Prototyp zur Verifizierung der korrekten Teileverwendung in der Produktion der Alukon KG

P r a x i s a r b e i t

an der Hochschule für angewandte Wissenschaften Hof
Fakultät Informatik
Studiengang Informatik-Bachelor

Vorgelegt bei
Prof. Dr. Göbel
Alfons-Goppel-Platz 1
95028 Hof

Vorgelegt von
Taylan Özdabak
Angerstraße 20
95152 Selbitz

Hof, 10.03.2025

Inhaltsverzeichnis

Abbildungsverzeichnis IV

Abkürzungsverzeichnis V

1. Einleitung 1

1.1 Überleitung von Bachelorarbeit 1

1.2 Aufbau der Arbeit 2

2 Grundlagen der Objekterkennung mit Maschinellem Lernen 3

2.1 Maschinelles Lernen und Objekterkennung 3

2.1.1 Modelle für die Objekterkennung 3

2.1.2 YOLO – Echtzeit-Objekterkennung 4

2.2 Train-Validation-Test-Split 5

2.2.1 Warum wird ein Datensatz in verschiedene Teile unterteilt? 6

2.2.2 Typische Verhältnisse der Aufteilung 7

2.3 Metriken zur Bewertung von Modellen 8

2.4 Roboflow – Datenvorbereitung für das Training 11

2.4.1 Roboflow und seine Funktionen 12

2.4.2 Warum Roboflow im Prototyp verwendet wurde 13

2.4.3 Datenexport für YOLO 13

3 Prototyp 14

3.1 Hintergrund 14

3.2 Initiale Idee 14

3.3 Datensammeln 15

3.4 Erkennungstypen 15

3.4.1 Statische Bildanalyse 16

3.4.2 Videodateiverarbeitung 16

3.4.3 Echtzeit-Kameraerkennung 17
3.5 Erster Prototyp 18
3.6 Zweiter Prototyp 21
3.7 Weitere Prototypen mit verschiedenen Modellen 25
3.7.1 Yolov8l-seg 25
3.7.2 Yolo11n-seg mit mehr Epochen 25
3.7.3 Mask R-CNN 27
3.7.4 Detectron2 28
3.8 Finaler Prototyp 29
3.8.1 Datensatz 29
3.8.2 Training 32
3.8.3 Fazit 37
3.9 Positionserkennung 38
3.10 Schlussfolgerung 40
4 Fazit und Ausblick 41
5 Literaturverzeichnis 43

Abbildungsverzeichnis

Abbildung 1: Erkennungsmatrix für YOLO-Modell 10
Abbildung 2: Beispiel für das Annotieren auf Roboflow 12
Abbildung 3: Trainingsresultate vom ersten Prototyp 18
Abbildung 4: Erkennungsresultate erster Prototyp 19
Abbildung 5: Fehlerhafte Erkennung 20
Abbildung 6: Trainingsresultate im Zweiten Versuch 21
Abbildung 7: Erkennungsresultate im Zweiten Versuch 22
Abbildung 8: Falsche Erkennung im zweiten Versuch 23
Abbildung 9: Fehlerkennung im Hintergrund 24
Abbildung 10: Resultate nach 150 Epochen 26
Abbildung 11: Resultate nach 300 Epochen 26
Abbildung 12: Axis Network Camera 29
Abbildung 13: Beispiel Datensatz im finalen Prototyp 31
Abbildung 14: Beispiel aus der ersten Einheit 32
Abbildung 15: Beispiel aus der zweiten Einheit 33
Abbildung 16: Beispiel aus der vierten Einheit 34
Abbildung 17: Beispiel aus der fünften Einheit 35
Abbildung 18: Positionserkennung der Traversen 39

Abkürzungsverzeichnis

CNN	Convolutional Neural Network
DL	Deep Learning
KG	Kommanditgesellschaft
KI	Künstliche Intelligenz
MAP	Mean average precision
ML	Machine Learning
R-CNN	Region-based Convolutional Neural Network
RPN	Region Proposal Networks
YOLO	You Only Look Once

1. Einleitung

1.1 Überleitung von Bachelorarbeit

In der an dieser Arbeit zugrundeliegenden Bachelorarbeit *„Vergleich und Evaluation verschiedener Identifikationsmethoden bei der Alukon KG“* wurde untersucht, welche Methode sich am besten zur Identifikation von Produktteilen eignet. Die Ergebnisse zeigten, dass die visuelle Erkennung die vielversprechendste Methode ist. Aufbauend auf diesen Erkenntnissen wird in der vorliegenden Praxisarbeit ein Prototyp entwickelt, der die visuelle Erkennung mithilfe von Objekterkennungsmodellen implementiert. Ziel ist es, eine Lösung zu erarbeiten, die die korrekte Verwendung von Bauteilen im Produktionsprozess automatisiert überprüft, um Fehler zu minimieren und die Effizienz zu steigern. Im Rahmen meines Bachelorstudiums der Informatik an der Hochschule Hof absolvierte ich dazu ein 18-wöchiges Praktikum bei der Alukon KG, einem führenden Hersteller von Rollladen-, Sonnen- und Insektenschutzsystemen. Seit ihrer Gründung im Jahr 1974 hat sich die Alukon KG von einem kleinen Handwerksbetrieb zu einem modernen Industrieunternehmen mit über 500 Beschäftigten an zwei Standorten, Konradsreuth und Haigerloch, entwickelt. Durch kontinuierliche Investitionen in moderne Fertigungstechnologien kann das Unternehmen sowohl Serienproduktionen als auch individuelle Kundenwünsche effizient umsetzen.

1.2 Aufbau der Arbeit

Die Praxisarbeit gliedert sich in die folgenden drei Unterpunkte:

1. Grundlagen der Objekterkennung mit Maschinellem Lernen – Eine grundlegende Erklärung der Objekterkennung und ihrer Bedeutung im Kontext von maschinellem Lernen.

2. Trainings Metriken und Roboflow – Erklärung der wichtigsten Metriken zur Bewertung von Modellen, wie Genauigkeit, Präzision und Recall, sowie die Rolle von Roboflow in der Datenvorbereitung für das Training des YOLO-Modells.

3. Prototyp – Entwicklung und Implementierung des Prototyps zur automatisierten Verifizierung der Teileverwendung.

2 Grundlagen der Objekterkennung mit Maschinellem Lernen

2.1 Maschinelles Lernen und Objekterkennung

Maschinelles Lernen (ML) ist ein Teilbereich der künstlichen Intelligenz (KI), der es Computern ermöglicht, Muster in Daten zu erkennen und daraus Vorhersagen zu treffen, ohne explizit programmiert zu sein[1]. Je nach Art der Daten und des Lernziels werden dabei unterschiedliche Ansätze verwendet, die sich in ihrer Methodik und Anwendung unterscheiden. Deep Learning (DL) ist eine spezialisierte Form des maschinellen Lernens, die auf künstlichen neuronalen Netzen basiert. Diese Netze sind inspiriert von der Struktur des menschlichen Gehirns und bestehen aus mehreren Schichten daher „Deep“. Besonders leistungsfähig sind Convolutional Neural Networks (CNNs), die speziell für die Verarbeitung von Bilddaten entwickelt wurden[2].

2.1.1 Modelle für die Objekterkennung

In der Objekterkennung geht es darum, bestimmte Objekte in Bildern oder Videos zu identifizieren und deren Position zu bestimmen. Dafür werden oft neuronale Netzwerke verwendet, insbesondere CNNs, die Bildmerkmale extrahieren und klassifizieren können.

Typische Schritte im Trainingsprozess eines Modells für die Objekterkennung sind:

1. **Datenvorbereitung:** Sammlung und Annotierung eines Datensatzes mit Bildern und entsprechenden Objektmarkierungen.
2. **Trainingsphase:** Das Modell lernt anhand eines großen Datensatzes, Muster und Merkmale in Bildern zu erkennen.

[1] Vgl. Goodfellow et al. (2016) S. 2f.
[2] Vgl. Goodfellow et al. (2016) S. 326

3. **Validierung:** Das trainierte Modell wird mit einem separaten Validierungsdatensatz getestet, um seine Genauigkeit und Robustheit zu überprüfen.
4. **Feinabstimmung und Hyperparameter-Optimierung:** Falls nötig, werden Anpassungen vorgenommen, um die Leistung des Modells zu verbessern.
5. **Testphase und Einsatz:** Das finale Modell wird auf einem unabhängigen Testdatensatz geprüft und kann anschließend für Echtzeitanwendungen genutzt werden[3].

2.1.2 YOLO – Echtzeit-Objekterkennung

YOLO (You Only Look Once) ist ein modernes Modell zur Objekterkennung, das sich durch hohe Geschwindigkeit und Genauigkeit auszeichnet. Im Gegensatz zu früheren Methoden, die Bilder in mehrere Regionen unterteilen und jedes Segment separat analysieren, betrachtet YOLO das gesamte Bild auf einmal und erkennt Objekte direkt. Dadurch kann YOLO in Echtzeit Objekte erkennen und lokalisieren, was es ideal für Anwendungen wie autonomes Fahren, Videoüberwachung oder Robotik macht. YOLO arbeitet mit einem Bounding-Box-Ansatz, bei dem das Modell ein Bild in ein Gitter unterteilt und für jede Zelle Vorhersagen über mögliche Objekte trifft. Diese Vorhersagen beinhalten[4]:

- Die Koordinaten der Bounding Box (x, y, Breite, Höhe)
- Den Klassennamen des Objekts
- Die Wahrscheinlichkeit, dass das Objekt tatsächlich vorhanden ist

Neben der klassischen Bounding-Box-Erkennung bietet YOLO auch Instanz Segmentierung, eine Technik, die nicht nur Objekte lokalisiert, sondern auch ihre genauen Konturen in Form von Pixel-genauen Masken erfasst. Dies ist besonders nützlich, wenn

[3] Vgl. Elgendy (2020) Kapitel 7
[4] Vgl. Redmon et al (2016) S. 1f.

sich mehrere Instanzen desselben Objekts überlappen. Dafür gibt es spezialisierte YOLO-Modelle mit der "-seg"-Erweiterung, die für Segmentierungsaufgaben optimiert sind. Die aktuellen YOLO-Versionen bieten verschiedene Varianten:

- YOLO11n-seg – Kleinste und schnellste Version mit geringem Rechenaufwand
- YOLO11s-seg – Etwas größer, bietet bessere Genauigkeit bei moderater Geschwindigkeit
- YOLO11m-seg – Mittlere Variante mit ausgewogenem Verhältnis zwischen Präzision und Geschwindigkeit
- YOLO11l-seg – Leistungsfähigeres Modell mit höherer Genauigkeit
- YOLO11x-seg – Größtes Modell mit maximaler Präzision, benötigt jedoch hohe Rechenleistung[5]

Dank dieser Weiterentwicklungen bietet YOLO eine noch höhere Erkennungsgenauigkeit und Flexibilität, sowohl für die klassische Objekterkennung als auch für Anwendungen mit detaillierter Segmentierung. Aktuelle Versionen wie YOLO11 bieten verbesserte Architektur, optimierte Algorithmen und eine höhere Erkennungsgenauigkeit im Vergleich zu früheren Versionen.

2.2 Train-Validation-Test-Split

Beim maschinellen Lernen wird ein Datensatz üblicherweise in drei Teile aufgeteilt: **Trainingsdaten (**Training Set**)**, **Validierungsdaten (**Validation Set**)** und **Testdaten (**Test Set**)**. Diese Unterteilung ist entscheidend, um sicherzustellen, dass das Modell nicht nur die Trainingsdaten auswendig lernt (Overfitting)[6], sondern auch auf neue, unbekannte Daten gute Vorhersagen treffen kann (Generalisierung)[7].

[5] Vgl. Seite „Instanz Segmentierung“ URL: https://docs.ultralytics.com/de/tasks/segment/ (Abgerufen am 05.11.2024, 14:13 UTC)
[6] Vgl. Murphy (2012) S. 22
[7] Vgl. Murphy (2012) S. 3

- **Trainingsdaten**: Der größte Teil der Daten wird zum Trainieren des Modells verwendet. Hier passt das Modell seine Gewichte an und lernt Muster in den Daten.
- **Validierungsdaten**: Während des Trainings werden diese Daten genutzt, um die Modellleistung zu überwachen und Hyperparameter wie die Lernrate oder Netzwerkarchitektur anzupassen.
- **Testdaten**: Dieser Datensatz wird erst nach dem Training verwendet, um die finale Leistung des Modells zu bewerten. Er simuliert echte Einsatzszenarien und darf während des Trainings nicht beeinflusst werden[8].

2.2.1 Warum wird ein Datensatz in verschiedene Teile unterteilt?

Die Aufteilung eines Datensatzes ist notwendig, um eine objektive Bewertung des Modells zu ermöglichen und Overfitting zu vermeiden. Würde das Modell nur mit einem einzigen Datensatz trainiert und getestet, bestünde die Gefahr, dass es sich zu stark an die Trainingsdaten anpasst und auf neuen Daten schlecht abschneidet. Durch die Trennung in Training, Validierung und Test kann man:

- Überprüfen, ob das Modell die richtigen Muster erkennt, statt sich nur die Trainingsdaten zu „merken“ (Generalisierung).
- Die Hyperparameter anhand der Validierungsdaten optimieren, um eine bessere Modellleistung zu erzielen.
- Sicherstellen, dass die finale Bewertung des Modells nicht durch bereits gesehene Daten beeinflusst wird.

[8] Vgl. Seite „Training, Validation, Test Split for Machine Learning Datasets” URL: https://encord.com/blog/train-val-test-split/ (Abgerufen am 31.10.2024. 09:43 UTC)

2.2.2 Typische Verhältnisse der Aufteilung

Die genaue Aufteilung eines Datensatzes hängt von der Größe des Datensatzes und der spezifischen Anwendung ab. Typische Verhältnisse sind:

- 70% Training – 20% Validierung – 10% Test (häufige Standardaufteilung)
- 80% Training – 10% Validierung – 10% Test (wenn viele Daten vorhanden sind)
- 60% Training – 20% Validierung – 20% Test (wenn feine Modellabstimmung notwendig ist)[9]

[9] Vgl. Seite „Training, Validation, Test Split for Machine Learning Datasets" URL: https://encord.com/blog/train-val-test-split/ (Abgerufen am 31.10.2024, 09:43 UTC)

2.3 Metriken zur Bewertung von Modellen

Bei der Bewertung von maschinellen Lernmodellen ist es wichtig, verschiedene Metriken zu betrachten, um die Leistung des Modells in verschiedenen Aspekten zu verstehen. Dies ist besonders entscheidend, wenn das Modell für Aufgaben wie die Klassifikation oder Objekterkennung verwendet wird. Zu den wichtigsten Metriken zählen Genauigkeit (Accuracy), Präzision (Precision), Recall (Sensitivität), F1-Score, Mean Average Precision (mAP) für Objekterkennung und die Konfusionsmatrix und werden im Folgenden genauer beschrieben[10].

Genauigkeit (Accuracy) ist eine der einfachsten und häufigsten Metriken zur Bewertung von Modellen. Sie gibt an, wie viele der Gesamtproben korrekt klassifiziert wurden. Die Genauigkeit wird berechnet als das Verhältnis der korrekt klassifizierten Instanzen zur Gesamtzahl der Instanzen im Datensatz. Allerdings kann die Genauigkeit bei unausgewogenen Datensätzen irreführend sein, da sie die wahren positiven und falschen negativen Werte nicht berücksichtigt.

Präzision (Precision) misst die Genauigkeit der positiven Vorhersagen eines Modells. Sie berechnet das Verhältnis der richtig positiven Vorhersagen zu der Gesamtzahl der als positiv klassifizierten Instanzen. Diese Metrik ist besonders wichtig, wenn falsche positive Vorhersagen erhebliche Konsequenzen haben.

Recall (Sensitivität) misst, wie gut ein Modell in der Lage ist, alle relevanten Instanzen zu finden, also die positiven Klassen. Es berechnet das Verhältnis der richtig positiven Vorhersagen zu der Gesamtzahl der tatsächlichen positiven Instanzen. Ein hoher Recall bedeutet, dass das Modell die meisten positiven Instanzen korrekt identifiziert, was wichtig ist, wenn das Verpassen von positiven Instanzen schwerwiegende Folgen hat.

Der **F1-Score** ist das harmonische Mittel von Präzision und Recall. Diese Metrik ist besonders nützlich, wenn ein Modell sowohl eine hohe Präzision als auch einen hohen

[10] Vgl. "Vertiefung der Leistungsmetriken" URL: https://docs.ultralytics.com/de/guides/yolo-performance-metrics/ (Abgerufen am 31.10.2024, 11:02 UTC)

Recall aufweisen soll. Ein hoher F1-Score zeigt an, dass das Modell sowohl gut darin ist, relevante Instanzen zu erkennen als auch ungenaue Vorhersagen zu vermeiden und hat dementsprechend einen hohen Recall- und Präzisionsswert.

Mean Average Precision (mAP) wird häufig in der Objekterkennung verwendet und bewertet die Genauigkeit des Modells in Bezug auf die gesamte Precision-Recall-Kurve. Der mAP-Wert wird über alle Klassen und alle Vorhersagen hinweg gemittelt und gibt somit einen Überblick darüber, wie gut das Modell bei der Erkennung und Lokalisierung von Objekten ist. Ein hoher mAP-Wert zeigt an, dass das Modell in der Lage ist, Objekte korrekt zu erkennen und zu lokalisieren.

Die **Konfusionsmatrix** ist eine sehr nützliche Methode zur detaillierten Bewertung der Modellleistung, insbesondere bei Klassifikationsaufgaben. Sie stellt eine Tabelle dar, die die tatsächlichen Klassen gegen die vom Modell vorhergesagten Klassen zeigt. Sie enthält die Werte für True Positives, True Negatives, False Positives und False Negatives, was hilft, nicht nur die Gesamtgenauigkeit des Modells zu verstehen, sondern auch, wie das Modell in Bezug auf jede einzelne Klasse abschneidet. Sie ermöglicht eine genauere Analyse der Fehlerarten und hilft, das Modell weiter zu verbessern.

Durch die Kombination dieser Metriken erhält man ein umfassenderes Bild von der Leistungsfähigkeit eines Modells, insbesondere bei komplexen Aufgaben wie der Objekterkennung, bei denen das richtige Verständnis der Fehlerarten entscheidend ist. Ein Beispielsgraph nach einem erfolgreichen Training eines YOLO-Modells schaut wie folgt aus:

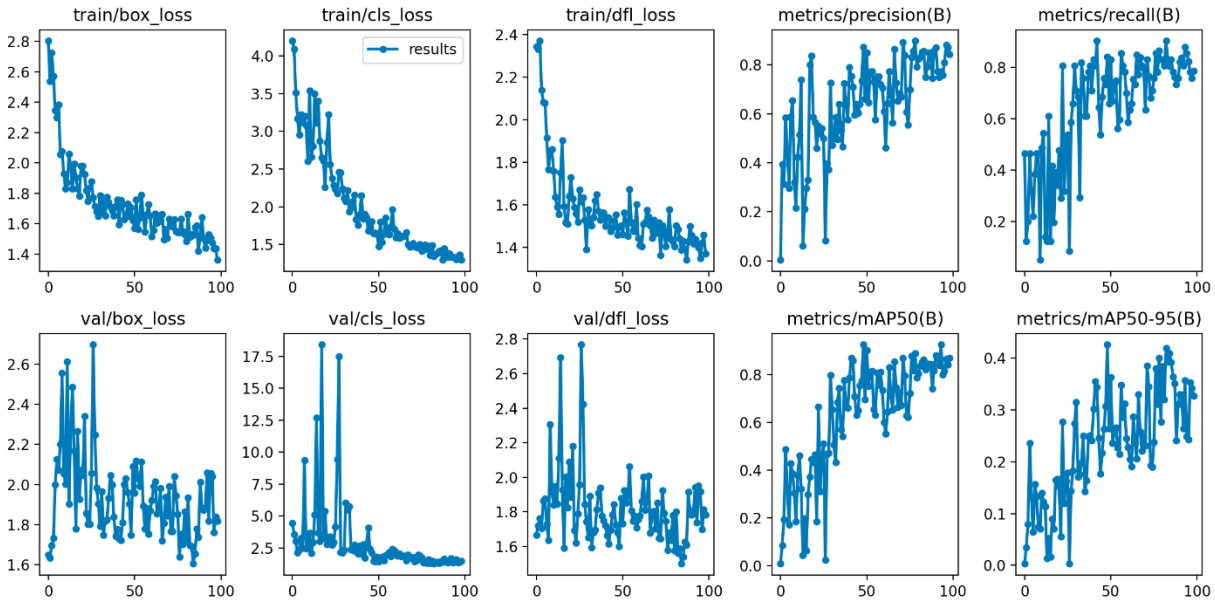

Abbildung 1: Erkennungsmatrix für YOLO-Modell

2.4 Roboflow – Datenvorbereitung für das Training

Roboflow ist eine fortschrittliche Plattform, die speziell entwickelt wurde, um die Datenerfassung, -vorbereitung und -verarbeitung für Objekterkennungsmodelle zu vereinfachen. Die Anwendung bietet eine breite Palette von Funktionen, die es Benutzern ermöglichen, ihre Bilddaten effizient zu verarbeiten und für das Modelltraining vorzubereiten. Was Roboflow besonders auszeichnet, ist seine benutzerfreundliche Oberfläche, die es auch Anfängern ermöglicht, ohne tiefgehende technische Kenntnisse komplexe Datenvorbereitungsprozesse durchzuführen. Durch die intuitive Gestaltung wird der gesamte Workflow – von der Sammlung der Daten bis hin zum Export – erheblich beschleunigt und vereinfacht. Im Vergleich zu anderen Plattformen bietet Roboflow eine cloudbasierte Lösung, die es ermöglicht, Daten in verschiedenen Formaten und aus verschiedenen Quellen zu integrieren. Dies erleichtert es, Datensätze zu erstellen und anzupassen, ohne dass umfangreiche Programmierkenntnisse erforderlich sind. Roboflow bietet auch eine Vielzahl von Anpassungsoptionen, die es Benutzern ermöglichen, die annotierten Datensätze für unterschiedliche Modellarchitekturen zu optimieren. Das macht die Plattform zu einem leistungsstarken Werkzeug für die Vorbereitung von Trainingsdaten in unterschiedlichen Anwendungsbereichen der Computervision. Roboflow sorgt dafür, dass alle Schritte der Datenaufbereitung effizient durchgeführt werden, von der Auswahl und Annotation der relevanten Objekte bis hin zur Erstellung von Datensätzen, die nahtlos mit Modellen wie YOLO genutzt werden können. Die Plattform ist darauf ausgelegt, eine nahtlose Integration in den gesamten Entwicklungsprozess zu ermöglichen, sodass Entwickler und Forscher sich auf die Modellarchitektur und -optimierung konzentrieren können, ohne sich um die mühsame Vorverarbeitung der Daten kümmern zu müssen. Durch das automatische Annotationssystem und die unterstützten Augmentierungsoptionen bietet Roboflow eine perfekte Lösung für die Erstellung großer, vielfältiger und gut annotierter Datensätze, die eine hohe Modellgenauigkeit gewährleisten [11].

[11] Vgl. Seite „Getting Started with Roboflow" URL: https://blog.roboflow.com/getting-started-with-roboflow/ (Abgerufen am 01.11.2024, 08:34 UTC)

2.4.1 Roboflow und seine Funktionen

Roboflow bietet eine umfassende Lösung für die Datenvorbereitung. Die Plattform ermöglicht es, Bilder hochzuladen, diese zu annotieren und anschließend zu augmentieren. Dies sind wichtige Schritte, um den Datensatz für das Training eines Objekterkennungsmodells wie YOLO vorzubereiten. Mit Roboflow können Benutzer Objekte in Bildern einfach mit Bounding Boxes, Polygonen oder Masken kennzeichnen. Diese Annotationen helfen dem Modell, zu lernen, welche Bildbereiche relevante Objekte enthalten. Zudem bietet Roboflow eine Vielzahl von Augmentierungsoptionen, mit denen bestehende Trainingsbilder künstlich vergrößert werden können. Transformationen wie Rotation, Skalierung, Spiegeln und Farbänderungen erhöhen die Vielfalt der Trainingsdaten und verbessern die Robustheit des Modells. Die Abbildung stellt ein Exemplar dar, in dem mithilfe vom Roboflow Werkzeug ein Auto annotiert wurde.

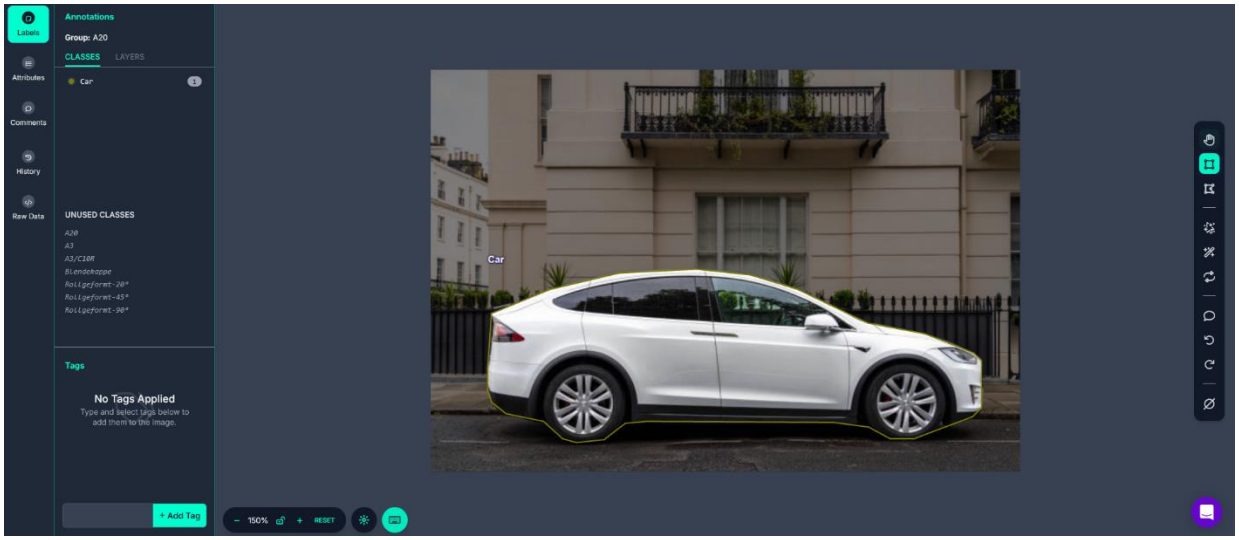

Abbildung 2: Beispiel für das Annotieren auf Roboflow

2.4.2 Warum Roboflow im Prototyp verwendet wurde

In diesen Prototypen wurde Roboflow genutzt, um die Bilddaten für das Training des YOLO-Modells vorzubereiten. Da der Erfolg eines Objekterkennungsmodells maßgeblich von der Qualität und Vielfalt der Trainingsdaten abhängt, war es entscheidend, eine Plattform zu wählen, die sowohl eine einfache Annotation als auch eine effektive Augmentierung der Daten ermöglicht. Roboflow hat sich als ideale Lösung herausgestellt, da es diese Schritte effizient durchführt. Besonders hervorzuheben ist, dass Roboflow auch eine kostenlose Variante anbietet, die trotz ihrer Zugänglichkeit eine ausgezeichnete Qualität und Funktionalität bietet. Diese Version hat es mir ermöglicht, die benötigten Daten schnell und effizient vorzubereiten, was die Entwicklung des Prototyps erheblich beschleunigt hat. Zudem sorgt Roboflow dafür, dass der gesamte Prozess der Datenvorbereitung intuitiv und benutzerfreundlich bleibt, was den gesamten Arbeitsablauf vereinfacht.

2.4.3 Datenexport für YOLO

Ein besonders nützliches Feature von Roboflow ist der Export der vorbereiteten Daten im richtigen Format für das Training von YOLO-Modellen. Nachdem die Daten annotiert und augmentiert wurden, ermöglicht Roboflow den Export der Bilder zusammen mit den dazugehörigen Label-Dateien. Diese Label-Dateien enthalten die Informationen zu den Bounding Box-Koordinaten und den entsprechenden Klassenlabels der Objekte in den Bildern. Im Fall von YOLO wird jedes Bild durch eine Textdatei begleitet, in der jede Zeile die Koordinaten einer Bounding Box, die Klasse, den x-Mittelwert, den y-Mittelwert, die Breite und die Höhe enthält. Roboflow stellt sicher, dass der Exportprozess für YOLO und auch andere Modelle reibungslos verläuft, sodass die Daten direkt für das Training des Modells genutzt werden können.

3 Prototyp

3.1 Hintergrund

Da das Praktikum, in welchem diese Arbeit verfasst wird, nur 18 Wochen dauerte, reicht es nicht aus, um eine vollständig funktionsfähige Software für alle Produkte der Firma Alukon KG zu implementieren. Um trotzdem greifbare Resultate zu erreichen, wurde ein Prototyp im kleineren Umfang implementiert. Dies hatte den Vorteil, dass wir uns im Rahmen der Entwicklung auf einen kleinen Rahmen von Produkten beschränken können, Damit konnte der Vergleich verschiedener Methoden vereinfacht.

3.2 Initiale Idee

Die grundlegende Idee dieses Prototyps ist es, ein Modell zur Instanz Segmentierung der auf der Traverse hängenden Produkte zu implementieren. Für den finalen Prototypen sollten zehn bis fünfzehn Klassen mit ca. 200 Bildern pro Klasse genügen. Somit sollten wir eine gute Menge an Bilder haben, um uns erste Einblicke in die Effizienz und Genauigkeit der Modelle zu verschaffen. Um erste Erkenntnisse mit den Prototypen zu gewinnen, wurden in der Arbeit nur bestimmte Produktteile für die visuelle Erkennung verwendet. Eine umfassendere Untersuchung hätte den zeitlichen Rahmen des Praktikums überschritten. Es wurden die folgenden Produktteile inkludiert:

- Blendkappen und Abdeckkappen
- Blenden in verschiedenen Winkelmaßen (90°, 45° und 20°)
- Häufig vorkommende Führungsschienen (A3, A20, A4, …)
- Grundschienen

Es war wichtig zu beachten, dass die Produktteile, die hier ausgewählt wurden, oft in der Produktion vorkommen. Dies ermöglicht ein leichtes Sammeln von Bildern der Traverse, da wir nicht warten müssen, bis ein bestimmtes Produktteil erscheint. Außerdem wurde nur eine kleine Anzahl an Produktteilen untersucht, da die Vielfalt an Produktteilen sehr groß

ist. Für einen Prototypen ist es nicht notwendig, so viele verschiedene Klassen zu berücksichtigen.

3.3 Datensammeln

Die Daten wurden per Hand im Betriebsbereich der Firma Alukon KG gesammelt, indem Bilder von den Traversen gemacht wurden. Um das Datensammeln zu vereinfachen, war die erste Idee keine Bilder zu machen, sondern Videos der Produkte zu filmen. Da ein Video technisch gesehen nur mehrere Bilder in einer Reihenfolge ist, kann man das Video später in mehrere Bilder extrahieren. Mit dieser Methode kann viel Zeit gespart werden, da wir nicht einzelne Bilder machen müssen. Jedoch hat sich im ersten Versuch an einen Prototyp herausgestellt, dass das Aufteilen der Videos in Bilder eine signifikante Abnahme in Bildqualität zur Folge hat. Nach dieser Erkenntnis wird deutlich, dass die Bilder einzeln geschossen werden müssen.

3.4 Erkennungstypen

Bei der Entwicklung eines Instanzsegmentierungsmodells zur Identifizierung von Produktteilen in der Alukon KG ist es entscheidend, verschiedene Methoden zur Vorhersage von Ergebnissen zu berücksichtigen. Dieser Abschnitt der Praxisarbeit wird drei unterschiedliche Ansätze zur Anwendung des Modells untersuchen: statische Bildanalyse, Videodateiverarbeitung und Echtzeit-Kameraerkennung. Jede Methode bietet einzigartige Vorteile und Herausforderungen im Kontext der Produktteilidentifikation.

3.4.1 Statische Bildanalyse

Die statische Bildanalyse beinhaltet die Anwendung des Prototyps auf einzelne Fotografien von Produktteilen. Diese Methode ist besonders nützlich für:

- Detaillierte Inspektion komplexer Baugruppen
- Qualitätskontrollprüfungen in bestimmten Produktionsphasen
- Erstellung einer Datenbank annotierter Produktbilder für zukünftige Referenzen

Bei der Verarbeitung statischer Bilder erzeugt das Modell eine Reihe von Masken oder Konturen, die jedes identifizierte Objekt umreißen, zusammen mit Klassenbezeichnungen und Konfidenzwerten. Somit wird die präzise Identifizierung und Messung einzelner Produktteile innerhalb des Bildes ermöglicht.

3.4.2 Videodateiverarbeitung

Die Videodateiverarbeitung erweitert die Fähigkeiten der statischen Bildanalyse auf eine Sequenz von Frames und ermöglicht es dem Modell, Produktteile über Zeitabschnitte zu analysieren[12]. Dieser Ansatz ist vorteilhaft für:

- Verfolgung von Teilebewegungen durch Montagelinien
- Identifizierung von Defekten in bewegten Produkten
- Analyse der Effizienz von Produktionsprozessen

Das Modell verarbeitet das Video einzeln pro Frame und wendet die Instanz Segmentierung auf jedes Bild in der Sequenz an. Dies führt zu einer neuen Videodatei mit hervorgehobenen erkannten Objekten und bietet eine umfassende Sicht auf die Teilidentifikation während der gesamten Aufnahme. Jedoch ist hier die Genauigkeit der Erkennung niedriger als bei der statischen Bildanalyse.

[12] Vgl. Szeliski (2010) S. 396ff.

3.4.3 Echtzeit-Kameraerkennung

Die Echtzeit-Kameraerkennung stellt die dynamischste Anwendung des Instanzsegmentierungsmodells dar. Diese Methode beinhaltet die Verarbeitung von Live-Video-Feeds von Kameras, die in der Produktionsumgebung installiert sind. Zu den wichtigsten Vorteilen gehören:

- Sofortiges Feedback zur Produktteilidentifikation
- Kontinuierliche Überwachung von Produktionslinien
- Möglichkeit, auf Probleme in Echtzeit zu reagieren

Für diesen Ansatz wird das Modell in einen Live-Videoeingang integriert, wie zum Beispiel mit einer IP-Kamera oder einer angeschlossene Gerätekamera. Das System verarbeitet den eingehenden Videostream in Echtzeit, wendet die Instanz Segmentierung auf jeden Frame an und liefert sofortige Ergebnisse. Die Effizienz dieser Erkennungsmethode hängt von der Qualität der verwendeten Kamera ab.

3.5 Erster Prototyp

Im ersten Versuch an einen Prototyp, wurde das Modell „yolo11n-seg“ von YOLO11 benutzt. Dieses Modell ist das leichtgewichtigste Modell für Instanz Segmentierung aus der Version YOLO11. Das Trainieren unseres Modelles verläuft mit „yolo11n-seg“ sehr schnell, jedoch ist dementsprechend die Genauigkeit geringer als bei anderen YOLO11 Modellvarianten. In diesem Versuch wurden 240 Bilder aufgeteilt auf vier Klassen benutzt. Die vier Klassen waren die folgenden: Blendekappe, Führungsschiene A20, Rollgeformte 20° und Rollgeformte 90°. Diese Klassen sind häufig auftretende Produkte in der Firma Alukon KG. Ein großes Problem bei diesem Versuch war die Qualität der Bilder. Dadurch, dass Videos der Traverse in Bilder aufgeteilt wurden, war die Bildqualität sehr schlecht und das Modell konnte dementsprechend keine guten Resultate erzielen. Hier wurden nur 50 Epochen an Trainings durchgeführt, da es nur als erster Einblick dienen sollte.

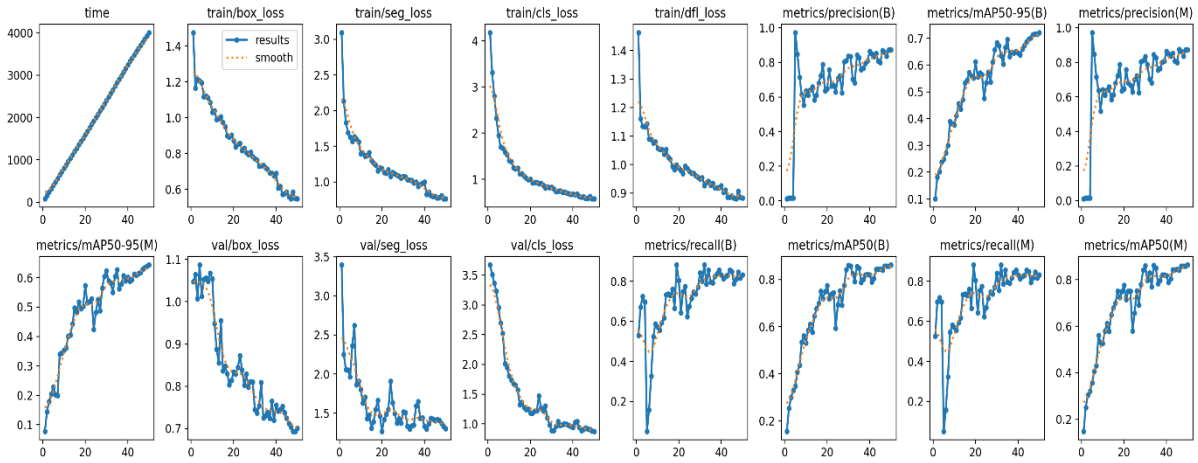

Abbildung 3: Trainingsresultate vom ersten Prototyp

Obwohl die Präzision und der Recall mit ca. 0,8 hoch sind, sind die tatsächlichen Resultate bei der Erkennung der Produktteile nicht sehr gut.

Ein Beispiel für die Erkennung ist die folgende Abbildung. Die Produkte an den Traversen werden zwar gut erkannt, jedoch gibt es auch falsche Erkennungen, wie z.B. die Blendkappen mit 0.44 im oberen rechten Bereich des Bildes. Hier wurde nämlich eine rechteckige Form im Hintergrund als Blendekappe erkannt.

Abbildung 4: Erkennungsresultate erster Prototyp

Es schaut zwar so aus, als ob das Modell schon sehr gut unsere Produkte erkennt, jedoch wird in Abbildung 5 klar, dass das Modell auch Objekte erkennt, die es gar nicht erkennen sollte. Ein Beispiel hierfür ist die Führungsschiene A20 mit einer Konfidenz von 0.57. In der nachfolgenden Abbildung handelt es sich hierbei um ein Plastikschild und nicht eine Führungsschiene.

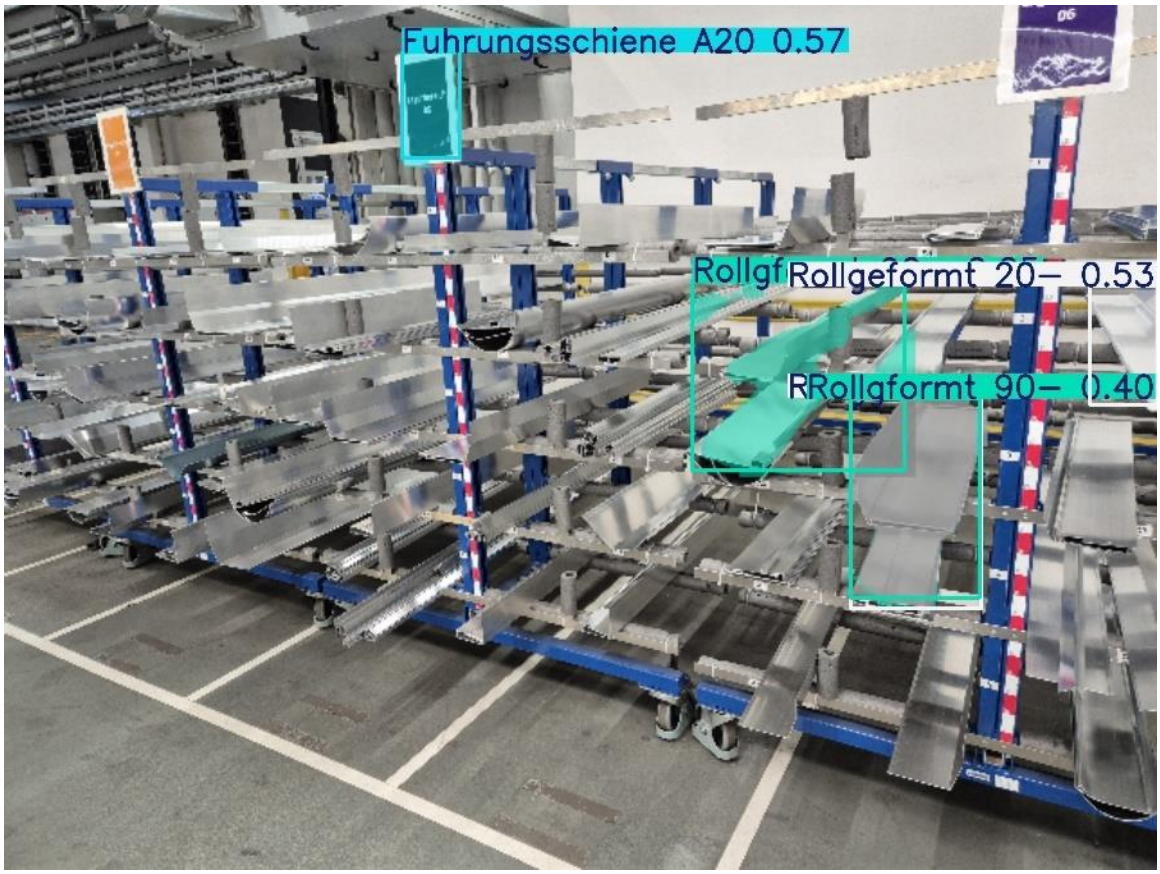

Abbildung 5: Fehlerhafte Erkennung

Trotz der schlechten Resultate für diesen Versuch war der erste Prototyp eine große Hilfe, um ein besseres Verständnis für die Vorgehensweise der nächsten Prototypen zu bekommen.

3.6 Zweiter Prototyp

Im Gegensatz zum ersten Versuch, wurde hier das „yolo11m-seg“ Modell von YOLO11 benutzt. Dieses Modell ist das mittelgroße Modell für die Instanz Segmentierung aus der YOLO11-Reihe. Aufgrund der höheren Anzahl an Parametern, welche beim Training verwendet werden, bietet dieses Modell eine höhere Genauigkeit bei der Segmentierung. Die Erhöhung der Parameter führt selbstverständlich auch zu einer längeren Trainingsdauer. In diesem Versuch wurde der Datensatz verbessert und die Bilder wurden alle per Hand im Produktionsbereich der Firma aufgenommen. Unser Datensatz besteht aus 699 Bildern und sieben Klassen. Die Klassen sind dieselben wie aus dem ersten Versuch, zusätzlich dazu wurden drei neue hinzugefügt: Führungsschiene A3, Führungsschiene A3/C10R und Rollgeformte 45°. Dadurch, dass jedes Bild per Hand aufgenommen wurde, hat sich die Bildqualität stark verbessert. Dementsprechend kamen hier wesentlich bessere Resultate als mit dem ersten Prototyp zustande. Resultate zu Vorschein. Jedoch gab es auch hier einige Probleme. Das erste Problem mit diesem Modell ist, dass eine der sieben Klassen ca. 100-mal öfter im Datensatz vorkommt als die anderen Klassen. Eigentlich sollten alle Klassen ungefähr gleich oft repräsentiert sein. Ein unbalancierter Datensatz kann dazu führen, dass das Modell die Minderheitsklasse schlechter erkennt und somit die Gesamtleistung des Modells beeinträchtigt. Obwohl der Datensatz nicht perfekt balanciert war, haben sich hier trotzdem gute Resultate ergeben.. Im Folgenden sind die Resultate nach 150 Epochen an Training zu sehen:

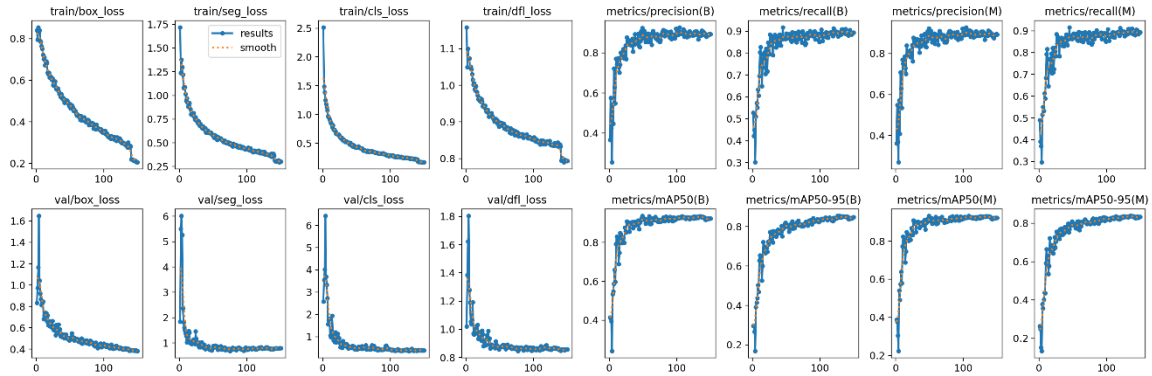

Abbildung 6: Trainingsresultate im Zweiten Versuch

Die Präzision und der Recall ist nun höher als im ersten Versuch. Es wird außerdem deutlich, dass auch 100 Epochen ausgereicht hätten, da keine signifikante Veränderung in den Epochen 100 bis 150 wahrgenommen werden kann.. Im zweiten Versuch ist die Erkennung um einiges besser als im ersten Versuch. Dies ist in der Abbildung 7 zu sehen. Die Konfidenzwerte sind höher und die Erkennung ist akkurater. Im Vergleich zur Abbildung 5 wird mit diesem Modell keine Blendekappe falsch erkannt, obwohl dasselbe Bild benutzt wurde.

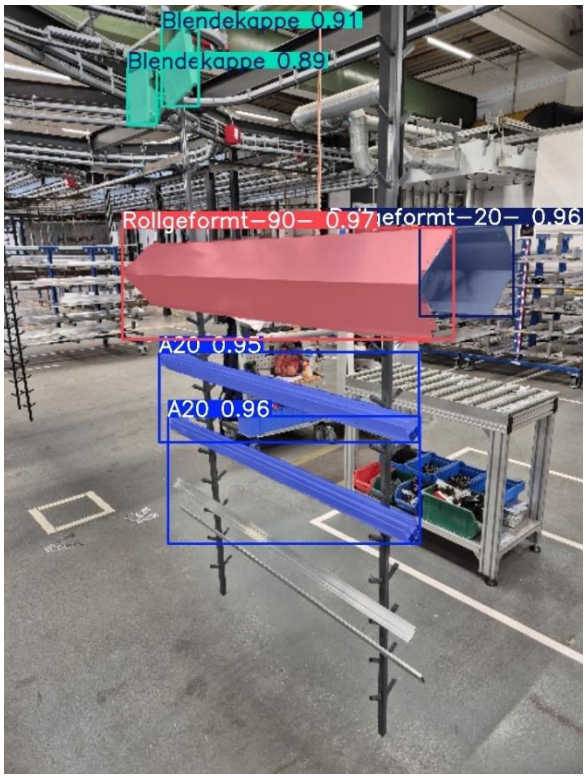

Abbildung 7: Erkennungsresultate im Zweiten Versuch

Das Modell kann außerdem Produkte in anderen Farben erkennen. Bis jetzt ist es beschränkt auf Farben wie braun, schwarz, weiß, dunkelblau und grau. Farben wie hellblau, grün oder orange werden nicht erkannt. Der genaue Grund dafür ist noch nicht klar. Möglicherweise liegt es daran, dass die dunkleren Farbtöne dem Aluminium der Produkte mehr ähnelt als die helleren Farben. Aufgrund der geringen Anzahl an Klassen tretet das Problem der Generalisierung auf. Das Modell versucht Produkte zu erkennen, welche nicht in unserem Modell als Klassen definiert sind. Solange ein Produkt den Produkten aus unseren Klassen ähnelt, versucht das Modell es zu erkennen. Dies wird gut deutlich in Abbildung 8. Hier werden Führungsschienen, die wir nicht als Klasse definiert haben, mit der Führungsschiene A20 verwechselt.

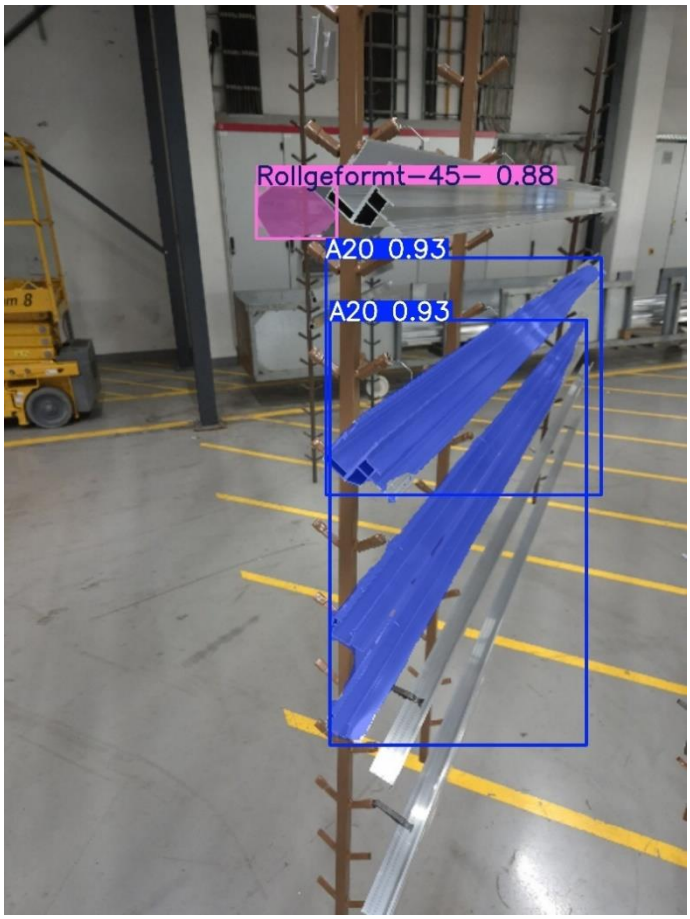

Abbildung 8: Falsche Erkennung im zweiten Versuch

Das größte Problem von den ersten beiden Versuchen, ist jedoch folgendes. Da der Produktionsbereich, in dem sich die Traversen befinden, sehr groß ist, ist im Hintergrund der Bilder fast immer viel unnötiges Hintergrundrauschen oder -informationen zu sehen, die für die Instanz Segmentierung nicht relevant sind. Die folgende Abbildung zeigt, dass ein Produkt, welches nicht auf der Traverse hängt, auch erkannt wird, weil keine Konfidenzschwelle gesetzt wurde. Dieses Problem kann behoben werden, indem die Konfidenzschwelle erhöht wird, welche zumindest eine temporäre, jedoch keine permanente Lösung für das Problem darstellt.

Abbildung 9: Fehlerkennung im Hintergrund

3.7 Weitere Prototypen mit verschiedenen Modellen

Neben den oberen drei beschriebenen Prototypen gab es noch weitere Versuche/Ideen für den Prototypen. Diese wurden erstellt, um verschiedene Modelle zu testen und zu vergleichen. Außerdem wurden hier einige Fragen aufgeklärt. Diese werden im Folgenden kurz erläutert:

3.7.1 Yolov8l-seg

Um den Unterschied zwischen Yolo11 und Yolov8 zu erkennen, wurde in diesem Versuch das „yolov8l-seg“ Modell benutzt. Der verwendete Datensatz war derselbe, wie im zweiten und dritten Versuch. Ein weiteres Ziel bei diesem Versuch war es, ein größeres Modell als in den oberen Versuchen zu benutzen. Es wurde Yolov8 anstatt Yolo11 ausgewählt, da der Speicherplatz von der zum Training verwendeten GPU nicht ausgereicht hat für “yolo11l-seg.pt“. Jedoch waren die Resultate hier sehr ähnlich zu den Resultaten aus dem zweiten Versuch und dementsprechend hat sich herausgestellt das kein signifikanter Unterschied besteht.

3.7.2 Yolo11n-seg mit mehr Epochen

Führen mehr Epochen automatisch zu besseren Endresultaten? Diese Frage sollte in diesem Versuch beantwortet werden. Um einen guten Vergleich zu kriegen, wurde das Modell erst mit 150 Epochen und dann noch einmal mit 300 Epochen trainiert. In beiden Fällen wurde der Datensatz mit 591 Bildern verwendet aber mit dem „yolo11n-seg“ Modell. In den nächsten beiden Abbildungen sind die Resultate dargestellt:

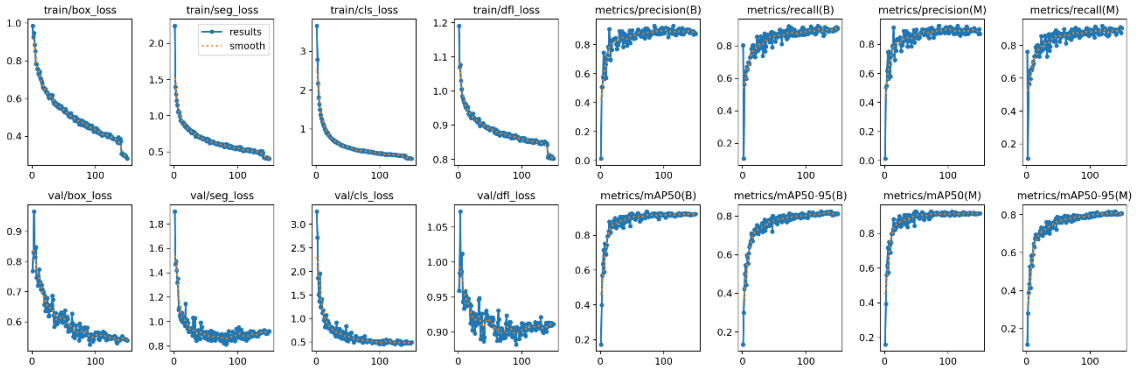

Abbildung 10: Resultate nach 150 Epochen

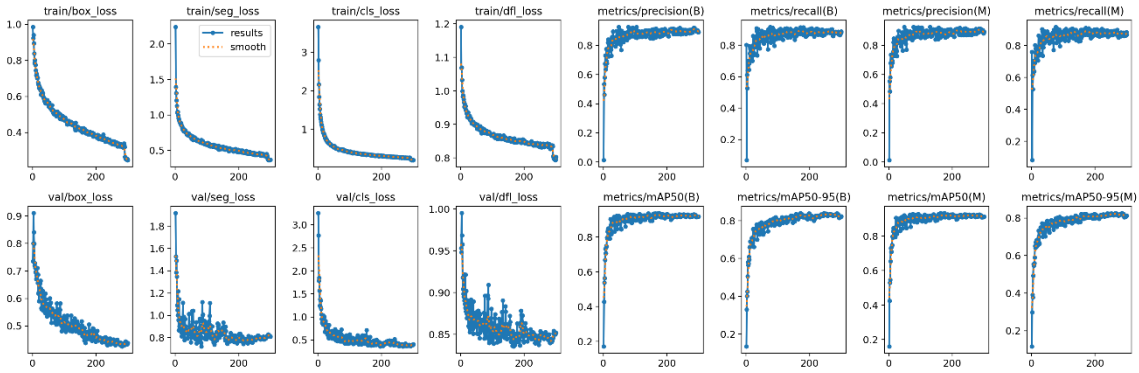

Abbildung 11: Resultate nach 300 Epochen

Es wird deutlich, dass sich die Resultate im Bereich 100 bis 150 nur sehr minimal verändern. Ab Epoche 150 findet keine Veränderung mehr statt. Somit ist nun klar, dass sich die Anzahl der Epochen nur bis zu einem bestimmten Punkt positiv auf die Trainingsresultate auswirkt. In den meisten Fällen reichen 100 bis 125 Epochen für ein Modell mit einem sehr spezifischen Anwendungsbereich wie diesen aus. Ein weiterer Nachteil von zu vielen Epochen ist das Auftreten vom sogenannten „Overfitting“. Hier fängt das Modell an, sich die Trainingsdaten einzuprägen, anstatt generalisierbare Muster zu lernen. Dies hat zur Folge, dass das Modell Schwierigkeiten haben wird, bestehende Produkte aus unbekannten Blickwinkeln zu erkennen.

3.7.3 Mask R-CNN

Das Region-based Convolutional Neural Network (R-CNN) ist eine Familie von Modellen zur Objekterkennung, die Bilder in Regionen unterteilen, um Objekte präzise zu lokalisieren. Während das ursprüngliche R-CNN noch langsam war, verbesserten spätere Versionen wie Faster R-CNN die Geschwindigkeit erheblich[13]. Mask R-CNN baut auf Faster R-CNN auf und erweitert es um die Fähigkeit zur Instanz-Segmentierung, wodurch nicht nur Objekte erkannt, sondern auch pixelgenaue Masken für jedes Objekt generiert werden[14]. Diese zusätzliche Funktionalität macht Mask R-CNN leistungsfähiger als frühere Modelle, bringt aber auch höhere Rechenanforderungen mit sich. Beim Training des Mask R-CNN-Modells traten bedeutende Herausforderungen auf. Trotz seiner fortschrittlichen Fähigkeiten zur Instanz-Segmentierung, die eine präzise Abgrenzung von Objekten in Bildern ermöglicht, war der Trainingsprozess sehr komplex und zeitaufwendig. Es dauerte einen ganzen Tag, bis das Mask R-CNN-Modell überhaupt zum Laufen gebracht wurde, und dennoch waren die Ergebnisse schlecht. Zum Beispiel dauerten 25 Epochen etwa 1 Stunde und 30 Minuten, während 150 Epochen für das YOLO11-Segmentation-Modell (yolo11m-seg) die gleiche Zeit in Anspruch nahmen. Die Präzision und der Recall des Mask R-CNN-Modells lagen bei etwa 0,6, was aufgrund der geringen Anzahl an Epochen erwartet wurde. Allerdings war das Modell aufgrund der Komplexität bei der Einrichtung nicht zufriedenstellend. Die Konfiguration von Mask R-CNN umfasst die Einrichtung des Region Proposal Network (RPN) sowie die Verwaltung des zusätzlichen Rechenaufwands, der mit der Generierung von Segmentierungs-Masken neben den Begrenzungsrahmen verbunden ist. Das RPN trägt dazu bei, den Rechenaufwand zu reduzieren, da es gezielt relevante Bereiche im Bild identifiziert, anstatt das gesamte Bild auf mögliche Objekte zu scannen. Dennoch bleibt der gesamte Prozess rechenintensiv und kompliziert. Obwohl Mask R-CNN leistungsstarke Segmentierungsfähigkeiten bietet, machen die Benutzerfreundlichkeit und Effizienz von YOLO es zu einer geeigneteren Wahl, um zuverlässige Ergebnisse zu erzielen, ohne sich mit den komplexen Anforderungen des Mask R-CNN-Trainings auseinanderzusetzen.

[13] Vgl. Elgendy (2020) S. 317ff.
[14] Vgl. Shanmugaman (2018) S. 154ff.

3.7.4 Detectron2

In unserem letzten Versuch wurde auch das Detectron2-Modell für die Bildsegmentierung getestet. Es ist ein leistungsstarkes Deep-Learning-Framework von Facebook AI Research für Objekterkennung, Instanz Segmentierung und Keypoint Detection. Es wird in vielen Industriellen und akademischen Anwendungen genutzt, z. B. für autonomes Fahren, medizinische Bildverarbeitung und Überwachungssysteme[15]. Allerdings traten bei diesem Versuch die gleichen Probleme auf wie beim Mask R-CNN. Die anspruchsvolle Trainingskomplexität und die notwendige Konfiguration machten die Implementierung herausfordernd und unterstrichen erneut die Praktikabilität von YOLO in diesem Kontext.

[15] Vgl. Seite "FAIR's platform for object detection and semantic segmentation" URL: https://paperswithcode.com/lib/detectron2 (Abgerufen am 05.11.2024, 09:18 UTC)

3.8 Finaler Prototyp

3.8.1 Datensatz

Die Idee bei diesen Prototypen war es, eine Kamera in der Produktion aufzustellen, um die Bilder für den Datensatz zu sammeln. Mithilfe von einem festen Winkel kann ein einheitlicher Datensatz erstellt werden. Generell ist ein einheitlicher Datensatz nicht geeignet für ein visuelles Erkennungsmodell, da es eigentlich gewünscht ist, dass das Modell aus verschiedenen Winkeln und mit verschiedenen Hintergründen trotzdem gut erkennen kann. Jedoch für diesen Fall bietet sich ein einheitlicher Datensatz an, da die Erkennung der Traversen immer an derselben Stelle passieren wird.
Im ersten Schritt wurde mithilfe vom Werkzeugbau ein dunkler Behang hinter den Traversengang aufgehangen. Mithilfe von diesem Behang, werden die Probleme aus den vorherigen Prototypen behoben. Dadurch, dass nun der Hintergrund dunkel ist, werden keine Objekte im Hintergrund ungewollt erkannt. Außerdem ist es so leichter die Traversen zu erkennen. Danach wurde die Kamera aufgebaut. Es wurde die folgende Kamera verwendet: „Axis Q1765-LE Network Camera“.

Abbildung 12: Axis Network Camera

Diese Kamera wird meistens für die Überwachung und Sicherheit von Firmen, Grundstücken, Häusern oder Geländern verwendet. Die Position der Kamera wurde so gewählt, dass sie schräg auf die Traverse schaut. Somit kann die Kontur der Produktteile gut erkannt werden. Die Kamera verfügt über einen eingebauten Bewegungssensor, welcher erst so eingestellt werden musste, dass er immer dann ein Bild macht, wenn eine Traverse im Bild sichtbar ist. Dadurch, dass die Kamera jedoch nicht unterscheiden kann zwischen Traverse und anderen Objekten, hat sie oft Bilder gemacht, wenn ein Mitarbeiter an der Kamera vorbeigelaufen ist. Damit diese Bilder nicht mit in dem Datensatz inkludiert werden, wurden Bilder mit Menschen mithilfe von einem Python Skript automatisch ausgefiltert. Das Python Skript hat das Yolov8 Modell benutzt und den „COCO"-Datensatz. Dieser Datensatz ist einer der größten und bekanntesten Datensätze für Aufgaben wie Objekterkennung, Bildsegmentierung und weiteres. Er enthält über 300 Tausend Bilder und wurde benutzt, um die Klasse „Human" in den Bildern zu erkennen. Das Skript iteriert durch die geschossenen Bilder und speichert die Bilder mit Menschen auf dem Bild in einen separaten Ordner. Somit ist der Datensatz um einiges einheitlicher und enthält nur die für den Prototypen wichtige Bilder.
Insgesamt wurden innerhalb von acht Tagen ca. 2500 Bilder mit der Kamera automatisch gemacht. Es wurden jedoch nur ca. 500 Bilder benutzt. Mithilfe von der Bildaugmentierung von Roboflow waren am Ende 1273 Bilder mit 19 Klassen im finalen Datensatz vertreten.

Ein typisches Bild aus dem Datensatz schaut wie folgt aus:

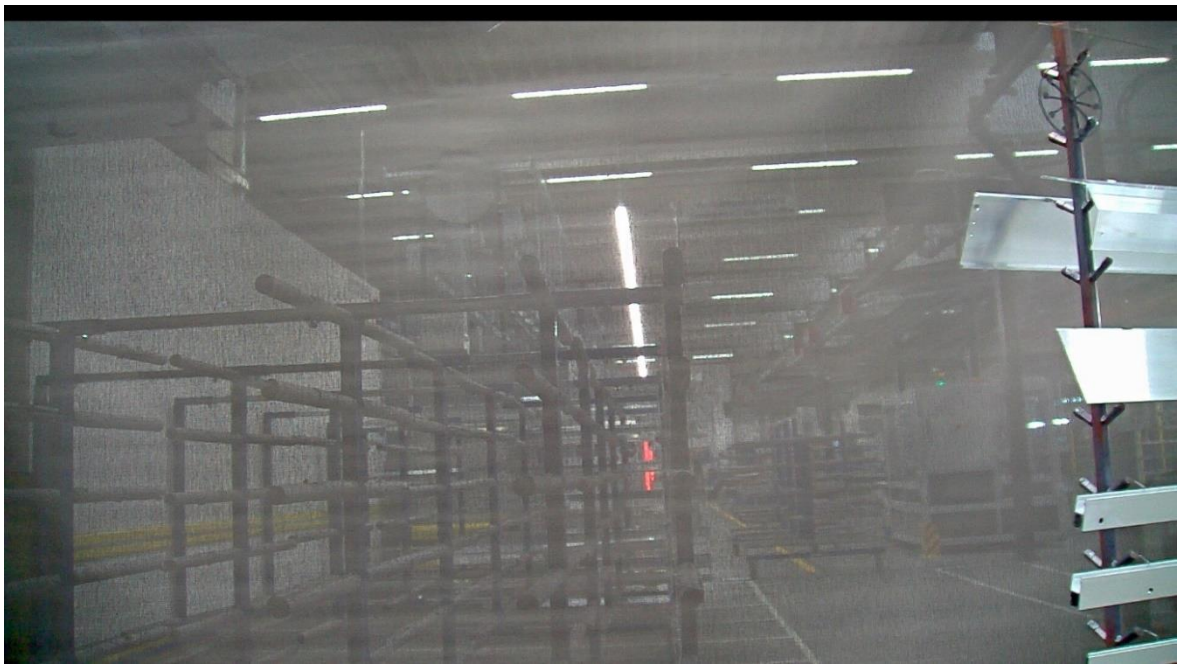

Abbildung 13: Beispiel Datensatz im finalen Prototyp

Aufgrund des Platzmangels und der begrenzten Zeit war es nur an dieser Stelle in der Produktion möglich, die Kamera zu platzieren. Im Idealfall müsste die Kamera etwas schräger auf die Traverse schauen, um einen besseren Winkel zu erzeugen.
Dadurch, dass bestimmte Produktteile öfter produziert werden, als andere, hat sich ein nicht balancierter Datensatz ergeben. Die A3-Führungsschiene, die flachen Blenden und die Grundschienen kamen deutlich häufiger vor als der Rest der Klassen.
Es ist außerdem noch wichtig zu bemerken, dass für diesen Prototypen nicht immer die richtigen Namen für die Produktteile verwendet wurden und es wurde auch nicht perfekt differenziert zwischen bestimmten Produktteilen. Manche Blenden sind zwar eigentlich etwas anders in der Form, jedoch wurden sie in eine Obergruppe eingeteilt. Der Grund dafür ist, dass es in diesem Fall wichtiger war, die Erkennung zu vergleichen, als großen Wert auf die korrekten Namen zu legen.

3.8.2 Training

Der Trainingsprozess wurde schrittweise ausgeführt und in sechs Trainingseinheiten aufgeteilt. Jede Trainingseinheit wurde mit dem „yolo11m-seg“ Modell trainiert. Pro Trainingseinheit wurden immer ca. 100 neue Bilder dem Datensatz hinzugefügt, bevor er trainiert wurde. Es wurde jede Einheit für 100 Epochen trainiert, ohne eine Konfidenzschwelle zu setzen.

i. Erste Einheit

In der ersten Einheit wurde das Modell mit 236 Bildern trainiert und hat ca. 20 Minuten gedauert. Die Resultate waren nicht gut. Die Erkennung war nicht akkurat und es wurde viel generalisiert, da der Datensatz nicht balanciert war. In der kommenden Abbildung ist zu erkennen, dass die Führungsschienen nicht korrekt erkannt werden und sogar mit einer 20° Blende verwechselt wurde. Es wurde direkt klar, dass mehr Bilder notwendig sind, um ein besseres Fazit zu ziehen.

Abbildung 14: Beispiel aus der ersten Einheit

ii. Zweite Einheit

Der zweite Versuch wurde mit 431 Bildern trainiert und hat 37 Minuten gedauert. Hier wurden kleine Verbesserungen deutlich, da die Erkennung genauer wurde. Die Generalisierung der Klassen war immer noch ein bestehendes Problem, da der Datensatz nicht gut aufgeteilt ist. Die Klassen, die am häufigsten im Datensatz vertreten sind, wie zum Beispiel die A3-Führungsschiene wurden schon mit hoher Genauigkeit erkannt. Hier wurde deutlich, dass ein balancierter Datensatz notwendig ist, um eine sehr gute Erkennung zu gewährleisten.
Die akkurate Erkennung der am häufigsten Produkte wird in der Abbildung 15 sehr gut deutlich. Diese Produkte haben ungefähr eine Erkennungsgenauigkeit von ca. 90%.

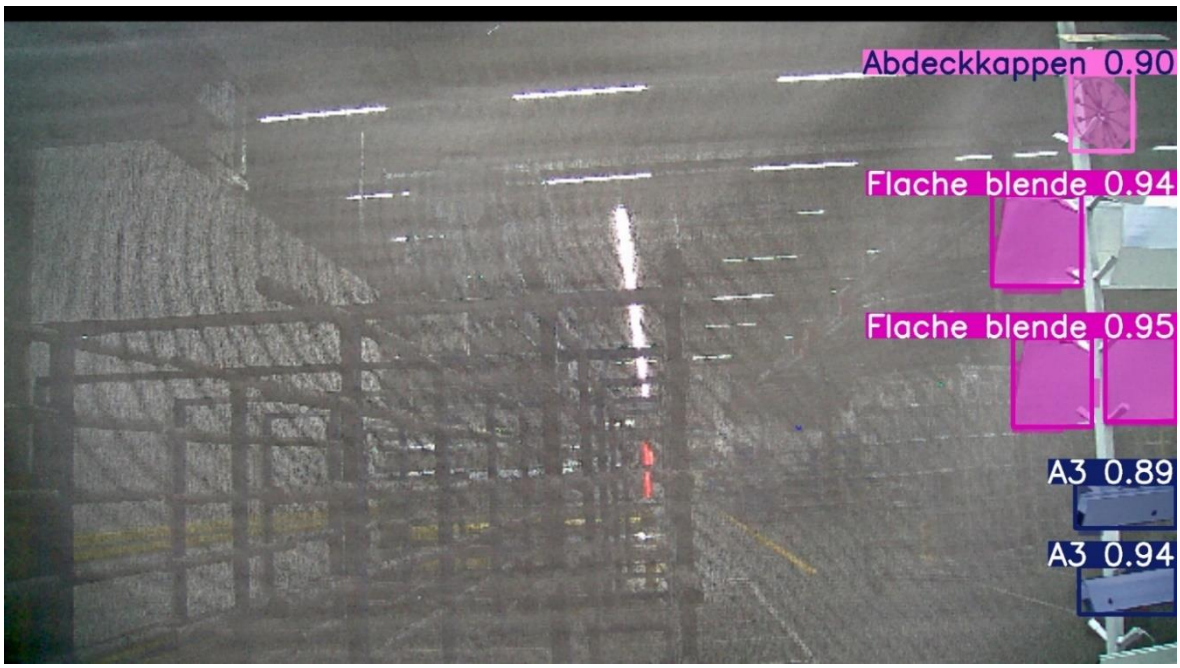

Abbildung 15: Beispiel aus der zweiten Einheit

iii. Dritte Einheit

Die dritte Einheit bestand aus 678 Bildern und wurde in 54 Minuten trainiert. Hier ist kein großer Unterschied zum zweiten Versuch deutlich. Sowohl die Erkennungsmatrix als auch die Erkennungen in den Testbildern sehen sehr ähnlich aus. Dementsprechend werden mehr Bilder hinzugefügt, um zu sehen, ob sich etwas verbessert.

iv. Vierte Einheit

Der nächste Versuch beinhaltete 931 Bilder und hat ca. 70 Minuten gedauert. Die Erkennungsgraphen sahen wieder ähnlich aus, jedoch hat sich die Erkennung an den Testbildern etwas verbessert. Es bestehen dieselben Probleme aus Versuch Zwei und Drei. Der Datensatz ist nicht balanciert und somit ist die Erkennung der weniger vertretenen Produktteile ungenauer. In der nächsten Abbildung wird deutlich, dass die Erkennung genau ist, obwohl die Produkte sich schon viel weiter links befinden und die Konturen nicht so gut zu erkennen sind.

Abbildung 16: Beispiel aus der vierten Einheit

v. Fünfte Einheit

In der vorletzten Einheit wurde das Modell mit 1123 Bildern in 1h20min trainiert. Die Blenden und Abdeckkappen werden nun sehr gut erkannt. Dasselbe gilt auch für die A3-Führungsschiene, jedoch gab es ein Paar Spezialfälle, in denen es nicht der Fall war. Es wurde häufiger zwischen Führungsschienen und Grundschienen generalisiert, da es aus dieser Perspektive für das menschliche Auge keinen Unterschied gibt.

Abbildung 17: Beispiel aus der fünften Einheit

Wie in Abbildung 17 zu sehen, wird die A3-Führungsschiene als eine Durchlaufschiene erkannt, was eine Falscherkennung ist.

vi. Letzte Einheit

Die sechste Einheit beinhaltete 1273 Bilder und hat 1h35min gedauert. Es war keine signifikante Veränderung zu sehen, da sowohl die Trainingsgraphen als auch die Erkennungen der Testbilder sehr ähnlich blieben. Das Problem der Generalisierung kann

nur behoben werden, indem die Produkte im Datensatz ausgewogen vertreten werden. Es kann temporär eine Konfidenzschwelle von 0.75 gesetzt werden, um die Fehlerkennungen zu beheben, jedoch ist dieser Ansatz keine gute Lösung.

vii. Zusätzliche Einheit

Zuletzt wurde noch ein Modell auf Roboflow trainiert. Hierfür wurde nicht YOLO11, sondern das Roboflow 3.0 Modell benutzt. Die Resultate fielen etwas schlechter aus als bei den vorherigen Versuchen. Auch der Erkennungsgraph war ungenauer als davor. Jedoch wurde durch das Trainieren auf Roboflow die Funktion vom „Auto-Labeling“ freigeschaltet. Das bedeutet, dass zukünftige Bilder, die in den Datensatz kommen sollen, mithilfe von künstlicher Intelligenz annotiert werden. Somit kann man sich sehr viel händische Arbeit sparen. Die künstliche Intelligenz versucht anhand von dem auf Roboflow trainierten Modell die Klassen durch eigenständiges Lernen zu annotieren. Nach dem automatischen Annotieren müssen die Annotationen von einem Menschen überprüft werden. Dies verringert die Fehlergenauigkeit, da der Mensch sich nicht viel anstrengen muss und nur die Annotationen überprüft. Da das Hinzufügen von neuen Bildern in den Datensatz nicht die bestehenden Probleme beheben wird, wurde die Entscheidung getroffen keine weiteren Versuche auszuüben.

3.8.3 Fazit

Als Schlussfolgerung lässt sich sagen, dass der Ansatz mit der Kamera und dem automatischem Datensammeln die beste Idee ist. Hier kann eine große Menge an Daten gesammelt werden, ohne dass jemand in der Produktion jede einzelne Traverse per Hand abfotografieren muss. Jedoch ist eine gute Umsetzung nicht möglich gewesen in dem Rahmen des Praktikums. Aufgrund von räumlichen und zeitlichen Begrenzungen haben sich nur diese Ergebnisse ergeben. Trotzdem sind die Ergebnisse sehr hilfreich für die Zukunft. Mit einer guten Umsetzung kann hier ein fast voll automatisches Erkennungssystem implementiert werden. Es ist notwendig, eine bessere Kameraqualität sicherzustellen und den Datensatz wirklich balanciert zu gestalten. Jede Klasse sollte ungefähr gleich oft vorkommen im Datensatz. Außerdem wären ein dunklerer Hintergrund und bessere Lichtverhältnisse wichtig für eine bessere Erkennung. Wenn diese Bedingungen gewährleistet sind, sollte die Erkennung sehr akkurat sein.
Die Funktion vom „Auto-Labeling“ weist sich als sehr hilfreich heraus und sollte definitiv mit eingebaut werden in eine vollständig funktionierende Implementierung. Somit können riesige Mengen an Bildern in kurzer Zeit annotiert werden, ohne dass jedes Bild per Hand beschriftet werden muss.

3.9 Positionserkennung

Neben der Produkterkennung war auch das Erkennen der Produktposition auf den Traversen ein Thema, das behandelt wurde. Da der Hauptfokus auf der Produkterkennung lag, sind die Ergebnisse in diesem Bereich aufgrund von Zeitmangel nicht so detailliert wie der Rest. Die ursprüngliche Idee bestand darin, die Positionen der Traversen mithilfe von Lasersensoren zu bestimmen. Allerdings erwies sich diese Methode als zu aufwendig für den zeitlichen Rahmen der Praxisarbeit, weshalb eine alternative Lösung gesucht wurde. Um die Position der Produktteile an den Traversen dennoch zu bestimmen, wurde ein Skript in Python unter Verwendung der OpenCV-Bibliothek entwickelt. Dieses Skript zeichnet Linien auf der Höhe der Traversenpositionen und nutzt anschließend die Instanz Segmentierung, um die x-Werte der Linien zu analysieren. Dies funktionierte sehr gut, da der Datensatz standardisiert war und die Traversenpositionen auf jedem Bild gleich waren. Auf dieser Basis konnte die genaue Position der Produktteile auf den Traversen berechnet werden.

Abbildung 18: Positionserkennung der Traversen

Grundsätzlich ist dies ein guter Ansatz, um die visuelle Erkennung mit der Positionserkennung zu kombinieren, ohne zusätzliche Geräte, wie Lasersensoren, besorgen zu müssen. Die genaue Position der Linien muss höchstwahrscheinlich noch etwas verfeinert werden, damit die Positionen sehr akkurat erkannt werden. Diese Methode funktioniert jedoch nur sehr gut, weil die Bilder immer aus demselben Winkel aufgenommen werden und dementsprechend die Positionen der Traverse von Bild zu Bild immer gleich bleiben. Sollte die Position, aus der die Kamera das Bild macht, sich verändern, dann muss das Skript so konfiguriert werden, dass die Linien genau auf der richtigen Höhe gezeichnet werden. Das gute an dieser Lösung ist das leichte zusammenbinden der Positionserkennung und der Produkterkennung, da beides mit der Programmiersprache Python implementiert wurde.

3.10 Schlussfolgerung

Der finale Prototyp mit dunklem Hintergrund und automatischer Datensammlung hat sich als die beste Lösung erwiesen. Er zeigte das größte Potential und erwies sich im Vergleich zu anderen Prototypen als leichter implementierbar. Außerdem ist das Sammeln von Bildern für den Datensatz hier am einfachsten, da die Kamera die Bilder automatisch mithilfe vom Bewegungssensor aufnimmt. Die erzielten Ergebnisse waren bereits vielversprechend und zeigten eine hohe Genauigkeit. Mit einer weiterführenden Optimierung und einem umfassenderen Ausbau könnte die Leistung des Systems noch weiter verbessert werden, sodass sich in Zukunft noch präzisere und zuverlässigere Ergebnisse erzielen lassen.

4 Fazit und Ausblick

Die Entwicklung der verschiedenen Prototypen war ein sehr interessantes und lehrreiches Projekt, das wertvolle Einblicke in das Thema maschinelles Lernen geliefert hat. Fast alle getesteten Prototypen erzielten gute Ergebnisse, wobei sich YOLO als das beste Modell für diese Anwendung herausgestellt hat. Die Arbeit an diesem Projekt hat nicht nur theoretisches Wissen vertieft, sondern auch praktische Erfahrung im Bereich der KI-gestützten Objekterkennung vermittelt. Um die Ergebnisse weiter zu optimieren, gibt es mehrere Möglichkeiten zur Verbesserung und Erweiterung:

i. Datensatz

Der Datensatz muss auf jedes einzelne Produktteil aus dem Produktkatalog erweitert werden. Jede Klasse sollte gleich oft vorkommen, damit ein balancierter Datensatz vorhanden ist. Dies sorgt für eine bessere Modellgenauigkeit.

ii. Modell

Leistungsstärkere Modelle wie z.B. „yolo11x-seg“ nutzen, um bessere Erkennungsgenauigkeit zu erreichen. Dementsprechend ist auch mehr Rechenleistung nötig, um solch ein Modell verwenden zu können.

iii. Kamera

Der Einsatz von Time-of-Flight Kameras, wie z.B. die Fempto Bolt Kamera von Orbbec. Sie ermöglichen eine präzisere Tiefenerkennung und sind weniger anfällig für schwierige Lichtverhältnisse. Dadurch können sie genauere Bilddaten liefern, was insbesondere für die Erkennung komplexer Formen und Strukturen von Vorteil ist. Ihre hohe Genauigkeit

und Robustheit machen sie zu einer effektiven Lösung für die automatische Datensammlung in der Produktionsumgebung[16].

iv. Vergleich mit Produktdaten

Entwicklung eines Programms, das Produktdaten, zum Beispiel dessen Name und Position, aus der Alukon Datenbank mit den erkannten Daten der visuellen Erkennung abgleicht. Diese Abfrage ist unbedingt zu implementieren, da die Visuelle Erkennung allein nicht als Ergebnis ausreichend ist.

v. Erkennungsbedingungen

Das Aufstellen einer Erkennungsstation, die darauf ausgelegt ist, perfekte visuelle Bedingungen für die Erkennung zu gewährleisten, ist definitiv von Vorteil, da somit die Genauigkeit der Erkennung verbessert werden kann. Eine Idee ist es, einen dunklen Tunnel aufzubauen, in den die Traverse läuft und stehenbleibt, bis die Erkennung fertig ist. Hier können mehrere Kameras benutzt und die Lichtverhältnisse optimiert werden.

Mit diesen Erweiterungen kann die Genauigkeit und Effizienz des Systems weiter gesteigert werden, sodass es in Zukunft eine noch größere praktische Relevanz in der Produktionsumgebung von Alukon erhält.

[16] Vgl. Hansard et al. (2012) Introduction

5 Literaturverzeichnis

Elgendy, Mohamed: Deep Learning for Vision Systems, Manning Publications, Shelter Island, NY, USA 2020

FAIR's platform for object detection and semantic segmentation“ URL: https://paperswithcode.com/lib/detectron2 (Abgerufen am 05.11.2024, 09:18 UTC)

Dwyer Brad / Gallagher James: “Getting Started with Roboflow” URL: https://blog.roboflow.com/getting-started-with-roboflow/ (Abgerufen am 01.11.2024, 08:34 UTC)

Goodfellow, Ian/Bengio, Yoshua/Courville, Aaron: Deep Learning. Adaptive Computation and Machine Learning, The MIT Press, Cambridge, MA, USA 2016

Hansard, Miles/Lee, Seungkyu/Choi, Ouk/Horaud, Radu: Time-of-Flight Cameras: Principles, Methods and Applications, Springer, London, UK 2012

„Instanz Segmentierung“ URL: https://docs.ultralytics.com/de/tasks/segment/ (Abgerufen am 05.11.2024, 14:13 UTC)

Murphy, Kevin P.: Machine Learning: A Probabilistic Perspective, The MIT Press, Cambridge, MA, USA 2012

Redmon, Joseph/Divvala, Santosh/Girshick, Ross/Farhadi, Ali: You Only Look Once: Unified, Real-Time Object Detection, URL: https://arxiv.org/abs/1506.02640 (Abgerufen am 30.10.2024, 11:15 UTC)

Shanmugamani, Rajalingappaa: Deep Learning for Computer Vision, Packt Publishing, Birmingham, UK 2018

Szeliski Richard: Computer Vision: Algorithms and Applications (Texts in Computer Science), Springer, London, UK 2010

Buhl Nikolaj: “Training, Validation, Test Split for Machine Learning Datasets” URL: https://encord.com/blog/train-val-test-split/ (Abgerufen am 31.10.2024. 09:43 UTC)

“Vertiefung der Leistungsmetriken” URL: https://docs.ultralytics.com/de/guides/yolo-performance-metrics/ (Abgerufen am 31.10.2024, 11:02 UTC)

Eidesstattliche Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde nach meiner besten Kenntnis bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Hof, den 10.03.2025 Unterschrift