Dipl.-Ing. **Ilkay Wunderlich**

ilkay.wunderlich@tu-dresden.de
Technical University Dresden, Faculty of Computer Science,
Institute of Computer Engineering
Dresden, Germany

Dipl.-Inform. **Michael Breiter**

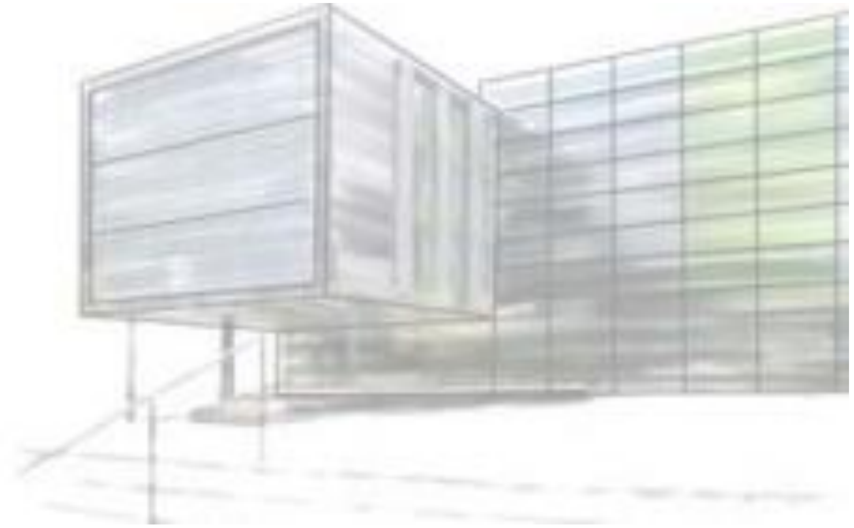
michael.breiter@eyyes.com
EYYES Deutschland GmbH
Aachen, Germany

Automated Image Annotation for Object Detection

The Seventh International Conference on Big Data, Small Data, Linked Data and Open Data, ALLDATA 2021
April 18, 2021 to April 22, 2021 - Porto, Portugal

Outline

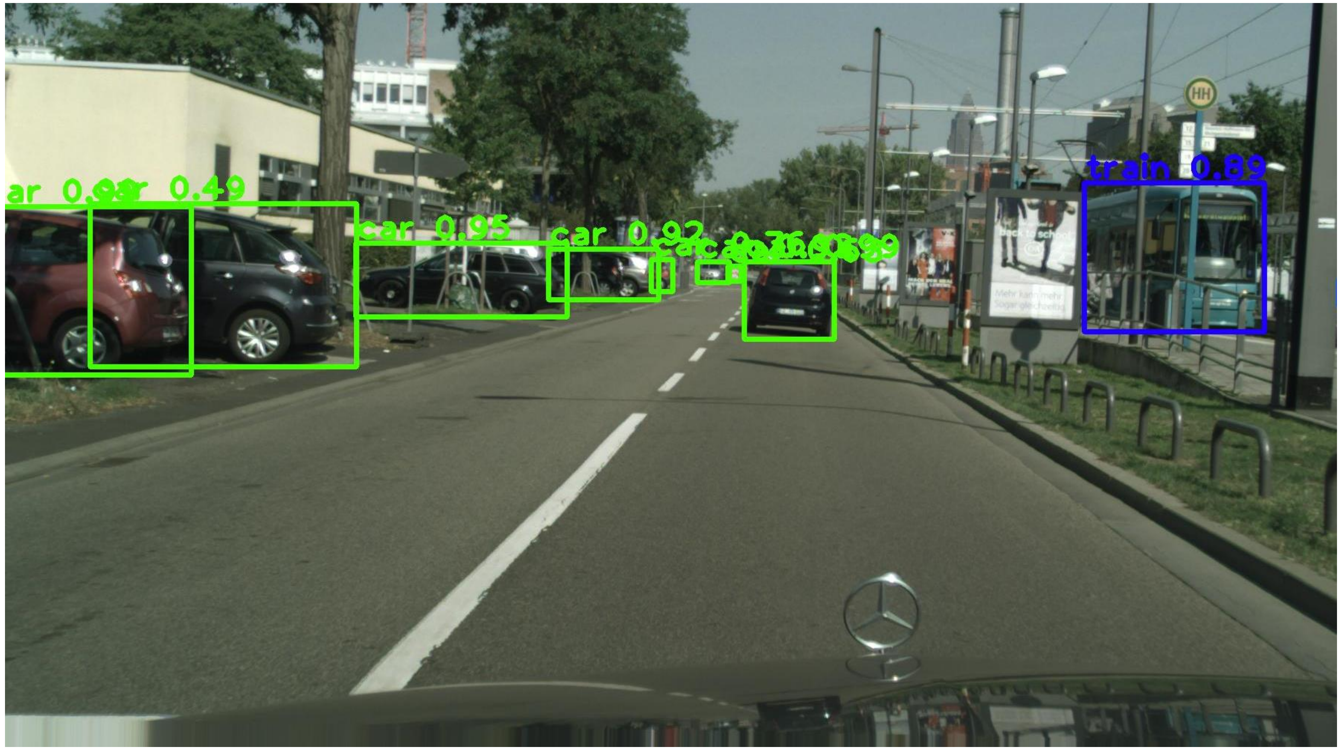
1. Background & Motivation
2. Fundamentals
3. Auto Annotation Workflow
   a) Stage 1: Image Detections
   b) Stage 2: Graph Construction
   c) Stage 3: Detection Combination
   d) Stage 4: Annotation Write Out
4. Evaluation
5. Example: RailEye 3D Annotation
6. Conclusion



1. Background & Motivation

Object Detection

- Important computer vision task
- Nowadays realized by Convolutional Neural Networks (CNNs)
  - ➢ Special kind of neural networks based on 2d convolutions with trainable filter kernels
- Wide use of object detection in many applications
  - Autonomous driving
  - Robot vision
  - Video surveillance
- Constant growth of publications

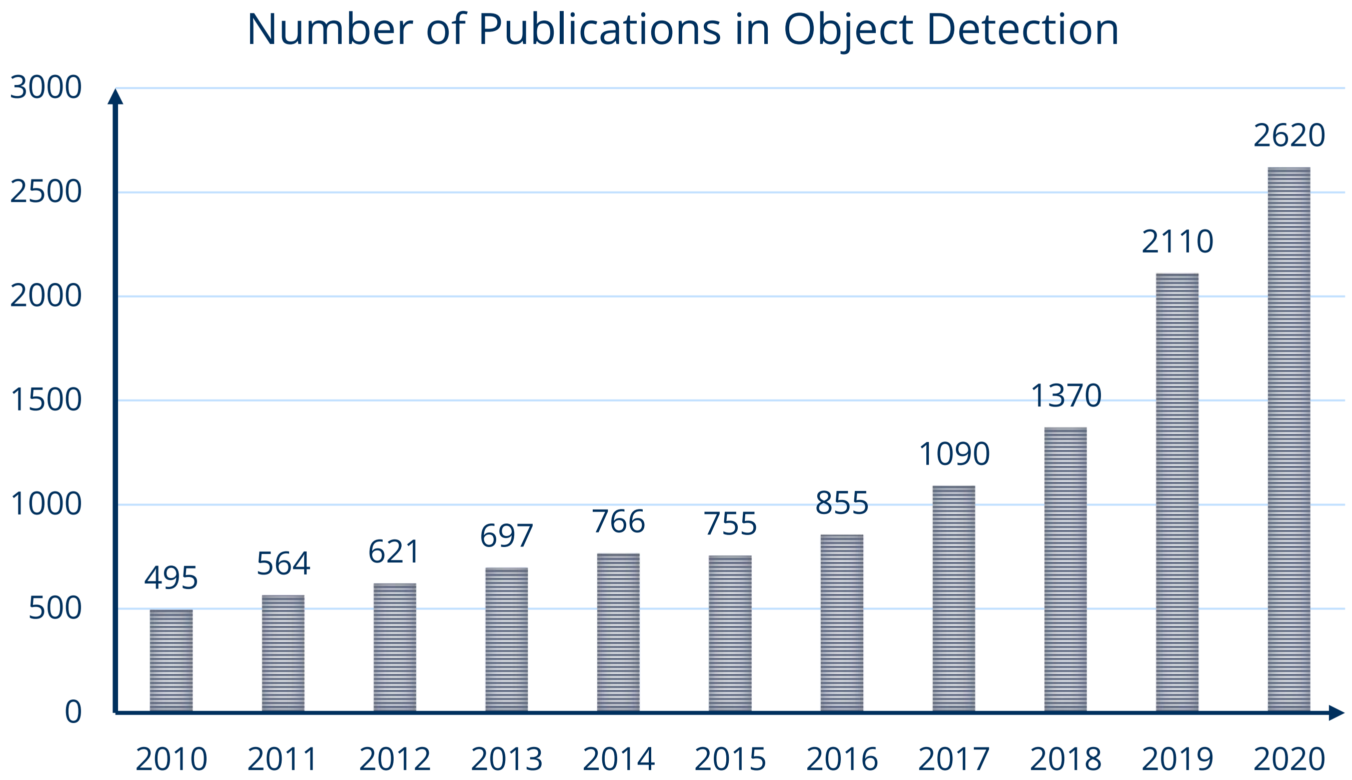


Object detection for an autonomous driving scenario.
Example image from the Cityscapes Dataset[1].

1. Background & Motivation

Object Detection

- Important computer vision task
- Nowadays realized by Convolutional Neural Networks (CNNs)
    - ➢ Special kind of neural networks based on 2d convolutions with trainable filter kernels
- Wide use of object detection in many applications
    - Autonomous driving
    - Robot vision
    - Video surveillance
- Constant growth of publications

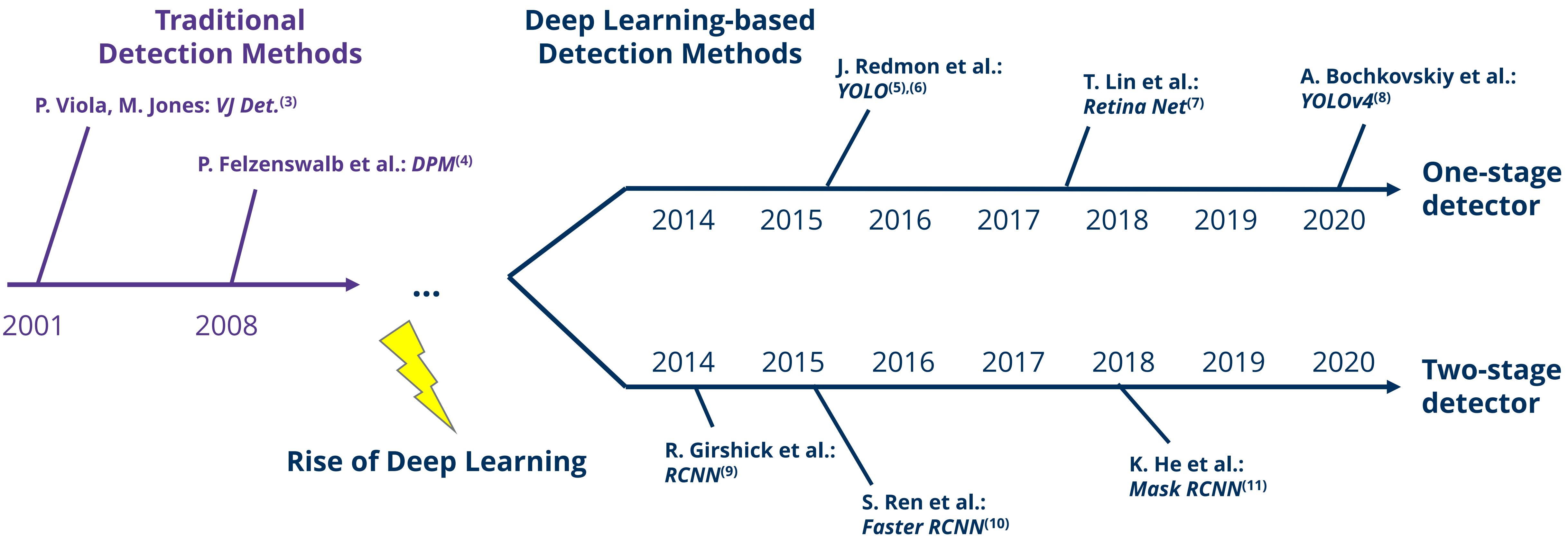


The increasing number of publications in object detection from 2010 to 2020.
Data from Google Scholar advanced search: “object detection” & “detecting objects”.

1. Background & Motivation

Object Detection

- Historic overview of object detection milestones: Z. Zou et al. *Object Detection in 20 Years: A Survey*[2]



1. Background & Motivation

Deep learning-based object detections methods

- Specialized training for modern object detectors needed
- Plenty of well-known object detection datasets available
  - **COCO**: Microsoft Common Objects in Context[12]
    - ➢ 80 classes (person, bicycle, car, … toothbrush)
  - **VOC**: Pascal Visual Object Classes[13]
    - ➢ 20 classes (aeroplane, bicycle, …, person, … tvmonitor)
- Annotation example: XML format in VOC:

```
<object>
        <name>sheep</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
                <xmin>25</xmin>
                <ymin>34</ymin>
                <xmax>419</xmax>
                <ymax>271</ymax>
        </bndbox>
</object>
```

class

bounding box

Algorithm based, Feature selection, Detection Cascades

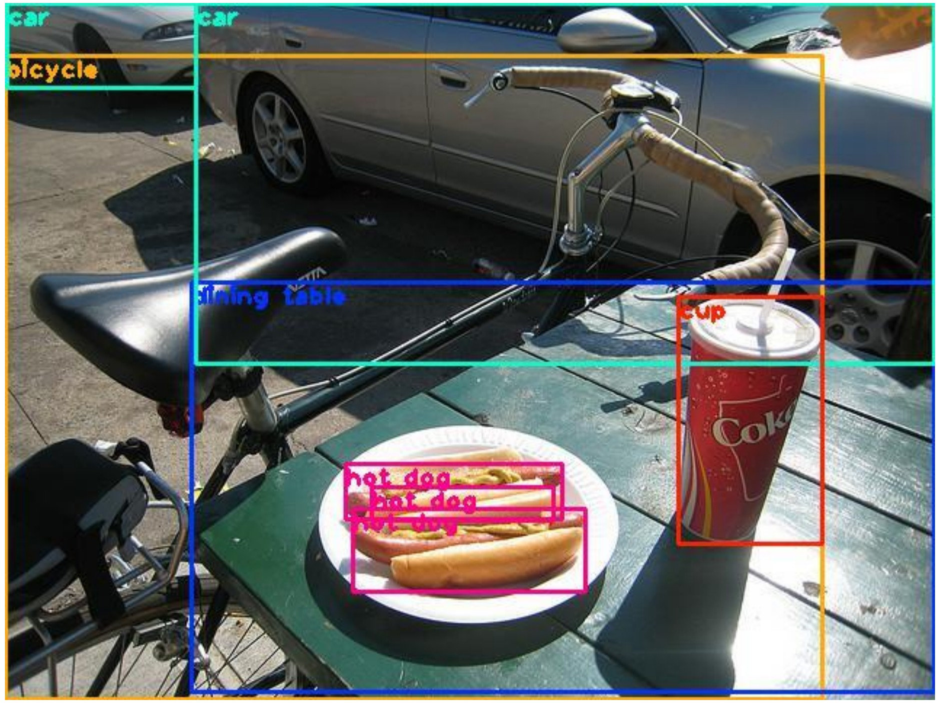
↓

Traditional Detection Methods
≤ 2014

Supervised Learning Image + Annotation

↓

Deep Learning-based Detection Methods
>2014



Bounding boxes of labeled Image
Example image from COCO validation set.

1. Background & Motivation

Deep learning-based object detections methods

- Specialized training for modern object detectors needed
- Plenty of well-known object detection datasets available
  - **COCO**: Microsoft Common Objects in Context[12]
    - 80 classes (person, bicycle, car, … toothbrush)
  - **VOC**: Pascal Visual Object Classes[13]
    - 20 classes (aeroplane, bicycle, …, person, … tvmonitor)
- Annotation example: XML format in VOC:

```
<object>
        <name>sheep</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
                <xmin>25</xmin>
                <ymin>34</ymin>
                <xmax>419</xmax>
                <ymax>271</ymax>
        </bndbox>
</object>
```

class

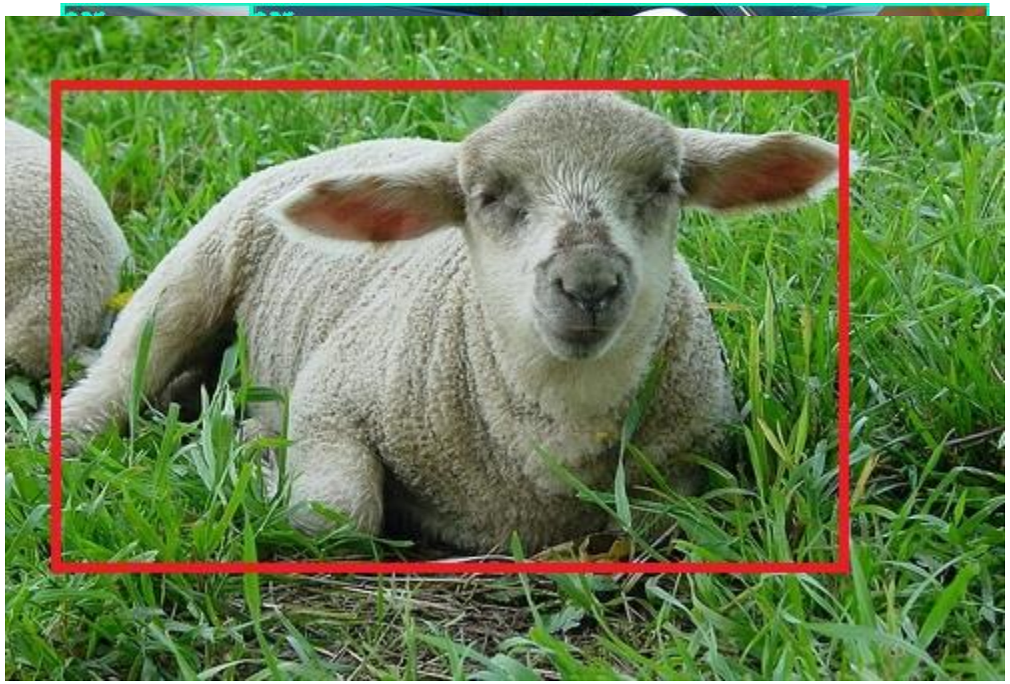
bounding box

Algorithm based,
Feature selection,
Detection Cascades

Traditional Detection Methods
≤ 2014

Supervised Learning
Image + Annotation

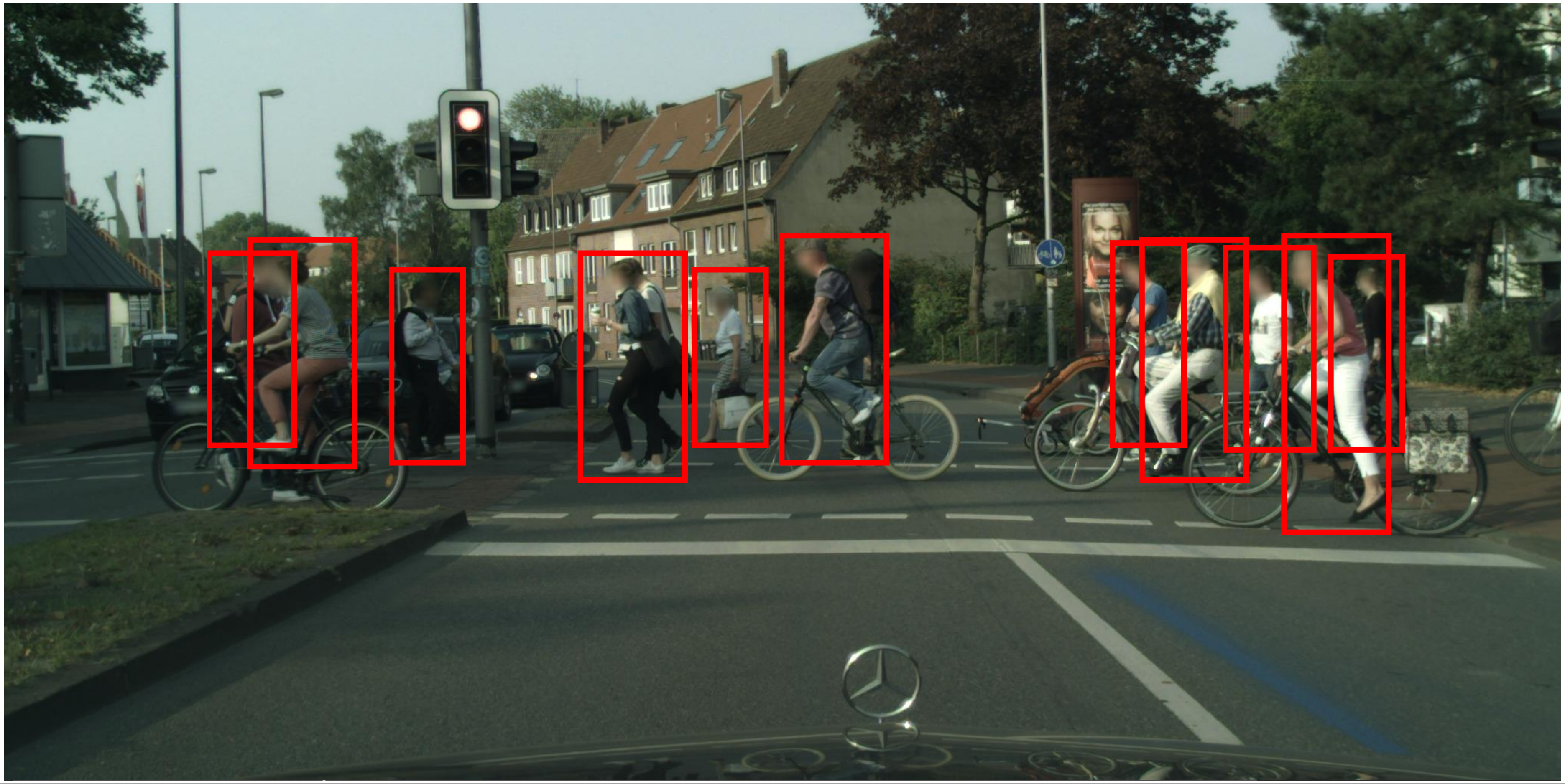
Deep Learning-based Detection Methods
>2014



Annotation example.
Example image from VOC validation Dataset.

1. Background & Motivation

How to get annotations from self recorded data?
=> Manual annotation! For instance, person annotation on this image:



Example image from CityScapes dataset[1]

**Use one (of many) annotation tool:
E.g.,
Make Sense AI[14]**

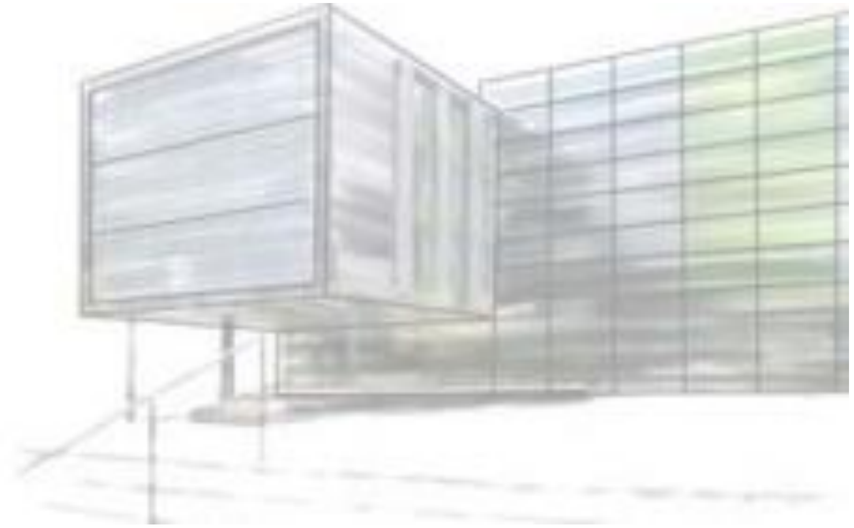
✓ **Non-Commercial tools available**

✓ **Easy to use**

× **Quick exhaustions**

× **Preciseness depending on annotator**

Outline



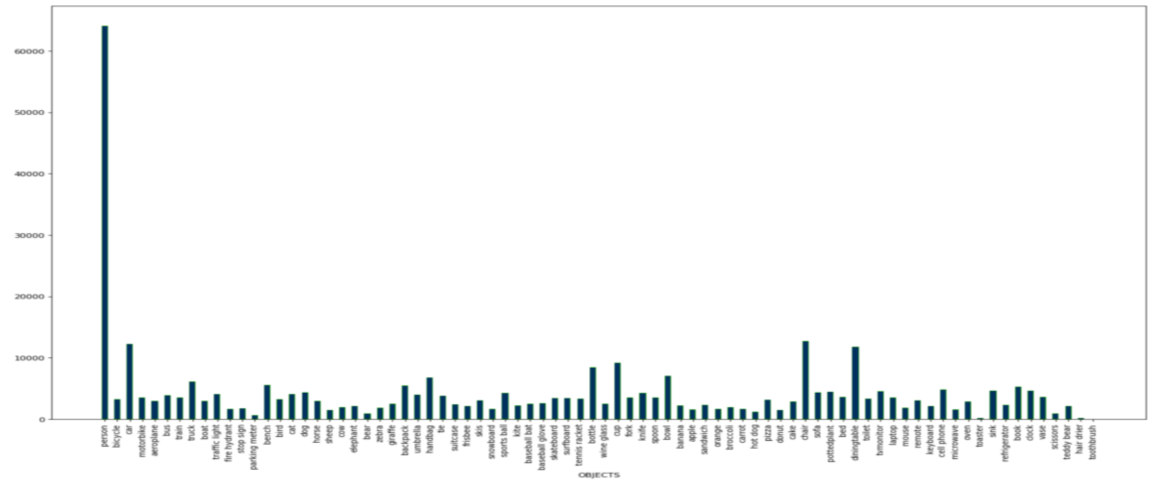
2. Fundamentals

Best practice results: Combination of three pretrained high quality object detectors

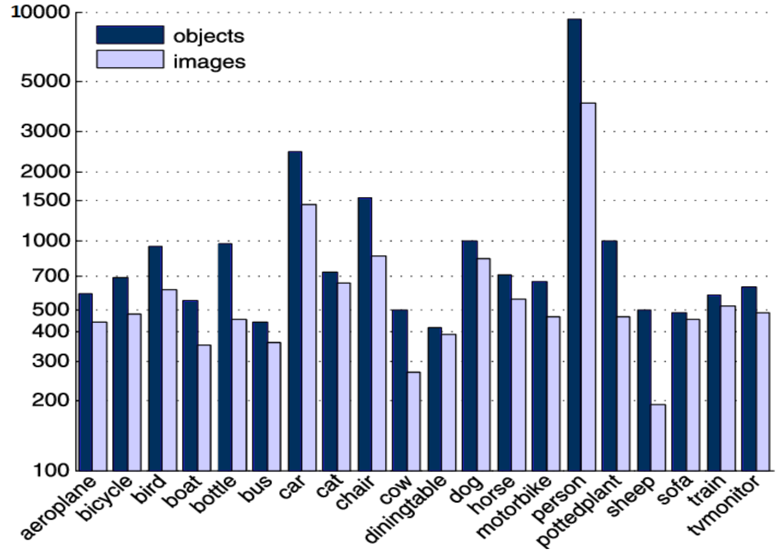
- **YOLOv3** trained on VOC[15]
  - ➢ Mean Average Precision mAP: 81.5
- **YOLOv4** trained on COCO[16]
  - ➢ mAP: 65.7
- **Mask-RCNN** trained on COCO[17]
  different augmentation techniques and different detector class
  - ➢ mAP: 62.3

High representation of person class

- ➢ Average precision of person way higher



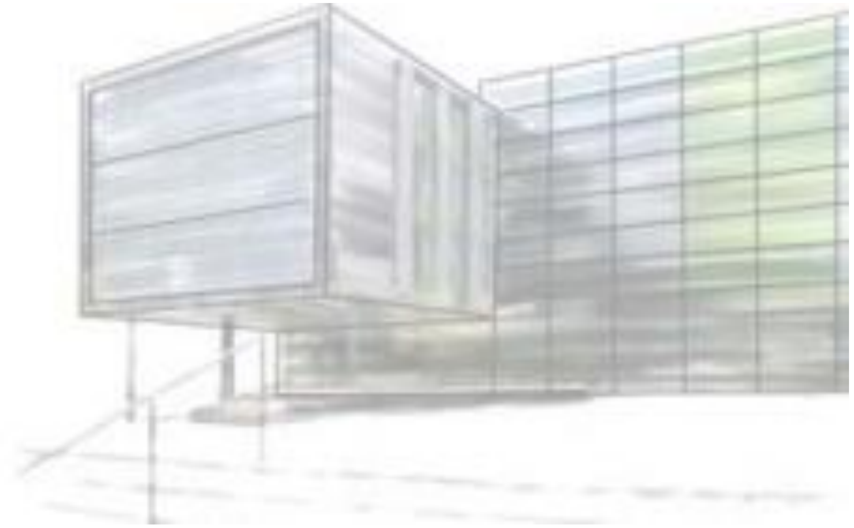
Class distribution for COCO train dataset.



Class distribution for VOC train dataset.

Outline

1. Background & Motivation
2. Fundamentals
3. Auto Annotation Workflow
   a) Stage 1: Image Detections
   b) Stage 2: Graph Construction
   c) Stage 3: Detection Combination
   d) Stage 4: Annotation Write Out
4. Evaluation
5. Example: RailEye 3D Annotation
6. Conclusion

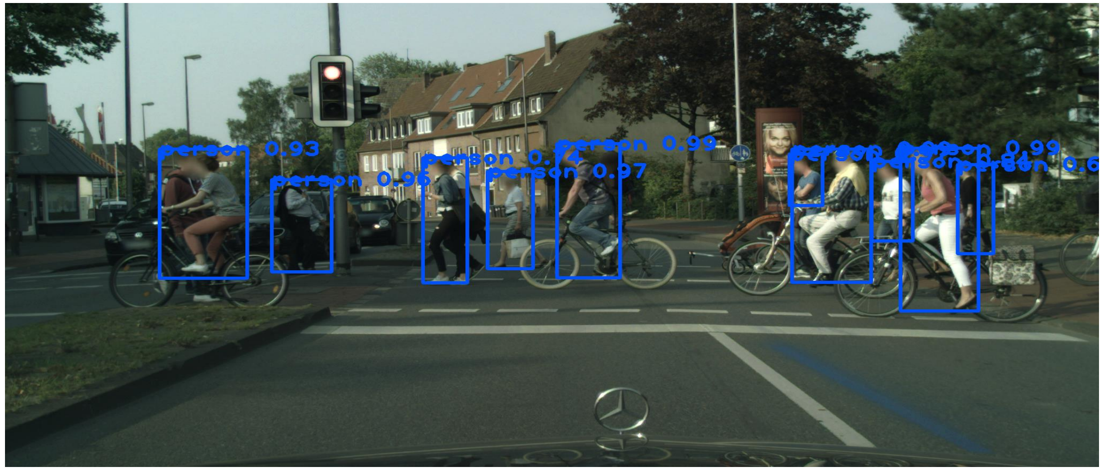


3. Auto Annotation Workflow

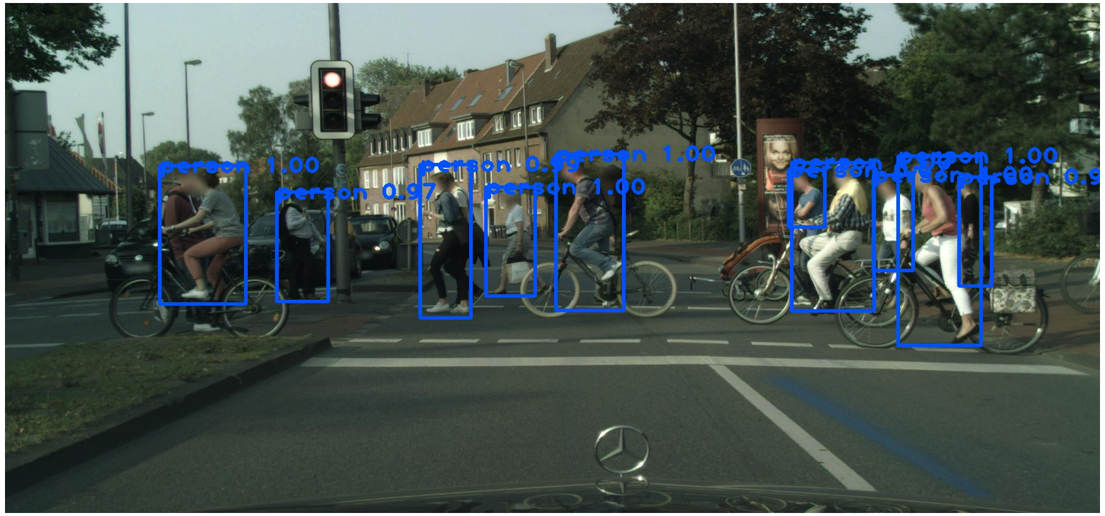
a) Stage 1: Image detection

- Task: Person annotation

1) Choose an image to annotate
2) Get detection from previously selected detectors
    - YOLOv3
    - YOLOv4
    - Mask-RCNN
3) Save bounding box coordinates and box scores



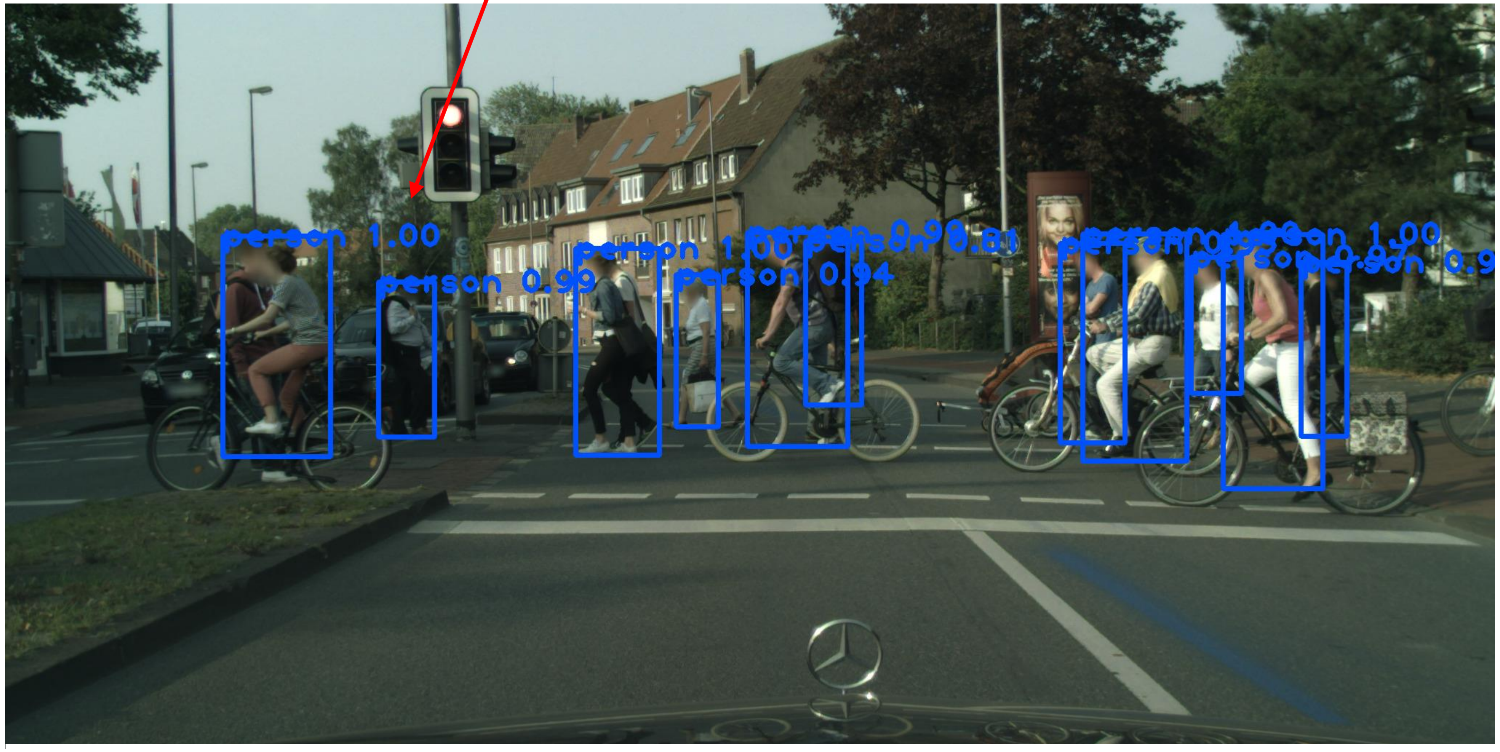
YOLOv4 detections.



YOLOv3 detections.

3. Auto Annotation Workflow

box score



Mask-RCNN detections.

a) Stage 1: Image detection

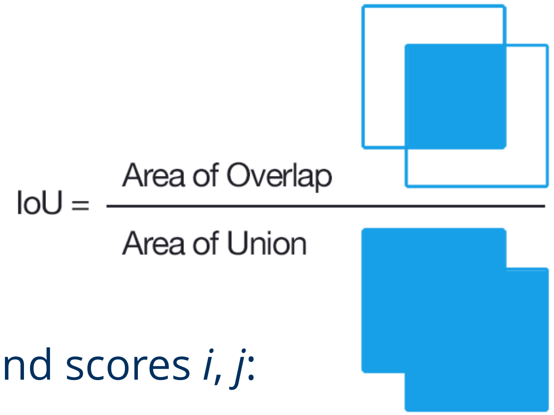
- Task: Person annotation

1) Choose an image to annotate
2) Get detection from previously selected detectors
   - YOLOv3
   - YOLOv4
   - Mask-RCNN
3) Save bounding box coordinates and **box scores**

3. Auto Annotation Workflow

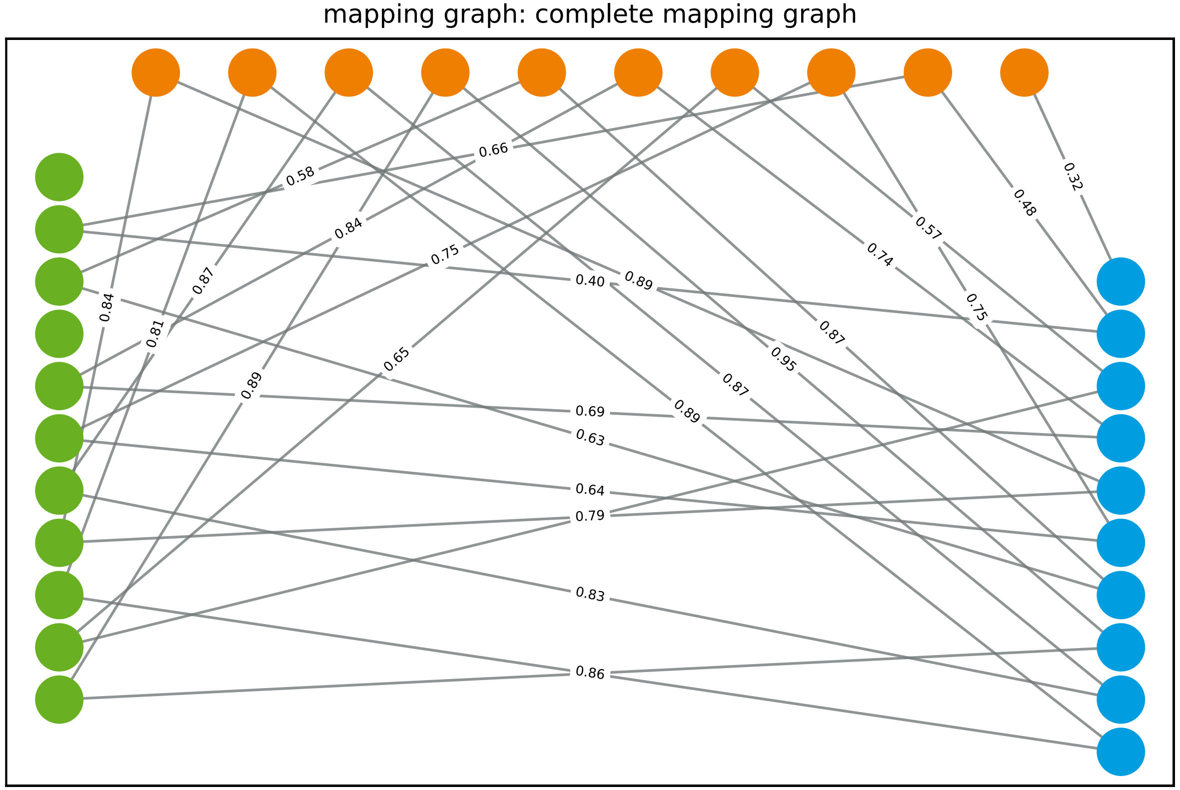
b) Stage 2: Graph Construction

1) Nodes: detected bounding boxes from **Stage 1**
   - YOLOv3
   - YOLOv4
   - Mask-RCNN
2) Edges:
   - Draw edge if intersection over union (IOU) is larger than an adjustable threshold T_{IOU}
   - Exclude nodes from same detector
   - E.g., $T_{IOU} > 0.5$



3) Edge weights:
   - Formula for boxes and scores *i*, *j*:

$$w_{i,j} = w_{j,i} = \mathrm{IOU}_{i,j} \cdot \mathrm{score}_i \cdot \mathrm{score}_j$$



Complete mapping graph.

3. Auto Annotation Workflow

c) Stage 3: Detection Combination

- Idea: Search for cycles and "lonely" nodes
  - **Three node cycle**
  - **Two node cycle**
  - **Lonely nodes**

1) Find three node cycle

2) Combine the three bounding box coordinates

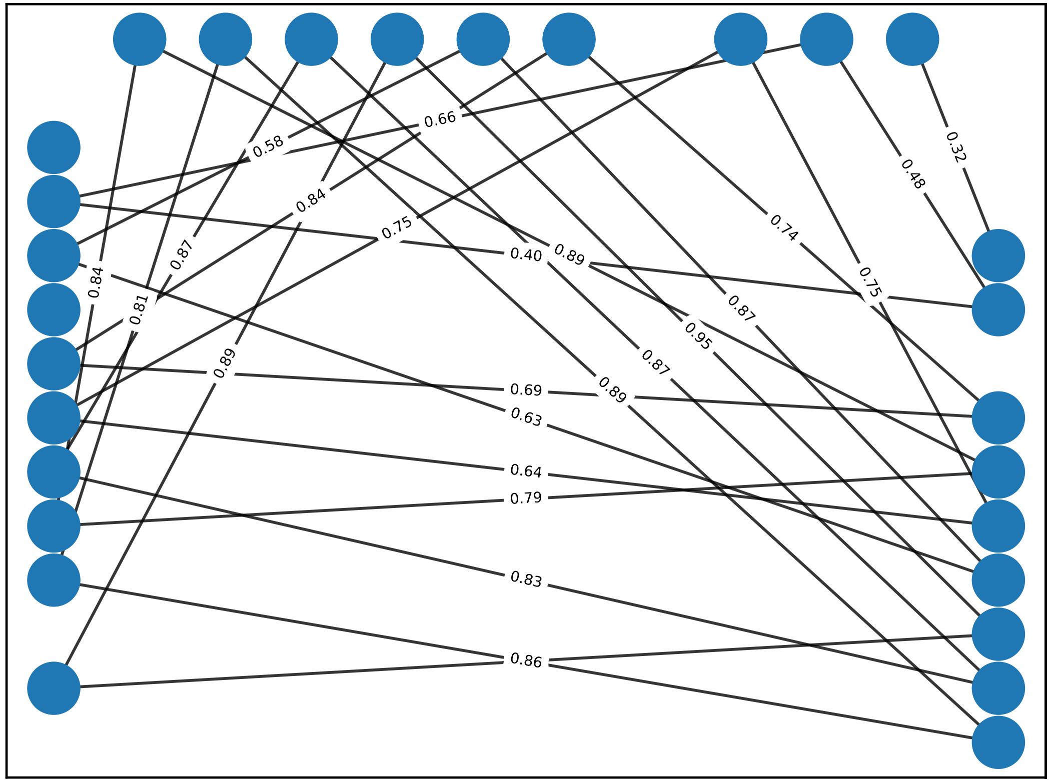
$$x_{\text{min,comb.}} = \text{mean}(x_{\text{min1}}, x_{\text{min2}}, x_{\text{min3}})$$

and analogously for y_{min}, x_{max}, y_{max}

Or: weighted mean with edge weights

3) Remove the nodes and edges of the three cycle

4) Go to 1) until all three node cycles are removed



iteration i=1

3. Auto Annotation Workflow

c) Stage 3: Detection Combination

- Idea: Search for cycles and "lonely" nodes
  - **Three node cycle**
  - **Two node cycle**
  - **Lonely nodes**

1) Find three node cycle

2) Combine the three bounding box coordinates

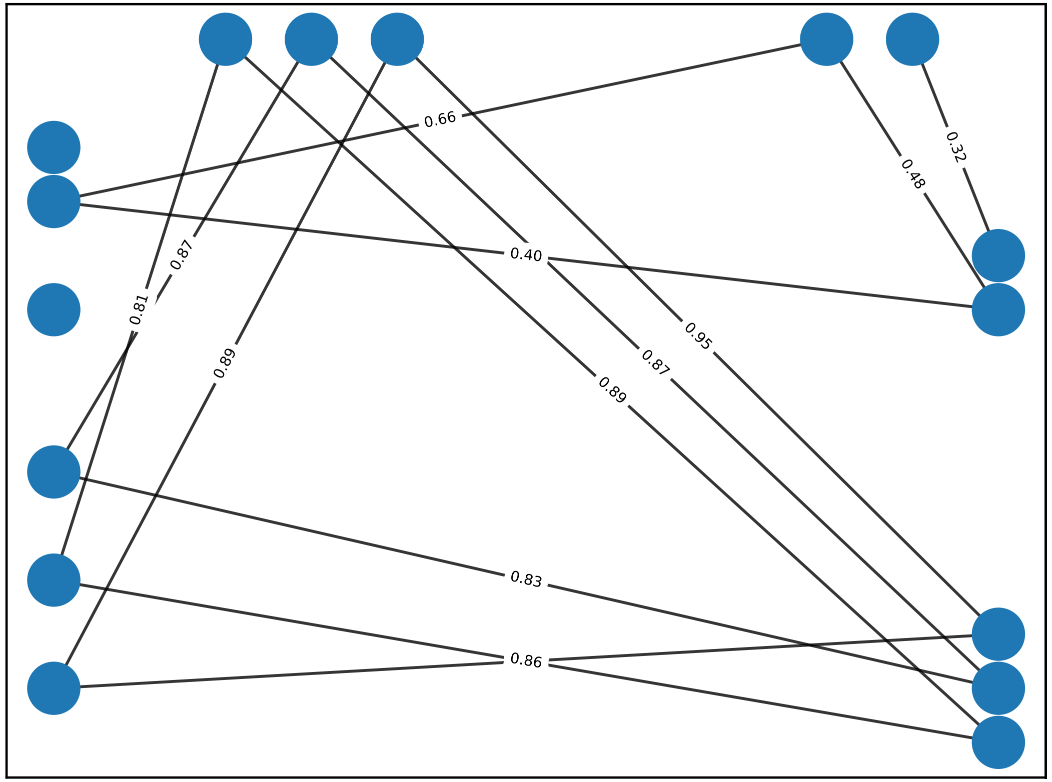
$$\mathrm{x}_{\mathrm{min,comb.}} = \mathrm{mean}(x_{\mathrm{min1}}, x_{\mathrm{min2}}, x_{\mathrm{min3}})$$

and analogously for y_{min}, x_{max}, y_{max}

Or: weighted mean with edge weights

3) Remove the nodes and edges of the three cycle

4) Go to 1) until all three node cycles are removed



iteration i=5

3. Auto Annotation Workflow

c) Stage 3: Detection Combination

- Idea: Search for cycles and “lonely” nodes
  - **Three node cycle**
  - **Two node cycle**
  - **Lonely nodes**

1) Find three node cycle

2) Combine the three bounding box coordinates

$$\mathrm{x}_{\mathrm{min,comb.}} = \mathrm{mean}(x_{\mathrm{min1}}, x_{\mathrm{min2}}, x_{\mathrm{min3}})$$

and analogously for y_{min}, x_{max}, y_{max}

Or: weighted mean with edge weights

3) Remove the nodes and edges of the three cycle

4) Go to 1) until all three node cycles are removed

iteration i=9

3. Auto Annotation Workflow

c) Stage 3: Detection Combination

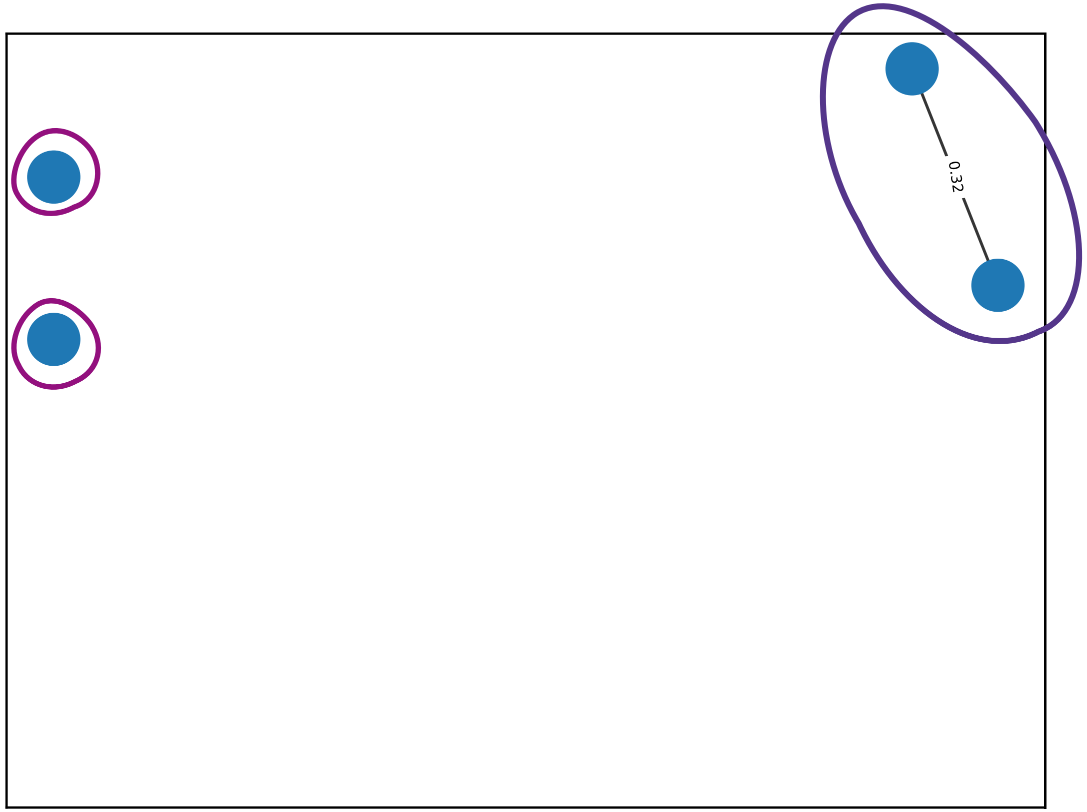
4) Go to 1) until all three node cycles are removed
5) Find **two node cycle**
6) Combine the three bounding box coordinates

$$\mathrm{x}_{\mathrm{min,comb.}} = \mathrm{mean}(x_{\mathrm{min1}}, x_{\mathrm{min2}})$$

   and analogously for y_{min}, x_{max}, y_{max}

   Or: weighted mean with edge weights

7) Go to 5) until all two node cycles are removed
8) Find the remaining **lonely nodes** and save their box coordinates as well

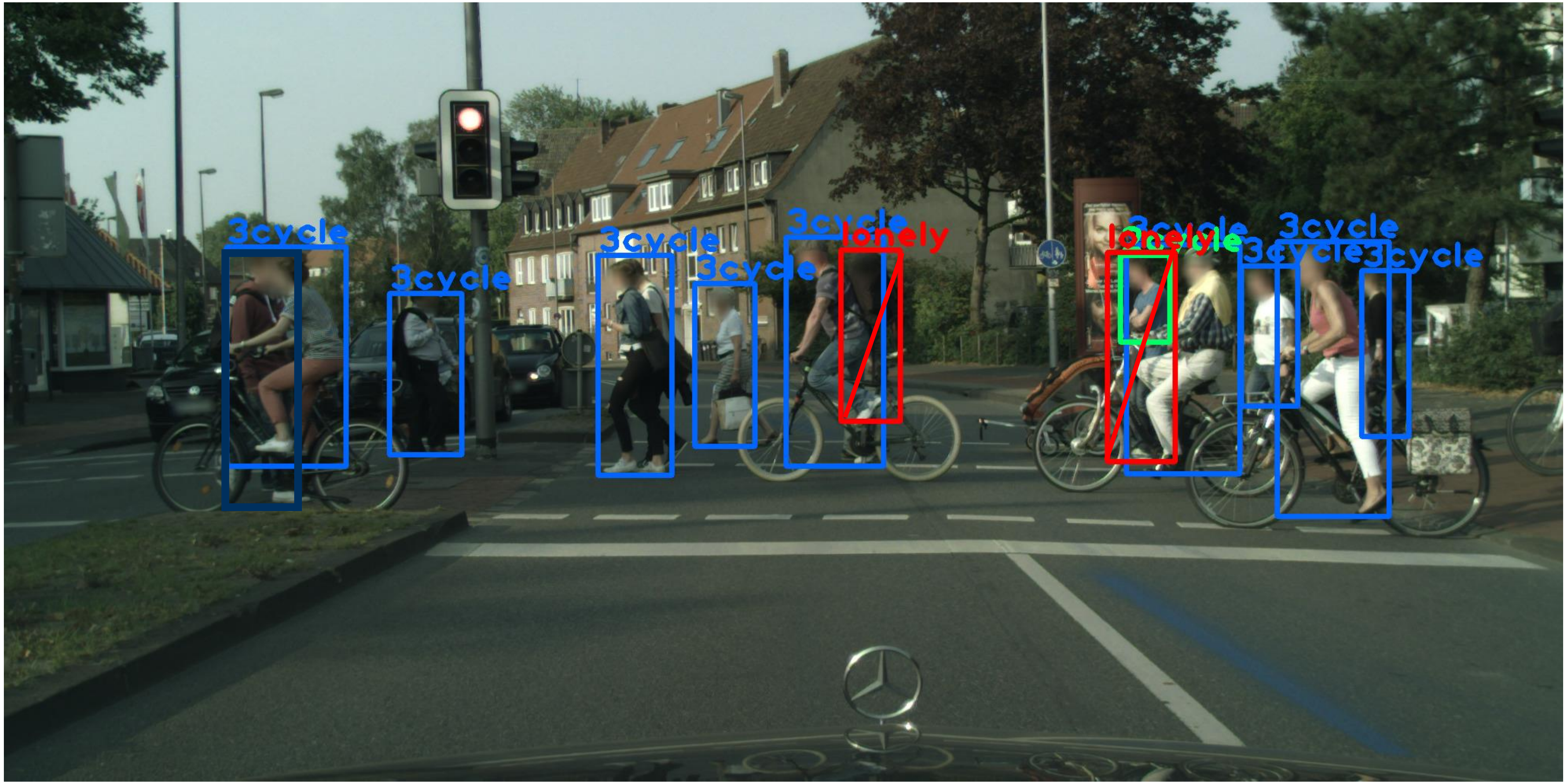


Remaining graph after three cycle node elimination

3. Auto Annotation Workflow

c) Stage 3: Detection Combination

- Result
- Adjust **missing bounding boxes**
- remove wrong **lonely boxes**



3. Auto Annotation Workflow

d) Stage 4: Annotation Write Out

```
<annotation>

        <filename>munster_000167_000019_leftImg8bit_blurred.jpg</filename>
        <path>D:\Data\example_images\</path>
        <source>
                <database>CityScapes</database>
        </source>
        <size>
                <width>2048</width>
                <height>1024</height>
                <depth>3</depth>
        </size>
        <object>
                <classes>
                        <name>person</name>
                </classes>
                <autoAnnotated>1</autoAnnotated>
                <level>3cycle</level>
                <bndbox>
                        <xmin>781</xmin>
                        <ymin>333</ymin>
                        <xmax>878</xmax>
                        <ymax>622</ymax>
                </bndbox>
        </object>

        ...
<annotation>
```

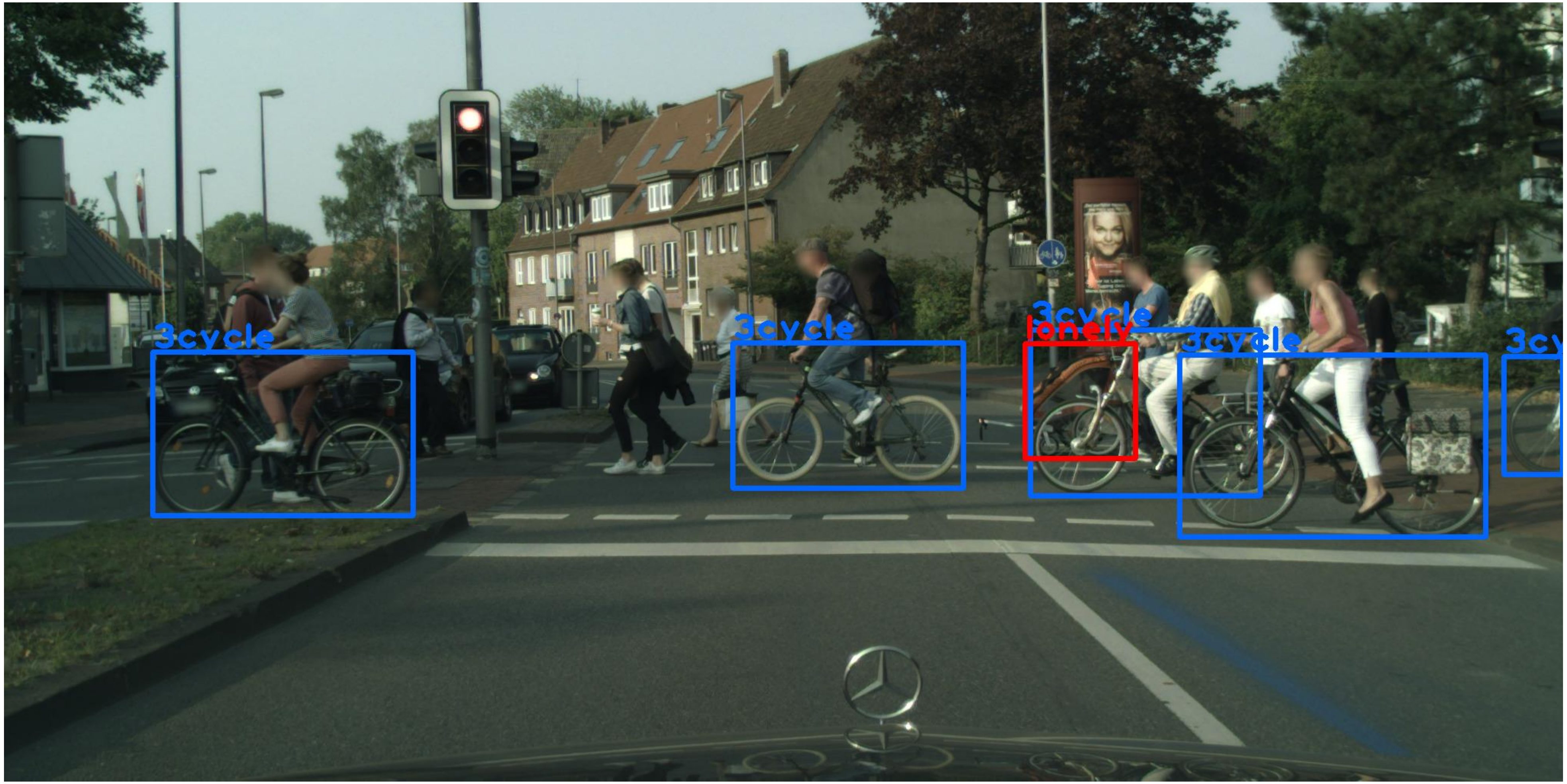
Hint the
auto annotation

Mark the bounding box
combination level

E.g., “3cycle” for a
three cycle combination

3. Auto Annotation Workflow

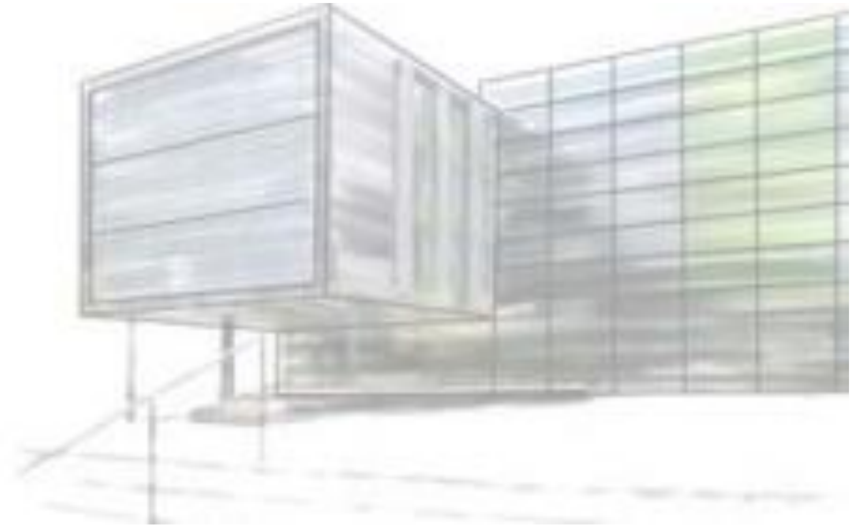
Bicycle auto annotation => same approach, just a different class



3. Auto Annotation Workflow

Traffic light
~~Bicycle~~ auto annotation => same approach, just a different class

Outline

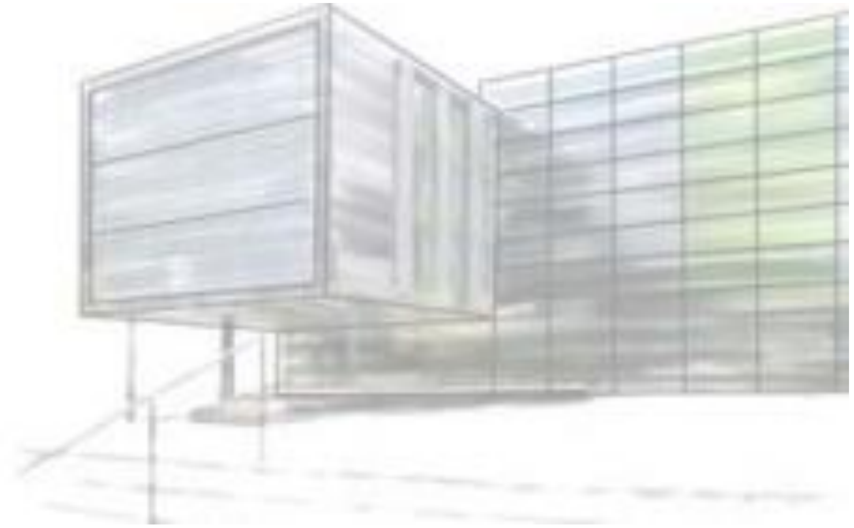


4. Evaluation

- Auto annotation of 500 images from CityScapes validation set
- Comparison with CityScapes annotations as the ground truth annotations
- Histogram of **correct**, **missing** and **wrong** auto annotations with respect to the bounding box image percentage
- Results:
  - **Missing** and **wrong** annotations less likely for large objects (relatively to image size)
  - Reflections and images can lead to **wrong** annotations
  - Generally, very high accuracy for objects larger than 0.5% of the image size

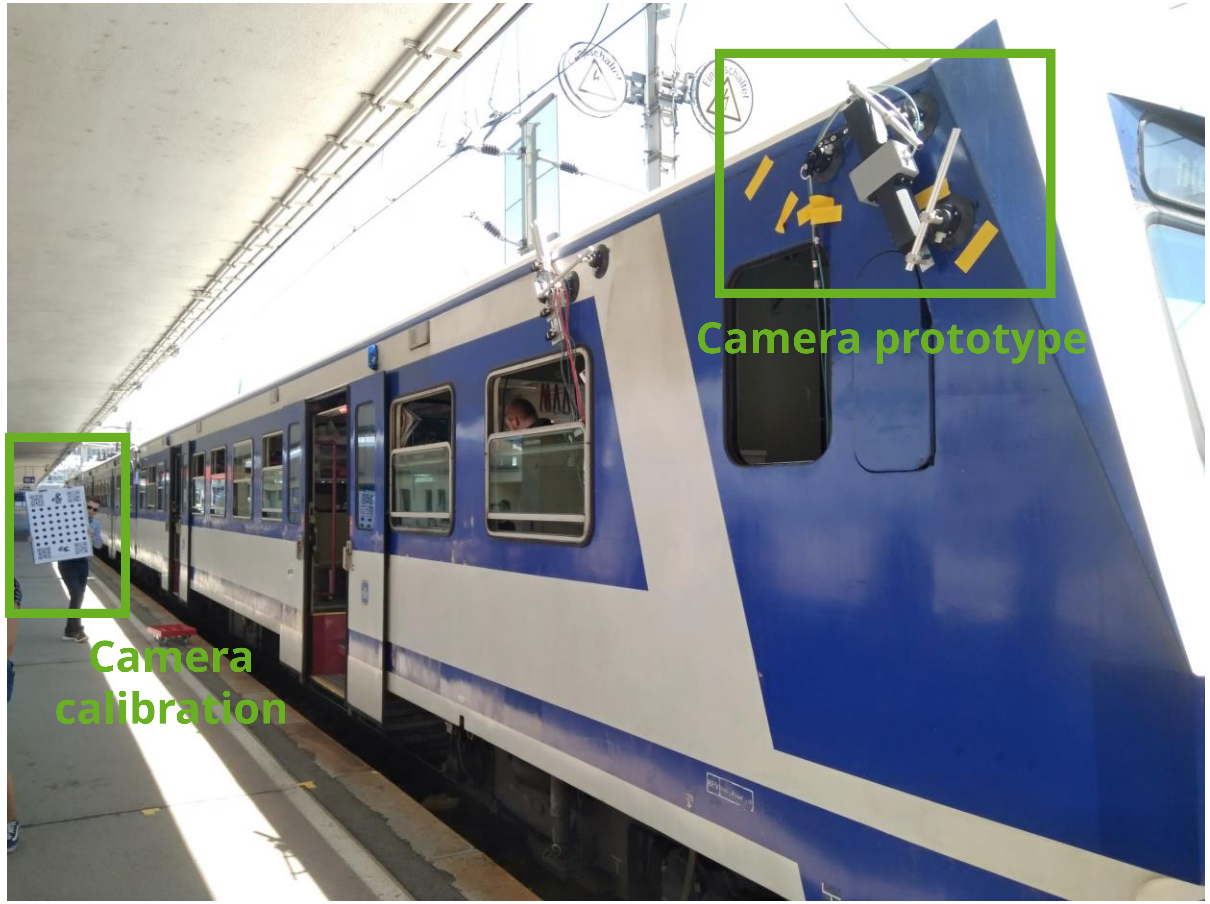
Outline

1. Background & Motivation
2. Fundamentals
3. Auto Annotation Workflow
   a) Stage 1: Image Detections
   b) Stage 2: Graph Construction
   c) Stage 3: Detection Combination
   d) Stage 4: Annotation Write Out
4. Evaluation
5. Example: RailEye 3D 3D Annotation
6. Conclusion



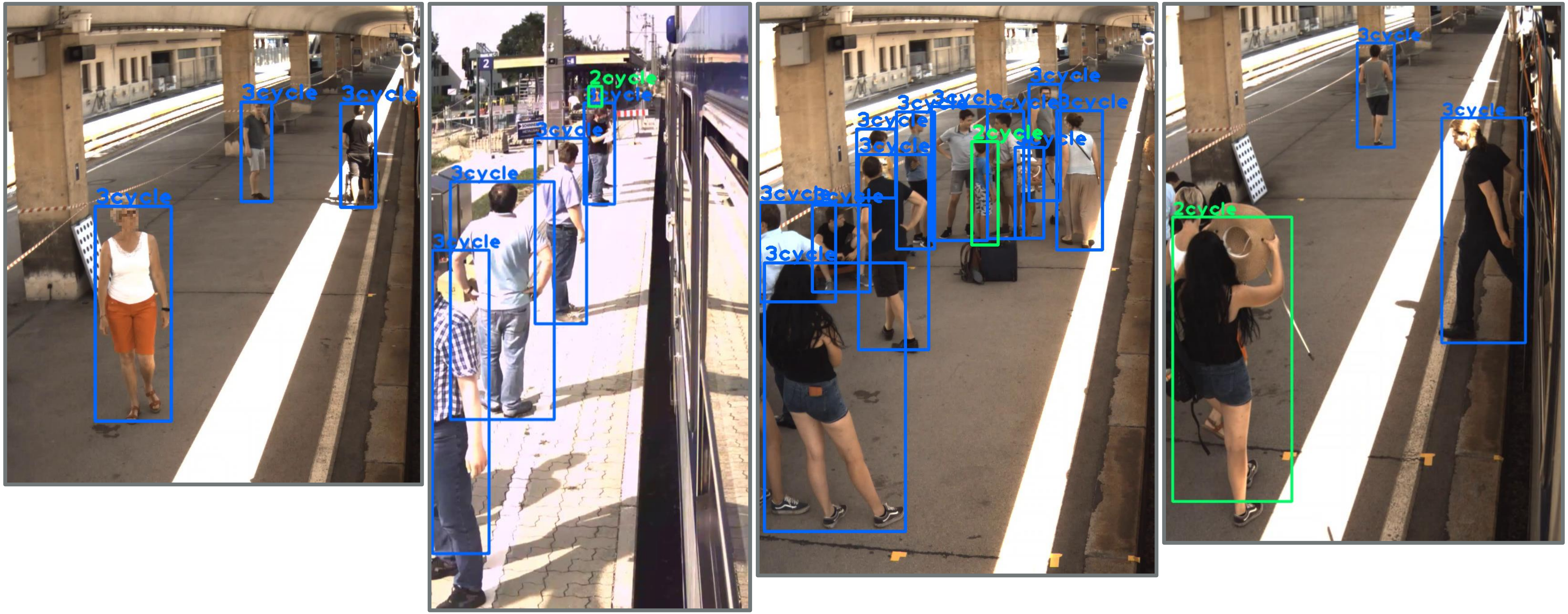
4. Example: RailEye 3D Annotation

- Developed by EYYES GmbH and promoted by the Austrian Research Promotion Agency: https://projekte.ffg.at/projekt/3200200
- Semi-automatic mechanism to enhance the public transport safety
- Surveillance of station platforms at rail stations
  - Monitoring of entrance and exit processes
  - Assurance that no persons are in dangerous areas when the train exits the station (E.g., behind the safety line)
- Realized by customized object detectors with person detection
- Annotated data with respect to the train camera perspective is required
  - ➢ Perfect use-case for the auto annotation
  - ➢ Manual adjustments if needed

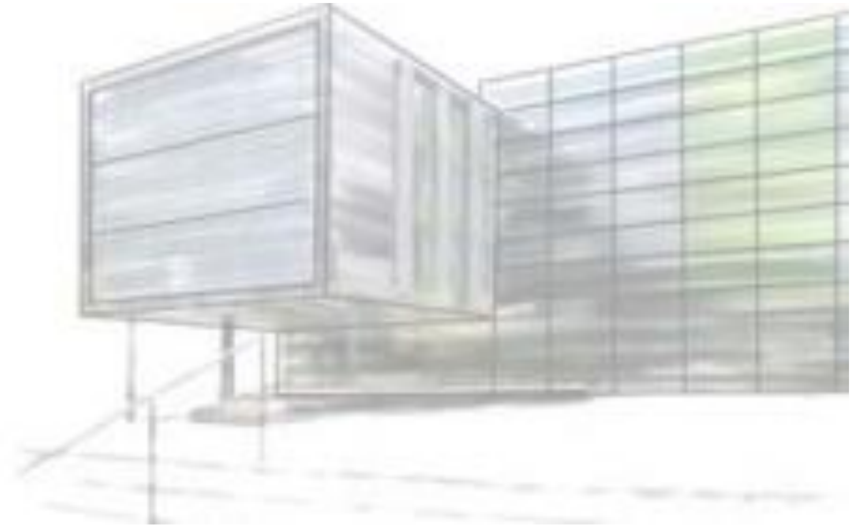


4. Example: RailEye 3D Annotation

Auto annotations



Outline



5. Conclusion

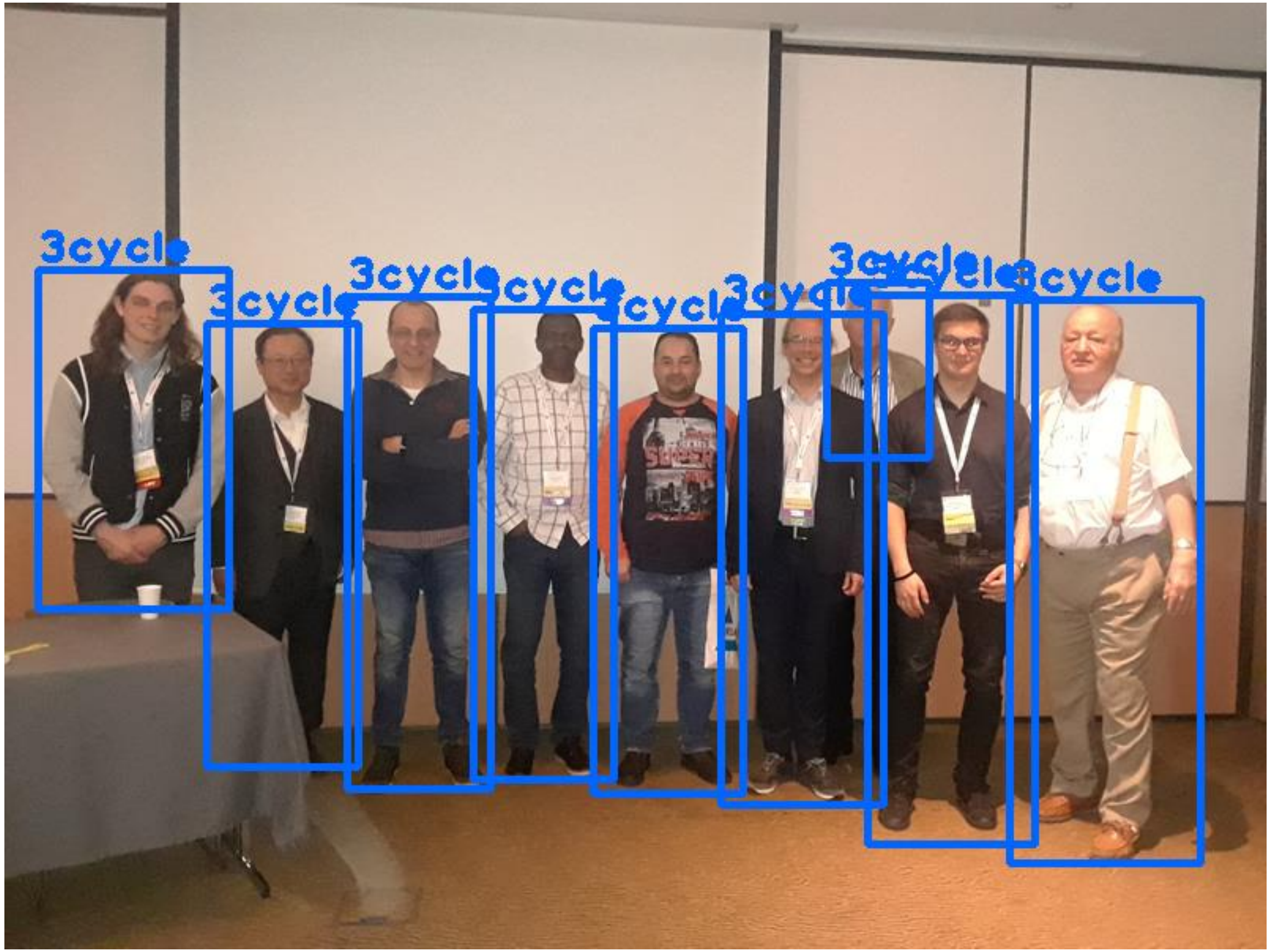
Current state

- Auto annotation workflow based on high quality and state of the art object detectors
- Graph-based combination of the bounding boxes of each detector
- Different levels of the bounding boxes based on the cycle length

Future work

- Further tests and development of a method to estimate the quality of the auto annotation
- Expanding the detector set with other detectors
- Additional methods to automatically check lonely nodes
- Reinforcement training with respect to the bounding box level
  (E.g., lesser loss penalty for lower-level bounding boxes)

Thank you for your attention



Auto annotation of the final photo of

“The Sixth International Conference on Big Data, Small Data, Linked Data and Open Data”

ALLDATA 2020
February 23, 2020 to February 27, 2020 - Lisbon, Portugal

References

(1) M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler et al., "*The Cityscapes Dataset for Semantic Urban Scene Understanding,*" arXiv: 1604.01685, 2016.

(2) Z. Zou, Z. Shi, Y.Guo and J. Ye, "*Object Detection in 20 Years: A Survey,*" arXiv: 1905.05055, 2019.

(3) P. Viola and M. Jones, "*Rapid object detection using a boosted cascade of simple features,*" doi: 10.1109/CVPR.2001.990517, 2001.

(4) P. Felzenszwalb, D. McAllester and D. Ramanan, "*A discriminatively trained, multiscale, deformable part model,*" doi: 10.1109/CVPR.2008.4587597, 2008.

(5) J. Redmon and A. Angelova, "*Realtime grasp detection using convolutional neural networks,*" IEEE International Conference on Robotics and Automation (ICRA), pp. 1316–1322, 2015.

(6) J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "*You Only Look Once: Unified, realtime object detection,*" IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788, 2016.

(7) T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "*Focal loss for dense object detection,*" IEEE transactions on pattern analysis and machine intelligence, 2018.

(8) A. Bochkovskiy, C.-Y. Wang and H.-Y M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv: 2004.10934, 2020.

References

(9) R. Girshick, J. Donahue, T. Darrell, and J. Malik, "*Rich feature hierarchies for accurate object detection and semantic segmentation,*" Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587, 2014.

(10) S. Ren, K. He, R. Girshick and J. Sun, "*Faster r-cnn: Towards real-time object detection with region proposal networks,*" in Advances in neural information processing systems, pp. 91–99, 2015.

(11) K. He and G. Gkioxari and P. Dollár and R. Girshick, "*Mask R-CNN,*" arXiv: 1703.06870, 2018.

(12) T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, et al., "*Microsoft coco: Common objects in context*," European conference on computer vision. Springer, pp. 740–755, 2014.

(13) M. Everingham, L. Van Gool, C. K. 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, 2010.

(14) P. Skalski, "Make Sense," GitHub: https://github.com/SkalskiP/make-sense/, 2019. website: https://www.makesense.ai/, retrieved: 02.2021.

(15) CV Gluon AI, website: https://cv.gluon.ai/model_zoo/detection.html#yolo-v3, retrieved: 02.2021

(16) A. Bochkovskiy, GitHub: https://github.com/AlexeyAB/darknet, retrieved: 02.2021

(17) W. Abdulla, "*Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow,*" GitHub: https://github.com/matterport/Mask_RCNN, retrieved: 02.2021