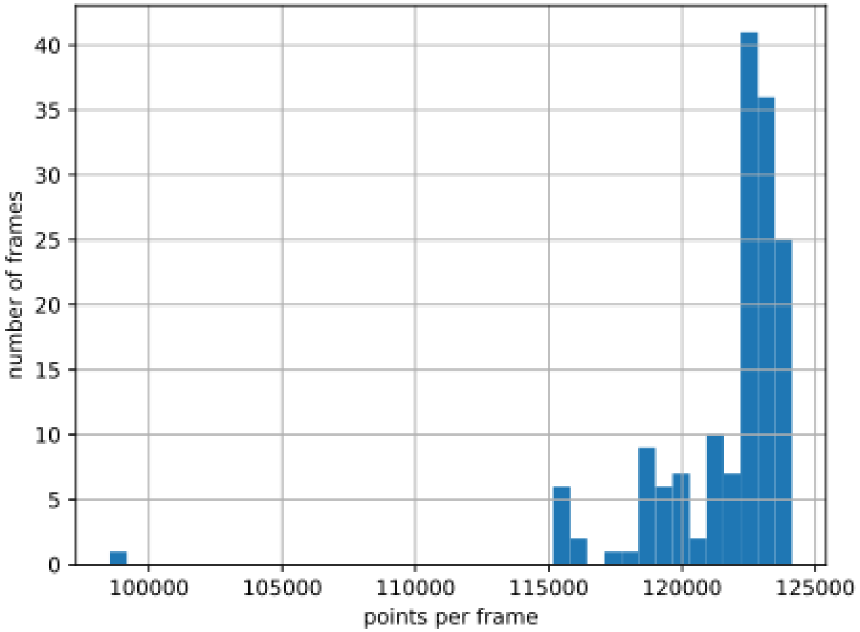

# Object detection:

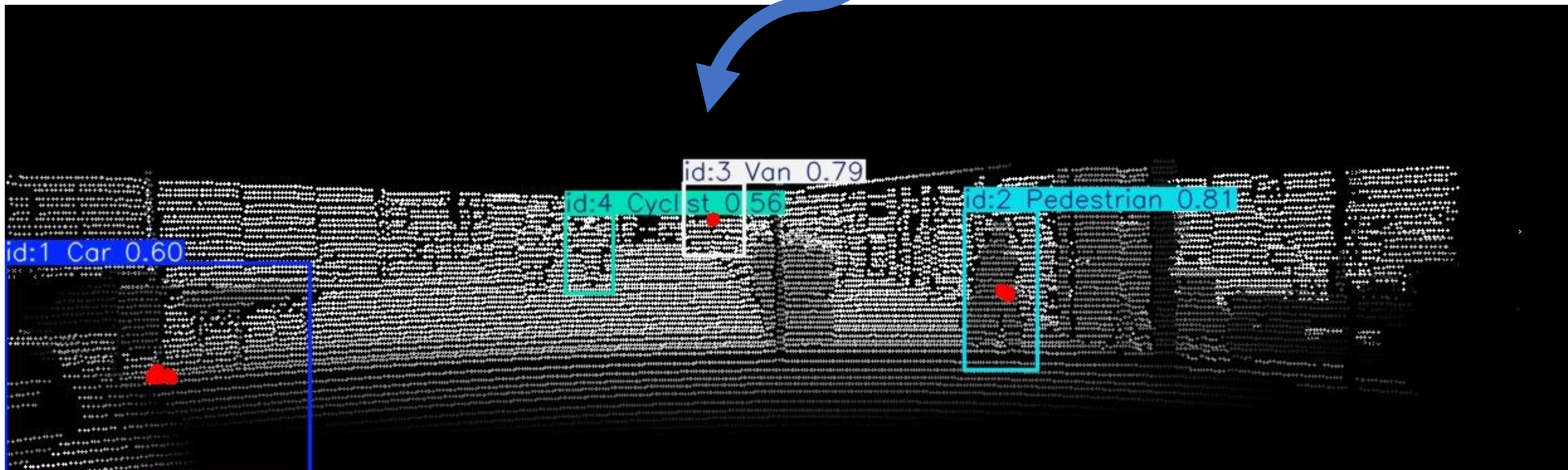
- In our context: 2D bounding boxes: (x, y, height, width)
- Each box gets: id, detected class, classification confidence score
- Done individually for each video frame
- Separately for camera and lidar
- Can be done out-of-the-box with pre-trained models
- Better results after training on KITTI itself
- Used model: You Only Look Once (YOLO) v8



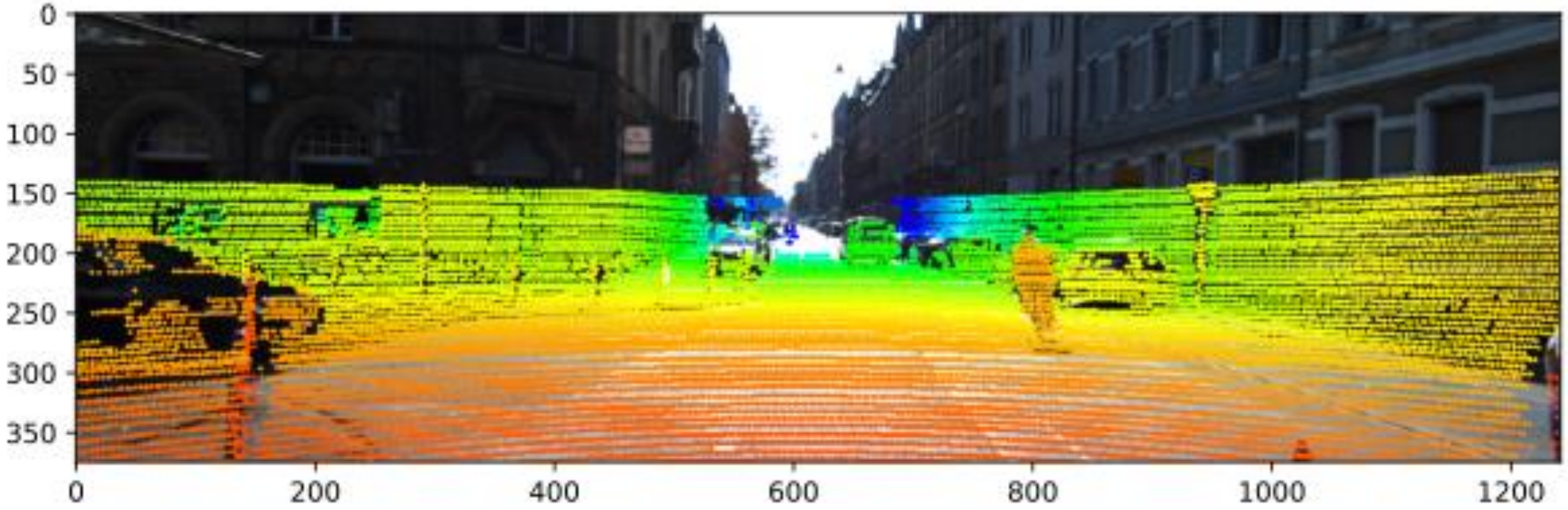

## Data preprocessing:

- Data: tracking dataset from KITTI
- Train & test split:
  - 17:4
  - by scenes, not by frames
    (otherwise tracking would be meaningless)
- KITTI labels → ultralytics format (we use their YOLO model)
- Lidar data: pointcloud → 2D projection (data shape & FoV coverage)

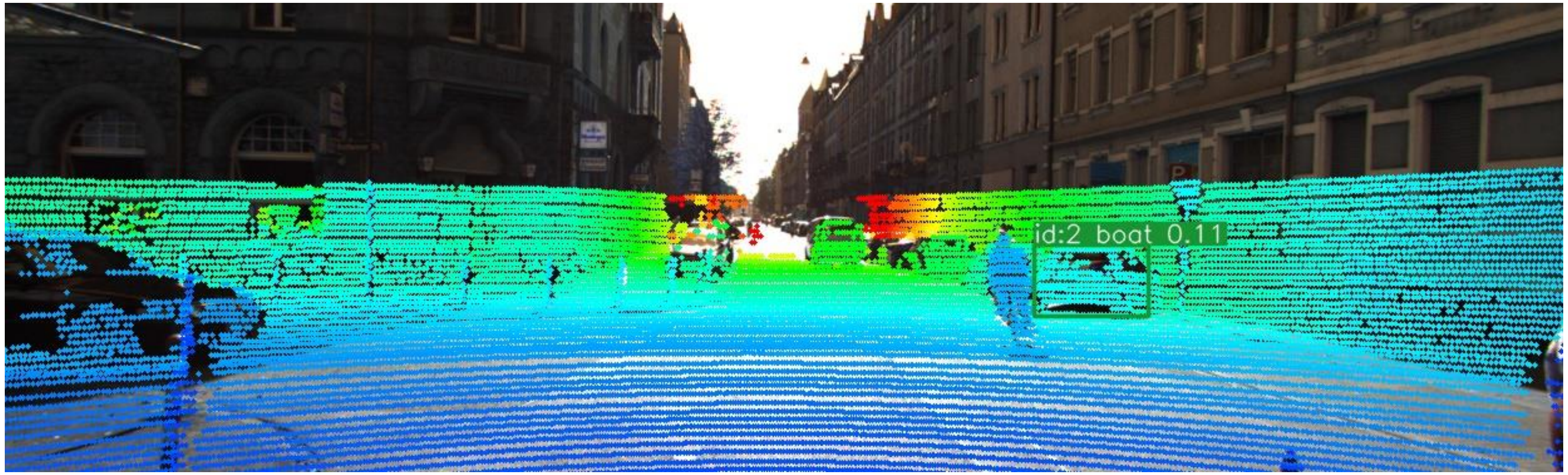


Why not just merge camera and lidar data into a single image like that?



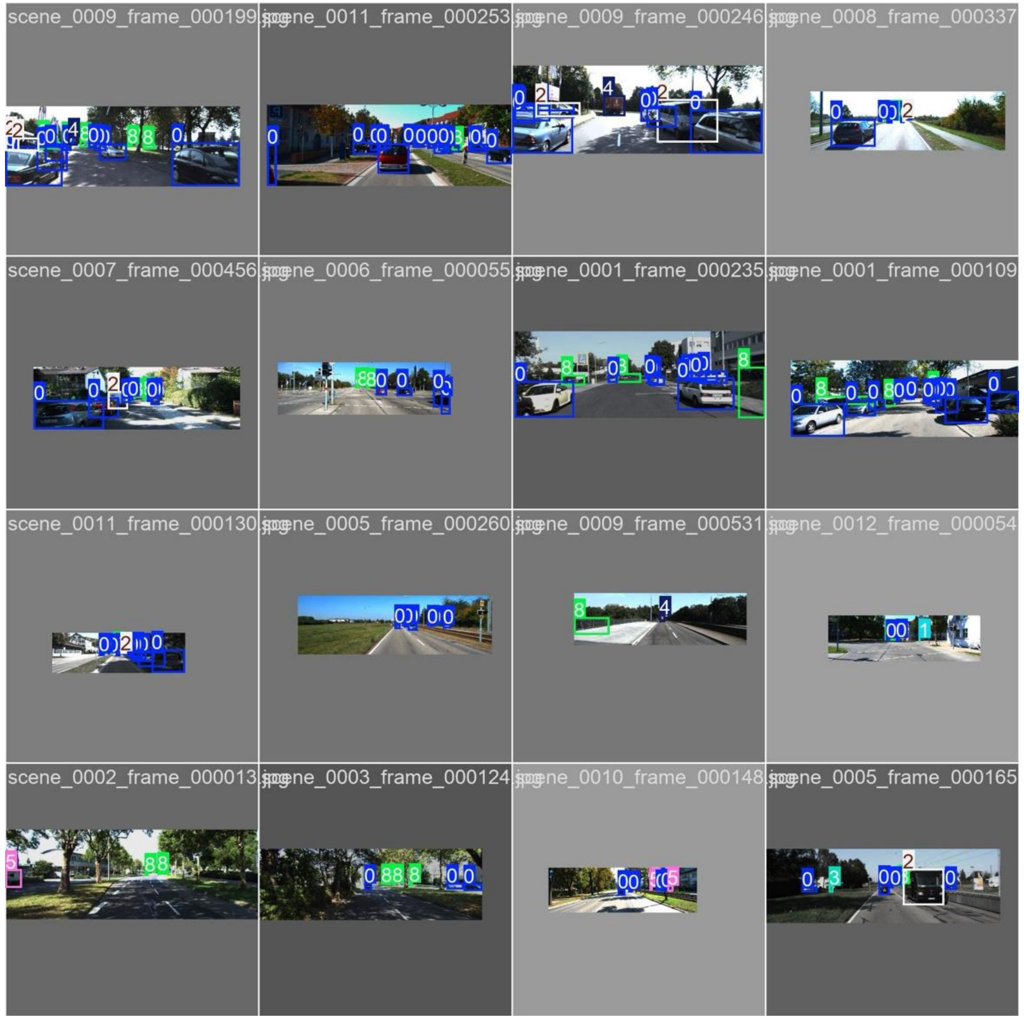

Well, you can but the performance is terrible:

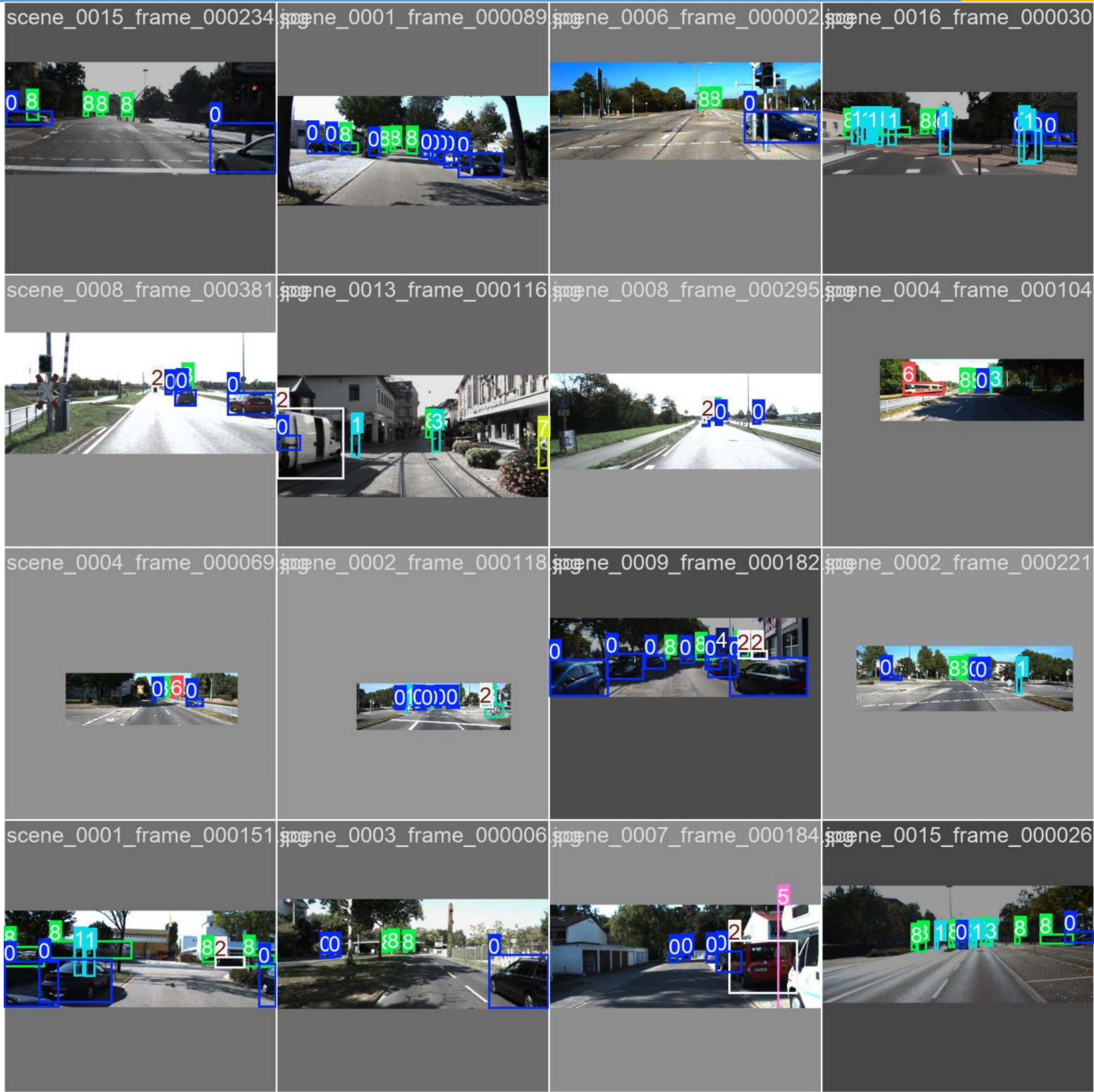


# Object detection: YOLO training (camera images)

## What the acutal training batches look like:

- Labels are encoded to ints
- Frames are shrinked, enlarged and moved for a more robust detection



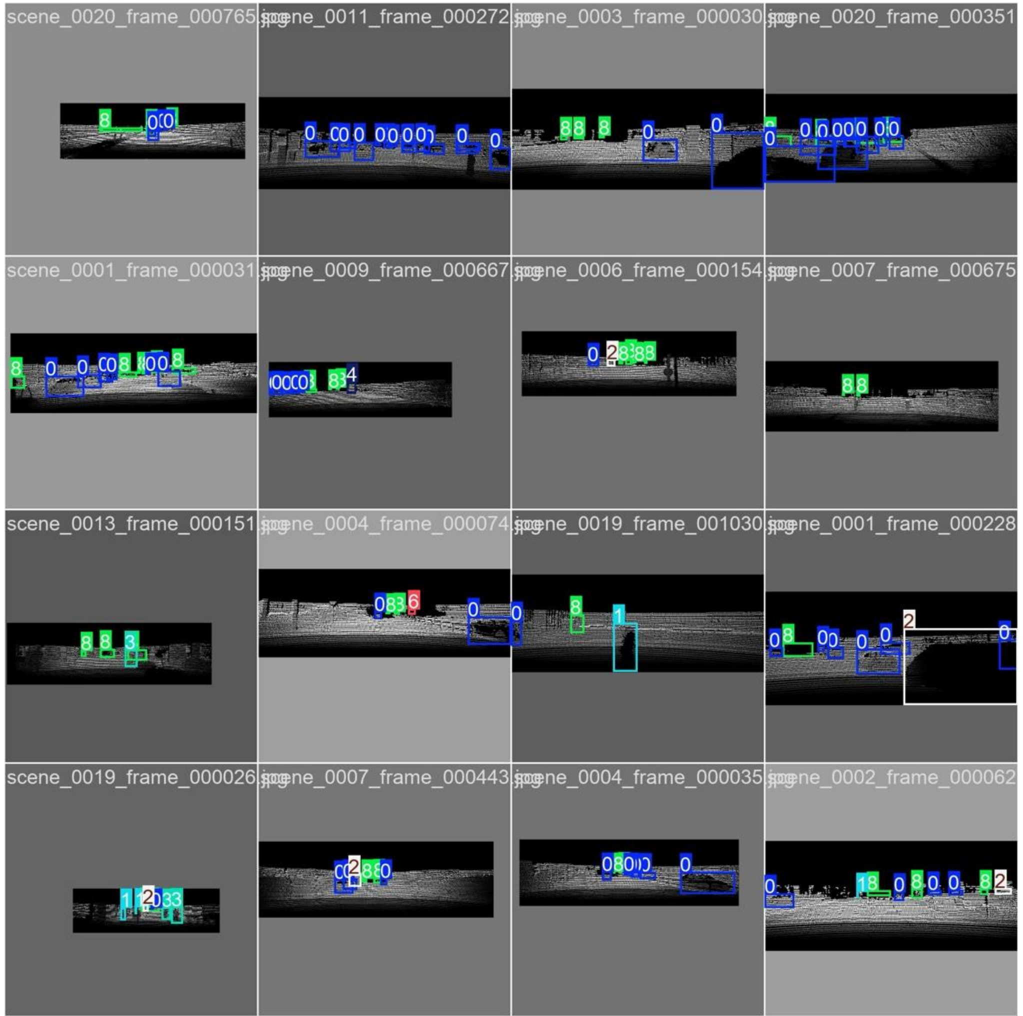



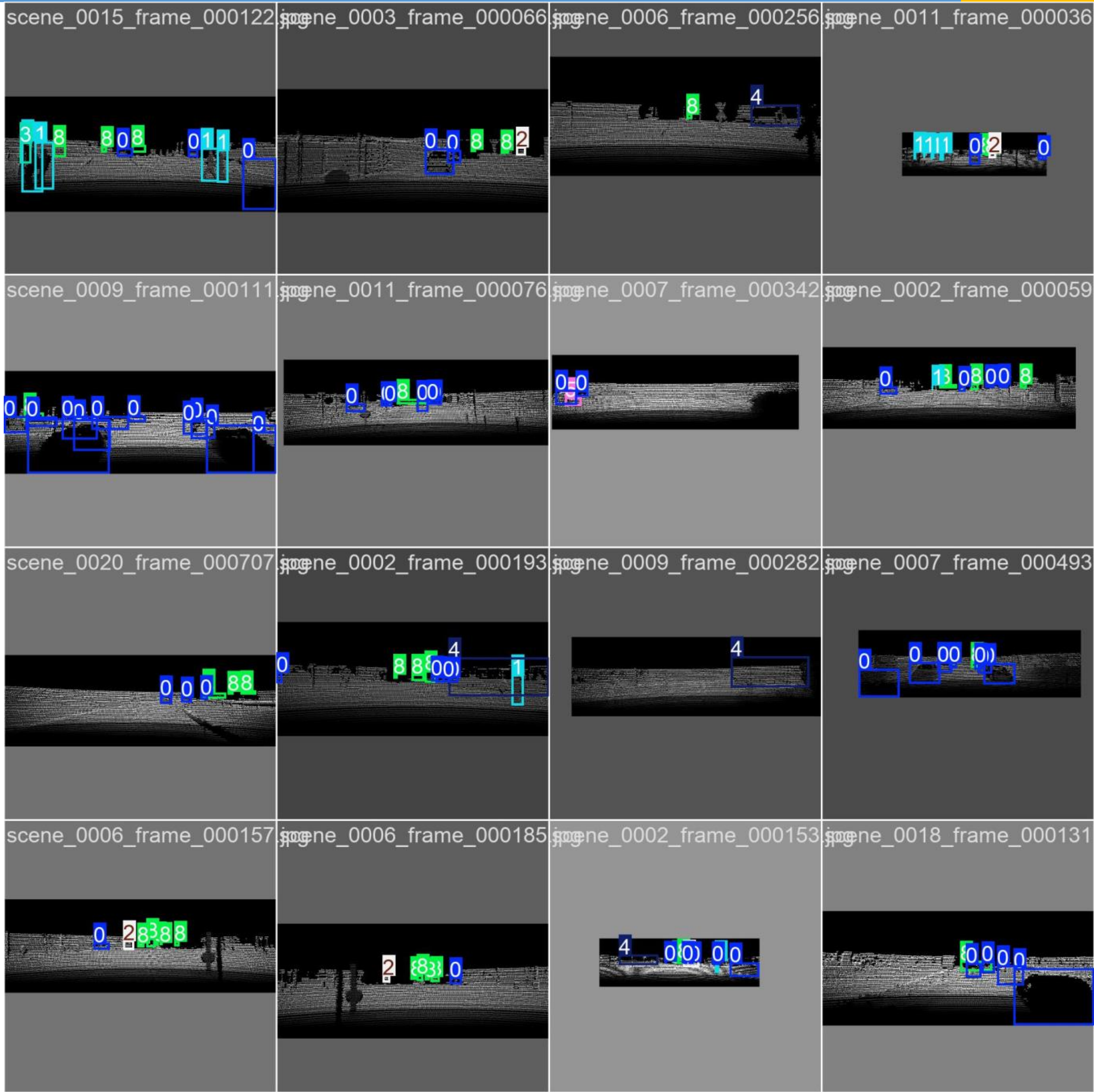
# Object detection: YOLO training (lidar projection images)



## What the acutal training batches look like:

- Labels are encoded to ints
- Frames are shrinked, enlarged and moved for a more robust detection



# Object detection: YOLO validation (camera images)



1st validation batch

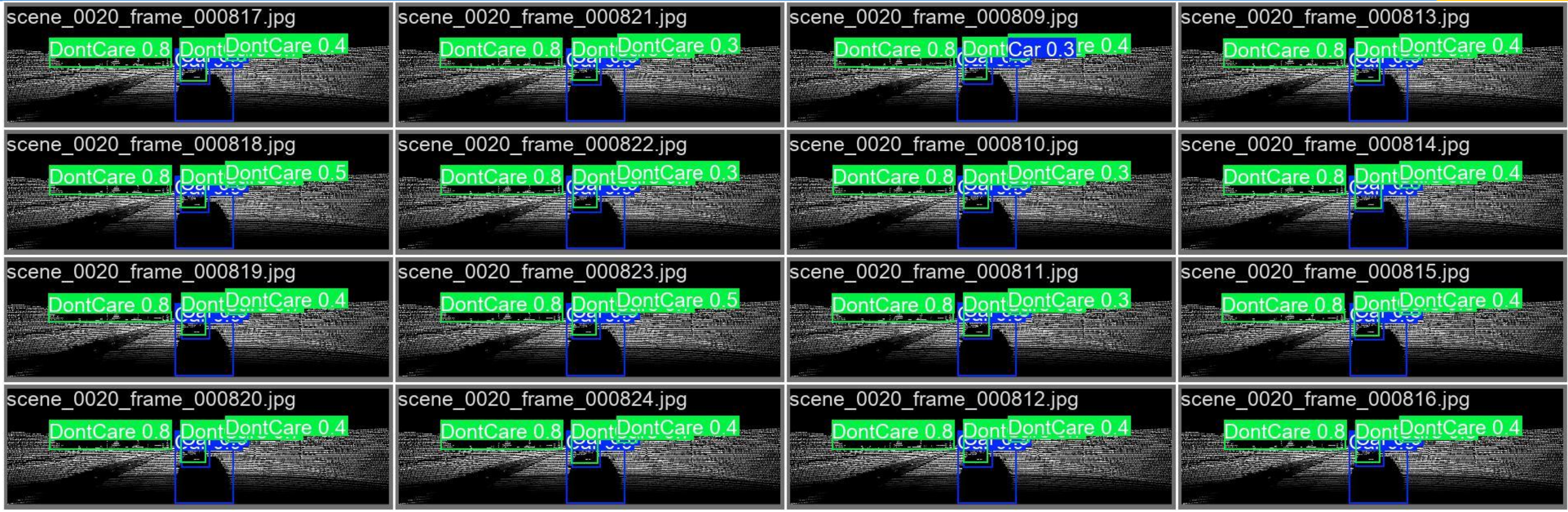
Predicted labels:

Annotated labels:
(~true)

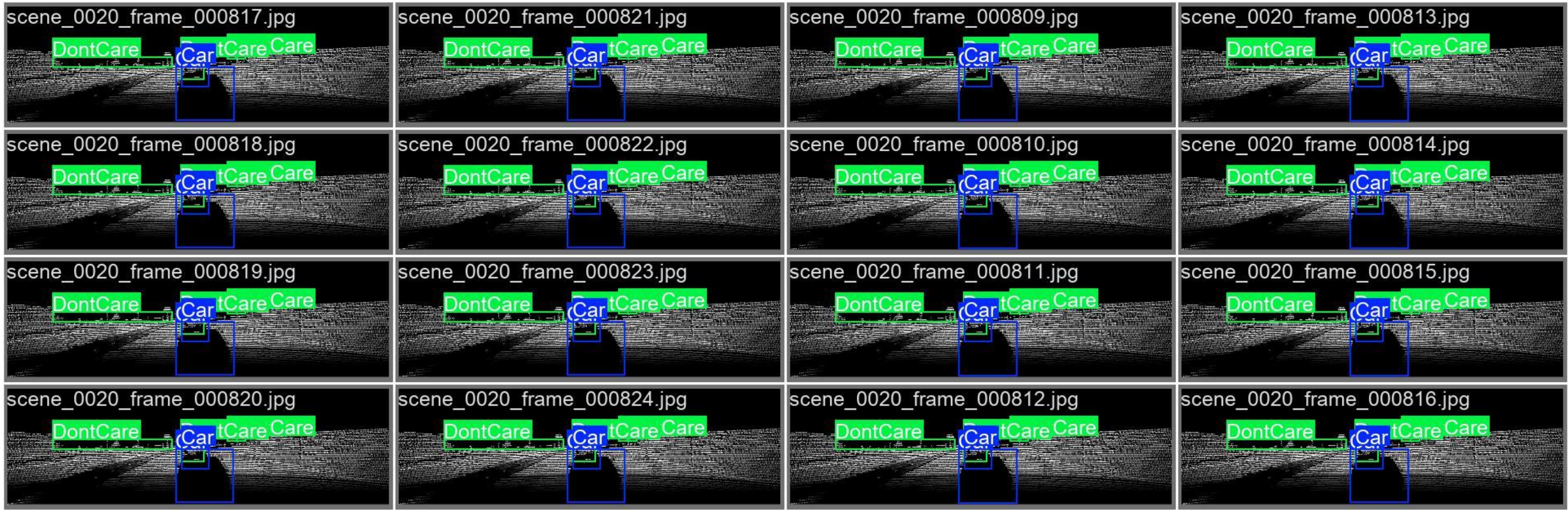
# Object detection: YOLO validation (lidar images)

1st validation batch

Predicted labels:



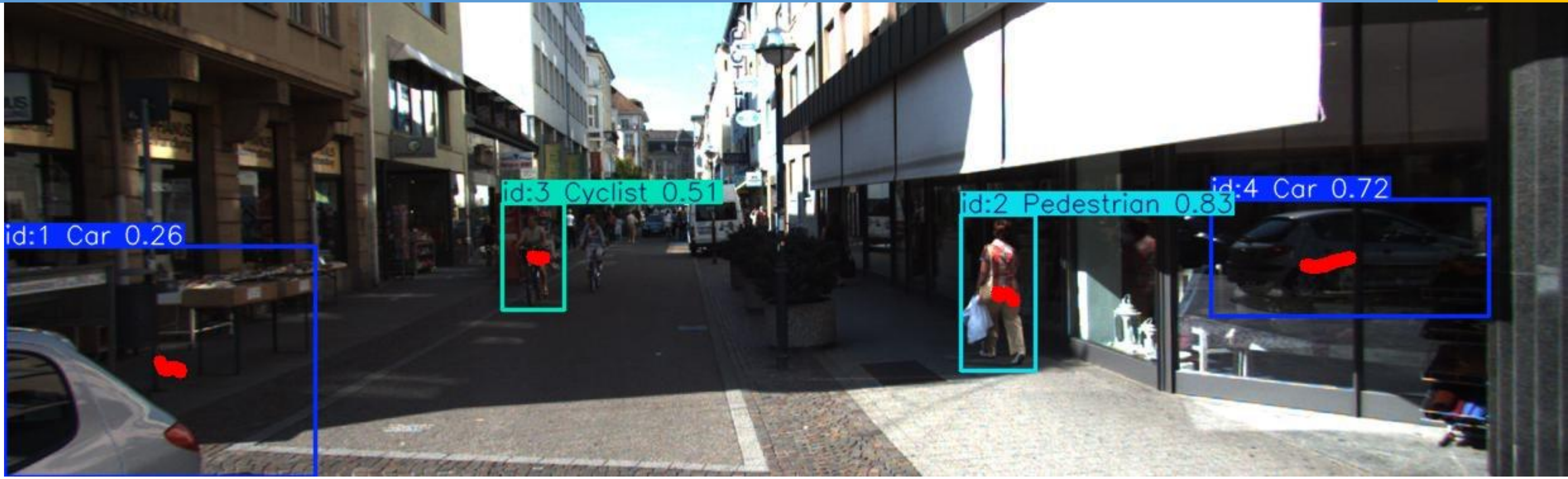

Annotated labels:
(~true)



# Object detection: YOLO validation (camera & lidar on same frame)

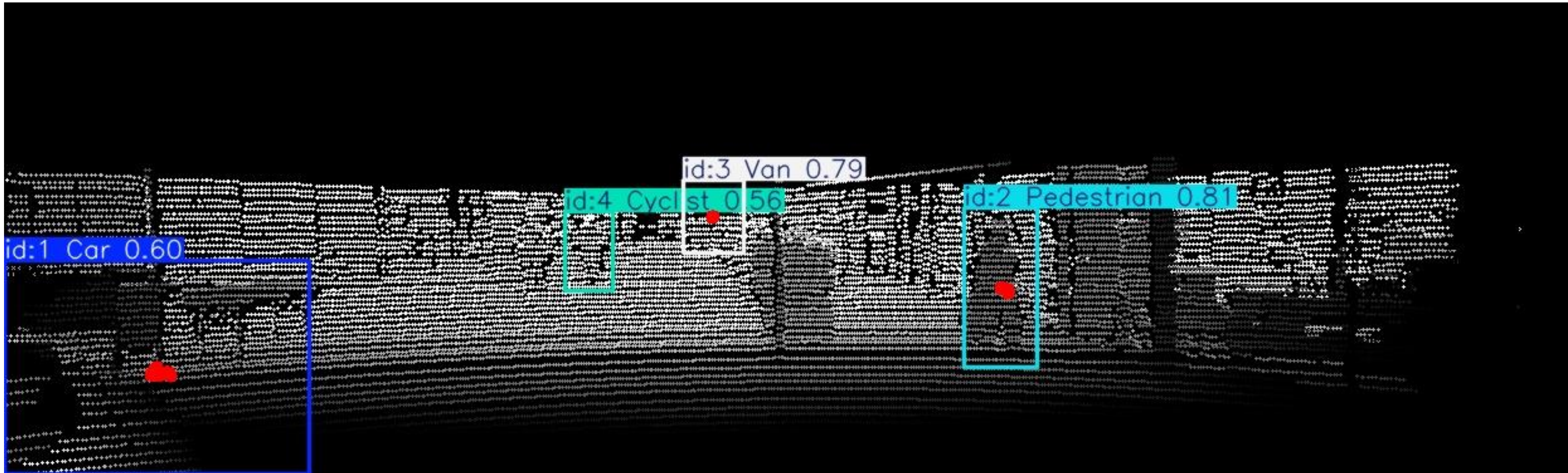


Camera:
- Slightly better confidence for pedestrian, could be just by chance



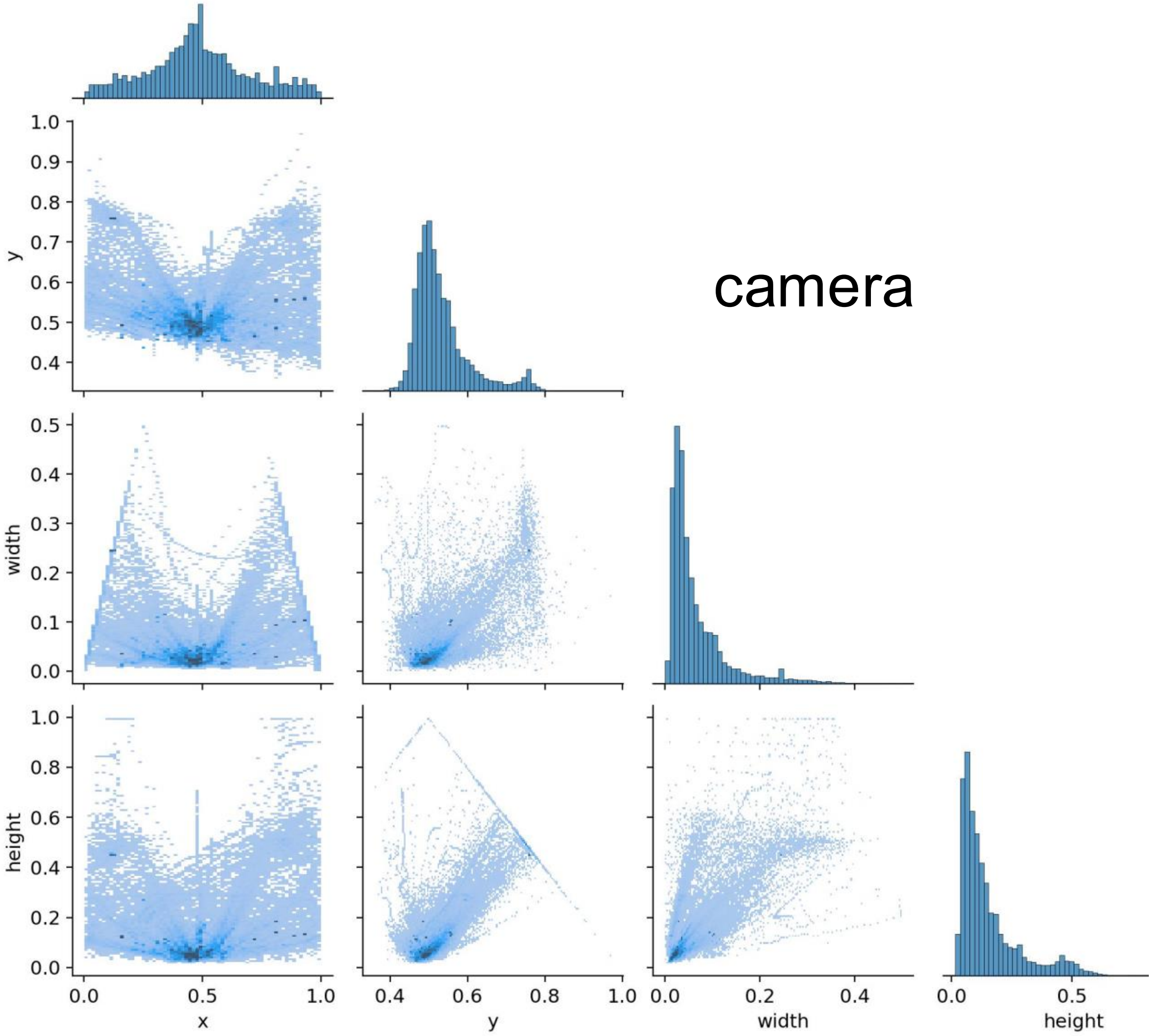

Those are with tracking switched on (in red), but let's pretend it's not there ;)

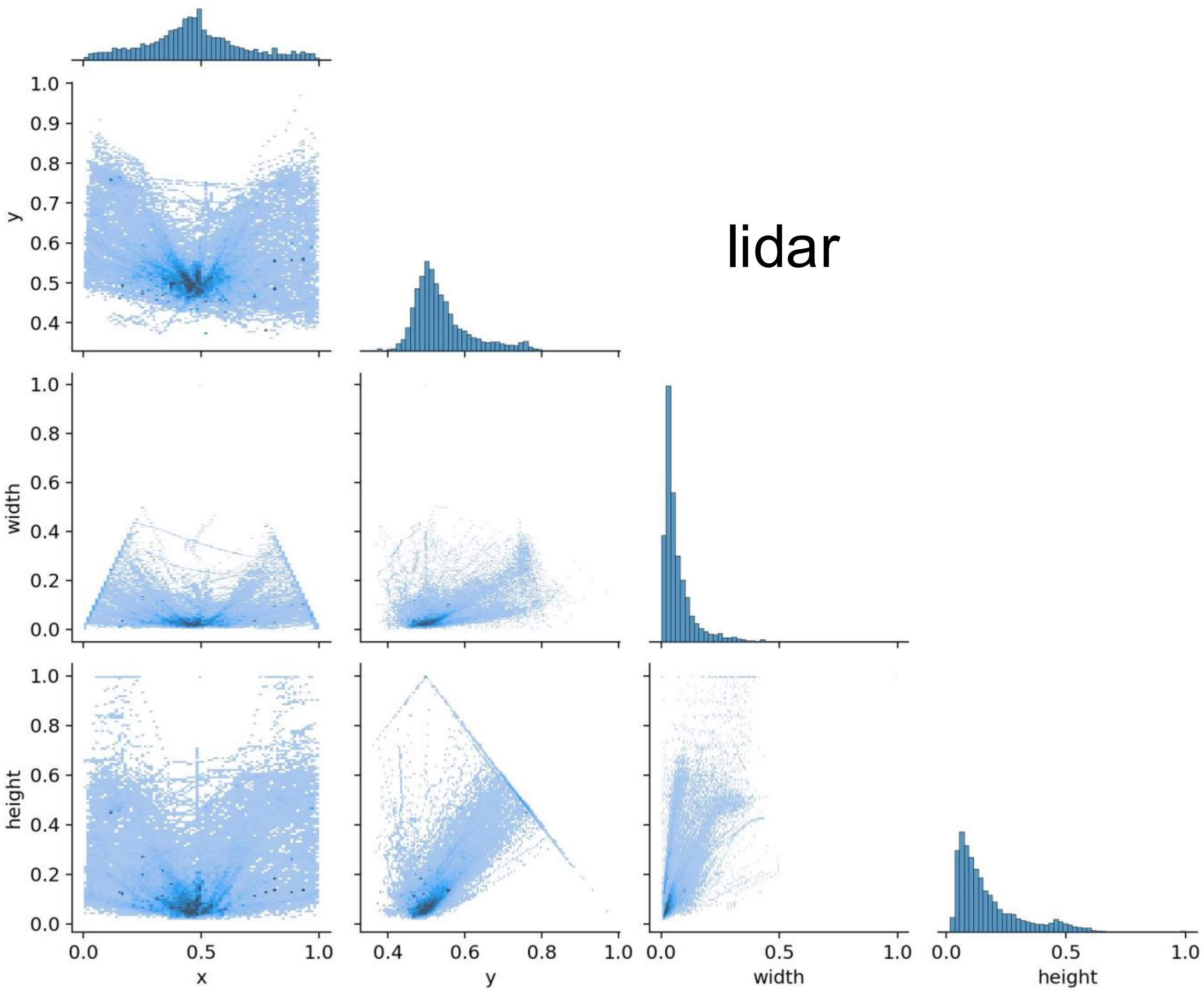
Combined even better?




Lidar:
- Seems overall better
- not fooled by reflection

Rather consistent distributions

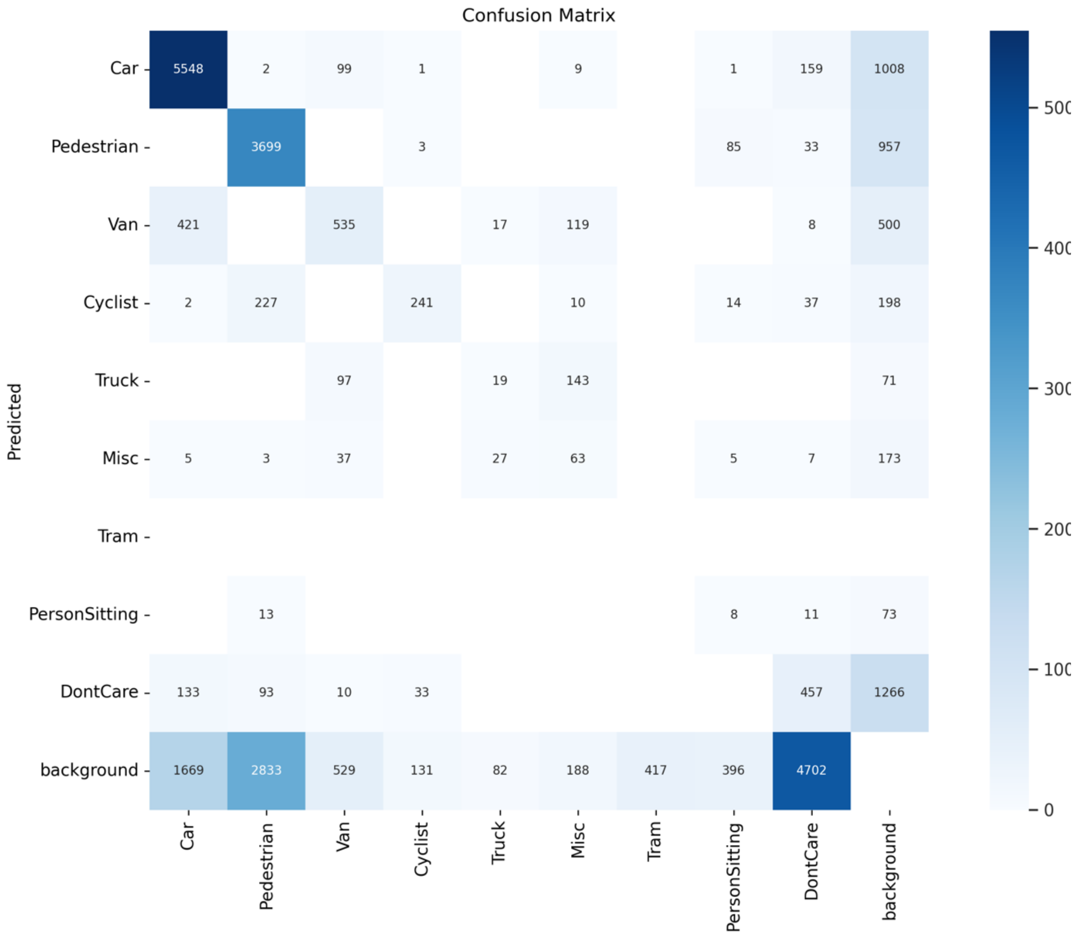


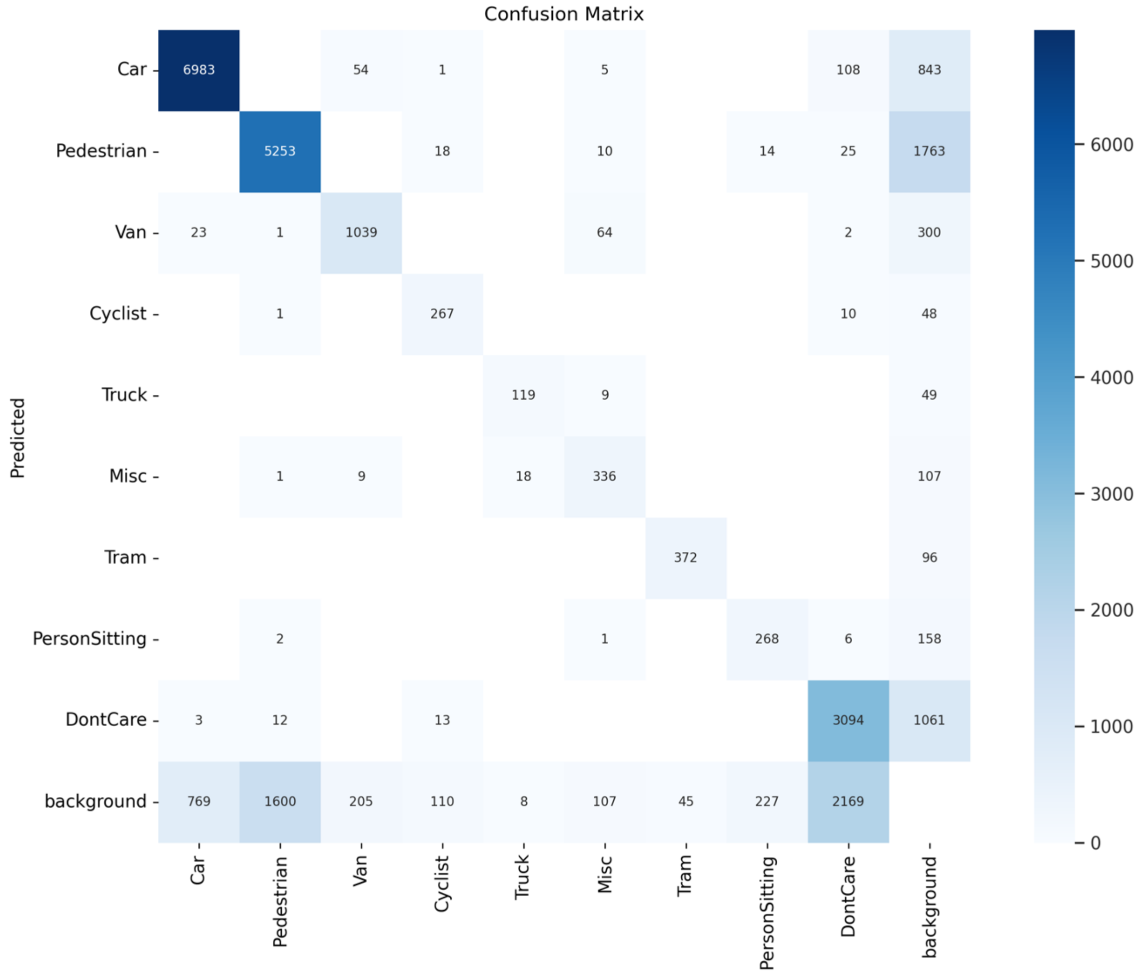
# Object detection: multiclass classification performance

## Confusion matrices:

### camera




### lidar

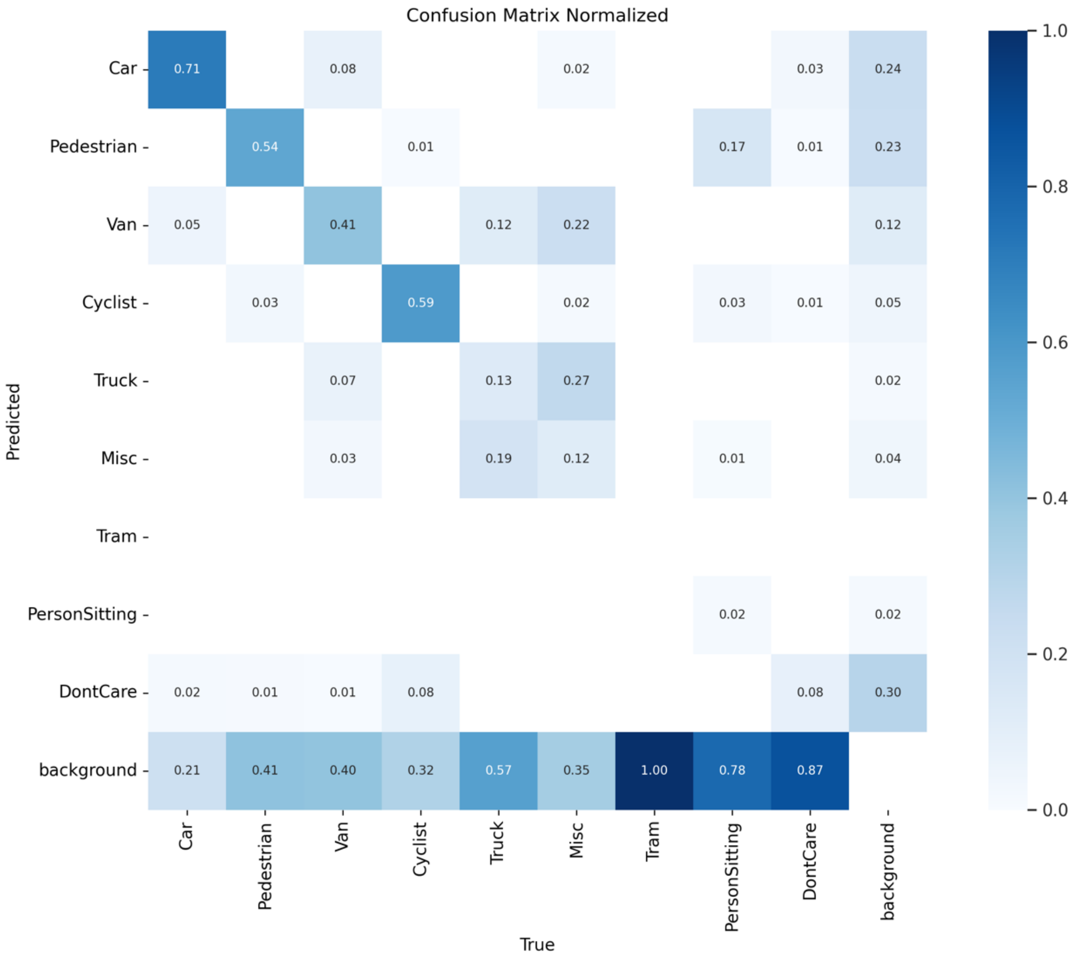


# Object detection: multiclass classification performance

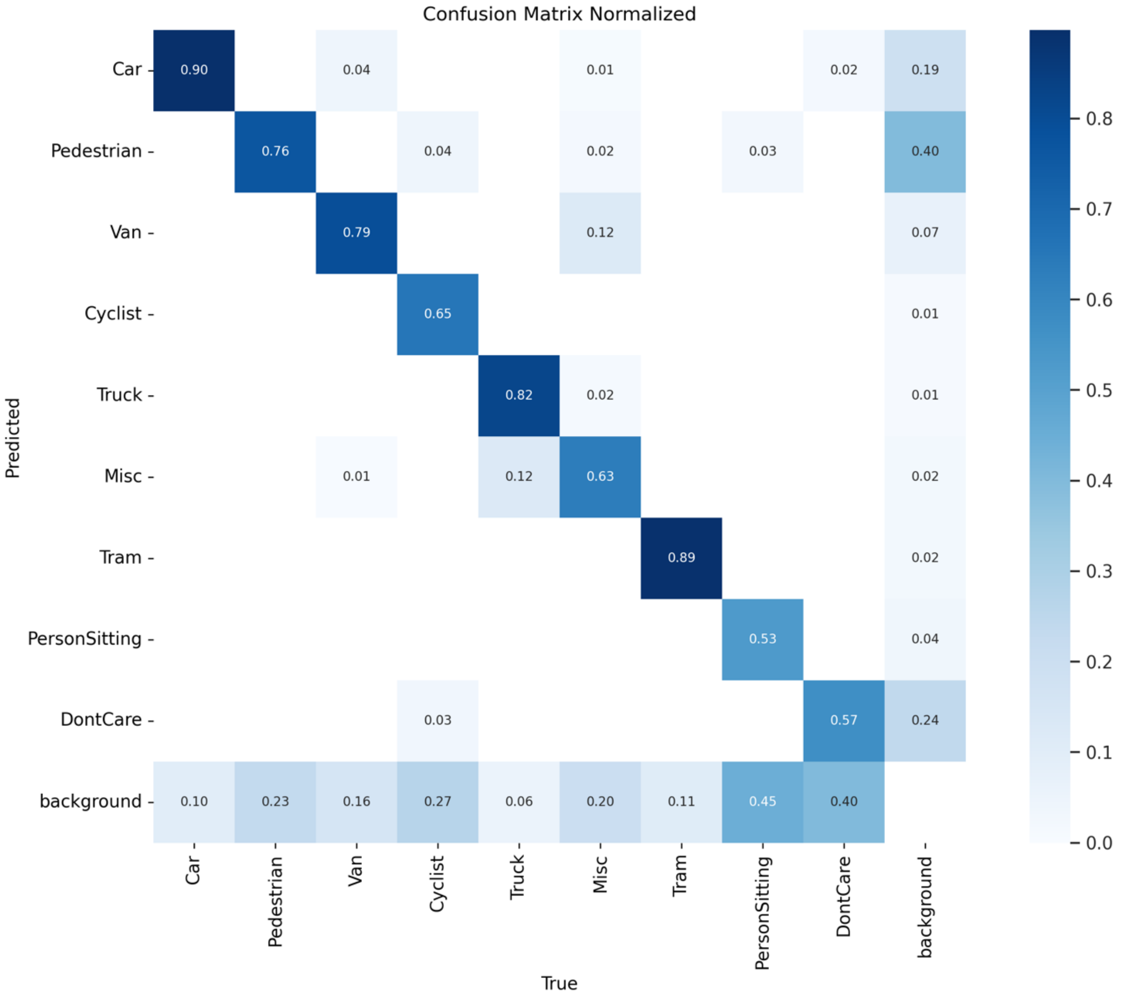


Normalized confusion matrices:

## camera



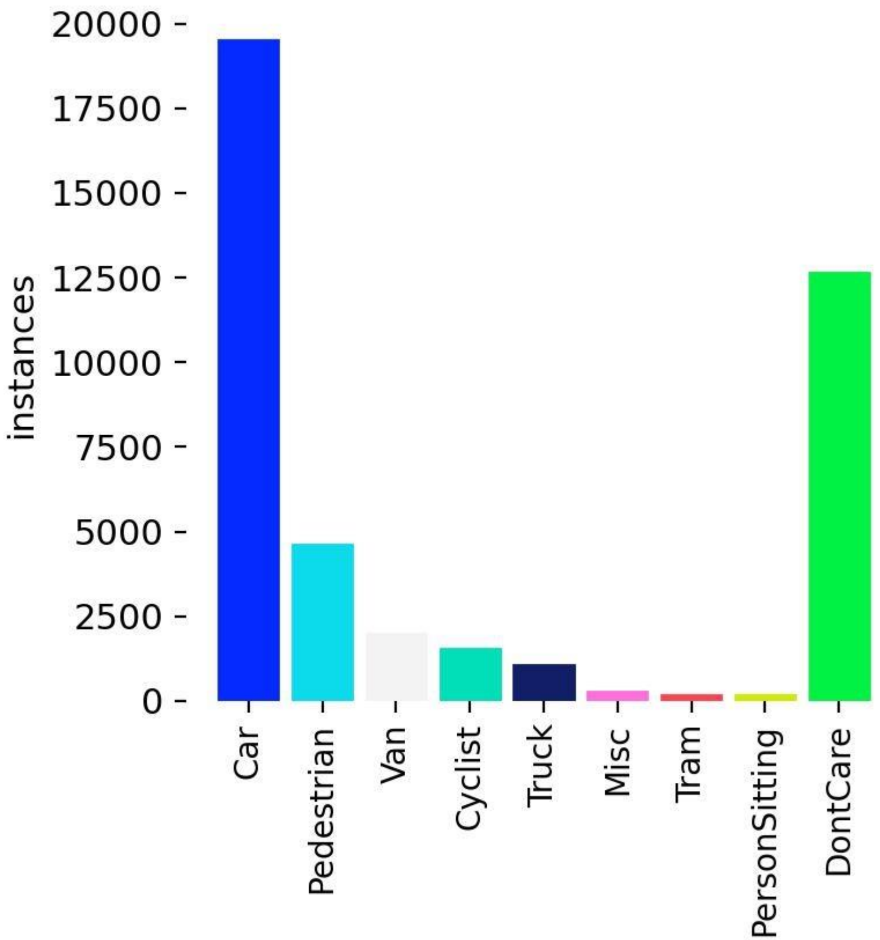

## lidar



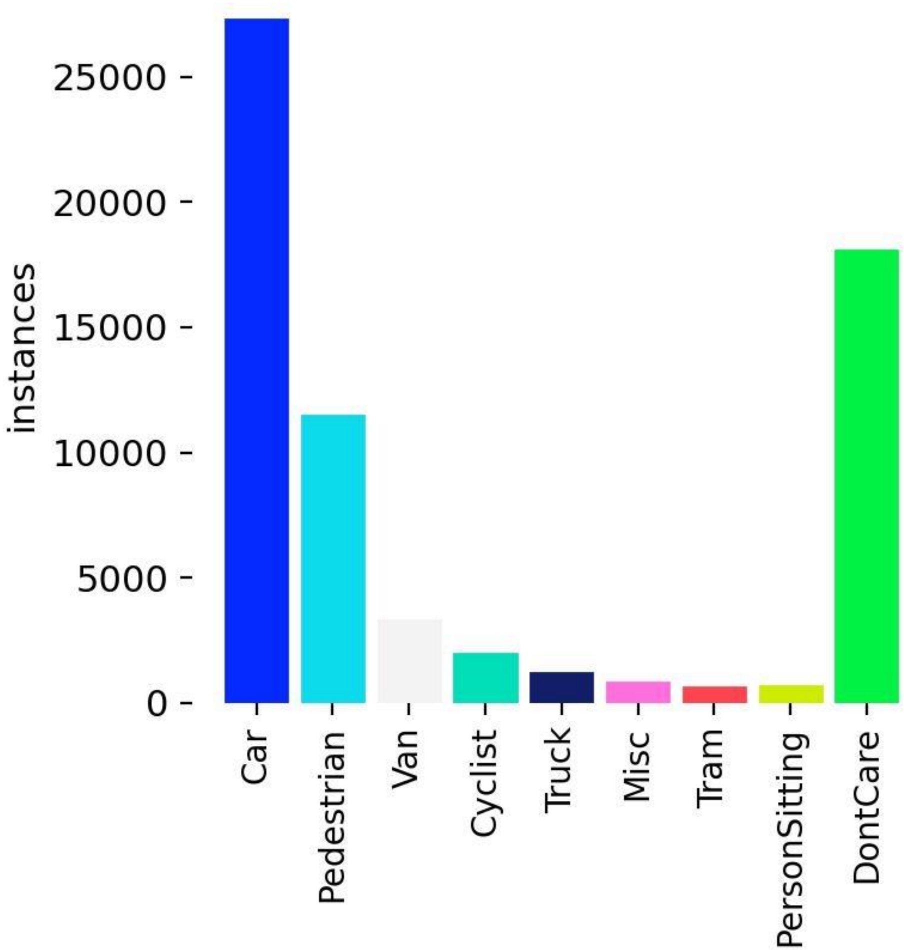
Normalized confusion matrices:

- ❖ Clearly not a balanced dataset
- ❖ Did not attempt to mitigate it (yet)

camera




lidar



# Object detection: multiclass classification performance

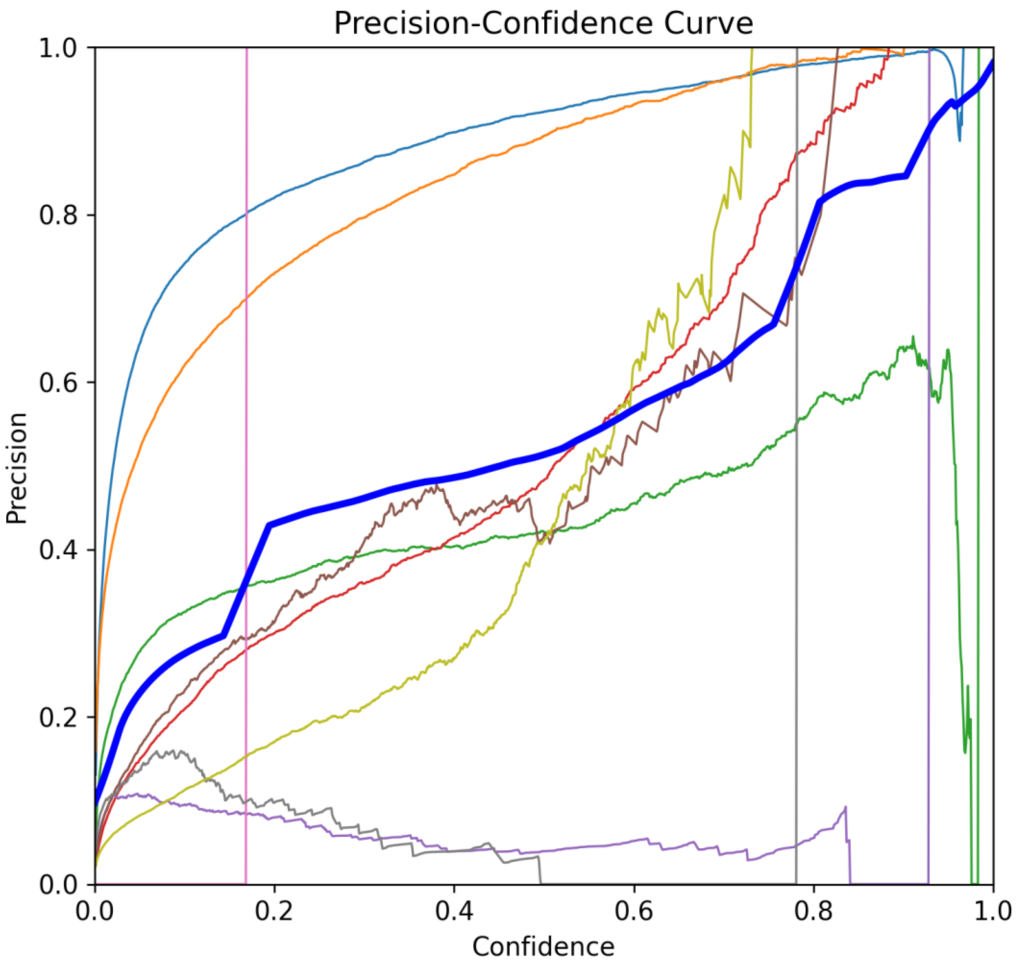
$C$ = likelihood that a prediction is correct

$$P = \frac{TP}{TP + FP}$$

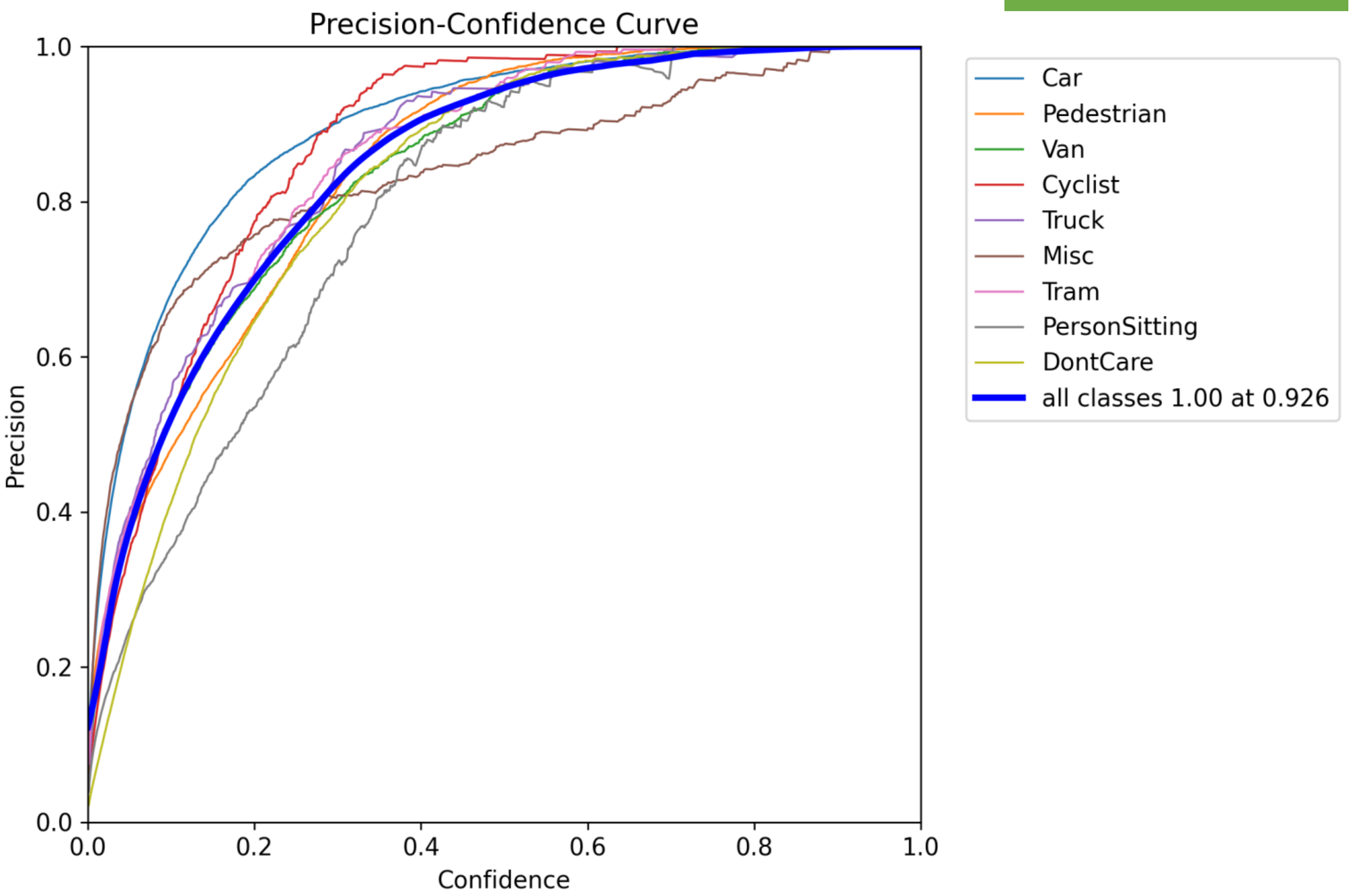
## Precision-confidence curves:

- For camera total ~ok, but class-wise mostly terrible
- For lidar we're underconfident: room for better calibration (ideally diagonal)

camera




lidar

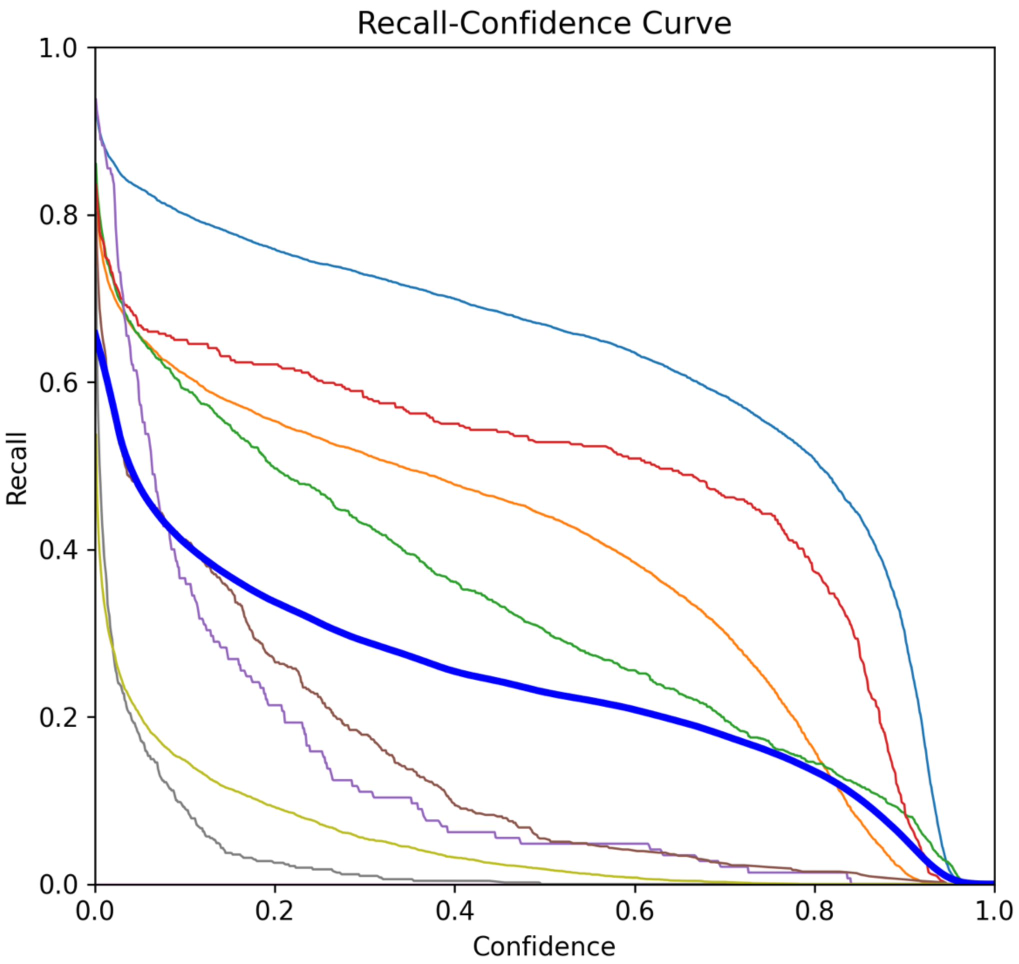


## Recall-confidence curves:

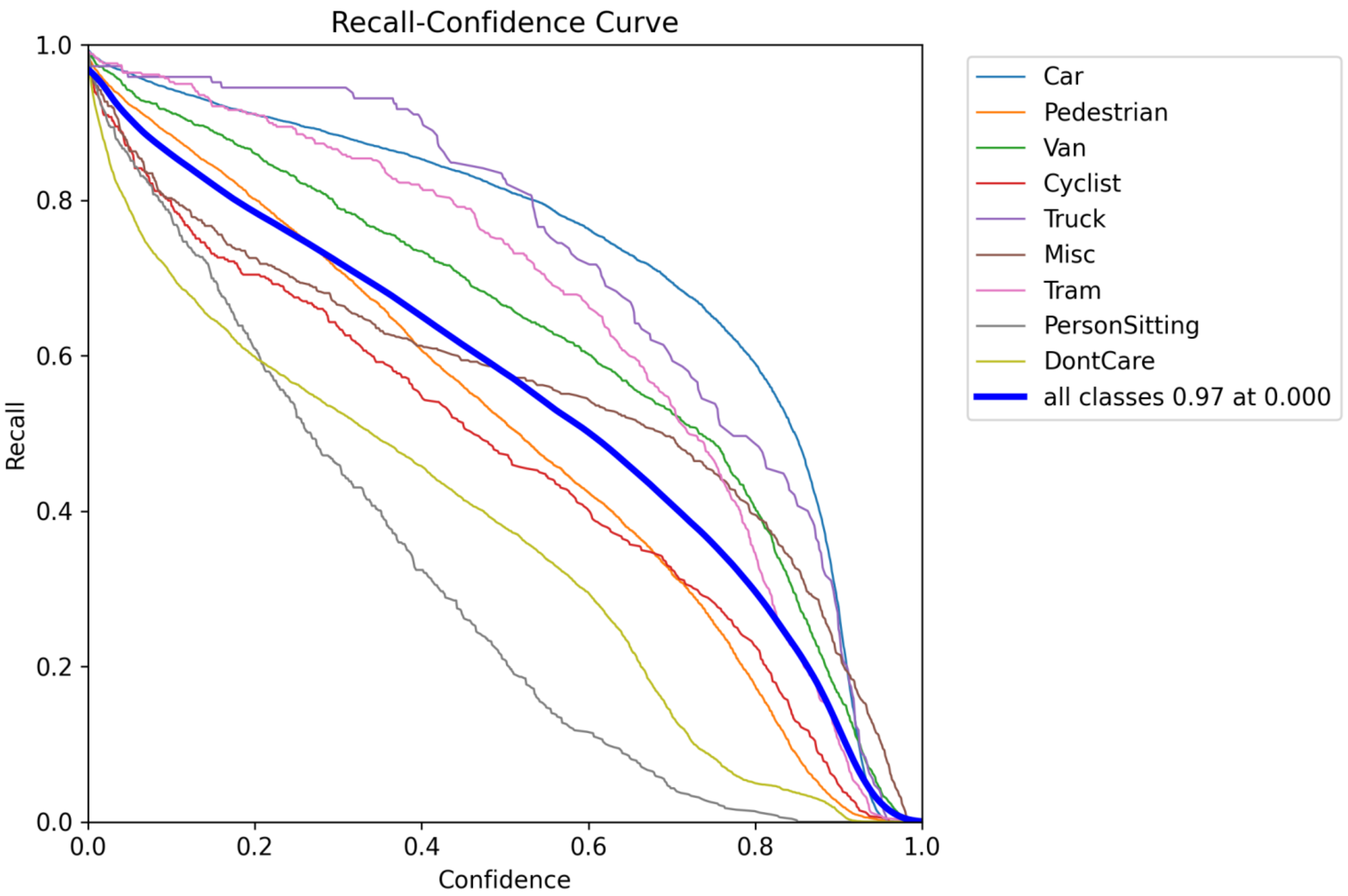
- Ideally: AUC=1
- Both with room for improvement, lidear clearly better

$$R = \frac{TP}{TP + FN} = \frac{TP}{P}$$

camera




lidar

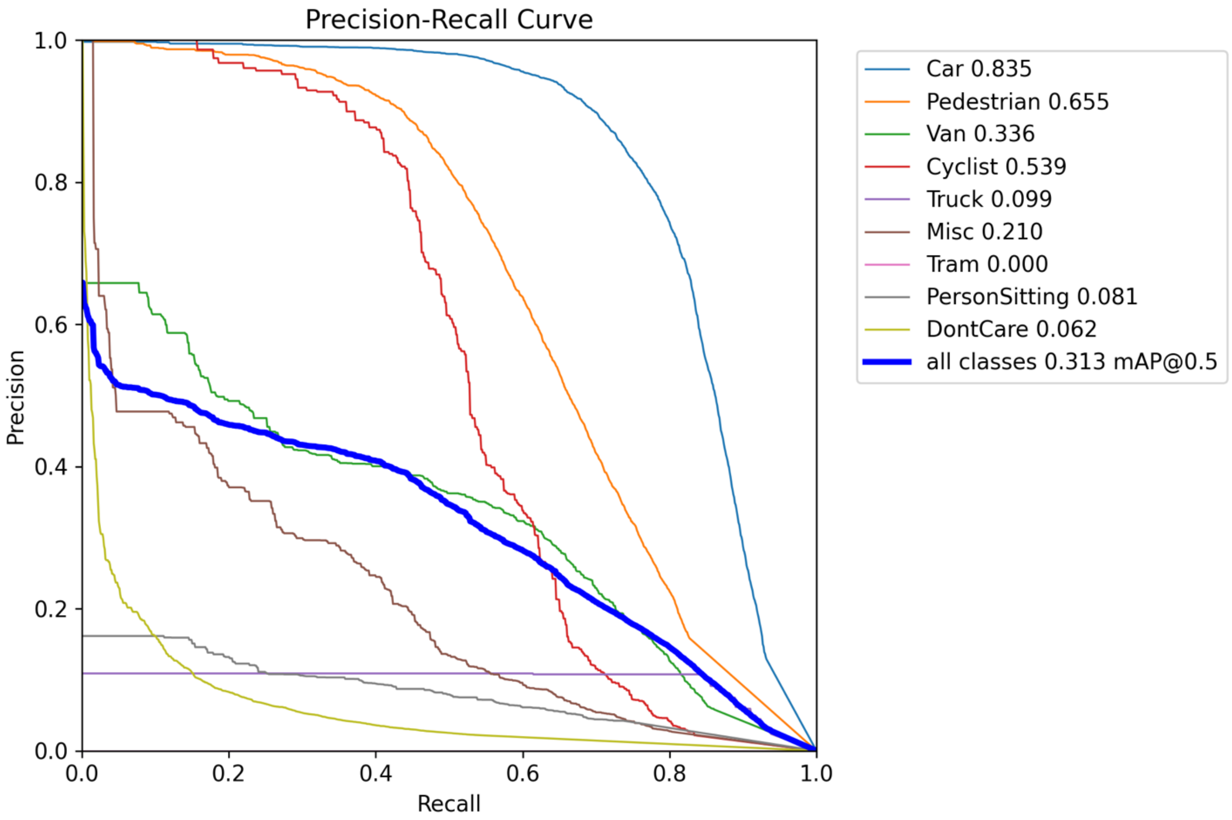


# Object detection: multiclass classification performance

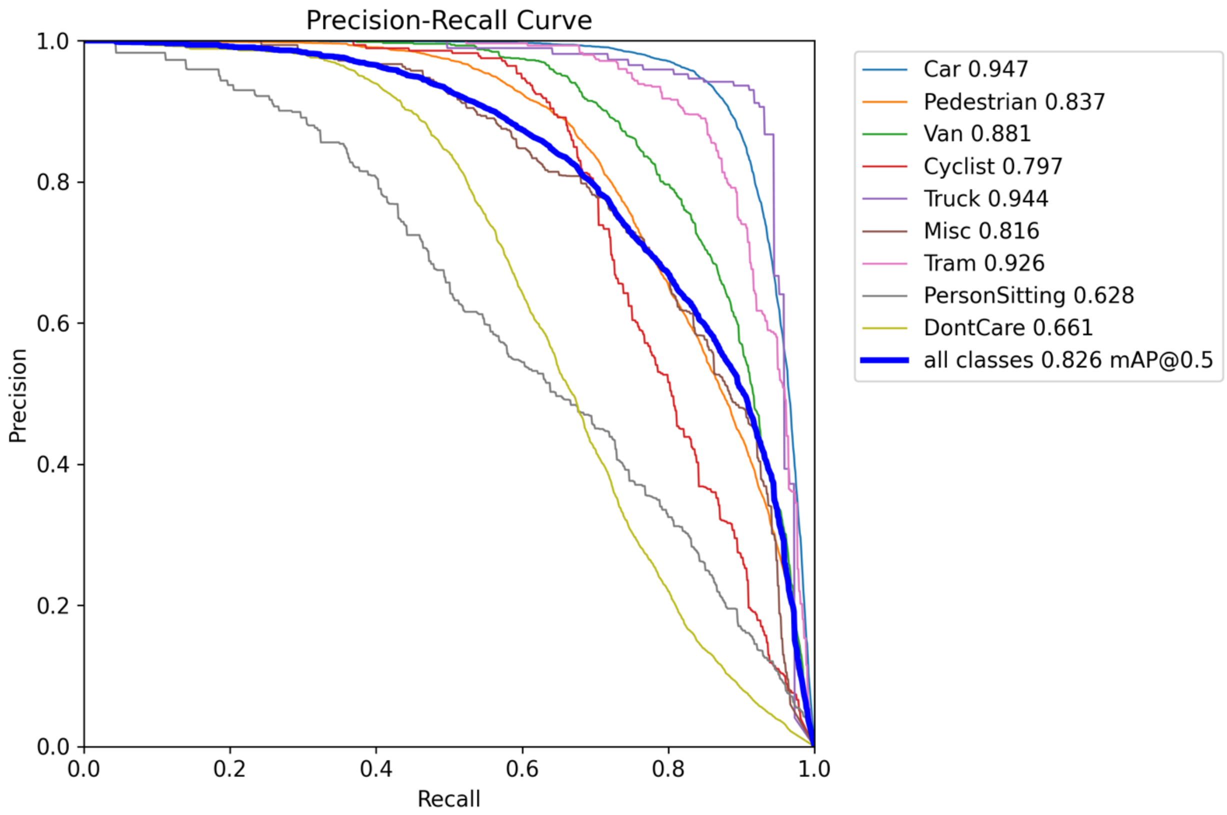
## Precision-recall curves:

- ❖ Lidar close to ideal, camera poor performance
- ❖ Interesting case: Truck
  (rare but very well detected by lidar, better than cars → big size effect?)

camera




lidar



# Object detection: multiclass classification performance

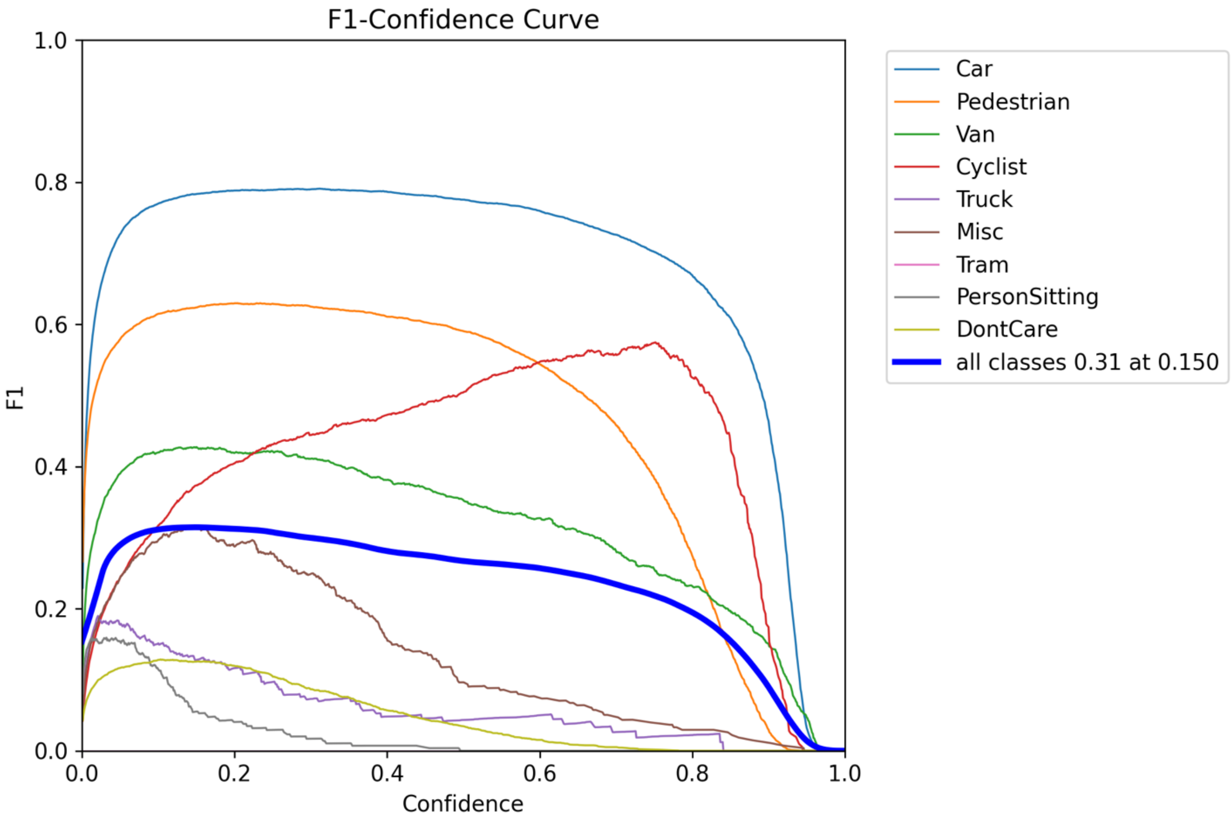


## F1-confidence curves:

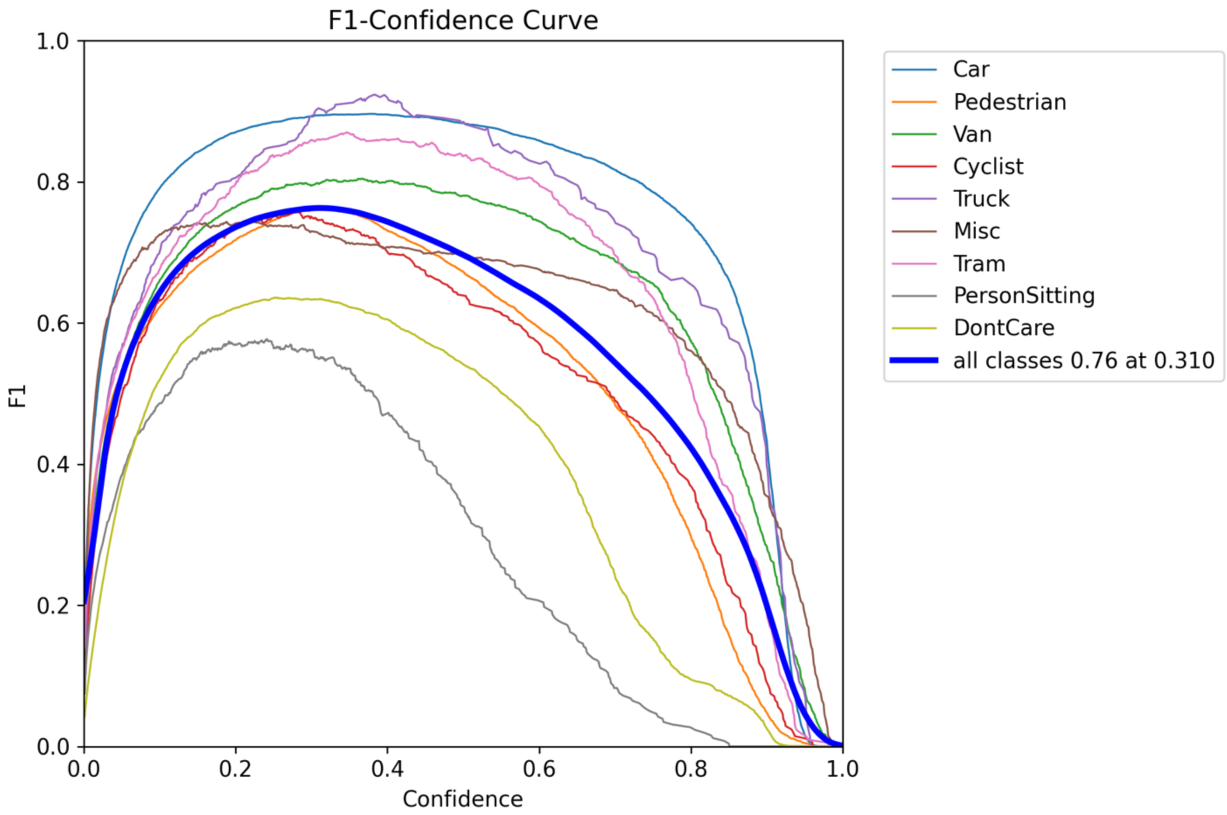
- curious case: cyclist opposite trend to all other classes for camera
- lidar again superior to camera

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R}$$

camera




lidar



# Outline

Introduction

Data used

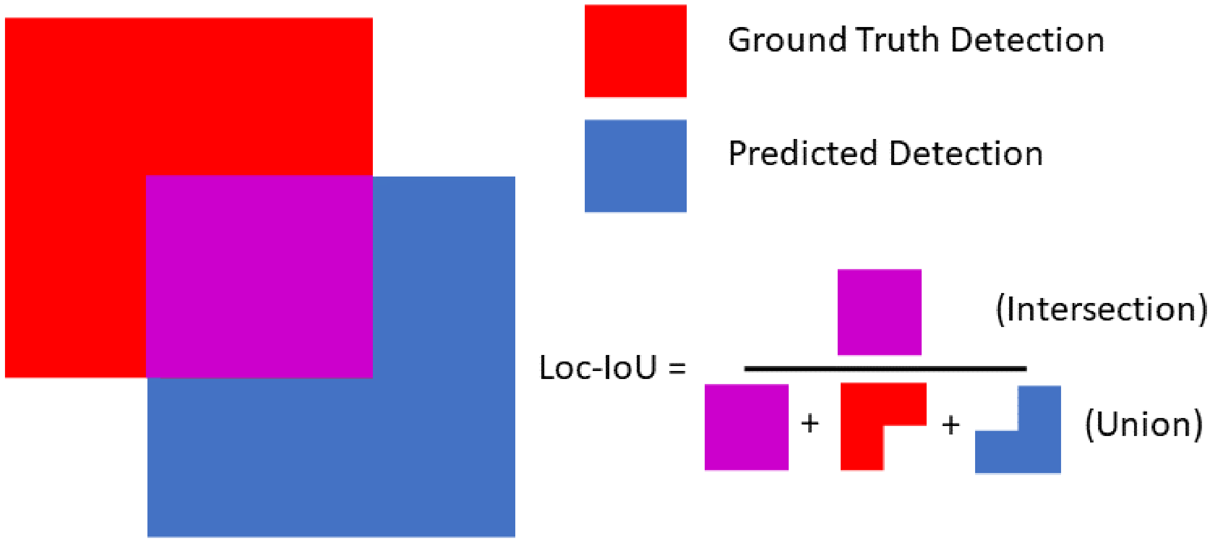
Object detection & tracking

detection | tracking

Label fusion

Summary

# First some definitions:




- Localization Intersection over Union (IoU)
  (How closely the boxes overlap)
  - used for thresholding (match/mismatch)

- Detection Accuracy (DetA):
  (How well tracker localises objects in each frame)

$$\mathrm{DetA} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

1 – ideal
0 – bad

## (some) standard metrics for tracking:

- Multiple Object Tracking Accuracy (MOTA):
  (basically error counting over ground truth)
  - each switch penalized only once for IDSW
  - FN & FP might dominate in crowded scenes
  - insensitive to detection accuracy changes (IoU threshold is fixed)

$$\mathrm{MOTA} = 1 - \frac{\sum(\mathrm{FN}_t + \mathrm{FP}_t + \mathrm{IDSW}_t)}{\sum \mathrm{GT}_t}$$

1 – ideal
0 or negative – bad

$\mathrm{GT}_t$ – total ground truth objects in frame $t$
IDSW – identity switches ()

- IDF1
  (focused on persistent correct identification)
  - More sensitive to tracking consistency
  - Balances precision and recall
  - Less affected by the total numer of objects than MOTA
  - can decrease when improving detection
  - insensitive to detection accuracy changes

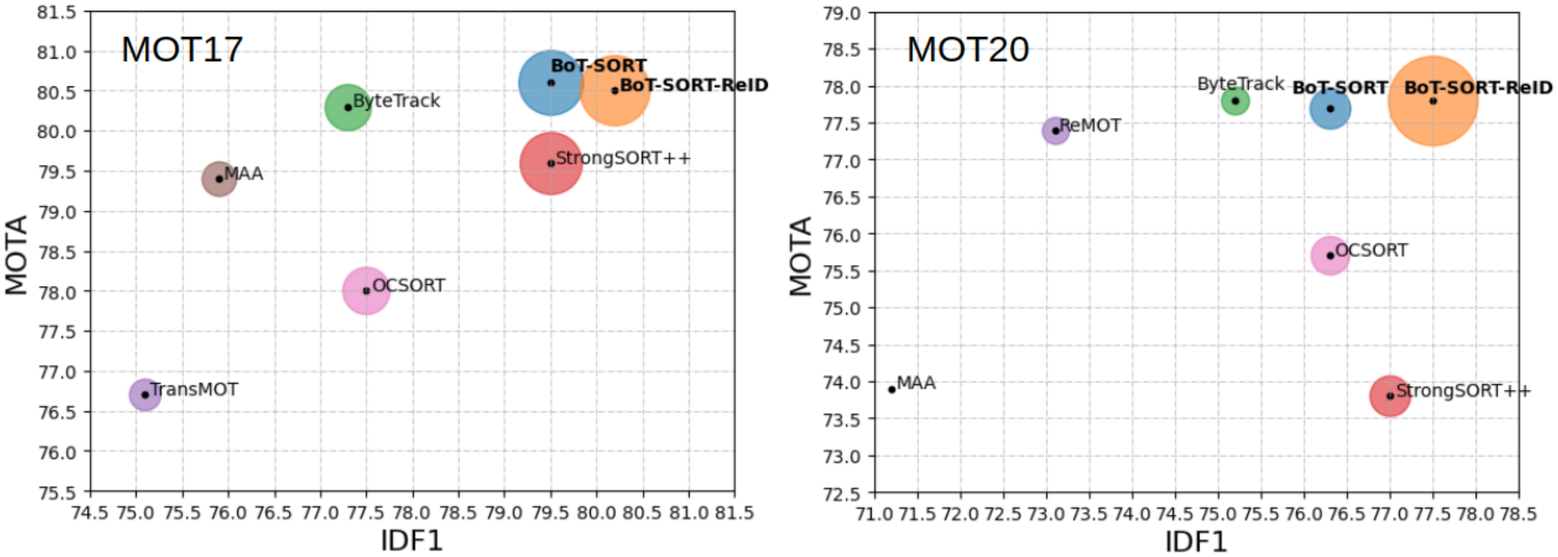
$$\mathrm{IDF1} = \frac{2 \cdot \mathrm{IDTP}}{2 \cdot \mathrm{IDTP} + \mathrm{IDFP} + \mathrm{IDFN}}$$

$\mathrm{IDTP}$ – correct trajectories
$\mathrm{IDFP}$ – fake trajectories
$\mathrm{IDFP}$ – untracked ground truth trajectories

There's also e.g. more robust Higher Order Tracking Accuracy (HOTA), but we don't need it for this talk

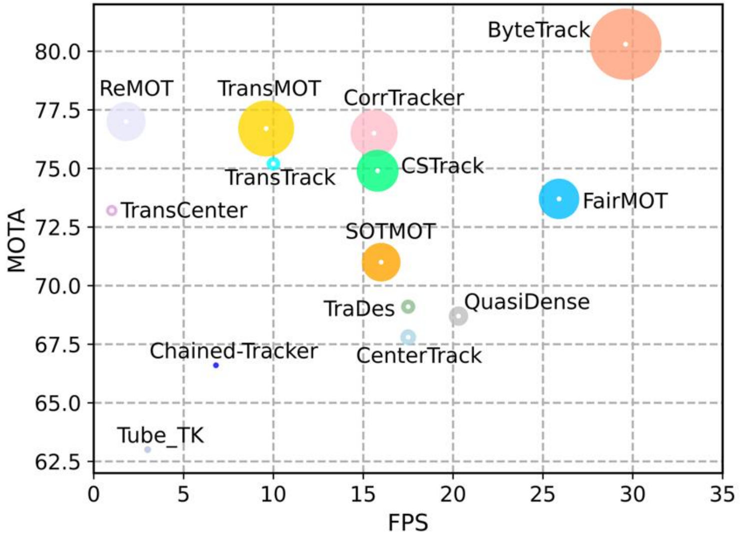
# 2 supported tracking algorithms in Ultralytics:

❖ BoT-SORT



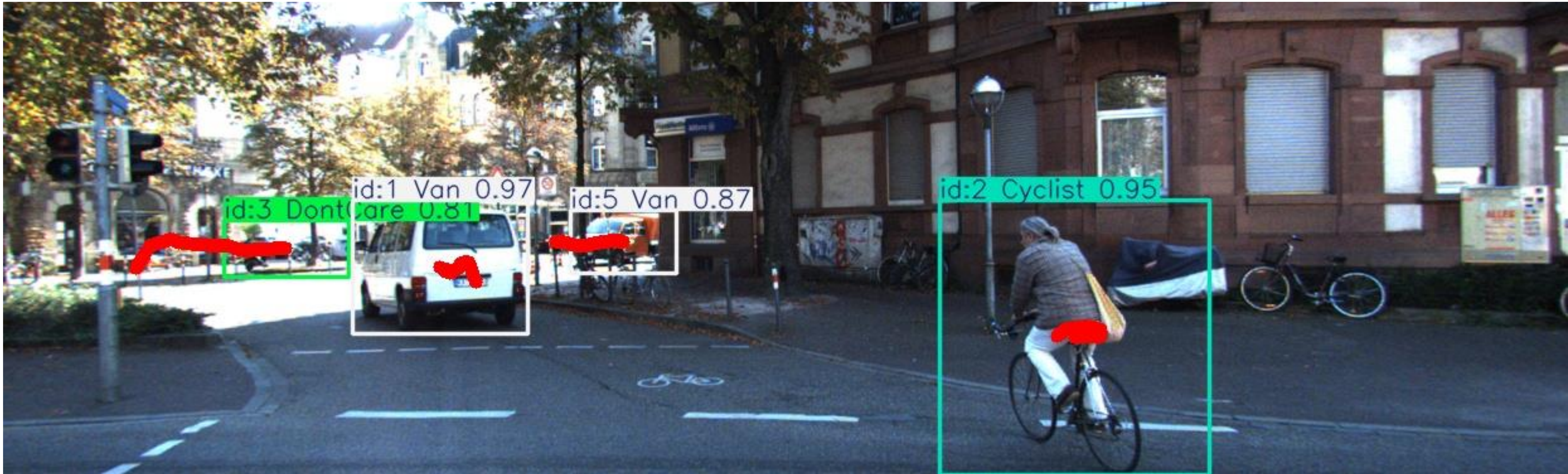

N. Aharon et al., BoT-SORT: Robust Associations Multi-Pedestrian Tracking, [arXiv:2206.14651](https://arxiv.org/abs/2206.14651)

❖ ByteTrack
(for now we actually use this one)




Y. Zhang et al.,
ByteTrack: Multi-Object Tracking by Associating Every Detection Box, [arXiv:2110.06864](https://arxiv.org/abs/2110.06864)

Tracking seems to work quite ok:



# Outline

- Introduction
- Data used
- Object detection & tracking
- Label fusion
- Summary

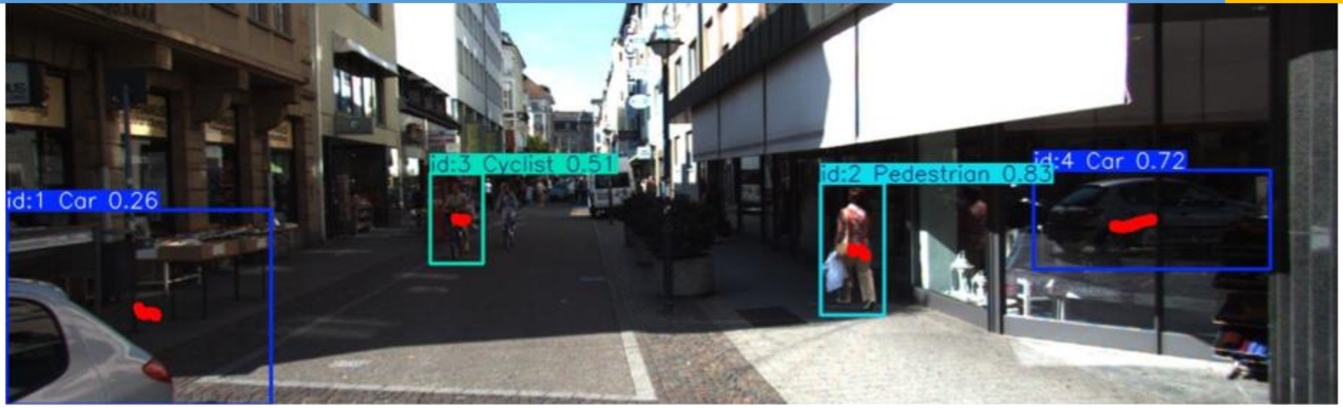
## Label Fusion:

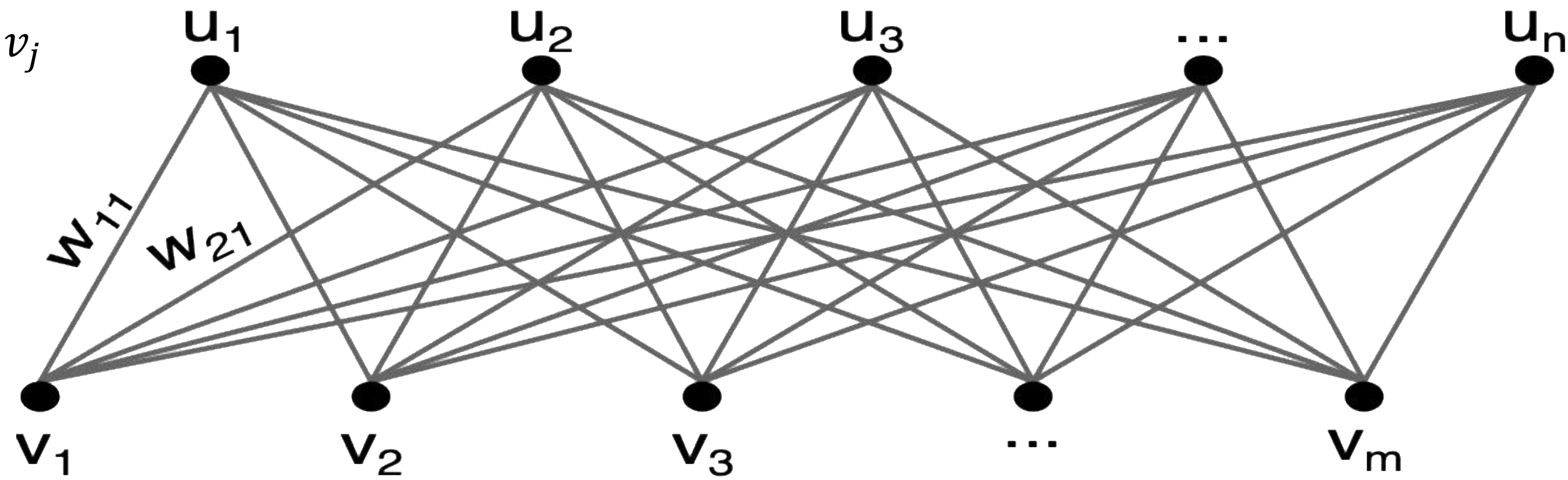
- can be represented as such a graph
- performed for each pair of subsequent video frames
- $w_{ij}$ measures similarity between $u_i$ and $v_j$
- goal: maximize $\sum w_{ij}$
- combinatorial optimisation problem

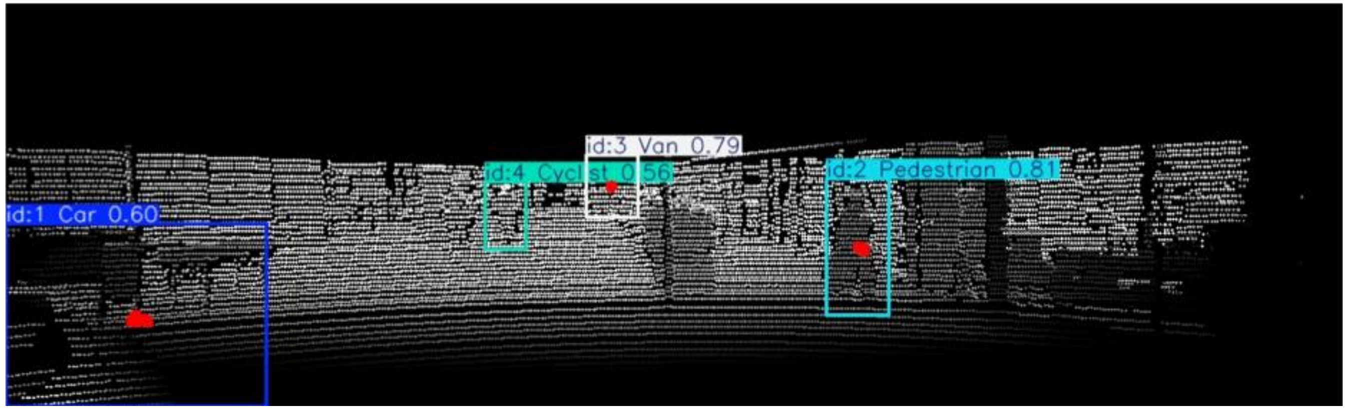
minimization problem with cost function $E$ in the form of the Ising model of a system of spins with values $\sigma = \pm 1$ :

$$\vec{\sigma}^* = \underset{\vec{\sigma}}{\operatorname{argmin}}\, E(\vec{\sigma})$$

$$E(\vec{\sigma}) = -\sum_{i \neq j} J_{ij}\, \sigma_i \sigma_j + \sum_i h_i \sigma_i$$



## Quadratic unconstrained binary optimization (QUBO) formulation:

we transform to binary variables: $q_i = \frac{\sigma_i + 1}{2}$ and get:

$$\vec{q}^* = \underset{\vec{q}}{\operatorname{argmin}}\, E'(\vec{q})$$

$$E'(\vec{q}) = -\sum_{i \neq j} a_{ij}\, q_i q_j - \sum_i b_i q_i$$

Our case (MOT) needs a bit more effort than Ising model:

$$M^* = \underset{M}{\operatorname{argmax}} \sum_{i,j} w_{ij}(m)$$

But after some rewriting we get QUBO formulation:

$$\vec{x}^* = \underset{\vec{x} \in \{x_{u,v} | (u,v) \in E\}}{\operatorname{argmin}} F(\vec{x})$$

$$F(\vec{x}) = F_w(\vec{x}) + \lambda F_U(\vec{x}) + \lambda F_V(\vec{x})$$

where:

$$x_{u,v} = \begin{cases} 1 \text{ for } (u,v) \in M \\ 0 \text{ for } (u,v) \notin M \end{cases}$$

$$F_w(\vec{x}) = -\sum_{u \in U} w(u,v) x_{u,v}$$

$$F_U(\vec{x}) = \sum_{u \in U} \left( \sum_{i<j} x_{u,v(i)}\, x_{u,v(j)} \right)$$

$$F_V(\vec{x}) = \sum_{v \in V} \left( \sum_{i<j} x_{u(i),v}\, x_{u(j),v} \right)$$

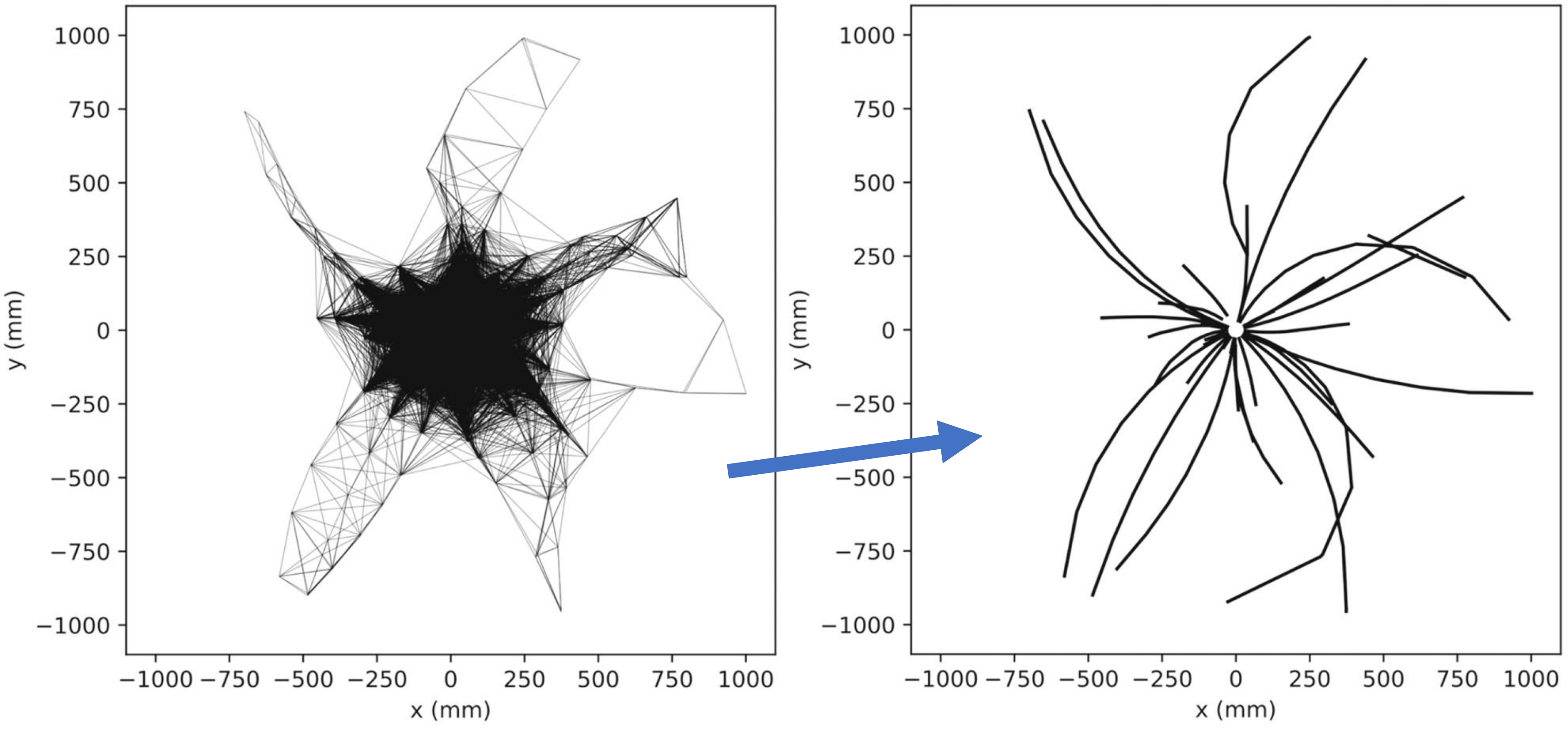
Y. Ihara, *Enhancing multiple object tracking accuracy via quantum annealing*, doi:10.1038/s41598-025-07492-7

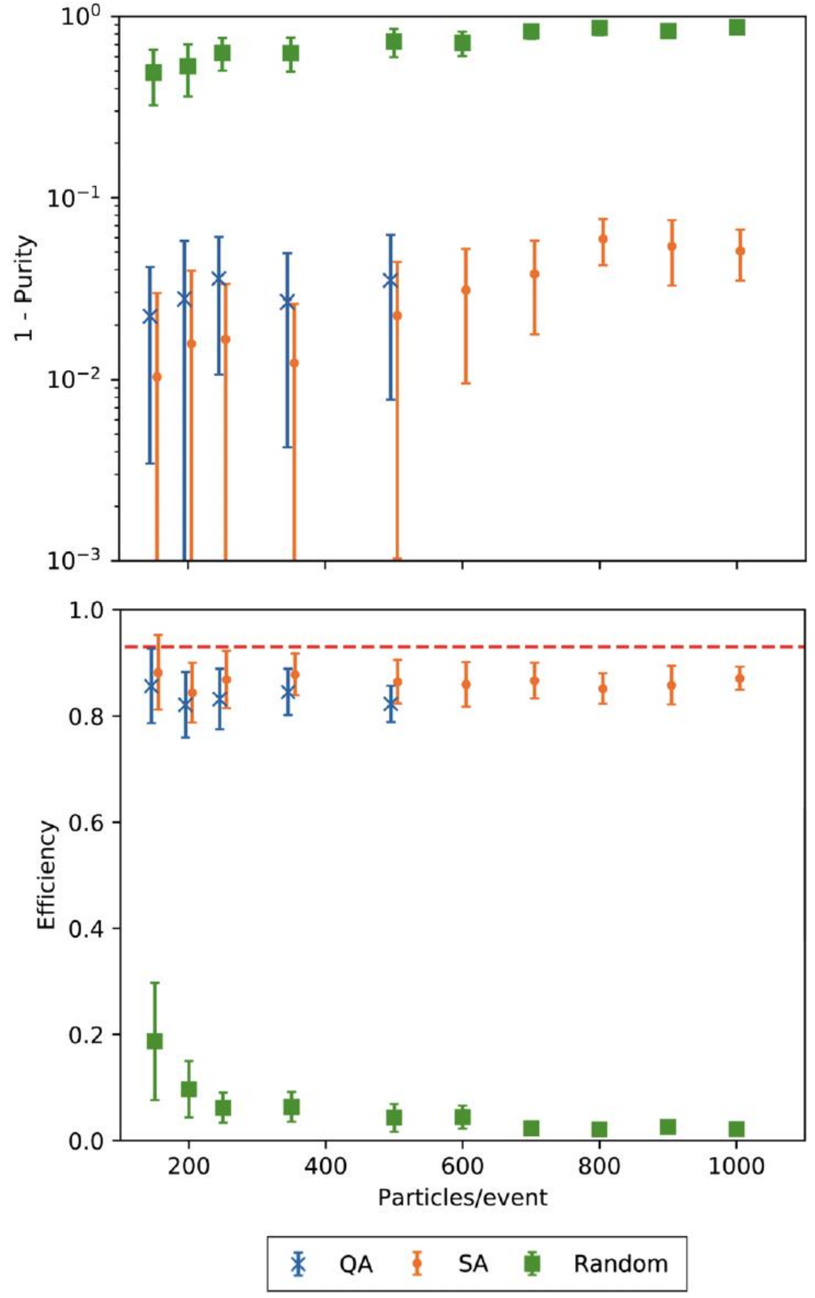
# Little detour: QUBO for particle physics ☺



## QUBO can be useful outside of automotive context:

- anywhere, where there’s combinatorics involved
- e.g. … in collisions at LHC




A. Zlokapa et al., *Charged particle tracking with quantum annealing optimization*, doi.org/10.1007/s42484-021-00054-w

Efficient way to solve QUBO problems: Quantum Annealing (QA)

Adiabatic Model of Quantum Computation (AQC):

1. Prepare the system in the ground state of a simple $H_0$
2. Adiabatically (slowly) evolve towards $H_p$
3. Measure the qubits, they should be in the ground state of $H_p$

(the adiabatic theorem)

Time-dependent quantum Ising Hamiltonian: $\hat{H}(t) = \left(1 - \frac{t}{\tau}\right)\hat{H}_0 + \frac{t}{\tau}\hat{H}_p$

for $\tau \gg t$, the final state will satisfy: $\hat{H}_p|\psi^{(p)}\rangle_0 = E_0|\psi^{(p)}\rangle_0$

How slow is slow enough? The evolution time must be roughly $T \gg \frac{1}{\Delta E_{\min}^2}$,
where $\Delta E_{\min}^2$ is the energy difference between $E_0$ and 1st excited state

# Quantum annealing



Quantum Annealing can be simulated or ran on real hardware.

There are a few companies on the market:

NEC

QILIMANJARO

QUANTUM · TECH

D:WAVE

The Quantum Computing Company™

# Outline

- Introduction
- Data used
- Object detection & tracking
- Label fusion
- Summary

## To sum up:

- We could succesfully detect and track multiple objects with KITTI + YOLO
- Lidar data clearly superior
- Hopefully there's still gain from complementary camera data
- Currently trying to settle for a particular option for $w_{ij}$

## Outlook:

- Implement the complete QUBO formulation
- Test solving the QUBO with simulated annealing
- Test on real hardware (D-wave?)

**Thank you for your attention!**

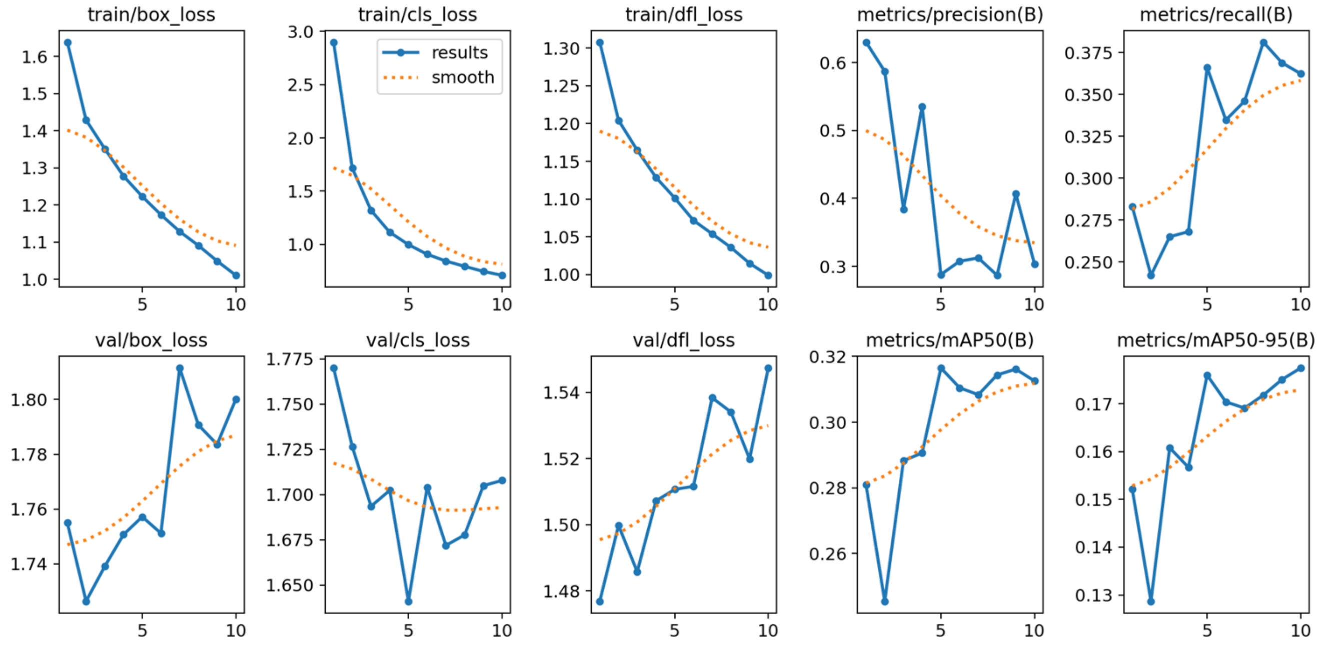
**Any questions/suggestions?**

# Backup

# Learning curves



camera




lidar