PARIS
LODRON
UNIVERSITÄT
SALZBURG

Object Detection

Wissenschaftliche Arbeitstechniken und Präsentation

Aeri Julian, Berger Simon, Buchinger Paul, Palma Andrè

12. Januar 2024

Paris Lodron Universität Salzburg

Inhaltsangabe

1. Einführung

2. YOLO

3. Verwendung

4. Beispiele

Inhaltsangabe

1. Einführung

2. YOLO

3. Verwendung

4. Beispiele

Object Detection und Object Recognition

Object Detection

- **kontrolliert** ob ein gegebenes Objekt überhaupt vorhanden ist im Bild [1]

Object Recognition

[1] Object detection and recognition in digital images, *B. Cyganek* (2013) [1]

Object Detection und Object Recognition

Object Detection

- **kontrolliert** ob ein gegebenes Objekt überhaupt vorhanden ist im Bild [1]
- Position und das Aussehen (Farbe, Größe, ...) spielen hierbei auch eine große Rolle

Object Recognition

[1] Object detection and recognition in digital images, *B. Cyganek* (2013) [1]

Object Detection und Object Recognition

Object Detection

- **kontrolliert** ob ein gegebenes Objekt überhaupt vorhanden ist im Bild [1]
- Position und das Aussehen (Farbe, Größe, ...) spielen hierbei auch eine große Rolle
- auch von mehreren Objekten gleichzeitig

Object Recognition

[1] Object detection and recognition in digital images, *B. Cyganek* (2013) [1]

Object Detection und Object Recognition

Object Detection

- **kontrolliert** ob ein gegebenes Objekt überhaupt vorhanden ist im Bild [1]
- Position und das Aussehen (Farbe, Größe, ...) spielen hierbei auch eine große Rolle
- auch von mehreren Objekten gleichzeitig

Object Recognition

- dient zum **identifiziert** und **klassifiziert** von erkannten Objekten

[1] Object detection and recognition in digital images, *B. Cyganek* (2013) [1]

Object Detection und Object Recognition

Object Detection

- **kontrolliert** ob ein gegebenes Objekt überhaupt vorhanden ist im Bild [1]
- Position und das Aussehen (Farbe, Größe, ...) spielen hierbei auch eine große Rolle
- auch von mehreren Objekten gleichzeitig

Object Recognition

- dient zum **identifiziert** und **klassifiziert** von erkannten Objekten
- kann auch verwendet werden um bestimmte Personen, oder Straßenschilder zu erkennen [1]

[1] Object detection and recognition in digital images, *B. Cyganek* (2013) [1]

Object Detection und Object Recognition

Object Detection

- **kontrolliert** ob ein gegebenes Objekt überhaupt vorhanden ist im Bild [1]
- Position und das Aussehen (Farbe, Größe, ...) spielen hierbei auch eine große Rolle
- auch von mehreren Objekten gleichzeitig

Object Recognition

- dient zum **identifiziert** und **klassifiziert** von erkannten Objekten
- kann auch verwendet werden um bestimmte Personen, oder Straßenschilder zu erkennen [1]

Beide Dinge kann der Mensch von Natur aus, um beispielsweise Gefahren zu erkennen und genauer zu identifizieren, oder Menschen wieder zu erkennen.

[1] Object detection and recognition in digital images, *B. Cyganek* (2013) [1]

Geschichte: Grundgedanken

Grundgedanken

- Berechnungsmodellen und Techniken für die Bestimmung, wo welche Objekte sind [2]

[2] Object Detection in 20 Years, *Z. Zou, K. Chen, Z. Shi, Y. Guo, J. Ye* (2023)[8]
[3] Computer Vision, *G. Stockman and L. G. Shapiro* (2001)[5]

Geschichte: Grundgedanken

Grundgedanken

- Berechnungsmodellen und Techniken für die Bestimmung, wo welche Objekte sind [2]
- Genauigkeit und Geschwindigkeit

[2] Object Detection in 20 Years, *Z. Zou, K. Chen, Z. Shi, Y. Guo, J. Ye* (2023)[8]
[3] Computer Vision, *G. Stockman and L. G. Shapiro* (2001)[5]

Geschichte: Grundgedanken

Grundgedanken

- Berechnungsmodellen und Techniken für die Bestimmung, wo welche Objekte sind [2]
- Genauigkeit und Geschwindigkeit
- Grundlage für viele andere Bereiche in der Computer Vision

[2] Object Detection in 20 Years, *Z. Zou, K. Chen, Z. Shi, Y. Guo, J. Ye* (2023)[8]
[3] Computer Vision, *G. Stockman and L. G. Shapiro* (2001)[5]

Geschichte: Grundgedanken

Grundgedanken

- Berechnungsmodellen und Techniken für die Bestimmung, wo welche Objekte sind [2]
- Genauigkeit und Geschwindigkeit
- Grundlage für viele andere Bereiche in der Computer Vision
- Automatisches Extrahieren von Informationen [3]

[2] Object Detection in 20 Years, *Z. Zou, K. Chen, Z. Shi, Y. Guo, J. Ye* (2023)[8]
[3] Computer Vision, *G. Stockman and L. G. Shapiro* (2001)[5]

Geschichte: 1990 bis Heute

1990er bis heute

- **1990 - 2014**: die *"traditional object detection period"*

Geschichte: 1990 bis Heute

1990er bis heute

- **1990 - 2014**: die *"traditional object detection period"*
- **2014 - heute**: die *"deep learning-based detection period"*

Geschichte: 1990 bis Heute

1990er bis heute

- **1990 - 2014**: die *"traditional object detection period"*
- **2014 - heute**: die *"deep learning-based detection period"*
- Entwicklung von Deep-Learning-Techniken trieb den Fortschritt bei der Objekterkennung sehr stark an

Geschichte: 1990 bis Heute

1990er bis heute

- **1990 - 2014**: die *"traditional object detection period"*
- **2014 - heute**: die *"deep learning-based detection period"*
- Entwicklung von Deep-Learning-Techniken trieb den Fortschritt bei der Objekterkennung sehr stark an

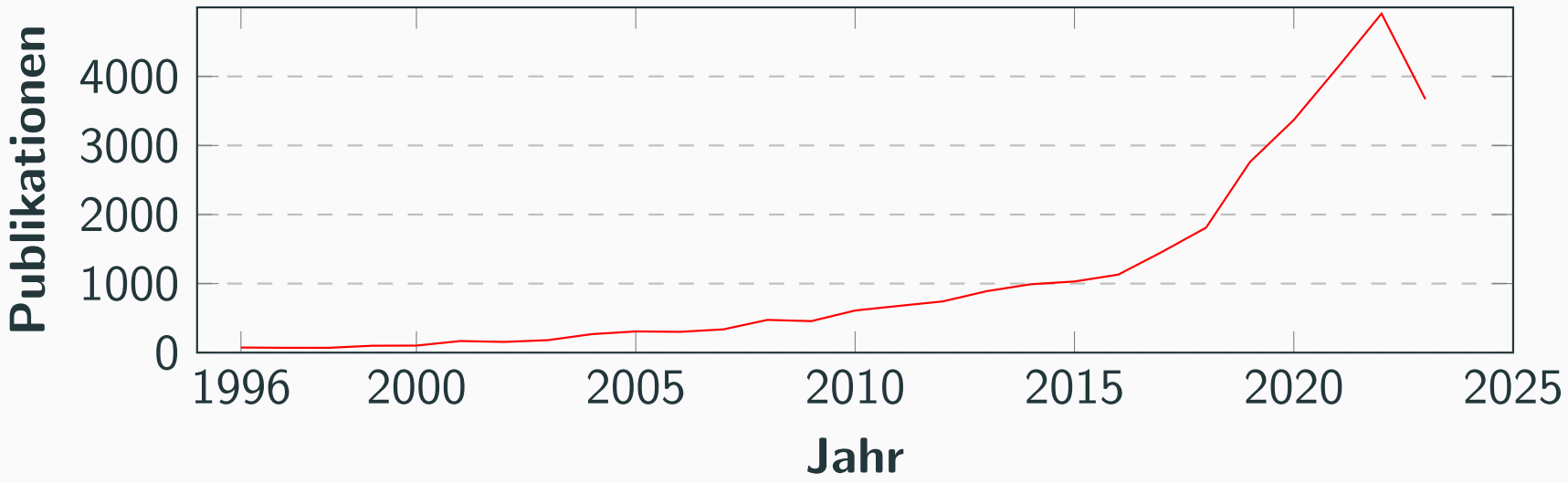


Abbildung 1: Darstellung der Publikationen laut *Google Scholar* im Bereich Object Detection, oder Detecting Objects pro Jahr

Meilensteine von 1990 - 2014

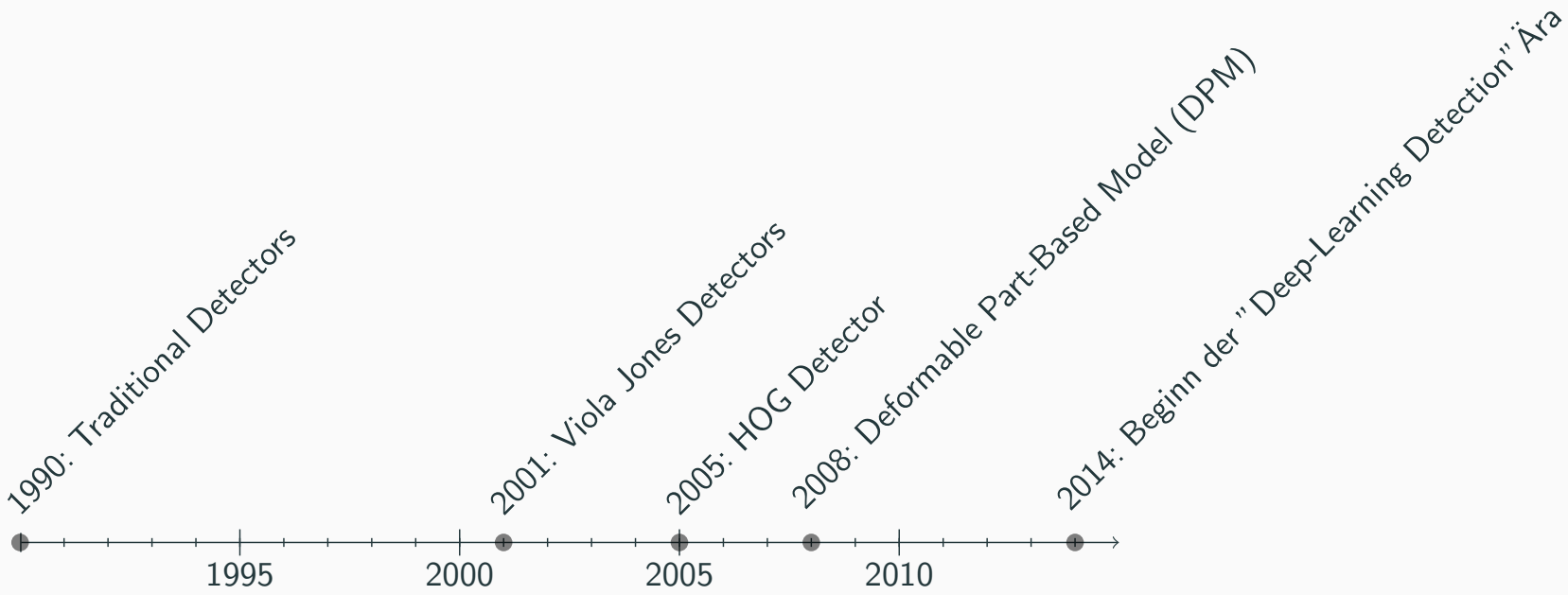


Abbildung 2: Zeitleiste von 1990 bis 2014, von den Entwicklungen im Bereich der Object Detection während der "traditionellen" Ära [4]

[4] Object Detection in 20 Years, *Z. Zou, K. Chen, Z. Shi, Y. Guo, J. Ye* (2023)[8]

Viola–Jones Algorithmus

Geschichte

- 2001 von *Paul Viola* und *Michael J. Jones* vorgestellt

[5] Robust real-time face detection, *Viola & Jones* (2004)[6]
[6] Experiments with a new boosting algorithm, *Y. Freund, R. E. Schapire, et al* [3]
[7] Rapid object detection using a boosted cascade of simple features, *Viola & Jones* [7]

Viola–Jones Algorithmus

Geschichte

- 2001 von *Paul Viola* und *Michael J. Jones* vorgestellt
- schnellen und robusten Visual Detection Algorithmus

[5] Robust real-time face detection, *Viola & Jones* (2004)[6]
[6] Experiments with a new boosting algorithm, *Y. Freund, R. E. Schapire, et al* [3]
[7] Rapid object detection using a boosted cascade of simple features, *Viola & Jones* [7]

Viola–Jones Algorithmus

Geschichte

- 2001 von *Paul Viola* und *Michael J. Jones* vorgestellt
- schnellen und robusten Visual Detection Algorithmus
- erster Algorithmus, der es schaffte, Gesichter schnell und akkurat zu erkennen[5]

[5] Robust real-time face detection, *Viola & Jones* (2004)[6]
[6] Experiments with a new boosting algorithm, *Y. Freund, R. E. Schapire, et al* [3]
[7] Rapid object detection using a boosted cascade of simple features, *Viola & Jones* [7]

Viola–Jones Algorithmus

Geschichte

- 2001 von *Paul Viola* und *Michael J. Jones* vorgestellt
- schnellen und robusten Visual Detection Algorithmus
- erster Algorithmus, der es schaffte, Gesichter schnell und akkurat zu erkennen[5]

Funktionsweise

- Operiert auf einem 386 x 288 Pixelbild

[5] Robust real-time face detection, *Viola & Jones* (2004)[6]
[6] Experiments with a new boosting algorithm, *Y. Freund, R. E. Schapire, et al* [3]
[7] Rapid object detection using a boosted cascade of simple features, *Viola & Jones* [7]

Viola–Jones Algorithmus

Geschichte

- 2001 von *Paul Viola* und *Michael J. Jones* vorgestellt
- schnellen und robusten Visual Detection Algorithmus
- erster Algorithmus, der es schaffte, Gesichter schnell und akkurat zu erkennen[5]

Funktionsweise

- Operiert auf einem 386 x 288 Pixelbild
- 3 Hauptmerkmale: *integral image*, *Feature - Auswahl mit AdaBoost*[6]und ein *kaskadierten Detektor*

[5]Robust real-time face detection, *Viola & Jones* (2004)[6]
[6]Experiments with a new boosting algorithm, *Y. Freund, R. E. Schapire, et al* [3]
[7]Rapid object detection using a boosted cascade of simple features, *Viola & Jones* [7]

Viola–Jones Algorithmus

Geschichte

- 2001 von *Paul Viola* und *Michael J. Jones* vorgestellt
- schnellen und robusten Visual Detection Algorithmus
- erster Algorithmus, der es schaffte, Gesichter schnell und akkurat zu erkennen[5]

Funktionsweise

- Operiert auf einem 386 x 288 Pixelbild
- 3 Hauptmerkmale: *integral image*, *Feature - Auswahl mit AdaBoost*[6] und ein *kaskadierten Detektor*
- kontrolliert die Helligkeitsunterschiede zwischen benachbarten Regionen, auf rechteckige Bereiche auf einem Bild[7]

[5] Robust real-time face detection, *Viola & Jones* (2004)[6]
[6] Experiments with a new boosting algorithm, *Y. Freund, R. E. Schapire, et al* [3]
[7] Rapid object detection using a boosted cascade of simple features, *Viola & Jones* [7]

Histograms of Oriented Gradients (HOG)

- Wurde 2005 von Navneet Dalal und Bill Triggs vorgestellt

Histograms of Oriented Gradients (HOG)

- Wurde 2005 von Navneet Dalal und Bill Triggs vorgestellt
- Bildmerkmalsextraktionstechnik in Bildverarbeitung und Computer Vision.

Histograms of Oriented Gradients (HOG)

- Wurde 2005 von Navneet Dalal und Bill Triggs vorgestellt
- Bildmerkmalsextraktionstechnik in Bildverarbeitung und Computer Vision.
- **Konzept:** Beschreibt Objekte durch Verteilung von Gradientenrichtungen.

Histograms of Oriented Gradients (HOG)

- Wurde 2005 von Navneet Dalal und Bill Triggs vorgestellt
- Bildmerkmalsextraktionstechnik in Bildverarbeitung und Computer Vision.
- **Konzept:** Beschreibt Objekte durch Verteilung von Gradientenrichtungen.
- **Vorteile:** Effektiv bei starrer Objektform.
- **Grenzen:** Empfindlich gegenüber Rotation, Skalierung, variable Form.

Visualisierung von Histogram of Oriented Gradients

Abbildung 3: Links: Originalbild. Rechts: Visualisierung mit HOG. Bildquelle: Sandro Halank, Wikimedia Commons, CC BY-SA 3.0.[8] [2]

[8] Verfügbar unter: `https://commons.wikimedia.org/w/index.php?curid=67282779`

Objekterkennungs Methoden mit CNN

CNN

- **Convolutional Neural Network** (=faltendes neuronales Netzwerk)

Objekterkennungs Methoden mit CNN

CNN

- **Convolutional Neural Network** (=faltendes neuronales Netzwerk)

R-CNN

- **Prinzip:** Lokalisierung potenzieller Objekte durch Regionenvorschläge.
- **Methode:** Einzelne Analyse jeder Region mit CNN.

Objekterkennungs Methoden mit CNN

CNN

- **Convolutional Neural Network** (=faltendes neuronales Netzwerk)

R-CNN

- **Prinzip:** Lokalisierung potenzieller Objekte durch Regionenvorschläge.
- **Methode:** Einzelne Analyse jeder Region mit CNN.

Fast R-CNN

- **Prinzip:** Schnellere R-CNN-Version mit integriertem CNN für Regionenvorschläge.
- **Methode:** Ganzbildanalyse und Regionenklassifizierung mit CNN.

Weiterentwicklungen und Zukunft

Mask R-CNN

- **Prinzip:** Erweiterung von Faster R-CNN für Bounding Boxes und Pixelmasken.
- **Methode:** Erzeugung von Bounding Boxes und detaillierten Masken zur präziseren Erkennung.

Weiterentwicklungen und Zukunft

Mask R-CNN

- **Prinzip:** Erweiterung von Faster R-CNN für Bounding Boxes und Pixelmasken.
- **Methode:** Erzeugung von Bounding Boxes und detaillierten Masken zur präziseren Erkennung.

SSD(Single Shot MultiBox Detector)

- Objekterkennung in einem Durchgang (Single Shot).
- Gleichzeitige Vorhersage mehrerer Bounding Boxes und Klassenwahrscheinlichkeiten.
- Basiert auf einem modifizierten Convolutional Neural Network (CNN).

Inhaltsangabe

1. Einführung

2. YOLO

3. Verwendung

4. Beispiele

YOLO

YOLO (You Only Look Once)

- populäres Modell zur Objekterkennung
- einmaliges Ausführen von einem einzelnen Neuronalen Netzwerks auf das Bild
    - dadurch sehr schnell, Bilder können in Echtzeit (45 FPS) bearbeitet werden

Entwickelt von einem Team rund um Joseph Redmon (University of Washington).

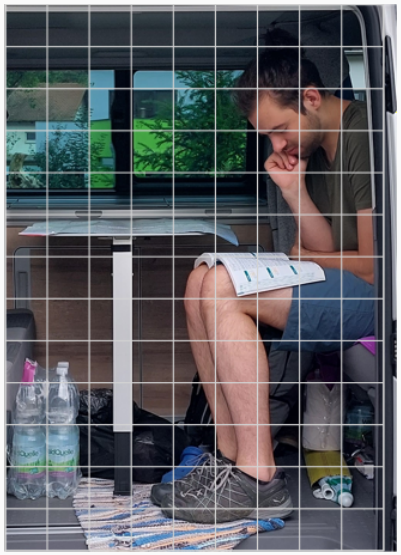
Funktionsweise YOLO

Abbildung 4: Grobe Funktionsweise von YOLO[9]

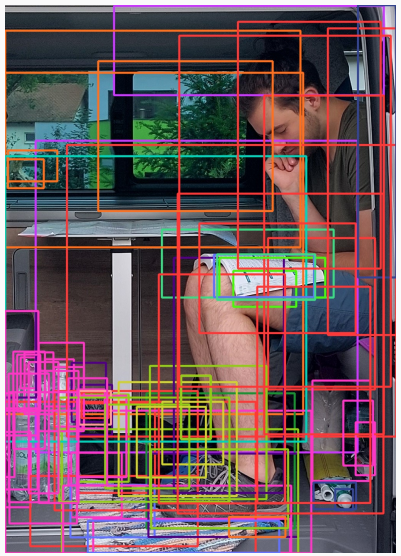
1. Größe des Input Fotos verkleinern.
2. Ausführen von einem einzelnen Neuronalen Netzwerks auf das Bild
3. Filtern der erkannten Objekte

[9] You only look once: Unified, real-time object detection, *Redmon et al.* (2016) [4]

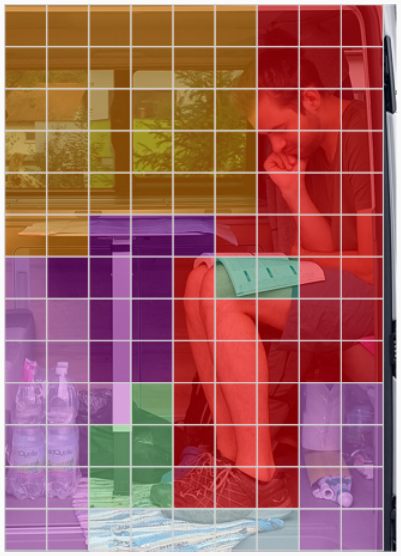
Modell



(a) grid



(b) bounding boxes



(c) probabilty map

(d) output

Funktionsweise YOLO II

Jede Bounding Box besitzt 5 Kennwerte:

- **(x,y)** - x,y-Koordinaten des Mittelpunkts, relativ zur Zelle an.
- **w** - Breite
- **h** - Höhe
- **Confidence** - gibt an wie Sicher sich das Modell ist, dass eine gegebene Bounding Box ein Objekt enthält

Jede Zelle, welche ein potenzielles Objekt enthält, berechnet die Klassenwahrscheinlichkeiten.

Aus der Klassenwahrscheinlichkeit einer Zelle, und der Confidence-Werte der Bounding Boxes, werden die klassenspezifischen Confidence-Werte der Bounding Boxes ermittelt.

Funktionsweise YOLO III

Abbildung 6: vorl. Ergebnis

Funktionsweise YOLO III

Abbildung 6: vorl. Ergebnis

Non-Max-Suppression:

- herausfiltern aller Bounding Boxes über einem gewissen 'Confidence Schwellenwert'
- solange noch mind. eine Box in der Input Menge enthalten ist:
  - füge die Box b, mit der höchsten Confidence, zum Ergebnis hinzu (und entferne sie aus dem Input)
    - überprüfe für jeden andere Box r: wenn $IOU_r^b \geq$ 'IOU Schwellenwert', entferne die Box r

Funktionsweise YOLO IV

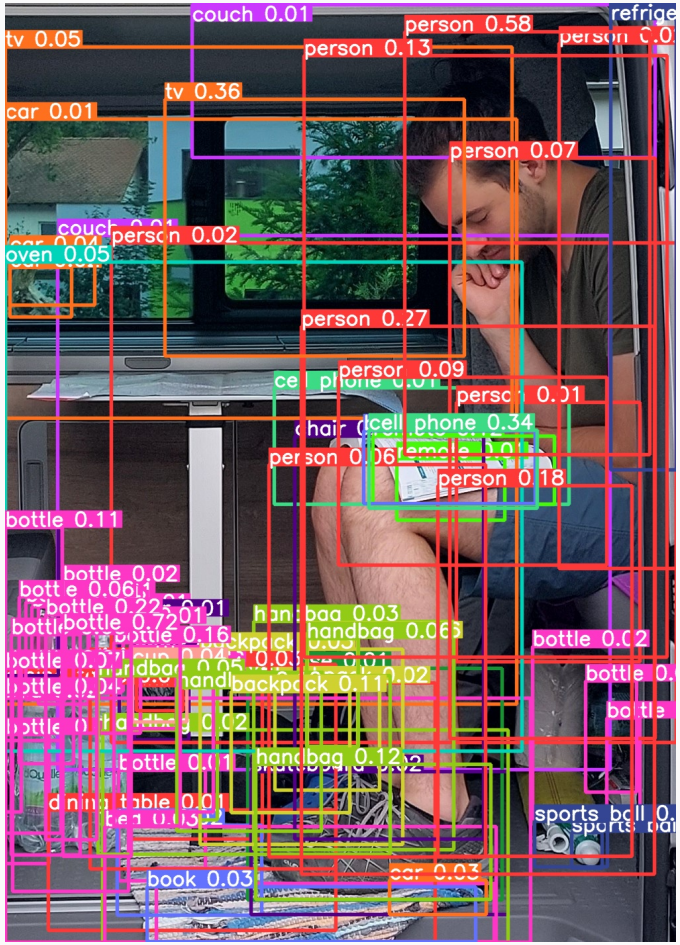


Abbildung 7: Ergebnis mit einem Confidence Schwellenwert von 0.01

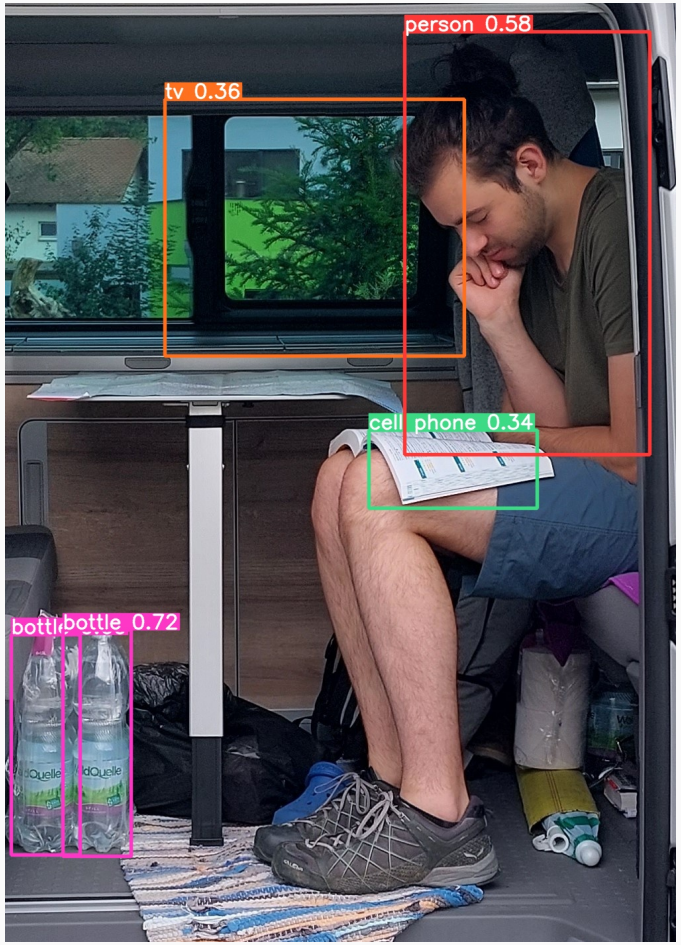
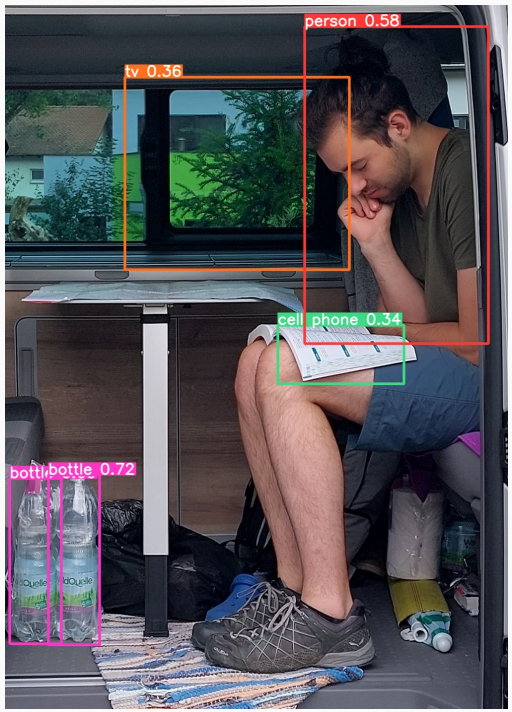


Abbildung 8: Ergebnis mit einem Confidence Schwellenwert von 0.3

Grenzen von YOLO

- falsche Lokalisierung oder Klassifizierung von Objekten



- viele kleine Objekte, die nah beieinander liegen

Abbildung 9: Erkennungsfehler mit YOLO[10]

[10] https://public.roboflow.com/object-detection/pklot

Vergleich Fast R-CNN und YOLO

(a) Fast R-CNN

(b) YOLO

Abbildung 10: Vergleich Fast R-CNN mit YOLO (mit PASCAL VOC 2007 Datensatz)[11]

[11] You only look once: Unified, real-time object detection, *Redmon et al.* (2016)[4]

Versionen von YOLO

Es gibt 8 Versionen (YOLOv1 bis YOLOv8).

wichtigsten Meilensteine:

- YOLOv3
- YOLOv5
- YOLOv8

Versionen von YOLO

Es gibt 8 Versionen (YOLOv1 bis YOLOv8).

wichtigsten Meilensteine:

- YOLOv3
- YOLOv5
- YOLOv8

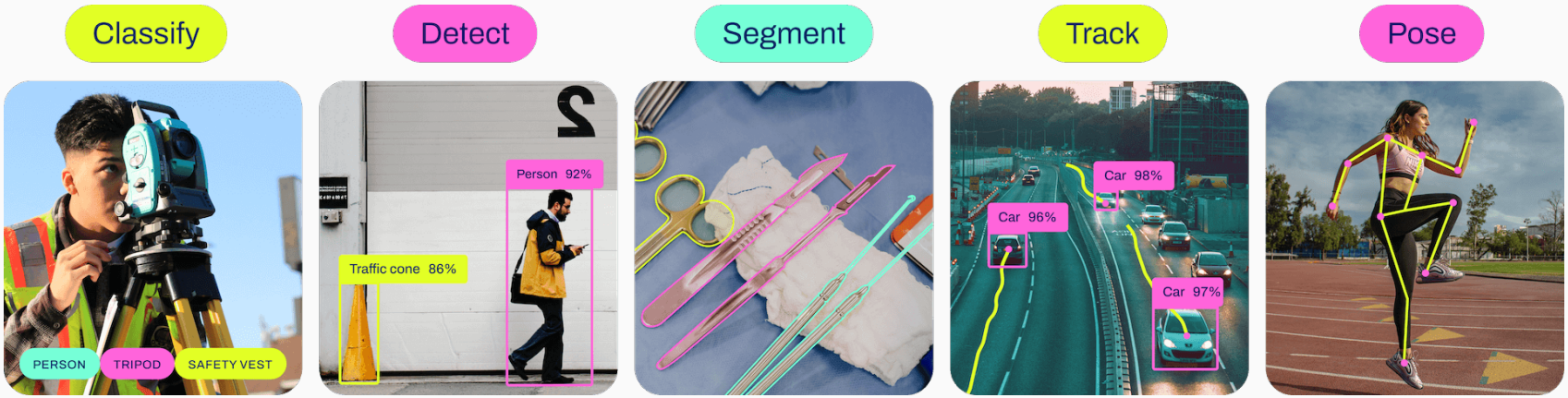


Abbildung 11: Neue Möglichkeiten mit YOLOv8[12]

[12] https://docs.ultralytics.com/tasks/

Inhaltsangabe

1. Einführung

2. YOLO

3. Verwendung

4. Beispiele

Verwendung

Wichtigste Funktionen:

- Trainieren: Vorgefertigtes Datenset oder eigenes Datenset
- Validation: Validierung der Ergebnisse des Models
- Detection: Verwendung des Models auf eine Quelle

Installation:

- YOLOv3 und YOLOv5:
  1. pip install ultralytics
  2. git clone https://github.com/ultralytics/yolov3 oder yolov5
  3. pip install -r requirements.txt

Parameter

Parameter die es beim Trainieren, Validation und Prediction gibt und wichtig sind:

1. **weights**: Das Model z.B: *--weights yolov3-tiny.pt*
2. **data**: Konfiguration vom Model z.B: *--data coco128.yaml*
3. **img**: Größe vom Bild beim Trainieren z.B: *--img 640*
4. **device**: Welches Gerät zum berechnen verwendet werden soll z.B:*--device 0*

Bei jeder Funktion gibt es Standard Werte, die man in der jeweiligen Python-Datei setzen kann.

Trainieren

- **YOLOv3 und YOLOv5:**
- Ausführung von train.py: python train.py --parameter value
- Parameter:
  1. **batch-size**: Wie viele Bilder auf einmal ins Model geladen werden z.B: *--batch-size 32*
  2. **epochs**: Wie viele Durchläufe das Training beendet wird z.B: *--epochs 100*
  3. **save-period**: Nach jeder X-ten Epoche wird das Model gespeichert z.B: *--period 3*
  4. **patience**: Nach wie vielen Epochen ohne Verbesserung abgebrochen werden soll z.B: *-patience 10*
- Aufruf:
  - *python train.py --weights yolov3-tiny.pt --img 320 --batch-size 16*

Validation

- **YOLOv3 und YOLOv5:**
- Ausführung von val.py: python val.py --parameter value
- Parameter nur in Validation:
  1. **batch-size**: z.B:*--batch-size 16*
  2. **confidence-threshold**: Die minimale Confidence, damit die Bounding Boxes gezeichnet werden z.B: *--conf-thres 0.3*
- Aufruf:
  - *python val.py --weights yolov5n.pt --conf-thres 0.001 --img 640*

Detection

- **YOLOv3 und YOLOv5:**
- Ausführung von detect.py: python detect.py --parameter value
- Parameter nur in Detection:
  - **source**: Quelle auf der man die Detection starten möchte.
    z.B: Webcam *--source 0*
    Ordner *--source path/to/something*
  - **confidence-threshold**: Die minimale Confidence, damit die Bounding Boxes gezeichnet werden z.B: *--conf-thres 0.3*
- Aufruf:
  - *python detect.py --weights yolov3-tiny.pt --source 0*

Inhaltsangabe

1. Einführung

2. YOLO

3. Verwendung

4. Beispiele

Beispiele I

Abbildung 12: Ergebnis mit YOLOv3-Tiny.pt3; Dauer: 124ms

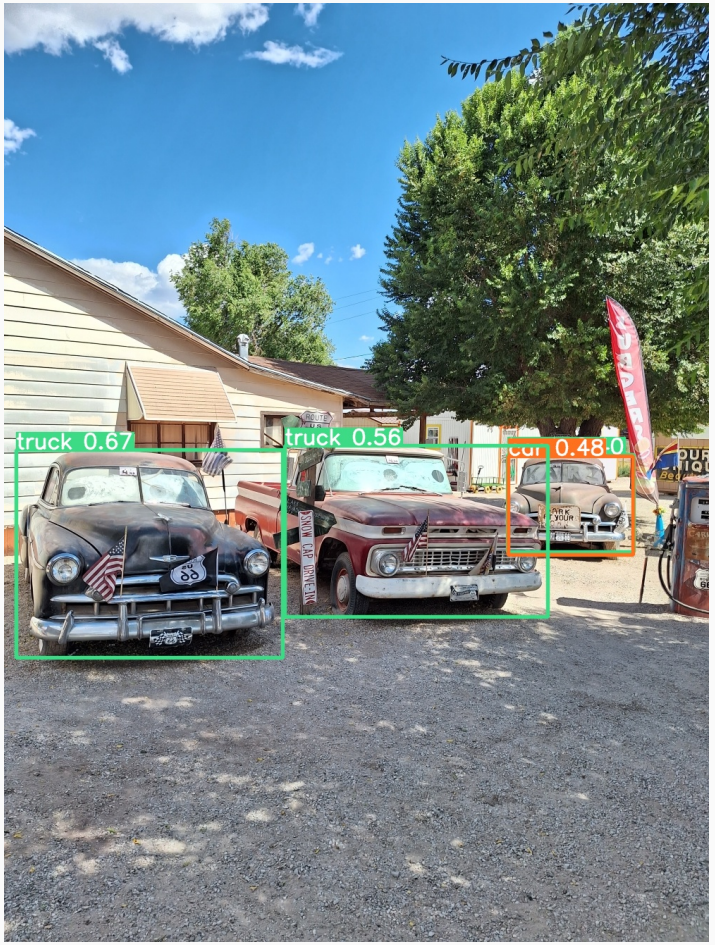


Abbildung 13: Ergebnis mit YOLOv5n.pt; Dauer: 73ms

Beispiele II

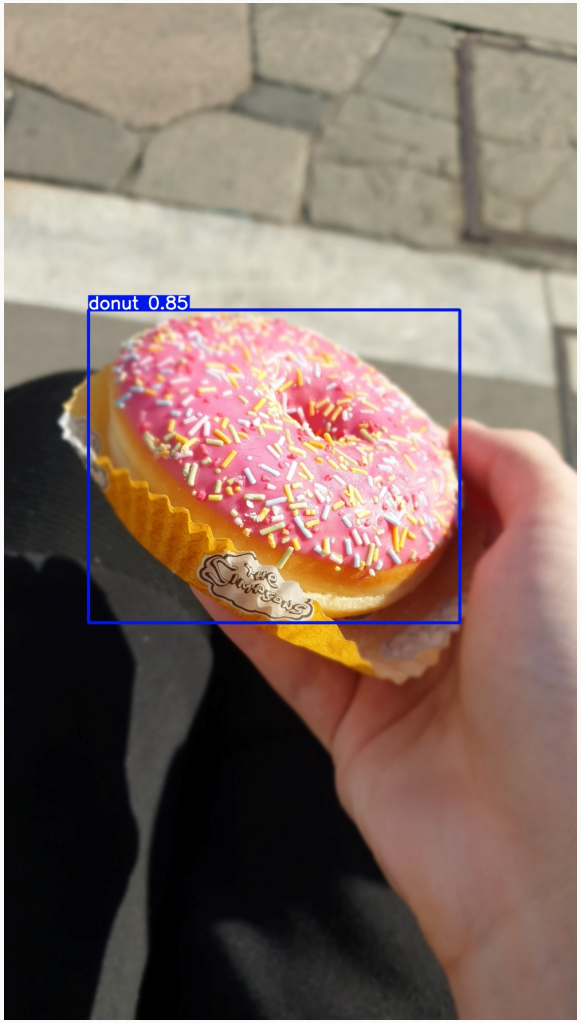


Abbildung 14: Ergebnis mit YOLOv3-Tiny.pt; Dauer: 89ms

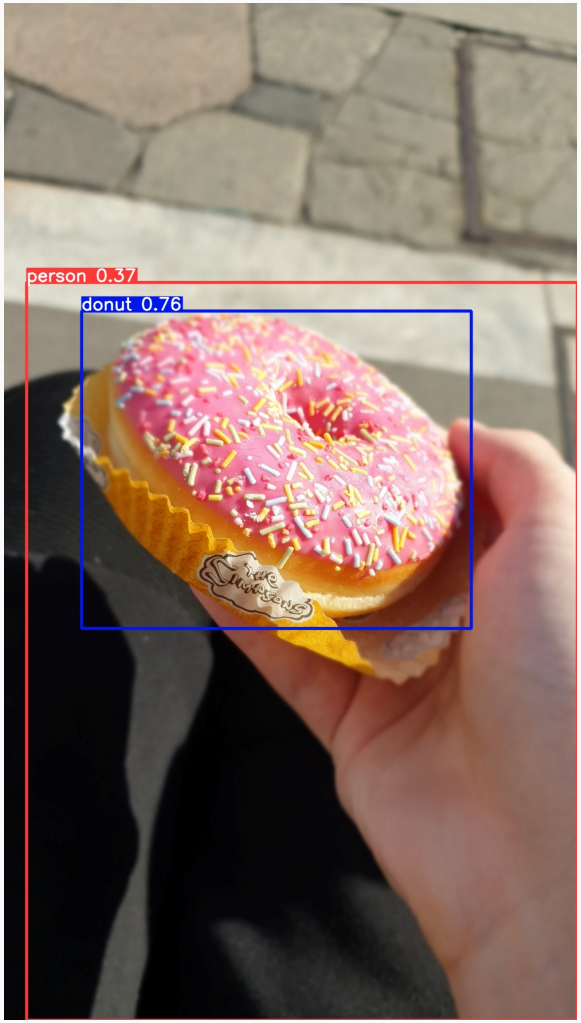
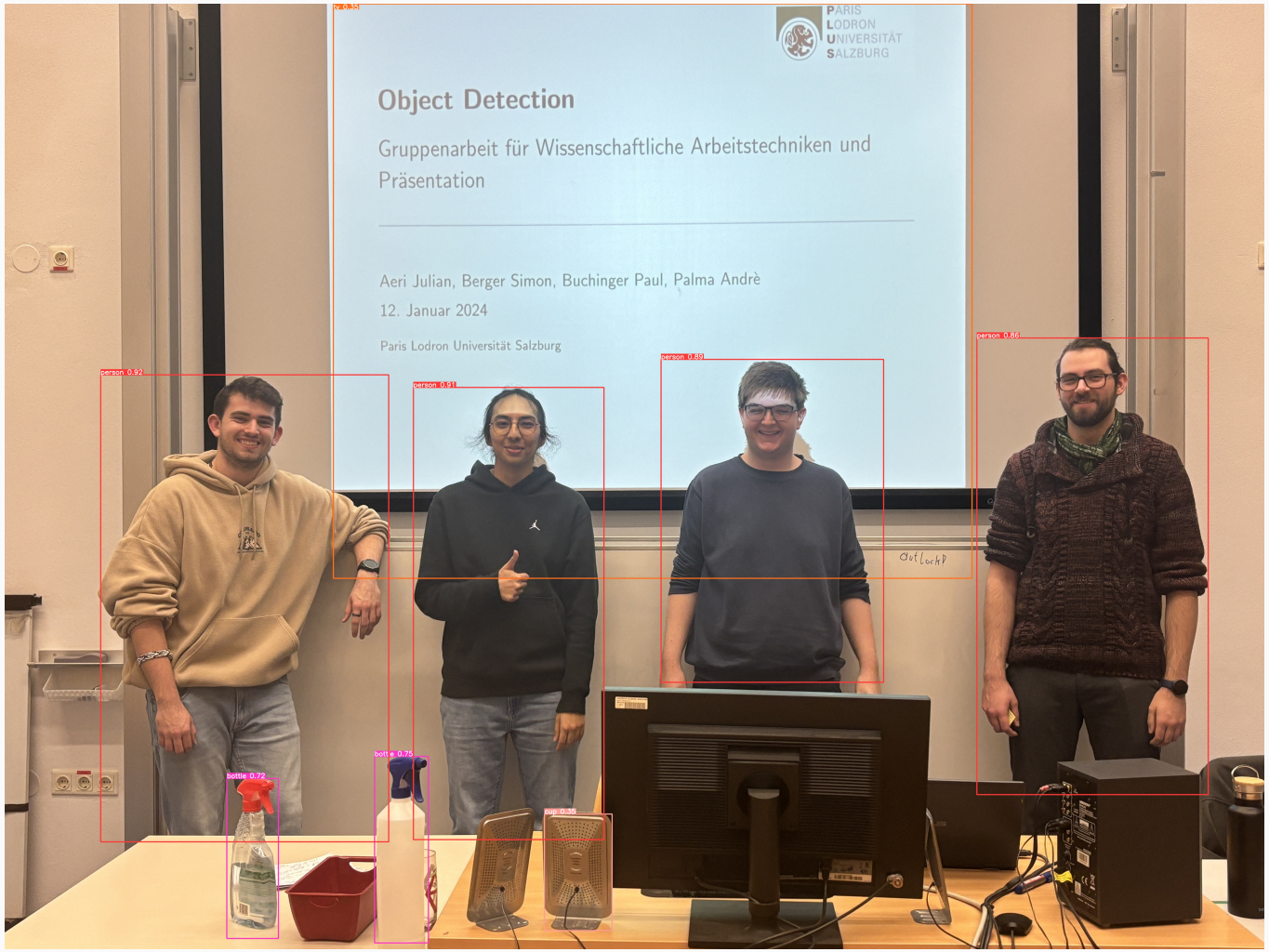


Abbildung 15: Ergebnis mit YOLOv5n.pt; Dauer: 66ms

Ende



Danke für Ihre Aufmerksamkeit!

Literaturverzeichniss i


B. Cyganek.
Object detection and recognition in digital images: theory and practice.
John Wiley & Sons, 2013.


N. Dalal and B. Triggs.
Histograms of oriented gradients for human detection.
In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.


Y. Freund, R. E. Schapire, et al.
Experiments with a new boosting algorithm.
In *icml*, volume 96, pages 148–156. Citeseer, 1996.

Literaturverzeichniss ii

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi.
You only look once: Unified, real-time object detection.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

G. Stockman and L. G. Shapiro.
Computer vision.
Prentice Hall PTR, 2001.

P. Viola and M. Jones.
Rapid object detection using a boosted cascade of simple features.
In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, 2001.

Literaturverzeichniss iii

P. Viola and M. J. Jones.
Robust real-time face detection.
International journal of computer vision, 57:137–154, 2004.

Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye.
Object detection in 20 years: A survey.
Proceedings of the IEEE, 111(3):257–276, 2023.