Universität Münster

ifgi
Institut für Geoinformatik
Universität Münster

Final Assignment
Study Project: Machine Learning meets Insect Monitoring

University of Münster
Institute for Geoinformatics

First lecturer:
Dr. Benjamin Risse

Second lecturer:
Sebastian Thiele

Semester:
WS 2023/2024

Submitted by:
Maximilian Elfers
??? ???

Timo Lietmeyer
459 169

Hendrik Lüning
514 910

Münster, March 2024

Contents

1 Introduction

2 Related Work

3 Fundamentals

3.1 Yolo

3.1.1 Der YOLO Algorithmus

Da der Blick des Menschen Objekterkennung, -einordnung und -wirkung intuitiv ermöglicht, ist es unserem Gehirn im Zusammenspiel mit unseren Augen möglich, schnell und genau zu sehen. Durch diese Fähigkeiten können wir mit nur wenig bewussten Gedanken komplexe Aufgaben, wie Fahrradfahren bewältigen, bei denen gleichzeitig mehrere Sinne beansprucht werden. [1].
Dem Computer kann dies mit schnellen und genauen Algorithmen zur Objekterkennung beigebracht werden. Aktuelle Systeme nutzen Klassifikatoren zur Objekterkennung. Dieser wird an verschiedenen Stellen in variablen Skalierungen im Testbild angewendet, um eine Klassifizierung eines Objektes zu ermöglichen [1].

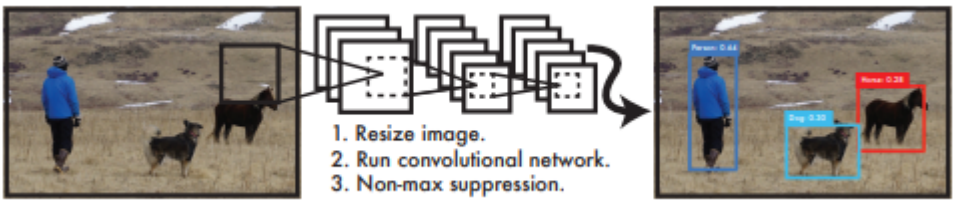

Figure 3.1: Das YOLO Objekterkennungsystem [1]

„You Only Look Once“ (YOLO) betrachtet Objekterkennung als einzelnes Regressionsproblem, indem direkt von Bildpixel zu Boundingbox Koordinaten und Klassenwahrscheinlichkeiten berechnet wird. Dieser Algorithmus analysiert nur einmal ein Bild und sagt direkt vorher, welche Objekte wo vorhanden sind. Dadurch ist die Komplexität des Aufbaus von YOLO sehr gering, wie in Abb. 3.1 zu sehen [1].
Die Performance zur Objekterkennung wird durch das Training von YOLO mit vollständigen Bildern gesteigert. Durch dieses vereinheitlichte Modell entstehen mehrere Vorteile gegenüber den traditionellen Objekterkennungssystemen [1].

Der erste Vorteil von YOLO ist die gesteigerte Performance. Dies wird dadurch ermöglicht, dass Objekterkennung auf Bildern als Regressionproblem betrachtet wird und deshalb keine komplexe Pipeline die Verarbeitung eines Bildes verlangsamt. [1]. Zweitens analysiert YOLO ein Bild global mit Vorhersagen zur Objekterkennung. Dadurch kann YOLO den Fehler beim Verwechseln von Hintergrund und Objekten im Vordergrund um die Hälfte im Vergleich zu Fast R-CNN verringern. Dies geschieht vor allem durch den größeren Kontext, den YOLO durch die Gesamtbildanalyse gewinnt [1].
Der dritte Vorteil ist das YOLO mit generalisierten Repräsentationen von Objekten trainiert wurde um die Fehlertoleranz bei der Anwendung auf neue Bereiche und unerwartete Eingaben zu vergrößern, aufgrund der Möglichkeit der hohen Verallgemeinerung [1].
Ein Nachteil von YOLO liegt in der Genauigkeit. Der Algorithmus hat Schwierigkeiten einige, insbesondere kleine, Objekte genau zu lokalisieren [1].
Da der Quellcode, mehrere vortrainierte Modelle und die Trainingsdaten von YOLO Open-Source sind und zum Download bereitstehen, ist dieser Algorithmus für den Rahmen dieser Arbeit leicht zugänglich und anwendbar [1].

YOLO unterteilt in Bild in $S \times S$ Rasterzellen. Wenn der Mittelpunkt eines Objektes in eine Rasterzelle fällt, ist diese für die Erkennung des Objektes zuständig. Boundingboxen und ihre jeweiligen Confidence Scores werden für jede Rasterzelle vorhergesagt [1]. Der Confidence Score beschreibt, wie sicher sich das Modell ist, dass die Boundingbox ein Objekt dieser Klasse enthält und für wie genau das Modell diese Vorhersage hält.
Dieser Wert enthält nicht nur die Wahrscheinlichkeit, dass diese Klasse in der Boundingbox vorkommt, sondern auch wie gut die vorhergesagte Box mit dem detektierten Objekt übereinstimmt. Ein Beispielablauf ist in Abb. 3.2 zu sehen.

Da jede Rasterzelle nur 2 Boundingboxenvorhersagen und eine Klasse haben kann, unterliegt YOLO einer räumlichen Einschränkung. Dies begrenzt die Anzahl der benachbarten Objekte, die das Modell vorhersagen kann. Außerdem ist es für das Modell schwierig, kleine Objekte, die in Gruppen auftreten, zu detektieren [1].
Eine weitere Herausforderung ist, dass Objekte mit neuen oder ungewöhnlichen Formen auftreten können und dadurch die Vorhersage erschwert wird. Da die Netzwerkarchitektur aus mehreren „Downsampling“-Schichten besteht, benutzt das Modell relativ grobe Features zur Vorhersage der Boundingboxen [1].
Außerdem sorgt das Training mit der YOLO spezifischen Verlustfunktion (für eine Erklärung s. Formel ??, S. ?? und Abb. ??, S. ??), die die Erkennungsleistung annähert, dafür dass Fehler bei kleinen Boundingboxen genauso wie bei großen Boundingboxen behandelt werden. Dies ist ein Nachteil, weil ein kleiner Fehler in einer großen Boundingbox meistens wenig Auswirkungen hat, aber ein kleiner Fehler in einer kleinen Boundingbox eine sehr viel größer Auswirkung auf die IOU

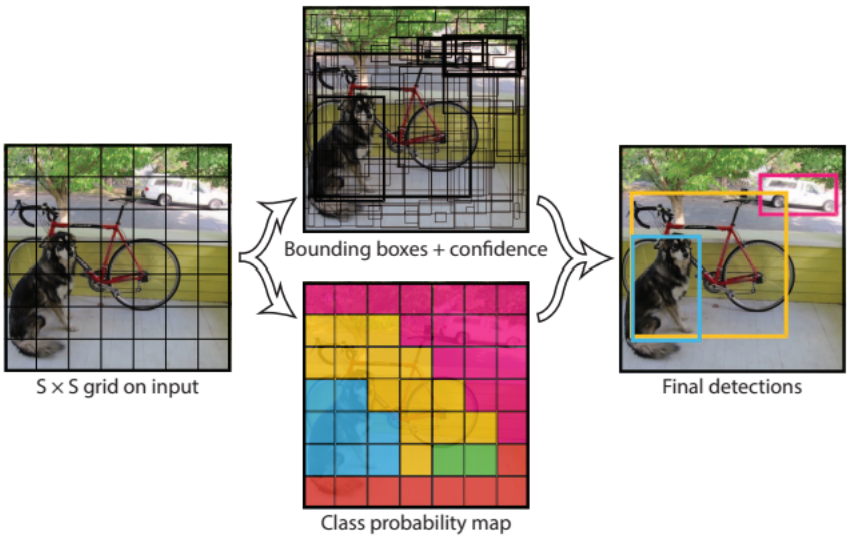

Figure 3.2: Das YOLO Modell[1]

hat. Falsche Lokalisierungen sind eine weitere Hauptfehlerquelle [1].
YOLO ist für den Anwendungszweck dieser Arbeit geeignet, weil der Algorithmus die entsprechende Performance und einfache Verfügbarkeit von trainierten Modellen bietet. Außerdem existieren mehrere weitere Versionen (s. Abb. **??**, S. **??**) von YOLO, die Vorteile in einzelnen Aspekten bieten [2].

Im Rahmen dieser Arbeit wird YOLOv8 von Ultralytics verwendet. Diese bietet Vorteile in der Performance und Genauigkeit der Objektdetektierung und wird im nächsten Kapitel genauer erläutert.

3.1.2 YOLOv8 von Ultralytics

Im Januar 2023 wurde von der Firma Ultralytics YOLOv8 veröffentlicht, welches auf YOLOv5 basiert. Diese Version beinhaltet 5 verschiedene Modelle (YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large), YOLOv8x (extra large)), die mit unterschiedlich großen Datensätzen trainiert wurden [2].
Ein Vorteil dieser YOLO Implementierung ist, dass verschiedene Varianten für Objektdetektion, -segmentierung und -verfolgung, sowie -klassifizierung existieren. In dieser Arbeit wird hauptsächlich die „-seg“-Variante verwendet, welches bereits eine Segmentierung der Umrisse der detektierten Objekte integriert hat. Die Architektur dieser Algorithmen kann man in 3 Teile aufteilen (s. Abb. 3.3). Diese sind der Backbone, der Neck und der Head [2].
Das Detektieren nützlicher Features vom Eingabebild geschieht im Backbone, welcher meist als CNN implementiert ist [2].

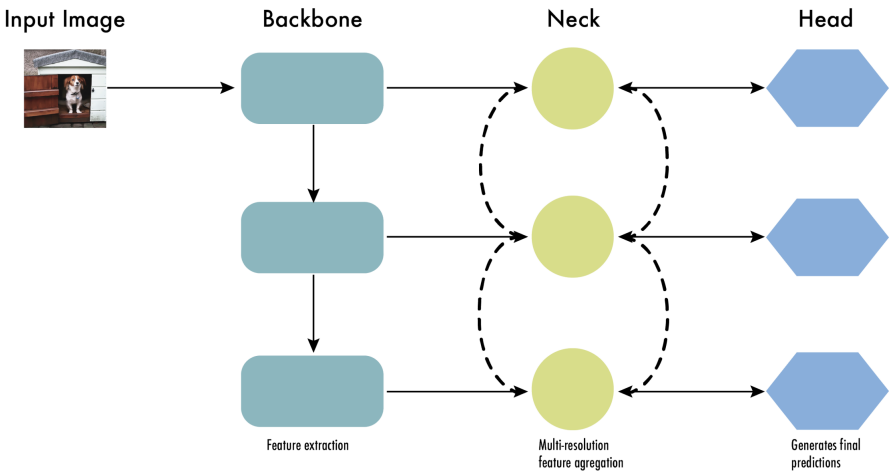

Figure 3.3: Architektur modernen Objektdetektoren [2]

Zwischen Backbone und Head wird der Neck eingesetzt, um die Features, die der Backbone ausgibt, zu aggregieren und zu verfeinern. Der Fokus liegt auf der Verbesserung der räumlichen und semantischen Informationen über die unterschiedlichen Skalierungen hinweg [2].
Die letzte Komponente ist der Head, welcher die Vorhersagen, aufgrund der von dem Backbone und Neck gelieferten Features, trifft. Hier werden meistens aufgabenspezifische Teilnetze eingesetzt, um Klassifizierung, Lokalisierung und auch sofortige Segmentierung durchzuführen. Aus den Features, die der Neck liefert, erstellt der Head Vorhersagen für jeden Objektkandidaten. Ein Post-Procressing Schritt, wie die Non-Maximum-Supression (NMS), filtert überlappende Vorhersagen heraus, sodass nur die sichersten Detektionen genutzt werden [2].
Da YOLOv8 auf YOLOv5 basiert, wird in diesem ein ähnlicher Backbone genutzt. Für die Architektur von YOLOv8 s. Abb. **??** (S. **??**).

Um die Performance insbesondere bei der Objekterkennung von kleineren Objekten zu verbessern, nutzt YOLOv8 CloU [3] und DFL [4] Verlustfunktionen für Boundingboxloss und binäre Kreuzentropie für den Klassifizierungsloss [2].

Mit dem YOLOv8-seg Modell wird auch eine Variante angeboten, die semantische Segmentierung ermöglicht. Dieses Modell wird in dieser Arbeit genutzt, da es den Anforderungen an Präzision und Performance entspricht. An den Einstellungen des YOLO-Modells, wie bspw. den Werten des Ausgabetensors, wird nichts geändert. Es werden die von Ultralytics vorgegebenen Standardeinstellungen und Modelle genutzt.

3.2 OpenCV

4 Methods

4.1 Contributions by group members

5 Results

6 Discussion

7 Contributions by group members

List of Figures

List of Tables

Bibliography

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2016, pp. 779–788, ISBN: 978-1-4673-8852-8. DOI: 10.1109/CVPR.2016.91 (cit. on pp. 4–6).

[2] J. Terven and D. Cordova-Esparza, "A Comprehensive Review of YOLO: From YOLOv1 to YOLOv8 and Beyond," pp. 1–33, 2023. arXiv: 2304.00501. [Online]. Available: http://arxiv.org/abs/2304.00501 (cit. on pp. 6, 7).

[3] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, no. 2, pp. 12 993–13 000, 2020, ISSN: 2159-5399. DOI: 10.1609/aaai.v34i07.6999. arXiv: 1911.08287 (cit. on p. 7).

[4] X. Li, W. Wang, L. Wu, *et al.*, "Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection," *Advances in Neural Information Processing Systems*, vol. 2020-December, pp. 1–14, 2020, ISSN: 10495258. arXiv: 2006.04388 (cit. on p. 7).

Declaration of Academic Integrity

We hereby confirm that this paper on

Final Assignment
Study Project: Machine Learning meets Insect Monitoring

is solely our own work and that we have used no sources or aids other than the ones stated. All passages in our paper for which other sources, including electronic media, have been used, be it direct quotes or content references, have been acknowledged as such and the sources cited.

Hendrik Lüning, Münster, March 12, 2024

Maximilian Elfers, Münster, March 12, 2024

Timo Lietmeyer, Münster, March 12, 2024