

# Todo list

Some todo . . . . .	VII
Nochmal neu formulieren? . . . . .	3
Nochmal neu formulieren? . . . . .	3
Tabelle mit Verteilung einfügen . . . . .	9
Ablaufdiagramm einfügen . . . . .	13
Bsp. Bild für jede Klasse zeigen (aus Paper nehmen!; Quellenverlinkung für text und bild nicht vergessen) . . . . .	14
Bsp. Bild für jede Klasse zeigen (aus Paper nehmen!; Quellenverlinkung für text und bild nicht vergessen); habe alle 16 Klassen für das Training genutzt, wichtig sind trotzdem nur Large Vehicle, Plane, Ship, small vehicle . . . . .	14
Beispielbilder einfügen . . . . .	15
Vergleichsbilder (Schiff und Auto Vergleich) einfügen . . . . .	15
Result.png einfügen? oder doch sein lassen? . . . . .	16
text\todo{Ref zur Formatbeschreibung einfügen} . . . . .	17
Verweis auf entsprechende Funktion . . . . .	18
ref einfügen . . . . .	19
ref einfügen . . . . .	19
ref einfügen . . . . .	21
ref funktion dafür . . . . .	21
ref funktion einfügen . . . . .	21
ab jetzt nur für analyse . . . . .	22

---

Vergleichsbild einfügen oder verweis . . . . .	23
Bild für Vergleich der BB Area einfügen (zw. obb, aab aab old) . . . . .	23
Bild einfügen best val dataset on val data . . . . .	24
Bild einfügen; best val dataset on test data . . . . .	24
Tabelle einfügen . . . . .	24
beide confusionsmatricen einfügen . . . . .	25
Differenzmatrix RGBIR vs RGIR einfügen . . . . .	25
Differenzmatrix RGBIR vs RGB einfügen . . . . .	25
Differenzmatrix RGBIR vs IRGB einfügen . . . . .	25
Differenzmatrix RGBIR vs RIRB einfügen . . . . .	26
Differenzmatrix RGBIR vs GBNDBVI und RGBNDVI einfügen . . . . .	26

# Deep Learning for Small Vehicle Detection and Classification on High-Resolution Multispectral Remote Sensing Imagery

MASTER'S THESIS  
in partial fulfilment of the requirements for the degree of  
MASTER OF SCIENCE

University of Münster  
Institute for Geoinformatics

Supervisor:  
*Sebastian Thiele*

First assessor:  
*Prof. Dr. Benjamin Risse*

Second assessor:  
*Dr. Christian Knoth*

Submitted by:  
*Timo Lietmeyer*

Münster, August 2025



# Deep Learning for Small Vehicle Detection and Classification on High-Resolution Multispectral Remote Sensing Imagery

---

Deep Learning zur Detektion und Klassifikation kleiner  
Fahrzeuge auf hochauflösenden multispektralen  
FERNERKUNDUNGSBILDERN



## Abstract

This document was compiled in final mode but can also be compiled in draft mode with an extra big margin for todo notes. To enable this, goto `preamble.tex` and replace the line `line draft=false` with `final`. Remember to switch it of when compiling your thesis for printing.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Some todo





# Contents

<b>1. Introduction</b>	<b>1</b>
1.1. Motivation . . . . .	1
1.2. Structure of the thesis . . . . .	2
<b>2. Fundamentals</b>	<b>3</b>
2.1. Computer Vision . . . . .	3
2.2. Machine Learning . . . . .	3
2.3. Deep Learning . . . . .	4
2.3.1. Backpropagation . . . . .	7
2.3.2. Vanishing . . . . .	8
2.3.3. Exploding Gradients . . . . .	8
2.3.4. Hyperparameter . . . . .	8
2.3.5. Overfitting . . . . .	8
2.4. YOLOv9 . . . . .	8
2.5. Evaluation Metrics . . . . .	8
2.5.1. N Fold Cross Validation . . . . .	8
2.5.2. Mean Average Precision . . . . .	9
<b>3. State of research</b>	<b>11</b>
<b>4. Methodology</b>	<b>13</b>
4.1. Preprocessing of input data . . . . .	13
4.2. YOLOv9 . . . . .	13
4.2.1. YOLOv9u . . . . .	13
4.2.2. YOLOv9e . . . . .	13
4.2.3. Hyperparameter . . . . .	13
4.3. Database . . . . .	13
4.3.1. Vehicle Detection on Aerial Images (VEDAI) Dataset . . . . .	13
4.3.2. Dataset for Object deTection in Aerial Images (DOTA) 1.5 . . . . .	14
4.4. Umsetzung . . . . .	14
4.4.1. PALMA . . . . .	14

4.4.2.	Challenges Preprocessing . . . . .	15
4.5.	Workflow . . . . .	15
4.5.1.	Axis Aligned vs. Oriented Bounding Boxes . . . . .	15
4.5.2.	Channel Permutation . . . . .	16
4.5.3.	DOTA Training . . . . .	16
4.5.4.	Ablation Studie . . . . .	16
<b>5.</b>	<b>Implementation</b>	<b>17</b>
5.1.	Preprocessing . . . . .	17
5.1.1.	DOTA Dataset . . . . .	17
5.1.2.	VEDAI Dataset . . . . .	18
5.1.3.	Fold Creation . . . . .	20
5.1.4.	Fold-Erstellung über create_own_folds . . . . .	20
5.2.	Python Pakete . . . . .	22
5.2.1.	Computer Vision 2 . . . . .	22
5.2.2.	Numpy . . . . .	22
5.2.3.	Scipy (nur Convexhull) . . . . .	22
5.2.4.	os . . . . .	22
5.2.5.	Matplotlib . . . . .	22
5.2.6.	Seaborn . . . . .	22
5.2.7.	pandas . . . . .	22
<b>6.</b>	<b>Results</b>	<b>23</b>
6.1.	Oriented Bounding Boxes and Axis Aligned Bounding Boxes . . . . .	23
6.2.	Comparison at Mean Average Precision 50-95 . . . . .	24
6.3.	Comparison with Confusion Matrices . . . . .	25
6.4.	Ablation Studies . . . . .	26
<b>7.</b>	<b>Discussion</b>	<b>27</b>
<b>8.</b>	<b>Conclusion and Outlook</b>	<b>29</b>
<b>A.</b>	<b>Appendix</b>	<b>31</b>
	<b>List of Figures</b>	<b>33</b>
	<b>List of Tables</b>	<b>35</b>

# 1 | Introduction

## 1.1. Motivation

Durch die leichte Verfügbarkeit von hochauflösenden Satelliten- und Luftbildern ist Detektion von Objekten weltweit möglich. Da außerdem die hohe Rechenleistung, die für maschinelles Lernen erforderlich ist, immer preiswerter wird, wird die Anwendung von künstlicher Intelligenz zur vollautomatischen Analyse immer leichter.

Das Monitoring von zerstörten (militärischen) Gerät oder Gebäudestrukturen in Konfliktregionen, wie Darfur [1] oder der Ukraine, kann durch künstliche Intelligenz unterstützt werden. Hier können Luftbilder gut genutzt werden, weil das Erstellen von Fotos vor Ort durch die Sicherheitslage am Boden in Konfliktregionen gefährlich sein kann.

Das Monitoring von militärischem Gerät an Ländergrenzen könnte benutzt werden um eine genauere Lageeinschätzung zu generieren, ob ein Konflikt bevorsteht. Es ist auch möglich die Arbeit von menschlichen Analysten unterstützen, da die Modelle Vorschläge für mögliche Detektionen und Klassifizierungen von Objekten liefern.

Maschinelles Lernen kann zur Automatisierung angewendet werden, um größere Datensätze zu verarbeiten. Eine Herausforderung hier besteht im Training der Algorithmen, da die Verfügbarkeit von Trainingsdaten limitiert ist.

Ein Anwendungsszenario wäre die Auswertung des menschlichen Mobilitätsverhaltens im Rahmen des Zählen von Autos auf großen öffentlichen Parkplätzen. Hier kann auch ein Tracking von Verkehrsaufkommen durch die KI landesweit erfolgen, wenn Fahrzeuge auf einem Satellitenbild detektiert werden können. Dieser Ansatz kann genutzt werden um abzuschätzen ob mehr Parkflächen erforderlich sind oder die Anzahl der aktuellen Flächen ausreicht. Weitere Fragestellungen wären, ob es möglich ist fahrende und stehende Autos zu unterscheiden.

Außerdem kann in dieser Arbeit evaluiert werden, ob die Präzision der Objekterkennung und -klassifizierung von Fahrzeugen in elektro-optischen, multi-spektralen Luftaufnahmen durch die Nutzung von mehr Bildkanälen (NIR, IR) verbessert werden kann. Dies kann durch einen Vergleich der Ergebnisse beim Trainieren desselben Deep Learning Modells geschehen, welches jeweils nur

mit 3 und einmal mit mehr Kanälen trainiert wurde. Da Fahrzeuge relativ klein auf hochauflösenden Satellitenbildern dargestellt werden, kann auch evaluiert werden, wie gut sich ein Deep Learning Modell zur Klassifizierung sehr kleiner Objekte eignet, bzw. ob ein weiterer Kanal die Präzision verbessert.

### **1.2. Structure of the thesis**

## 2 | Fundamentals

### 2.1. Computer Vision

Nochmal neu formulieren?

Computer Vision ist ein Teilbereich der künstlichen Intelligenz, der eng mit der Bildverarbeitung und dem maschinellen Lernen verknüpft ist. Dabei wird die Rohdatenerfassung durch Verfahren erweitert, die digitale Bildverarbeitung, Mustererkennung, maschinelles Lernen und Computergrafik kombinieren. Ziel ist es, Maschinen die menschliche Fähigkeit zu verleihen, aus Bildern Informationen zu extrahieren und zu interpretieren [2]. Zur Informationsgewinnung und zur Simulation menschlicher visueller Wahrnehmung werden Algorithmen sowie optische Sensoren eingesetzt [3]. Es lässt sich zwischen Bildbeschaffung und Bildanalyse unterscheiden. Wichtige Komponenten der Bildanalyse sind unter anderem:

- **Bilderzeugung:** Das Speichern eines Objekts als digitales Bild.
- **Bildverarbeitung:** Verbesserung der Bildqualität zur Erhöhung des Informationsgehalts.
- **Bildsegmentierung:** Trennung des Objekts vom Hintergrund.
- **Bildvermessung:** Ermittlung signifikanter Bildpunkte (sogenannter Features).
- **Bildinterpretation:** Ableitung semantischer Informationen aus den Bilddaten [4].

### 2.2. Machine Learning

Nochmal neu formulieren?

Maschinelles Lernen beschreibt einen Ansatz in der Informatik, bei dem Algorithmen auf Basis von vorhandenen Daten selbstständig Muster erkennen, um ein definiertes Ziel zu erreichen – etwa die Klassifikation von Informationen. Im Gegensatz zu traditionellen Algorithmen, die feste, vom Menschen vorgegebene Regeln befolgen, entwickelt ein lernfähiger Algorithmus eigene Verarbeitungsstrategien. Die Entscheidungslogik entsteht dabei nicht durch manuelle Programmierung, sondern durch die Analyse von Daten und das Erkennen wiederkehrender Strukturen [5]. Der Begriff „Lernen“ bezieht sich allgemein auf den Prozess, bei dem Wissen oder Fähigkeiten

durch Erfahrung, Versuch und Irrtum, Anleitung oder Beobachtung erworben werden. Übertragen auf den maschinellen Kontext bedeutet dies, dass Maschinen durch wiederholte Analyse und Rückmeldung in der Lage sind, aus Beispielen zu generalisieren und neue, unbekannte Daten zu verarbeiten. Maschinelles Lernen kann daher als rechnergestützte Nachbildung menschlicher Lernprozesse verstanden werden, bei der Algorithmen in der Lage sind, abstrahierte Regeln aus Rohdaten zu entwickeln und auf neue Situationen anzuwenden [6], [7]. Diese Fähigkeit eröffnet neue Möglichkeiten bei der Lösung komplexer Probleme, die sich mit konventionellen, regelbasierten Methoden nur schwer oder gar nicht bearbeiten lassen [8].

### 2.3. Deep Learning

- Teilgebiet von Maschine Learning
- Funktionsweise inspiriert von menschlichem Gehirn (neuronalen Prozessen) / simuliert den Prozess, der in den zentralen sensorischen Regionen des menschlichen Gehirns abläuft
- Funktioniert ohne strikte von Menschen entworfene Regeln, sondern nutzt große Datenmengen um Muster zu erkennen um die Eingabe auf bestimmte Eigenschaften zu analysieren
- DL nutzt zahlreiche Algorithmentschichten (artificial neural Networks, ANN), von denen jede die zugelieferte Information (einzigartig) interpretiert
- Herkömmliche ML Techniken benutzen: Pre Processing, Feature Extraction, Feature selection, learning and classification
- DL kann das Lernen von Feature Sets für mehrere Aufgaben automatisiert
- DL kann Lernen und Klassifizieren in einem Schritt vornehmen (Single Shot)
- Vorteile Deep Learning
  - Universeller Lernansatz (viele Anwendungsbereiche)
  - Robustheit: keine Präzise entworfenen Merkmale, stattdessen optimierte Merkmale in einem automatisierten Prozess nutzen
  - Verallgemeinerung: Verschiedene Datentypen und Anwendungen können dieselbe DL Technik verwenden, (Transferlernen, TL); nützlicher Ansatz für Probleme bei denen die Daten nicht ausreichend sind
  - Skalierbarkeit: Einfache Erweiterung, wenn mehr Knoten benötigt werden
  -

- DL Techniken haben drei Kategorien: Unsupervised, partially supervised (Semi-supervised), supervised
  - Deep Supervised Learning
    - Der Algorithmus bekommt eine Sammlung von Eingabe- und Ausgabedaten
    - Intelligenter Agent schätzt  $\hat{y}_t = f(x_t)$ , wenn die Eingabe  $x_t$  ist und erhält  $tz(\hat{y}_t, y_t)$  als Verlustwert
    - Netzparameter werden neu berechnet, bis die Schätzung besser zu den bekannten Ausgaben passt
    - Agent erhält die Fähigkeit die richtigen Lösung für die Abfragen aus der Umgebung zu erhalten
    - Mehrere Möglichkeiten zum überwachten Lernen für DL: rekurrent neuronale Netze (RNNs), konvolutionale neuronale Netze (CNNs) und tiefe neuronale Netze (DNNs)
  - Deep Semi supervised learning
    - Lernprozess auf halb beschrifteten Datensätzen
    - Vorteil: Minimierung der benötigten Menge an gelabelten Daten
    - Nachteil: irrelevante Eingangsmerkmale in den Trainingsdaten können zu falschen Entscheidungen führen
    - Bsp: Klassifizieren von Textdokumenten
  - Deep unsupervised Learning
    - Lernprozess ohne gelabelte Daten
    - Agent lernt signifikante Merkmale oder die innere Repräsentation um nicht identifizierte Strukturen oder Beziehungen in den Eingabedaten zu entdecken
    - Hauptnachteil: keine genauen Informationen über die Datensortierung vom Algorithmus lieferbar und rechenintensiv
    - Bsp: Clustering
- Arten von DL Netzwerken: RbNNs, RNNs und CNNs; Konzentration auf CNNs im folgenden Abschnitt da hohe Bedeutung
  - CNN ist bekanntester und am häufigsten Verwendeter Algorithmus bei DL.

- Struktur von CNNs wurde von Neuronen in menschlichen und tierischen Gehirn inspiriert
- [8] hat 3 Vorteile des CNNs identifiziert: gleichwertige Darstellungen, spärliche Interaktion und gemeinsame Nutzung von Parametern
- drei Dimensionen einer Eingabe  $x$  jeder Schicht in einem CNN : Höhe, Breite, Tiefe; wobei Höhe gleich der Breite ist; Tiefe wird auch als die Anzahl der (Bild-) Kanäle bezeichnet
- Mehrere Kernel (Filter), die in jeder Faltungsschicht integriert sind, werden mit  $k$  bezeichnet und haben ebenfalls drei Dimensionen ( $n \times n \times q$ ); Ähnlich zum Inputbild
- es gilt jedoch  $n \ll m \wedge q \leq r$ ; Kernel bilden Grundlage für lokale Verbindungen, die ähnliche Parameter (Bias  $b^k$  und Gewicht  $W^k$ ) zur Erzeugung von  $k$  Feature Maps  $h^k$  mit einer jeweiligen Größe von  $(m - n - 1)$ . Diese werden im Input Layer verwendet.
- Faltungsschicht berechnet Punktprodukt zwischen Eingabe und den Gewichten wie in 2.1

$$h^k = f(W^k * x + b^k) \quad (2.1)$$

- Nächster Schritt ist Down Sampling jeder Feature Map in den Sub Sampling Schichten, um die Netzparameter zu verringern, was den Trainingsprozess beschleunigt. Außerdem wird das Lösen Problem der Überanpassung dadurch ermöglicht.
- Alle Feature Maps werden durch eine Pooling Funktion (max oder average) auf einen angrenzenden Bereich der Größe  $p \times p$  angewendet, wobei  $p$  die Kernelgröße ist. Letztendlich erhalten die FC Schichten die Merkmale der mittleren und unteren Ebene und erstellen die Abstraktion der oberen Ebene, welches die letzte Schicht darstellt
- Die Klassifizierung wird mit der abschließenden Schicht, wie einer Support Vector Machine (SVM) erzeugt, um für die gegebene Instanz die Wahrscheinlichkeit einer bestimmten Klasse in einem Score darzustellen
- Vorteile CNN Einsatz:
  - \* Bessere Generalisierung des Netzwerkes und Verhinderung der Überanpassung durch Weight Sharing Feature
  - \* Ausgabe des Modells ist flexibel organisiert als auch flexibel von den extrahierten Merkmalen abhängig durch das Lernen von zwei Merkmalsextraktionsschichten und einer Klassifizierungsschicht



- \* CNNs in großem Maßstab können sehr einfach entwickelt werden
- \* CNN Architecture
- \* **Convolutional Layer:** Kollektion von Convolutional Filtern (sog. Kernels); Eingabebild wird durch diesen Filter zur Output Feature Map Generierung genutzt
- \* **Kernel definition:** Grid of discrete numbers, which each value is called kernel weight. Werte werden zufällig initialisiert zum Beginn des CNN Trainingsprozesses; Gewichte werden in jeder Trainingsepoche optimiert, sodass der Kernel lernt relevante Features zu extrahieren
- \* Convolutional Operation: Übersprungen
- \* **Pooling Layer:** Hauptaufgabe: Sub Sampling der Feature Maps, oder verkleinern der großen Feature Maps um kleiner Feature Maps zu erzeugen; meistgenutzt max, min und GAP Pooling als Methode dafür
- \* **Activation Function (non linearity):** Mapping the input to the output is core; input Value is the weight summation of the neuron input along with its bias (if present); Activation Function entscheidet ob eine Neuron aktiviert wird (feuert) oder nicht, mit Referenz zu einem Teil des Inputs durch das Erstellen des dazugehörigen Outputs (d. Klasse?); (nicht lineare) Aktivierungsfunktion muss differenzierbar sein, um Backpropagation zu ermöglichen
- \* ReLu: Meist eingesetzt Aktivierungsfunktion; konvertiert alle Ergebnisse in Positive Zahlen; (NACHTEILE NICHT ERWÄHNEN? DYING RELU?)
- \* **Fully Connected Layer:** Am Ende jeder CNN Architecture; jedes Neuron ist mit allen Neuronen des letzten Layers verbunden, deshalb fully connected (FC) Layer
- \* **Loss Function:** loss funktion berechnet den predicted error über die Training samples; error ist die difference zwischen dem Aktuellen Output und dem vorhergesagten; dieser wird optimiert -> first parameter: estimated Output (Prediction); second parameter: actual output (label); Euclidean Loss Function bei Regressionssproblemen (meist auch mean square error genannt)

### 2.3.1. Backpropagation

- [8]

- Feedforward neural Network um Input  $x$  und Output  $y$  zu produzieren.  $x$  enthält die initiale Information für die Hidden Units in jedem Layer, sodass Final  $y$  rauskommen kann (forward propagation)
- back propagation ist ein einfaches Verfahren um (Kosten (Berechnungszeit/Methode?)-) zurück durch das Netzwerk fließen zu lassen um einen Gradienten zu berechnen

### 2.3.2. Vanishing

### 2.3.3. Exploding Gradients

### 2.3.4. Hyperparameter

### 2.3.5. Overfitting

## 2.4. YOLOv9

## 2.5. Evaluation Metrics

### 2.5.1. N Fold Cross Validation

- Cross validation to ensure robustness and better comparison of different models
- create own folds, as the ones provided by the paper had the same images in training and validation data
- 6 Folds
- 5 for Training and Validation, 1 for Test (Fold 5)
- Good object distribution between the folds
- 207-221 Images per Fold, where 3 or 7 Images are only background
- Sorting method similar to bucketsort
- Each fold gets an image at the beginning and the class that is least common in each fold is identified
- then the next image is taken and the objects of the classes are counted (how often each class appears in the image). The image is placed in the fold with the fewest objects of that class

- If there are the same number, priority is given to rarity
- \_\_\_\_\_

Tabelle mit  
Verteilung  
einfügen

### 2.5.2. Mean Average Precision

**mAP@50**

**mAP@90-95**



### 3 | State of research

Mit Luftbildern oder Satellitenbildern können bereits Massengräber detektiert werden [9]. Dieser Ansatz kann in nahezu Echtzeit oder in der Vergangenheit angewendet werden [9]. Er basiert auf Satellitenbildern, die aus verschiedenen Bändern bestehen. Man kann ihn auch zur Detektion von geheimen Massengräbern nutzen [9]. Möglicherweise ist ein Umbau zur Detektierung von Massengräbern in der Ukraine möglich [9].

Weitere Ansätze umfassen die Erkennung von Einzelgräbern mithilfe von Air Borne Imagery [10].

Außerdem wurden Satellitenbilder von chinesischen Krematorien genutzt, um die offiziellen Daten zur Covid Mortalität der Regierung zu überprüfen. Dies konnte in Kombination mit Augenzeugenbefragungen dafür genutzt werden, die offiziellen Zahlen zur Sterblichkeit anzuzweifeln [11].

Mit Satellitenbildern konnten bereits Zerstörungen von Hüttenstrukturen in Darfur detektiert werden [1].

RGB Bilder von Drohnen können in den meisten Szenarien gut für Objekterkennung eingesetzt werden. Wärme (IR) kann die Möglichkeiten zur Objekterkennung in der Nacht oder bei verdeckten Objekten erweitern. Ein Problem ist der Mangel an verfügbaren Trainingsdaten für IR Bilder von Drohnen.



## 4 | Methodology

Ablaufdiagramm  
einfügen

Um Objekte auf hochauflösenden Satellitenbildern zu detektieren, bietet sich der 'You Only Look Once' (YOLO) Algorithmus an. Dieser Algorithmus, der in der Version 9 im Jahr 2024 veröffentlicht wurde, ermöglicht eine schnelle und präzise Objekterkennung. YOLO basiert auf einer einzigen neuronalen Netzwerkarchitektur, die das Bild in Raster unterteilt und für jedes Rasterfeld Vorhersagen über die Position und Klasse von Objekten trifft.

Die Hauptvorteile von YOLO sind seine Geschwindigkeit und Genauigkeit, die es ermöglichen, große Datensätze effizient zu analysieren. Für diese Arbeit wird YOLO verwendet, um Fahrzeuge auf Luftbildern zu detektieren und zu klassifizieren. Dabei könnten zusätzliche multispektrale Kanäle wie NIR oder IR integriert werden, um einen Vergleich zu dem reinen 3 Kanal RGB Training zu ermöglichen. Die Vergleichbarkeit kann gewährleistet werden, wenn die gleichen Trainings- und Testdaten verwendet werden, bei denen der einzige Unterschied das weitere Band ist.

### 4.1. Preprocessing of input data

### 4.2. YOLOv9

#### 4.2.1. YOLOv9u

#### 4.2.2. YOLOv9e

#### 4.2.3. Hyperparameter

### 4.3. Database

#### 4.3.1. Vehicle Detection on Aerial Images (VEDAI) Dataset

Das 'Vehicle Detection in Aerial Images' (VEDAI) Dataset [12] aus dem Jahr 2015 bietet sich als Datengrundlage an, da es hochauflösende Luftbilder enthält, die speziell für die Fahrzeugerkennung

geeignet sind [13]. Es umfasst annotierte Daten, die Fahrzeuge in unterschiedlichen Szenarien, Größen und Orientierungen zeigen. Außerdem ist es ein Benchmark für die Detektion von sehr kleinen Objekten. Das Dataset enthält sowohl RGB-Bilder als auch multispektrale Daten, was es ideal für die Untersuchung der Auswirkungen zusätzlicher Kanäle wie NIR oder IR auf die Objekterkennung macht. Es sind mehr als 3700 Objekte in ungefähr 1200 Bildern annotiert. Diese Objekte sind in 9 Klassen (Boat, Camping Car, Car, Pickup, Plane, Tractor, Truck, Van und Others) unterteilt und der Hintergrund der Objekte ist abwechslungsreich, was die Robustheit des trainierten Modelles erhöht.

Die Auflösung der Bilder liegt bei 12.5 cm  $\times$  12.5 cm pro Pixel, was ausreichend ist um einzelne Fahrzeuge zu erkennen. Die Bilder mit einer Größe von 1024  $\times$  1024 Pixeln sind bei einer Befliegung im Jahr 2012 in US Amerikanischen Bundesstaat Utah aufgenommen worden.

Bsp. Bild für jede Klasse zeigen (aus Paper nehmen!; Quellenverlinkung für text und bild nicht vergessen)

Die Datenaufbereitung umfasst Schritte wie das Unterteilen der Bilder in Trainings- und Testdaten, sowie die Aufbereitung der Annotationen für den YOLO Algorithmus. Nach diesen Schritten kann das Training auf dem High-Performance-Cluster (HPC) PALMA der Uni Münster erfolgen. Anhand der Ergebnismatrix wird dann ein Vergleich der beiden Modelle erfolgen, da der einzige Unterschied das weitere Band ist.

### 4.3.2. Dataset for Object deTection in Aerial Images (DOTA) 1.5

- DOTA bietet sich als pretrained Model Datengrundlage für die Permutationsexperimente an, da es viele verschiedenen Klassen auf Satellitenbildern enthält
- Bildgröße von 800  $\times$  800 bis 20.000  $\times$  20.000 Pixel
- 3 Channel (Red, Green, Blue) und Grayscale Images (Panchromatic Band von GF2 und JL1 Satelliten)
- Satellite Images von Google Earth und anderne Quellen

Bsp. Bild für jede Klasse zeigen (aus Paper nehmen!; Quellenverlinkung für text und bild nicht vergessen); habe alle 16 Klassen für das Training genutzt, wichtig sind trotzdem nur Large Vehicle, Plane, Ship, small vehicle

## 4.4. Umsetzung

### 4.4.1. PALMA

- UV als Python Umgebung auf Palma genutzt
- Bash Scripte geschrieben um die Modelle auf diversen GPUS (HGX, etc) zu trainieren



- Hersteller: MEGWARE[14]
- 16.272 Cores
- 77.568 GB Memory
- 444 Nodes
- Processor: Intel Xeon Gold 6140 18C @ 2.30GHz (Skylake)
- Interconnect 100Gbit/s Intel Omni-Path
- GPFS Storage: 2,4 PB
- Linpack Performance: Rmax: 800 TFlop/sRpeak: 1,277 TFlop/s
- OS: Rocky Linux 9

#### 4.4.2. Challenges Preprocessing

##### VEDAI Dataset Challenges

- Label müssen in yolov9 obb format konvertiert werden
- kleine Anzahl (7/3757) war kleiner als 0 oder größer als 1
- Lösung mit Exception Handling und Runden der Werte
- 

Beispielbilder  
einfügen

## 4.5. Workflow

### 4.5.1. Axis Aligned vs. Oriented Bounding Boxes

- Vergleich zwischen Axis Aligned und Oriented Bounding Boxes
- YOLOv9 arbeitet ursprünglich nur mit aab Boxen
- YOLOv9u kann mit obb arbeiten, da Codebasis von YOLOv8 von Ultralytics, was obb unterstützt
- 
- 

Vergleichsbilder  
(Schiff und  
Auto Vergleich)  
einfügen

- mehr Blankspace bei axis aligned Bbs
- Concentration of the box on the actual object, significantly less surrounding area outlined.  
No overlap between bounding boxes (Bei Ship bb)
- 

### 4.5.2. Channel Permutation

- Folgende Permutation werden im Rahmen der Arbeit evaluiert: RGBIR, IRGB, RIRB; RGB, RGIR, RGBNDVI, GBNDBVI

### 4.5.3. DOTA Training

- DOTA als Pretrained Modell für Channel Permutation, map50-95 around 0.4 and training over around 800 epochs

Result.png ein-  
fügen? oder  
doch sein  
lassen?

•

### 4.5.4. Ablation Studie

- Ablation Study für R, G, B, IR, NDVI durchgeführt

## 5 | Implementation

Die genutzte Programmiersprache ist Python. Die Scripte für PALMA wurden in Bash geschrieben.

### 5.1. Preprocessing

#### 5.1.1. DOTA Dataset

Im Rahmen der Datenvorverarbeitung wurden ausschließlich die originalen Rohbilder des DOTA-Datensatzes verwendet, ohne dass ein Zuschchnitt oder eine anderweitige Modifikation der Bildinhalte vorgenommen wurde. Der Fokus der Verarbeitung lag auf der Umwandlung der vorhandenen Labeldaten in das YOLOv9-0BB-Format, welches polygonale Objektrumrandungen in Form einer Punktsequenz  $(x_1, y_1, x_2, y_2, \dots)$  beschreibt (vgl. ).

Für jedes Bild stellt DOTA eine separate Textdatei bereit, die alle zugehörigen Annotationen enthält. Zu Beginn wurde eine standardisierte Ordnerstruktur nach dem YOLO-Schema erstellt, bestehend aus den Unterverzeichnissen `train/images`, `train/labels`, `val/images` und `val/labels`. Die Annotationsdateien wurden daraufhin zeilenweise eingelesen und verarbeitet.

Jede Zeile, die ein einzelnes Objekt beschreibt, wurde zunächst an die jeweilige Bildgröße angepasst, wobei die absoluten Koordinaten der Objektrahmen in relative Werte normalisiert wurden. Da die Auflösung der Bilder im DOTA-Datensatz stark variiert (zwischen  $800 \times 800$  und  $20,000 \times 20,000$  Pixeln), war eine dynamische Skalierung erforderlich. Die konvertierten Informationen wurden anschließend in das YOLOv9-Format überführt, wobei die Klassen-ID jeweils am Anfang der Zeile stand. Die so erzeugten Labeldateien wurden unter Beibehaltung des ursprünglichen Dateinamens gespeichert.

Insgesamt wurden sämtliche 16 im DOTA-Datensatz enthaltenen Objektklassen berücksichtigt und entsprechend konvertiert. Die Bilddaten selbst blieben unangetastet und wurden in ihrer Originalform auf den HPC Cluster PALMA übertragen. Aufgrund der hohen Anzahl an verfügbaren Annotationen gestaltete sich der Konvertierungsprozess zeitintensiv.

text\todo{Ref  
zur  
Formatbeschreibung  
einfügen}

### 5.1.2. VEDAI Dataset

Die Verarbeitung des VEDAI-Datensatzes erfolgt über eine zentrale Hauptmethode, in der mehrere konfigurierbare Parameter in Form von booleschen Variablen bereitgestellt werden. Besonders wichtig ist hierbei der Parameter `oriented`, welcher angibt, ob die Bounding Boxes (BBs) als achsenparallele Rechtecke oder als orientierte Rechtecke im OBB-Format generiert werden. Zusätzlich kann über `bool_create_yaml` gesteuert werden, ob für jedes erzeugte Trainingsset eine entsprechende YAML-Datei automatisch mitgeneriert werden soll.

Ein weiterer Parameter, `merge_ir_bool`, ermöglicht es, das Infrarotbild (IR) in die erzeugten Bilddaten zu integrieren. Mithilfe des Parameters `namestring` lässt sich jedem erzeugten Trainingsset ein individueller Name zuweisen. Entscheidend für die Zusammensetzung der Bildkanäle ist das Dictionary `perm_object`, in dem über boolesche Werte festgelegt wird, welche Kanäle (z. B. R, G, B, IR, NDVI) in das finale Bild aufgenommen werden sollen.

In der zentralen Hauptfunktion wird anschließend bestimmt, welche Teilmengen des Datensatzes erzeugt werden sollen. Dies betrifft u. a. ablation-Sets, perm-Datasets oder aab\_vs\_obb-Konfigurationen. Für beide erstgenannten Varianten folgt der Ablauf einem konsistenten Schema: Die Parameter `oriented` und `merge_ir_bool` werden aktiviert, ein Speicherpfad für das erzeugte Dataset definiert, und das `perm_object` wird entsprechend der gewünschten Kanalzusammensetzung konfiguriert. Beispielsweise wird für die Kombination `rgir` lediglich R, G und IR im `perm_object` auf `true` gesetzt. Im Anschluss wird mittels einer `for`-Schleife über die Folds 0 bis 5 iteriert und für jeden Fold mit der Methode `create_fold_cross_validation` das entsprechende Dataset aus Trainings-, Validierungs- und Testdaten generiert (vgl. ).

Verweis auf  
entsprechende  
Funktion

**Erzeugung von Folds via `create_fold_cross_validation`** Diese Methode übernimmt die vollständige Erstellung der Ordnerstruktur im für YOLO erforderlichen Format (`train/images`, `train/labels`, `val/images`, `val/labels`, `test/images`, `test/labels`). Zudem wird eine YAML-Datei erzeugt, die sowohl die Pfade zu den Bilddateien als auch die Anzahl der enthaltenen Kanäle enthält. Ein Trainingsdatensatz setzt sich stets aus drei Komponenten zusammen:

1. **Trainingsdaten:** Bestehen aus den Bildern der übrigen Folds (also alle außer dem aktuellen Fold und Fold 6).
2. **Validierungsdaten:** Entsprechen den Bildern des jeweils aktuellen Folds.
3. **Testdaten:** Bestehen stets aus den Bildern von Fold 6.

Der aktuell ausgewählte Fold wird aus der Liste mit allen Folds entfernt, um Überschneidungen mit Trainings-, Validierungs- und Testdaten auszuschließen. Anschließend wird die zentrale Datei

mit allen Labelinformationen des Datensatzes eingelesen. Über eine for-Schleife werden alle Folds durchlaufen, die nicht dem aktuellen Fold entsprechen. Dabei wird jeweils die Datei mit den Bildnamen eingelesen und für jedes dieser Bilder die Funktion `create_image_and_label` aufgerufen.

Diese Funktion liest für jede Zeile das zugehörige RGB- und IR-Bild ein und filtert die Labels so, dass nur die zum aktuellen Bild gehörenden Annotationen weiterverarbeitet werden. Das neue Bild wird anschließend in den jeweiligen Zielpfad kopiert, wobei intern die Funktion `merge_RGB_IR_image` zur Anwendung kommt. Diese Funktion stellt sicher, dass nur die im `perm_object` spezifizierten Kanäle in die Zieldatei übernommen werden (vgl. [Referenz zur Funktion](#)). Im Anschluss wird die Funktion `create_label_file` aufgerufen, um das zugehörige Label-File im gewünschten Format zu erzeugen (vgl. [Referenz zur Funktion](#)).

ref einfügen

ref einfügen

Nach dem Abschluss der Verarbeitung aller Trainingsfolds wird abschließend sowohl der Validierungsfold (aktueller Fold) als auch der Testfold (immer Fold 6) mithilfe der Funktion `create_image_and_label` separat verarbeitet. Nach deren erfolgreichem Durchlauf gilt das Datenset für die jeweilige Kanal-Kombination als vollständig erstellt.

**Fusion von RGB- und IR-Bildern (`merge_RGB_IR_image`)** Diese Methode lädt die entsprechenden RGB- und IR-Bilder und erzeugt basierend auf der Vorgabe im `perm_object` eine kombinierte Bilddarstellung. Die Verschmelzung erfolgt mit Hilfe von `cv2.merge`. Falls gefordert, wird innerhalb dieser Funktion zusätzlich ein NDVI-Bild erzeugt. Zur numerischen Stabilisierung bei der NDVI-Berechnung wird der Nenner ( $IR + R$ ) auf mindestens 0,01 gesetzt, um Division durch Null zu vermeiden. Die NDVI-Werte werden anschließend skaliert und als 8-Bit-Bild zurückgegeben. Vor dem Zusammenführen werden die RGB-Kanäle immer einzeln extrahiert. Die Funktion `copy_image` übernimmt dann das Schreiben der erzeugten Bilder an den vorgesehenen Zielpfad.

**Erzeugung von Label-Dateien (`create_label_file`)** Zur Erstellung der Label-Datei wird zunächst der Pfad zur Ausgabedatei bestimmt und das zugehörige Bild eingelesen, um die Bilddimensionen für die spätere Normalisierung zu ermitteln. Anschließend werden alle übergebenen Label-Informationen durchlaufen. Für jedes Label wird mithilfe der Funktion `get_bounding_box_in_px` die Bounding Box in Pixelkoordinaten berechnet.

Im nächsten Schritt wird die Zielzeile im YOLO-Format erstellt: Die Klasse wird dabei in eine numerische ID zwischen 0 und 8 überführt (mittels einer internen Mapping-Funktion), gefolgt von den acht Koordinatenpunkten der Bounding Box, jeweils durch Leerzeichen getrennt.

Im Fall von `oriented = true` werden die acht Punkte einer rotierbaren Box gespeichert. Ist `oriented = false`, können die Koordinaten entweder in klassischer YOLO-Darstellung (zentrumsbasiert mit Höhe und Breite) oder als achsenparalleles Rechteck im OBB-Format gespeichert werden. Die Umwandlung erfolgt hier durch unterschiedliche Konvertierungsfunktionen. Abschließend wird die Zeile in die Datei geschrieben und der Prozess für das nächste Label wiederholt.

**Berechnung der Bounding Box in Pixeln (`get_bounding_box_in_px`)** Diese Funktion dient der Extraktion und Transformation der Labelkoordinaten aus dem ursprünglichen DOTA-Format. Zunächst wird der Label-String anhand von Leerzeichen in seine Bestandteile zerlegt. Die Position (x, y) sowie die Orientierung befinden sich an den Stellen 1 bis 3 des Arrays, während sich die Klassen-ID an Position 12 befindet. Die Eckpunkte der Bounding Boxen (x-Werte: Positionen 4–8, y-Werte: 8–12) werden extrahiert und daraus die Seitenlängen berechnet.

Die längsten Seiten des Objekts werden identifiziert und als Fahrzeuglänge, die kürzeren als Fahrzeugbreite angenommen. Aus den längsten Seiten wird ein Richtungsvektor berechnet, und der Winkel der Fahrzeugausrichtung über den Arctan bestimmt. Die vier Eckpunkte der orientierten Box werden schließlich unter Berücksichtigung von Fahrzeugmitte, Länge, Breite und Rotationswinkel berechnet.

Ist `oriented = true`, so werden die vier Eckpunkte als ganzzahlige Koordinaten-Tupel zurückgegeben. Ist `oriented = false`, wird aus den minimalen und maximalen x- und y-Werten der rotierten Box eine achsenparallele Bounding Box erzeugt und zurückgegeben.

### 5.1.3. Fold Creation

#### 5.1.4. Fold-Erstellung über `create_own_folds`

Die Erstellung der Folds erfolgt über die zentrale Hauptfunktion `main`, welche auf einer Vorverarbeitungsfunktion (`preproc`) basiert. Diese Funktion liest Textdateien ein, in denen jeder Bildname zeilenweise enthalten ist. Ziel des gesamten Skripts ist es, solche Textdateien zu generieren, welche später zur Trainings-, Validierungs- und Testdatenerstellung genutzt werden. Zunächst wird eine Liste aller im Quellverzeichnis vorhandenen Bilder erstellt. Für jedes Bild wird dabei zusätzlich die Anzahl der Objekte pro Klasse gespeichert. Anschließend wird die Methode `create_own_folds` aufgerufen, wobei die Anzahl der Folds flexibel gewählt werden kann – standardmäßig werden fünf Folds genutzt.

Die Methode `create_own_folds` erhöht die gewünschte Fold-Anzahl um eins, um einen separaten Testfold zu berücksichtigen. Danach wird ein leeres mehrdimensionales Array erzeugt, das der

neuen Anzahl an Folds entspricht. Die übergebene Bildliste wird einmal zufällig durchmischt, bevor den einzelnen Folds initial je ein Bild zugewiesen wird. Im Anschluss wird über eine separate Funktion () ermittelt, wie viele Objekte jeder Klasse in jedem Fold enthalten sind, um eine gleichmäßige Verteilung sicherzustellen.

ref einfügen

Die eigentliche Verteilung erfolgt iterativ in einer `while`-Schleife, die so lange läuft, bis alle Bilder verteilt sind. Innerhalb dieser Schleife wird für jedes Bild analysiert, welche Klasse im Bild am seltensten vorhanden ist(). Wird genau eine seltene Klasse identifiziert, so wird das Bild jenem Fold zugewiesen, der aktuell die geringste Anzahl von Objekten dieser Klasse enthält. Falls mehrere seltene Klassen vorhanden sind, wird anhand einer vordefinierten Frequenzbewertung entschieden. Jede Klasse wird dabei mit einem Gewicht von 0 (häufig) bis 8 (selten) bewertet. Es wird die Klasse mit dem höchsten Frequenzwert ausgewählt, und entsprechend der geringsten Objektanzahl dieser Klasse wird das Bild einem Fold zugewiesen. Bilder ohne Objekte werden separat in einem Array für leere Bilder gespeichert.

ref funktion dafür

Nachdem alle Bilder mit Objekten verteilt wurden, erfolgt die Zuweisung der leeren Bilder. Dazu wird zunächst eine ganzzahlige Division der Anzahl leerer Bilder durch die Anzahl der Folds durchgeführt, um eine möglichst gleichmäßige Grundverteilung zu ermöglichen. Der verbleibende Rest wird anschließend über eine zusätzliche Schleife auf jene Folds verteilt, die aktuell die geringste Bildanzahl enthalten.

Am Ende der Verteilung erfolgt eine erneute Zählung der Objekte in jedem Fold. Die finalen Fold-Zuweisungen werden anschließend als Textdateien gespeichert – jeweils eine Datei pro Fold (insgesamt sechs Dateien, Fold 0 bis Fold 5), die ausschließlich die entsprechenden Bildnamen enthalten.

**Zählung der Objekte in den Folds (`count_objects_in_fold_arr`)** Zur Auswertung der Objektverteilung innerhalb der Folds wird die Methode `count_objects_in_fold_arr` verwendet. Diese Funktion erzeugt eine Datenstruktur, die für jeden Fold die Anzahl der enthaltenen Objekte pro Klasse speichert. Die Zählung erfolgt durch eine doppelt geschachtelte Schleife, in der alle Bilder jedes Folds durchlaufen und die Objekte gezählt werden. Am Ende liefert die Funktion eine vollständige Übersicht der Objektverteilung als Grundlage für weitere Entscheidungen.

#### **Identifikation der kleinsten Klassenverteilung (`get_indices_of_folds_with_smallest_object_count`)**

Die Methode `get_indices_of_folds_with_smallest_object_count` dient dazu, für eine gegebene Objektklasse den oder die Folds mit der geringsten Anzahl dieser Klasse zu ermitteln. Basierend auf den Zähldaten aus `count_objects_in_fold_arr` wird über alle Folds iteriert und geprüft, welcher Fold die minimalen Objektzahlen für die entsprechende Klasse enthält. Dabei wird eine

ref funktion einfügen

Liste mit allen zutreffenden Indizes zurückgegeben, sofern mehrere Folds denselben Minimalwert aufweisen.

**Bestimmung der seltensten Klasse im Bild (`get_smallest_class_in_image`)** Die Methode `get_smallest_class_in_image` analysiert ein gegebenes Bild und gibt jene Objektklasse zurück, die in diesem Bild am wenigsten vertreten ist. Diese Information dient als Ausgangspunkt für die optimale Zuweisung des Bildes zu einem Fold und trägt zur Balance der Objektverteilung bei.

## 5.2. Python Pakete

### 5.2.1. Computer Vision 2

### 5.2.2. Numpy

### 5.2.3. Scipy (nur Convexhull)

### 5.2.4. os

ab jetzt nur  
für analyse

### 5.2.5. Matplotlib

### 5.2.6. Seaborn

### 5.2.7. pandas




## 6 | Results

- Konsistenz aller Modelle über die 5 Folds der Kreuzvalidierung ist gut, was eine robuste Trainierbarkeit bedeutet

### 6.1. Oriented Bounding Boxes and Axis Aligned Bounding Boxes

- Training Results by the mAP50-95 of all 5 Folds for every model
- Left Side (Blue): YOLOv9 (without obb) and with axis aligned Bounding Boxes
- Middle (Orange): YOLOv9u with oriented bounding boxes
- Uses Ultralytics as Backend for obb support
- Right (Green): YOLOv9u with axis aligned Bounding Boxes in the oriented Bounding Box format
- Result -> aab performs better

-  This is probably due to the fact that small changes in the orientation of the bounding boxes lead to a greater deviation of the mean average precision
- Oriented Bounding boxes are smaller than the axis aligned bounding boxes -> higher inaccuracy of mAP50-95 (see next slide)
- Red box at the right picture has a small deviation from the correct (blue) label

Vergleichsbild  
einfügen oder  
verweis


-  Oriented Bounding Boxes has an area from around 600 to 900 pixels
- Axis aligned Bounding boxes from around 750 to 1300 px
- -> higher inaccuracy of mAP50-95

Bild für Ver-  
gleich der BB  
Area einfügen  
(zw. obb, aab  
aab old)

- For the channel permutation I used the obb model

### 6.2. Comparison at Mean Average Precision 50-95

Bild einfü-  
gen best val  
dataset on val  
data

- 
- See the performance of the best validation model on the validation data (per fold and model)
- RGBIR has the best performance followed by irgb
- 3. place are rgb and rirb
- Rgir and both ndvi models has the poorest map values

Bild einfü-  
gen; best val  
dataset on test  
data

- 
- Best Validation Models on the test data (Fold 5)
- Red Dots are the best validation model at the validation data on the test data
- (means the highest whisker from the previous slide)
- Difference between the quantils, rgir performs at best
- Then the rirb model and rgb model

Tabelle einfü-  
gen

- 
- Difference Between best mAP Fold at the validation data
- From 0 to 4
- Smaller difference at the mAP Performance at the test data
- Fold 2 and 3
- Now looking at the confusion and difference matrices

## 6.3. Comparison with Confusion Matrices

beide confu-  
sionsmatricen  
einfügen

- 
- Normalisierte Konfusions Matrix mit 9 Klassen und der Einteilung Background (Hintergrund)
- Right: Confusion Matric of RGBIR Modell on Fold 2:
- Good Performance at Cars, Trucks, Ships, Tractors, Camping Cars, Pick Up and planes
- Many confusion at vehicle with background
- Nearly the same for RGIR at Fold 3
- Both modells detected nearly all Planes
- Difference Matrics at next slide

- 
- At all Difference Matrics is red for the better performance of the RGBIR Model and blue for the other modell
- RGBIR detected Trucks, Ships and Tractor more accurate than the RGIR Model
- RGBIR seems more robust for common ground vehicles
- Both modells detect all planes (in short every modell detect all planes)
- Both modells misclassifier background as any class (RGBIR is better at vehicle and Camping Car)

Differenzmatrix  
RGBIR vs  
RGIR einfügen

- 
- RGBIR is better for detecting Ships (12%) and Vehicles (15%), slightliy degrades performance of car detection
- Vehicle is much better than in RGB Model (15%)
- IR input helps for common vehicle recognition (vehicle includes Excavators, construction equipment, etc.)
- RGB is better in identifying ship and vehicle as background

Differenzmatrix  
RGBIR vs RGB  
einfügen

- 
- RGBIR recognize Ships 21 % better than the irgb model

Differenzmatrix  
RGBIR vs  
IRGB einfügen

- Particularity stronger for camping car and van classification (6%)
- IRGB has more background-ship and background-vehicle confusion

Differenzmatrix  
RGBIR vs  
RIRB einfügen

- 
- RGBIR detects Ships and vans better than RIRB Model
- RIRB misclassified van and ship as background more often than RGBIR

Differenzmatrix  
RGBIR vs  
GBNDBVI und  
RGBNDVI  
einfügen

- 
- RGBIR have a better performance in Ship, Tractor and Vehicle
- Both ndvi models show a tendency for confusing background with other classes

### 6.4. Ablation Studies

## 7 | Discussion



## 8 | Conclusion and Outlook

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

This is the second paragraph. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

And after the second paragraph follows the third paragraph. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

After this fourth paragraph, we start a new paragraph sequence. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

**Definition 8.0.1** ( $\sigma$ algebra)

Let  $X$  be some set. A subset  $\Sigma \in \mathcal{P}(X)$  is called a  $\sigma$ -algebra if it satisfies the following three properties:

1.  $X \in \Sigma$
2.  $A \in \Sigma \implies X \setminus A \in \Sigma$
3.  $A_1, A_2, \dots \in \Sigma \implies A_1 \cup A_2 \cup \dots \in \Sigma$

For a full list of environments see `preamble.tex`



## A | Appendix

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.



## List of Figures



**List of Tables**



## Bibliography

- [1] C. Knoth and E. Pebesma, "Detecting dwelling destruction in darfur through object-based change analysis of very high-resolution imagery," *International Journal of Remote Sensing*, vol. 38, pp. 273–295, 1 Jan. 2017, ISSN: 13665901. DOI: 10.1080/01431161.2016.1266105. [Online]. Available: <https://www.tandfonline.com/doi/abs/10.1080/01431161.2016.1266105> (cit. on pp. 1, 11).
- [2] V. Wiley and T. Lucas, "Computer vision and image processing: A paper review," *International Journal of Artificial Intelligence Research*, vol. 2, p. 22, 1 Jun. 2018. DOI: 10.29099/ijair.v2i1.42 (cit. on p. 3).
- [3] S. Matiacevich, D. C. Cofré, P. Silva, J. Enrione, and F. Osorio, "Quality parameters of six cultivars of blueberry using computer vision," *International Journal of Food Science*, vol. 2013, 2013, ISSN: 23145765. DOI: 10.1155/2013/419535 (cit. on p. 3).
- [4] D. Mery, F. Pedreschi, and A. Soto, "Automated design of a computer vision system for visual food quality evaluation," *Food and Bioprocess Technology*, vol. 6, pp. 2093–2108, 8 Aug. 2013, ISSN: 19355130. DOI: 10.1007/s11947-012-0934-2 (cit. on p. 3).
- [5] S. H. Shetty, S. Shetty, C. Singh, and A. Rao, "Supervised machine learning: Algorithms and applications," *Fundamentals and Methods of Machine and Deep Learning: Algorithms, Tools, and Applications*, pp. 1–16, Jan. 2022. DOI: 10.1002/9781119821908.CH1 (cit. on p. 3).
- [6] P. Fischer, "Algorithmisches lernen," 1999. DOI: 10.1007/978-3-663-11956-2. [Online]. Available: <http://link.springer.com/10.1007/978-3-663-11956-2> (cit. on p. 4).
- [7] U. Braga-Neto, "Fundamentals of pattern recognition and machine learning," *Fundamentals of Pattern Recognition and Machine Learning*, pp. 1–357, Jan. 2020. DOI: 10.1007/978-3-030-27656-0/COVER (cit. on p. 4).
- [8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, <http://www.deeplearningbook.org> (cit. on pp. 4, 6, 7).
- [9] M. Kalacska and L. Bell, "Remote sensing as a tool for the detection of clandestine mass graves," *Canadian Society of Forensic Science Journal*, vol. 39, pp. 1–13, 1 Jan. 2006, ISSN: 0008-5030. DOI: 10.1080/00085030.2006.10757132 (cit. on p. 11).

- [10] G. Leblanc, M. Kalacska, and R. Soffer, "Detection of single graves by airborne hyperspectral imaging," *Forensic Science International*, vol. 245, pp. 17–23, Dec. 2014, ISSN: 0379-0738. DOI: 10.1016/J.FORSCIINT.2014.08.020 (cit. on p. 11).
- [11] *Corona in china: Satellitenbilder von krematorien deuten auf deutlich mehr tote hin - der spiegel*, 2023. [Online]. Available: <https://www.spiegel.de/ausland/corona--satellitenbilder--chinesischer--krematorien-deuten-auf--deutlich--mehr--tote-hin--a%5C--c4305852%5C--d092%5C--4e54%5C--a210%5C--c88b820d564c> (cit. on p. 11).
- [12] *Vehicle detection in aerial imagery (vedai) : A benchmark*. [Online]. Available: <https://downloads.greyc.fr/vedai/> (cit. on p. 13).
- [13] S. Razakarivony, F. Jurie, S. Razakarivony, and F. Jurie, "Vehicle detection in aerial imagery : A small target detection benchmark," *Journal of Visual Communication and Image Representation*, 2015. [Online]. Available: <https://hal.science/hal-01122605v2> (cit. on p. 14).
- [14] Center for Information Technology, University of Münster, *Hpc documentation palma ii hardware*, Accessed: 2025-07-30. [Online]. Available: <https://palma.uni-muenster.de/documentation/> (cit. on p. 15).



# Declaration of Academic Integrity

I hereby confirm that this thesis, entitled

*Deep Learning for Small Vehicle Detection and Classification on High-Resolution Multispectral Remote Sensing Imagery*

is solely my own work and that I have used no sources or aids other than the ones stated. All passages in my thesis 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. I am aware that plagiarism is considered an act of deception which can result in sanction in accordance with the examination regulations.

---

Timo Lietmeyer, Münster, August 1, 2025

I consent to having my thesis cross-checked with other texts to identify possible similarities and to having it stored in a database for this purpose.

I confirm that I have not submitted the following thesis in part or whole as an examination paper before.

---

Timo Lietmeyer, Münster, August 1, 2025