INSTITUT FÜR INFORMATIK
Fachbereich 12 – Informatik und Mathematik

Forschungsprojekt

Einfluss von Bildannotationstechniken auf die Genauigkeit von Machine Learning-Modellen zur Objekterkennung

Alicia Wirth

Abgabe am 20. Oktober 2021

betreut von
Dr. Karsten Tolle
Datenbanken und Informationssysteme

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Aufgabenstellung 1
1.2 Struktur der Arbeit 2

2 Theoretische Grundlagen **3**
2.1 Prinzipien der Bildannotation 3
2.2 Arten der Bildannotation 4
2.3 Techniken der Bildannotation 5

3 Implementierung **8**
3.1 Wahl und Erhalt des Datensatzes 8
3.2 Manuelle Annotation 10
3.3 Training der Modelle 11
3.4 Ergebnisse und Evaluation 12

4 Fazit **15**

Appendix **I**

Literaturverzeichnis **IV**

Abbildungsverzeichnis

3.1 Prozess des Erhalts der Bilddaten bis zur Annotation. 9
3.2 Mit *CVAT* annotierte Münzen. Links: *Greif*. Rechts: *Löwe* und *Schlange*. 11
3.3 `loss` der Modelle. Blau: Mit *Polygon* annotiert. Rot: Mit *Bounding Box* annotiert. 12
3.4 Vorhersage dreier Münzen. Oben: *Polygon*-Modell. Unten: *Bounding Box*-Modell. 13

Tabellenverzeichnis

2.1 Die Bildannotationsarten und ihre beispielhaften Ausgaben. 5
2.2 Häufig genutzte Bildannotationstechniken und ihre Anwendung. 6

3.1 Die zehn häufigsten Tiere und Objekte der SQL-Datenbank von *Corpus Nummorum*. 8

1 Einleitung

Garbage in – Garbage out beschreibt die direkte Abhängigkeit der Qualität des maschinellen Outputs von der Qualität der Input-Daten. Mangelt es den Input-Daten an Qualität, führt dies zu einem unzuverlässigen Output [KR18]. Auch im Bereich des Machine Learnings kann jenes Idiom zum Tragen kommen: Die Performanz und Genauigkeit eines Modells sind stark von den Daten abhängig, mit denen es trainiert wurde. Dabei ist nicht nur eine hinreichend große Anzahl von Daten relevant. Auch qualitative Markierungen mit Ground Truth Labeln sind höchst essentiell [SJZ21].

Mit stets steigender Relevanz und Interdisziplinarität von Machine Learning-Methoden, unter anderem im Bereich von Computer Vision, welchem durch etliche Konferenzen, Experimenten und Workshops Ausdruck verliehen wird [KAH20], sollte jenem Umstand große Beachtung geschenkt werden. Prominente Anwendungsfälle sind beispielsweise das Autonome Fahren [GTCM20] oder die Unterstützung von Ärzt*innen im Erkennen von Krankheiten [MSG+20, SJZ21].

Auch im Bereich der Numismatik spielen Machine Learning-Methoden in Kombination mit Computer Vision eine Rolle, beispielsweise bei der Klassifikation in Münztypen [ZKS08]. Insbesondere mit der stets wachsenden Menge numismatischer Daten im Zuge der Digitalisierung [Gru18] steigt die Relevanz maschineller Unterstützung zur Bewältigung und Analyse. Mit ihrer Hilfe kann die numismatische Forschung beschleunigt werden – sei es durch automatische oder vorschlagsorientierte Klassifikation [ZKS08].

Folglich ist eine qualitative Annotation von Daten zum Trainieren von Machine Learning-Modellen in Bereichen, die über die Informatik hinausgehen, nicht nur essentiell, sondern unabdingbar.

1.1 Aufgabenstellung

Zum Ziel des Forschungsprojektes steht der Vergleich des Einflusses zweier Bildannotationstechniken auf die Genauigkeit von Machine Learning-Modellen zur Detektion von Tieren oder Objekten auf antiken, griechischen Münzen der Datenbank von *Corpus Nummorum*[1].
Entsprechend steht neben einer grundlegenden Recherche der Bildannotation und dem Herausfiltern zweier zum Anwendungsfall passender Annotationstechniken auch die Annotation

[1] Corpus Nummorum, `https://www.corpus-nummorum.eu/`, besucht am 07.09.2021

des Datensatzes, das Training zweier Modelle und die Evaluation der endgültigen Performanzunterschiede zur Aufgabe.

Bei der Wahl der Münzprägungen als Datensatz für Training, Validierung und Testen darf sich auf drei Fälle fokussiert werden. Diese spiegeln die Schwierigkeit der Erkennung der geprägten Tiere oder Objekte aus menschlicher Sicht wider. Für jeden Fall soll ein auf einer Münze präsentes Tier oder Objekt gewählt werden.

1.2 Struktur der Arbeit

Zunächst werden die theoretischen Grundlagen der Bildannotation auf Basis einer wesentlichen Recherche vermittelt. Diese umfassen sowohl allgemeine Aspekte und Automatisierungsmöglichkeiten, als auch prominente Annotationsarten und -techniken.

Anschließend wird die praktische Umsetzung des Projekts aufgeführt. Zuerst werden auf Münzen geprägte Tiere oder Objekte den drei Fällen entsprechend gewählt und als Datensatz angesammelt. Danach wird jener Datensatz manuell und mit zweierlei Annotationstechniken mit dem freien Tool *CVAT*[2] annotiert. Das Training beider Modelle erfolgt mittels des *Mask R-CNN* Frameworks[3], woraufhin erzielte Ergebnisse evaluiert werden.

Schlussendlich wird ein Fazit gezogen und mögliche zukünftige Auswirkungen präsentiert.

[2]CVAT, Online: `https://cvat.org/`, GitHub: `https://github.com/openvinotoolkit/cvat`, besucht am 07.09.2021

[3]Mask R-CNN, `https://github.com/matterport/Mask_RCNN`, besucht am 07.09.2021

2 Theoretische Grundlagen

In diesem Kapitel werden die für das Projekt relevanten theoretischen Grundlagen aufgeführt. Diese umfassen die generellen Prinzipien der Bildannotation, aber auch mögliche Annotationsarten, wie die *Object Detection*, und Annotationstechniken, wie *Bounding Boxen*. Zusätzlich werden auch mögliche Automatisierungsmöglichkeiten und damit einhergehende Problematiken erwähnt.

2.1 Prinzipien der Bildannotation

Durch die Bildannotation werden detaillierte textuelle Beschreibungen oder Schlüsselworte mit Bildern assoziiert. Relevant ist sie nicht nur für einen vereinfachten Zugriff und damit eine effektivere Suche von Bildern [Han08], sondern auch zur Implementierung von überwachten Machine Learning-Methoden, bei denen Datensätze mit qualitativen Ground Truth Labeln annotiert werden müssen [SJZ21].

A survey of methods for image annotation [Han08] des Autors Allan Hanbury folgend, stellt die Bildannotation einen von zwei Ansätzen zur Assoziation von Texten und Bildern in der Computer Vision-Literatur dar. Als anderer Ansatz gilt die Kategorisierung, bei welcher jedes Bild einer vordefinierten Kategorie zugeordnet wird [Han08].

Die Durchführung der Bildannotation kann vollautomatisiert, semi-automatisiert über Vorschläge oder gänzlich manuell erfolgen. Eine manuelle Annotation ist sowohl zeit-, als auch kostspielig und kann durch ihre Subjektivität zu Mehrdeutigkeit einzelner Bilder führen [CZF+18, SJZ21]. Insbesondere die vollautomatisierte Annotation bildet den zeitweisen Gegenstand der Forschung im Bereich der Computer Vision [SJZ21].

Es wird zwischen verschiedenen Arten von Informationen zur Assoziation mit Bildern oder Videos unterschieden. Die *inhaltsunabhängigen Metadaten* beschreiben den Inhalt des Bildes oder Videos nicht direkt, sondern damit verbundene Informationen, wie zum Beispiel den Namen des Autors oder das Erstellungsdatum. *Inhaltsabhängige Metadaten* hingegen beziehen sich auf Merkmale niedriger oder mittlerer Ebenen, wie zum Beispiel die Farbe oder Textur [Han08].

Erst die *inhaltsbeschreibenden Metadaten* beschreiben die Semantik des Inhalts und die Beziehung zwischen realen und Bildelementen. Auch Aspekte der Emotionen und tiefere Be-

deutungen durch visuelle Zeichen und Szenen finden dabei Beachtung [Han08]. Zur Lösung der Aufgabe aus Kapitel 1.1 und damit der Beschreibung von Inhalten, die auf Münzen geprägt sind, sind letztere Informationen relevant und zu annotieren. Jedoch gibt es auch dabei unterschiedliche Vorgehensweisen.

Zunächst existiert die Option des *Freitextes*, die keiner vordefinierten Struktur folgt und beispielsweise eine Beschreibung durch ganze Sätze erlaubt. Mit *Schlüsselworten* hingegen werden subjektive Assoziationen anhand eines einzelnen Begriffs beschrieben [SJZ21, Han08].

Darüber hinaus können Beschreibungen auch mittels *Taxonomien* oder *Ontologien* umgesetzt werden. Bei *Taxonomien* werden Begriffe in hierarchischen Bäumen strukturiert, während bei *Ontologien* Konzeptualisierungen spezifiziert werden, die zum Vorschlagen von Labeln benutzt werden können. Zuletzt besteht die Möglichkeit, *Instanzlabel* einzusetzen, um der Bildbeschreibung eine tiefere Bedeutung zu geben, wie zum Beispiel durch das Hinzufügen des Namens eines bestimmten Hundes [SJZ21].

In der Praxis gibt es eine Vielzahl von kostenlosen und kostenpflichtigen Tools zur manuellen, semi-automatisierten oder vollautomatisierten Annotation von Bildern mit den unterschiedlichsten Techniken. Auch Exportformate sind in großer Breite gegeben. Die wohl typischsten sind *Pascal VOC*, *COCO* und *YOLO* [SJZ21].

2.2 Arten der Bildannotation

Abhängig von dem Ziel, welches es mit den annotierten Bildern zu erreichen gilt, ist eine passende Annotationsart zu wählen. *Image Annotation for Computer Vision* [CLO20] folgend, existieren primär vier Annotationsarten, die als nächstes genauer aufgeführt werden.

Bei der *Image Classification* wird versucht zu identifizieren, ob bestimmte, ähnliche Entitäten in einem ausgewählten Bild eines Datensatzes präsent sind. Weitere Informationen, wie zum Beispiel die Position der Entität im Bild oder die Anzahl der Entitäten, sind dabei nicht von Bedeutung.

Dagegen werden jene Informationen bei der *Object Detection* relevant. Es wird versucht, entsprechende Entitäten in einem Bild eines Datensatzes zu erkennen und sie korrekt zu labeln. Dabei können mehrere gleiche oder ungleiche Entitäten in einem einzelnen Bild gelabelt werden. Entsprechend ist nicht nur ihre Präsenz, sondern auch ihre Position und Anzahl relevant.

Die *Segmentierung* geht einen Schritt weiter. Nun wird auch der Kontext relevant, in dem sich die Entitäten im Bild befinden. Jedem Pixel eines Bildes oder eines ausgewählten Bildteiles wird ein Label zugeordnet.

Bei der *Semantischen Segmentierung* werden dabei gleiche Entitäten gruppiert, sodass ihre Anzahl nicht mehr zu ermitteln ist. Dafür können eventuell die Größe und Form der Entitäten ausgemacht werden. Hingegen wird bei der *Instanzsegmentierung* jede Entität individuell gelabelt und nicht nach Entitätstypus gruppiert. Es sind Präsenz, Position, Anzahl, Größe und Form zu identifizieren. Beide Segmentierungsarten können zur *Panoptischen Segmentierung* kombiniert werden.

Abschließend werden bei der *Boundary Recognition* Grenzen von Entitäten, Linien und Kurven erkannt. Insbesondere zur Bestimmung von Fahrspuren und Bürgersteigen beim Autonomen Fahren kommt diese Annotationsart zum Einsatz.

Mögliche Ausgaben der einzelnen Arten können der folgenden Tabelle 2.1 entnommen werden. Diese dient dem leichteren und schnelleren Verständnis.

Bildannotationsart	**Beispielhafte Ausgabe**
Image Classification	„X ist im Bild“
Object Detection	„X ist an diesen zwei Positionen im Bild“
Semantische Segmentierung	„All diese Pixel gehören zu Gruppe X, die anderen Pixel gehören zu Gruppe Y“
Instanzsegmentierung	„Diese Pixel gehören zu X_1, diese Pixel gehören zu X_2, diese Pixel gehören zu X_3,...“
Panoptische Segmentierung	„Diese Pixel gehören zu X_1, diese Pixel gehören zu X_2, die anderen Pixel gehören zu Gruppe Y“
Boundary Recognition	„Diese Linien grenzen X ein“

Tabelle 2.1: Die Bildannotationsarten und ihre beispielhaften Ausgaben.

2.3 Techniken der Bildannotation

Ähnlich dem vorgegangenen Kapitel 2.2 ist auch die Wahl einer passenden Bildannotationstechnik relevant. Diese hängt von dem spezifischen Anwendungsfall ab und kann sowohl Kosten, als auch Nutzen der Annotation stark beeinflussen [MJTS19]. Die gängigsten Annotationstechniken werden im Folgenden aufgeführt.

Die vermutlich am häufigsten in der Literatur genutzte Annotationstechnik ist die *Bounding Box*. Dabei wird eine Box um die zu annotierende Entität gezogen. In der Regel hat diese eine rechteckige Form, jedoch gibt es beispielsweise auch runde, elliptische oder gar dreidimensionale Ausführungen (sogenannte *3D Cuboids*) [SJZ21, CLO20]. Den meisten Sinn finden *Bounding Boxen* bei zu annotierenden Entitäten, die symmetrisch sind oder wenn deren genaue Form weniger von Interesse ist [CLO20].

Bei der *Key-Point-* oder *Landmark*-Annotation werden Punkte auf relevante Merkmale gesetzt. Beispielsweise können Charakteristika wie Augen, Nase und Mund annotiert werden, welche im Bereich der Gesichtserkennung Anwendung finden. Anhand der Körperposition und -ausrichtung können zum Beispiel Hände getrackt werden [SJZ21, CLO20].

Demgegenüber steht die *Polygon*-Annotation. Zu annotierende Entitäten werden genau nachgefahren, indem ihre höchsten Punkte markiert und Kanten beschriftet werden. In der Regel werden *Polygone* angewendet, wenn Entitäten eine unregelmäßigere Form vorweisen [SJZ21, CLO20].

Während die *Polygon*-Annotation geschlossene Formen erzeugt, bleiben die kontinuierlichen Linien der *Polyline*-Annotation offen. Entsprechend wird sie für das Arbeiten mit offenen Formen verwendet, wie zum Beispiel bei der Markierung von Fahrspuren für das Autonome Fahren [SJZ21, CLO20].

Darüber hinaus gibt es eine Vielzahl weiterer Annotationstechniken, von denen bestimmte Anwendungsfälle profitieren können. So gibt es beispielsweise zur Annotation von Videomaterial, bei welchem sich Entitäten über mehrere Bilder bewegen, Techniken wie das *Tracking* [CLO20]. Eine Zusammenfassung der Annotationstechniken ist in Tabelle 2.2 aufzufinden.

Bildannotationstechnik	**Anwendung**
Bounding Box	Box um eine Entität ziehen; sowohl in 2D und 3D, als auch mit verschiedenen Formen möglich
Key-Point-/Landmark-Annotation	Punkte auf relevante Merkmale setzen
Polygon-Annotation	Entität genau nachfahren
Polyline-Annotation	Wie Polygon, nur keine geschlossene Form

Tabelle 2.2: Häufig genutzte Bildannotationstechniken und ihre Anwendung.

Bedeutsam bei der Wahl einer passenden Annotationstechnik ist jedoch nicht nur eine genaue Betrachtung des Anwendungsfalles und der vorliegenden Daten selbst, sondern auch eine Kosten-Nutzen-Analyse. So nehmen einige Techniken mehr Zeit in Anspruch, können aber gleichermaßen die Genauigkeit von Machine Learning-Modellen steigern.

In *Comparing the Effects of Annotation Type on Machine Learning Detection Performance* [MJTS19] von Mullen et al. wird insbesondere der Einfluss von *Bounding Boxen*, *Polygonen* und *Key-Points* auf die Performanz von Machine Learning-Modellen untersucht. Den Autor*innen folgend dauert ihre Annotation mit *Polygonen* ganze sechsmal und die mit *Bounding Boxen* ganze zweimal länger als jene mit *Key-Point*. Im Endeffekt kann kein signifikanter Einfluss der untersuchten Annotationstechniken auf die Genauigkeit ihrer Objekterkennungsmodelle festgestellt werden [MJTS19].

Jedoch sollte der Anwendungsfall der Publikation beachtet werden: Das Ziel ihrer Model-

le ist, Autos auf Satellitenbildern zu erkennen. Die rechteckige und symmetrische Form der Autos kann dabei offensichtlich bereits gut von *Bounding Boxen* erfasst werden. Insbesondere unregelmäßige Formen mit einer größeren Anzahl von False Positive Pixeln innerhalb der Annotation könnten zu einem Vorteil der *Polygon*-Annotation führen. Genauer wird dieser Umstand in Kapitel 3 untersucht.

3 Implementierung

Das folgende Kapitel erläutert die praktische Implementation zur Lösung der Aufgabenstellung 1.1. Dazu wird sowohl die Gewinnung des Datensatzes, die manuell durchgeführte Annotation der Daten und das Training anhand eines Machine Learning-Frameworks, als auch eine Evaluation der erzielten Ergebnisse aufgeführt.

3.1 Wahl und Erhalt des Datensatzes

In einem ersten Schritt müssen Münzen mit passenden Prägungen als Datensatz für Training, Validierung und Testen der in Kapitel 1.1 gestellten Aufgabe entsprechend gewählt und erlangt werden. Zur Verfügung stehen sowohl eine SQL-Datenbank, als auch die Webseite und ein RDF-Dump von *Corpus Nummorum*.

Um sich einen groben Überblick zu verschaffen und eine hinreichend große Anzahl an Daten von Tieren oder Objekten zur Generierung sinnvoller Ergebnisse und Spielraum nutzen zu können, bietet sich das Durchforsten der SQL-Datenbank an. Tabelle 3.1 zeigt die zehn am häufigsten auf Münzen geprägte Tiere und Objekte. Auf Gipsabdrücke wird dabei explizit verzichtet.

Tier	**Anzahl**
Horse	797
Eagle	775
Serpent	751
Lion	652
Griffin	490
Bull	314
Tunny	308
Ram	258
Goat	255
Dolphin	248

Objekt	**Anzahl**
Head	7554
Bust	3185
Paludamentum	1938
Cuirass	1927
Helmet	1075
Patera	837
Spear	714
Sceptre	575
Shield	555
Cornucopia	513

Tabelle 3.1: Die zehn häufigsten Tiere und Objekte der SQL-Datenbank von *Corpus Nummorum*.

Indes soll nicht nur die Anzahl der vorhandenen Prägungen pro Kategorie zur Wahl der Daten relevant sein, sondern auch der Schwierigkeitsgrad ihrer Identifikation aus menschlicher Sicht.

Dafür werden drei Fälle definiert, denen jeweils ein individuelles Tier oder Objekt zugewiesen werden soll.

Fall 1: *Einfach identifizierbare Tiere oder Objekte.* Auf Münzen geprägte Tiere oder Objekte dieses Falles gleichen sich stark in ihrer Darstellung über jede Münze. Sowohl ihre Größe, als auch ihre grobe Position und ihr dargestellter Ausschnitt unterscheiden sich kaum.
Die Entscheidung fällt in diesem Fall auf mit einem ***Greif*** geprägte Münzen.

Fall 2: *Schwer identifizierbare Tiere oder Objekte.* Sowohl die Größe, als auch die Darstellungsweise von geprägten Tieren oder Objekten dieses Falles verändern sich über verschiedenen Münzen. Häufig sind sie schwer von anderen Entitäten des Münzdesigns zu unterscheiden oder gar mit ihnen verworren.
Für diesen Fall wird sich für mit einer ***Schlange*** geprägte Münzen entschieden.

Fall 3: *Kombination aus Fall 1 und Fall 2.* Hierbei wird sich aus oben angebrachten Aspekten für mit einem ***Löwe*** geprägte Münzen entschieden.

Zur Erlangung der Bilder der entsprechenden Münzen kommt in einem ersten Versuch die SQL-Datenbank und der RDF-Dump von *Corpus Nummorum* zum Einsatz. Dabei sollen die Münz-IDs und Münzseiten (*obverse* oder *reverse*) mittels SQL-Statement angesammelt und anschließend ihre Bild-URIs aus dem RDF-Dump heruntergeladen werden. Ein geschriebenes Python-Skript automatisiert das Vorgehen. Trotz des Ausschlusses unerwünschter Gipsabdrücke im SQL-Statement, zieht diese Herangehensweise jene oder kaputte Bilder aus dem RDF-Dump mit sich.

Stattdessen verwendet ein neues Python-Skript nun die SQL-Datenbank und Webseite von *Corpus Nummorum*. Weiterhin werden Münz-IDs und Münzseiten, die keine Gipsabdrücke sind, über ein SQL-Statement erhalten.

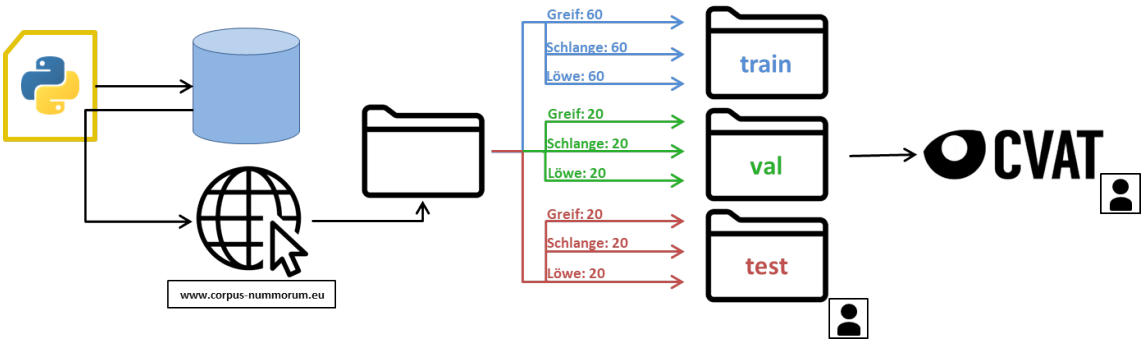


Abbildung 3.1: Prozess des Erhalts der Bilddaten bis zur Annotation.

Doch nun werden diese verwendet, um den Link zur entsprechenden Webseite auf `https://www.corpus-nummorum.eu/` zu generieren und deren HTML-Code abzugreifen. Über die richtige Stelle im HTML-Code kann nun an Bild-URIs gelangt und diese heruntergeladen werden. Dieser Vorgang wird für alle drei Fälle wiederholt.

Abschließend können jeweils 100 Bilder pro Fall manuell im Verhältnis von 6:2:2 jeweils in `train`-, `val`- und `test`-Ordner einsortiert werden. Es wird darauf geachtet, gut erhaltene Münzen mit erkennbaren und möglichst diversen Prägungen zu wählen, um den zuvor definierten Fällen zu entsprechen. Erst dann kann mit der Annotation der Daten begonnen werden. Abbildung 3.1 zeigt den vollständigen Prozess.

3.2 Manuelle Annotation

Zur Durchführung der Bildannotation muss sich zunächst für zwei zu vergleichende Annotationstechniken nach Kapitel 2.3 entschieden werden. Ihre häufige Verwendung in der Literatur, Einfachheit und guten Ergebnisse machen die Annotation mit *Bounding Boxen* interessant. Demgegenüber steht eine zeitaufwendigere Annotation mit *Polygonen*, die eng an der zu annotierenden Entität liegt und in der Theorie bessere Ergebnisse bei unregelmäßigen Formen erzielen sollte [CLO20]. Insbesondere im Hinblick auf die Formen der Entitäten der in Kapitel 3.1 definierten Fälle, erscheint auch diese Annotationstechnik interessant.

Bereits Veröffentlichungen wie *Comparing the Effects of Annotation Type on Machine Learning Detection Performance* [MJTS19] von Mullen et al. untersuchen den Einfluss von *Polygonen* und *Bounding Boxen*. Dabei liegen jedoch symmetrische Formen zu Grunde, die das fast ähnlich gute Abschneiden von beiden Techniken erklären könnten. Durch dieses Projekt soll dieser Unterschied genauer untersucht werden.

Weiterhin muss sich für ein Tool zur Bildannotation entschieden werden. Dafür werden mehrere freie Tools betrachtet und auf ihre Aktualität, Annotationstechniken mit Fokus auf *Bounding Boxen* und *Polygonen*, Automatisierungsmöglichkeiten und Exportformate untersucht. Eine vollständige Liste findet sich im Kapitel Appendix.

Die Wahl fällt schließlich auf das Open Source Tool *Computer Vision Annotation Tool* oder *CVAT* von Intel, welches beide Annotationstechniken und darüber hinaus gewisse Automatisierungen bietet. Es ist online oder durch eine lokale Installation zugänglich.
Schließlich sind insgesamt 300 Münzbilder mit *Greifen*, *Schlangen* und *Löwen* manuell und eng mit *Polygonen* zu annotieren. Abbildung 3.2 zeigt die Annotation einer Münze geprägt mit einem *Greifen*, sowie einer Münze mit einem *Löwen* und einer *Schlange*.

Die Annotation mit *Bounding Boxen* kann von der mit *Polygonen* direkt abgeleitet werden, sodass kein weiterer manueller Annotationsdurchlauf vernommen werden muss. Das gewählte

Abbildung 3.2: Mit *CVAT* annotierte Münzen. Links: *Greif*. Rechts: *Löwe* und *Schlange*.

COCO-Exportformat[1] generiert die *Bounding Box*-Annotationen automatisch.

3.3 Training der Modelle

Schließlich können mit den im vorgegangenen Kapitel annotierten Münzen zwei Modelle trainiert werden – eines auf Basis von *Bounding Boxen* und eines auf Basis von *Polygonen*. Um einen aussagekräftigen Vergleich ziehen zu können, sollten beide Modelle demselben Framework zugrunde liegen. Viele Machine Learning-Frameworks unterstützen jedoch keine Annotationen mit *Polygonen* als Input und fordern eine Konvertierung in *Bounding Boxen*. Dies würde offensichtlich das Ziel des Forschungsprojektes verfehlen.

Nicht so das *Mask R-CNN* Framework, welches einen Schritt über die *Object Detection* hinaus geht und eine *Instanzsegmentierung* ermöglicht [HGDG17]. Die verwendete Implementierung basiert auf Python 3, Tensorflow und Keras, womit jeder Instanz eine *Bounding Box* und Segmentierungsmaske generiert werden kann.

Für den vorliegenden Anwendungsfall wird sich an der Implementierung von Dhruvil Shah [Sha20] orientiert. Der auf GitHub vorliegende Code zur Erkennung von Pferden und Menschen wird für auf Münzen geprägte *Greifen*, *Schlagen* und *Löwen* angepasst. Dazu muss insbesondere die Methode `load_custom` der Klasse `CustomDataset` konfiguriert werden. Dem zugrunde liegt vor allem das für die Annotationen gewählte *COCO*-Exportformat. In der herkömmlichen Implementierung wurde mit dem *JSON*-Format des *VGG Image Annotator (VIA)* gearbeitet, sodass anders auf die einzelnen Werte im Exportformat zugegriffen werden muss.

Darüber hinaus sollen auch mit *Bounding Boxen* annotierte Daten als Input verwendet werden können. Mit einem zusätzlichen `If`-Zweig und dem Zugriff auf den `bbox`- anstelle des `segmentation`-Schlüssels des *COCO*-Exportformats, kann dies leicht umgesetzt werden. Der

[1] COCO-Exportformat, `https://cocodataset.org/#format-data`, besucht am 24.09.2021

Einfachheit halber werden die *Bounding Boxen* ähnlich den *Polygonen* als Maske interpretiert – schließlich liegt das Interesse des Forschungsprojektes dennoch bei der einfachen *Object Detection* und nicht der *Instanzsegmentierung*, weswegen die generierten Masken schlussendlich ignoriert werden können.

Letztlich werden die Modelle mit allen erhaltenen Münzbildern auf Basis von *COCO weights* sowie den beiden Annotationstechniken mit 15 Epochen, 60 `VALIDATION_STEPS` und 180 `STEPS_PER_EPOCH` aufgrund mangelnder GPU-Ressourcen trainiert.

3.4 Ergebnisse und Evaluation

Nachdem beide Machine Learning-Modelle unter Verwendung von *Bounding Boxen* oder *Polygonen* trainiert wurden, können ihre Differenzen und damit ihr Einfluss analysiert und evaluiert werden. Dazu bietet es sich an, die bereits von *Mask R-CNN* festgehaltenen Evaluationsmetriken mittels *Tensorboard*[2] zu betrachten.

Aus dem Code der *Mask R-CNN*-Implementation ist zu erkennen, dass sich der generelle `loss` der Modelle aus den fünf Metriken der ersten Etappe `rpn_class_loss` und `rpn_bbox_loss` sowie der zweiten Etappe `mrcnn_class_loss`, `mrcnn_bbox_loss` und `mrcnn_mask_loss` des Frameworks summiert. Nähere Informationen können in der entsprechenden Veröffentlichung [HGDG17] von He et al. erhalten werden.

Entsprechend Kapitel 3.3 wurden *Bounding Boxen* einfachheitshalber als Masken interpretiert, welche offenkundig sowohl leichter, als auch besser zu bestimmen sind als jene der *Polygon*-Annotationen. Dies verfälscht mit dem `mrcnn_mask_loss` auch den `loss`. Aufgrund dessen und da eine *Instanzsegmentierung* kein Teil der Aufgabe ist, kann der `mrcnn_mask_loss` subtrahiert werden. Es ergibt sich folgender `loss`-Graph 3.3.

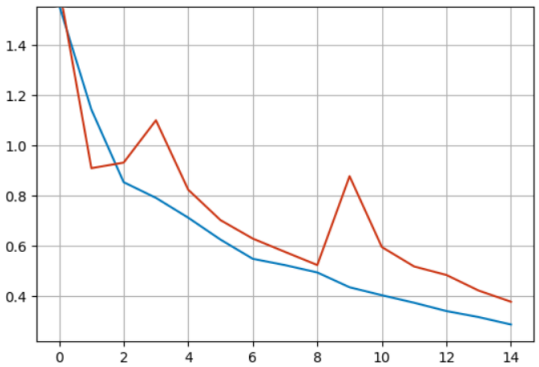


Abbildung 3.3: `loss` der Modelle. Blau: Mit *Polygon* annotiert. Rot: Mit *Bounding Box* annotiert.

[2]Tensorboard, `https://github.com/tensorflow/tensorboard`, besucht am 19.10.2021

Nach 15 Epochen weist das mit *Polygon*-Annotation trainierte Modell einen `loss` von 0.2864 und das mit *Bounding Box*-Annotation trainierte Modell einen `loss` von 0.3772 auf. Auch in fast allen anderen Epochen ist das *Polygon*-Modell überlegen. Einen genaueren Einblick eröffnen die einzelnen Summanden der `loss`-Metrik im Kapitel Appendix. Bereits anhand von ihnen kann erkannt werden, dass das *Polygon*-Modell in beiden Etappen sowohl bezüglich der Klassifikation mit dem `class_loss`, als auch der Objekterkennung mit dem `bbox_loss` bessere Werte erzielt.

Darüber hinaus können zusätzlich von den Modellen prognostizierte Münzentitäten des noch nicht verwendeten `test`-Ordners genauer erörtert werden. Es ist wenig überraschend, dass die schlechtesten Ergebnisse beider Modelle bei *Schlangen* oder auch den schwer zu identifizierbaren Tieren oder Objekten aus Fall 2 auftreten. Lediglich eine einzelne, aber unterschiedliche Münze geprägt mit einer *Schlange* kann von beiden Modellen korrekt klassifiziert werden.

Hingegen können *Greifen* und *Löwen* von beiden Modellen weitaus einfacher mit stets weniger als sieben fehlerhaften Vorhersagen identifiziert werden. Das *Polygon*-Modell schneidet jedoch auch hier durchschnittlich etwas besser ab. Beide Entitäten werden im vorhandenen Datensatz nur miteinander verwechselt, nicht aber mit einer *Schlange*. Diese wird allerdings von den Modellen minimal häufiger mit einem *Löwen*, statt einem *Greifen* vertauscht. Abbildung 3.4 zeigt eine beispielhafte Vorhersage beider Modelle für drei Münzen.

Abbildung 3.4: Vorhersage dreier Münzen. Oben: *Polygon*-Modell. Unten: *Bounding Box*-Modell.

Der wohl größte Kontrast zwischen den Modelle bildet jedoch die Tatsache, dass das mit *Polygon*-Annotation trainierte Modell dazu neigt, mehr als eine tatsächlich präsente Entität zu identifizieren. Dabei ist in der Regel jedoch eine der prognostizierten Entitäten immer tatsächlich auf der Münze abgebildet. So wird beispielsweise für die Münze mit der ID 6500 sowohl ein *Greif*, als auch ein *Löwe* erkannt, auch wenn tatsächlich nur ein einzelner *Greif* abgebildet ist. Ähnlich werden für die Münze mit der ID 19148, welche mittig in Abbildung 3.4 vertreten ist, zwei *Schlangen* erkannt, auch wenn nur eine geprägt ist. Im Gegensatz dazu neigt das *Bounding Box*-Modell eher dazu, fälschlicherweise keinerlei Entität identifizieren zu können.

Interessanterweise kann kein eindeutiger Zusammenhang zwischen den beiden Sonderfällen des *Polygon*- und *Bounding Box*-Modells ermittelt werden. So sind die Münzen mit fälschlich mehrfach auftretenden, vorhergesagten Entitäten des *Polygon*-Modells vom *Bounding Box*-Modell in fast gleichen Teilen korrekt, falsch oder ohne Entität vorhergesagt. Ähnliches gilt für die Vorhersagen der Münzen mit fehlenden Entitäten des *Bounding Box*-Modells, die dem *Polygon*-Modell folgend allerdings häufiger mehrfach auftretende Entitäten vorweisen.

Trotz allem ähneln sich die meisten Vorhersagen beider Modelle. Eine genaue Liste mit den einzelnen Münzen des `test`-Ordners, deren Klassen und Vorhersagen beider Modelle ist diesbezüglich im Kapitel Appendix aufzufinden.

4 Fazit

Die stets steigende Relevanz und Interdisziplinarität von Machine Learning-Methoden fordern neben einer großen Anzahl an Daten insbesondere eine qualitative Markierung jener Daten mit Ground Truth Labeln zur Generierung eines zuverlässigen maschinellen Outputs [SJZ21, KR18]. Auch im Bereich der Numismatik spielt dies eine Rolle, wodurch die Aufgabenstellung 1.1 des Forschungsprojekts gebildet wird. Demnach soll der Einfluss von Bildannotationstechniken auf die Genauigkeit von Machine Learning-Modellen zur Objekterkennung auf antiken Münzen untersucht werden.

Nachdem die theoretischen Grundlagen der Bildannotation umfänglich erarbeitet worden sind, werden in einem ersten Schritt der Datensatz gewählt und angesammelt. Drei Fälle werden aus menschlicher Sicht unterschieden: *Einfach identifizierbare Entitäten* wie geprägte *Greife*, *schwer identifizierbare Entitäten* wie geprägte *Schlangen* und eine *Kombination aus Fall 1 und Fall 2* wie geprägte *Löwen*. Folglich können den Fällen entsprechend jeweils 100 Münzen über die SQL-Datenbank und Webseite von *Corpus Nummorum* heruntergeladen und auf Ordner verteilt werden. Aufgrund ihrer hohen Verheißungen und Präsenz in der Literatur werden die Münzen anschließend mit dem Tool *CVAT* jeweils mit *Bounding Box* und *Polygon* manuell annotiert. Die *Bounding Box*-Annotation lässt sich dabei automatisch aus der *Polygon*-Annotation bestimmen.

Mit dem Machine Learning-Framework *Mask R-CNN* wird dann das Training zweier Modelle über 15 Epochen unternommen, die schließlich im Vergleich stehen sollen. Die Ergebnisse zeigen, dass der `loss` des *Polygon*-Modells dem des *Bounding Box*-Modells um circa 9% überlegen ist. Durch eine genauere Untersuchung anhand der Vorhersagen beider Modelle über Test-Münzen, kann dieses Ergebnis bestätigt werden. Jedoch neigt das *Bounding Box*-Modell dazu, keinerlei Entität zu erkennen, wohingegen das *Polygon*-Modell mehrere Entitäten erkennt, obwohl nur eine präsent ist. Größtenteils treffen beide Modelle allerdings ähnliche Vorhersagen.

Schlussfolgernd bringt das Training mit der *Polygon*-Annotation zwar bessere Ergebnisse, nimmt aber im Vergleich zur *Bounding Box*-Annotation auch weitaus mehr Zeit in Anspruch. So sind für die *Bounding Box*-Annotation der Münzen höchstens zwei Klicks nötig, während für eine enge *Polygon*-Annotation weit über 100 Klicks nötig werden können.

Darüber hinaus könnte die Genauigkeit der Modelle bereits durch eine Erweiterung um mehrere Epochen und dem Hinzunehmen einer größeren Datenmenge gesteigert werden, sodass

ihr Unterschied nicht mehr derart ins Gewicht fällt oder sich gar ausgleicht. Insbesondere wäre es interessant zu untersuchen, wie weit beide Modelle trainiert werden und welche Ergebnisse sie vorweisen könnten, bevor es zu Overfitting kommt. Dies würde jedoch den eingeschränkten Rahmen des Forschungsprojekts sprengen.

Trotzdem sollte beachtet werden, dass die manuelle Annotation von zwei Personen vernommen wurde, was sich offensichtlich stärker in der *Polygon*-Annotation niederschlägt. Durch subjektive Wahrnehmungsunterschiede und deren Annotation könnten Differenzen entstanden sein, die das mit *Polygon*-Annotation trainierte Modell verschlechtern haben könnten. Sollte die Genauigkeit eines mit *Bounding Box*-Annotation trainierten Modells über dies nicht mehr weiter, aber notwendigerweise zu verbessern sein, lohnt sich ein Blick zur zeitaufwendigen *Polygon*-Annotation.

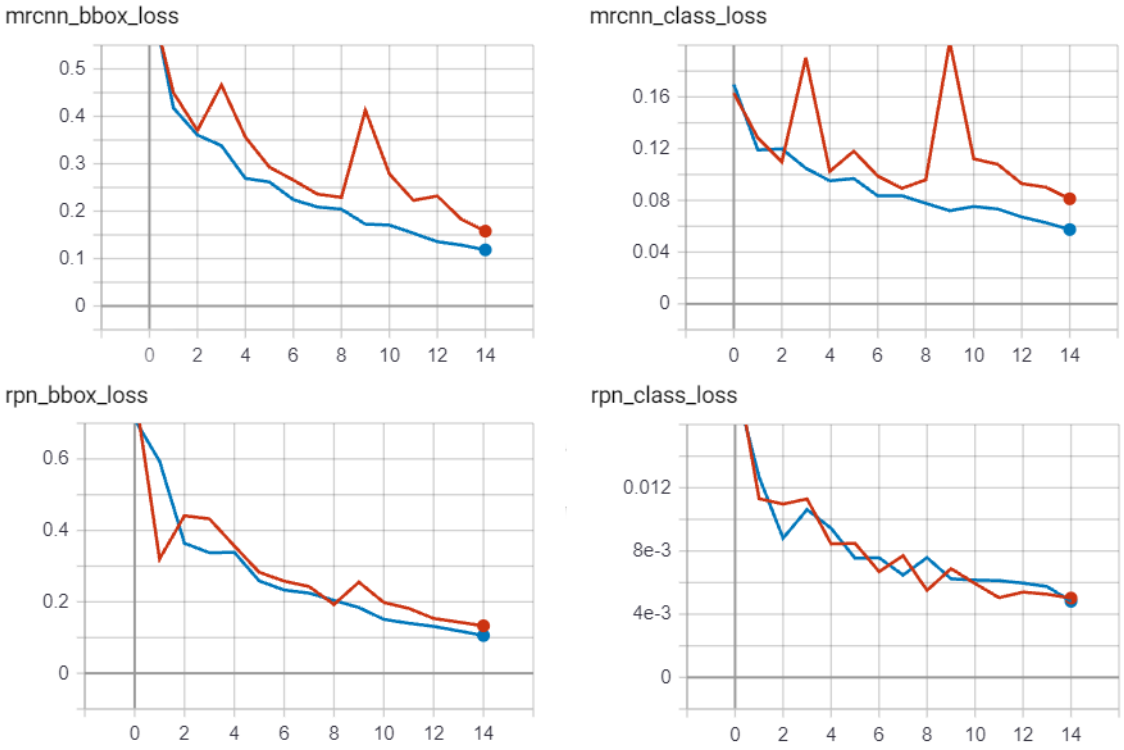
Letztlich ist jedoch ein genaues Abwiegen der Kosten-Nutzen-Kalküle für jeden spezifischen Anwendungsfall von Nöten.

Appendix

Die in dem Forschungsprojekt erwähnten praktischen Implementierungen sind in Kooperation mit Huy Luong im Rahmen seiner Masterarbeit erarbeitet worden und können über den folgenden Link heruntergeladen werden. Sowohl das Python-Skript zum Erlangen der Münzbilder mittels der Datenbank und Webseite von *Corpus Nummorum*, als auch den für das Training der Modelle verwendeten Code und Datensatz sind dabei enthalten.

```
https://github.com/aliciawrt/CN-Detection-Annotation
```

Summanden der `loss`-Metrik



Blau: Mit *Polygon* annotiert. Rot: Mit *Bounding Box* annotiert.

Annotationstools

Name	Link	Aktiv	Letztes Update	Video	Bounding Box	Polygon	Andere Annotationsarten	Automatisierungsgrad	Exportformate
Alturos.ImageAnnotation	https://github.com/AlturosD	Ja	07.02.2020	Nein	Ja	Nein		Manuell	YOLO
Anno-Mage	https://github.com/virajmav	Ja	12.04.2020	Nein	Ja	Nein		Manuell, semi-automatisiert	CSV, XML
Annotorious	https://github.com/recogito/	Ja	31.08.2021	Nein	Ja	Ja		Manuell, semi-automatisiert	Selbst über API zu programmieren
CATMAID	https://github.com/catmaid/	Ja	21.09.2020	Nein					JSON
COCO Annotator	https://github.com/jsbroks/c	Ja	11.06.2021	Nein	Ja	Ja	Point	Manuell, semi-automatisiert	COCO
CVAT	https://github.com/openvino	Ja	26.08.2021	Ja	Ja	Ja	Point, Cuboid, Polyline, Class Labels	Manuell, semi- und vollautomatisiert	VOC, COCO, YOLO und 12 weitere
deepdabel	https://github.com/ivetchma	Ja	13.08.2021	Nein	Ja	Nein	Nein	Manuell, Model inference (pre-trained model trained on a small subs	VOC, YOLO, COCO, KITTI, Google Cloud Platform
ImageTagger	https://github.com/bit-bots/i	Ja	22.06.2021	Nein	Ja	Ja	Line, Point, Ball	Manuell (semi-automatisiert in progress)	unklar
imglab	https://github.com/NaturalIn	Ja	02.12.2020	Nein	Ja	Ja	Circle, Point	Manuell, semi-automatisiert	VOC, COCO und 2 weitere
labelImg	https://github.com/tzutalin/l	Ja	25.07.2021	Nein	Ja	Nein		Manuell	VOC, YOLO,
LabelMe	https://github.com/wkentaro	Ja	27.07.2021	Ja	Ja	Ja	Circle, Polyline, Point, Class Labels	Manuell	VOC, COCO
LOST	https://github.com/l3p-cv/lo	Ja	19.03.2021	Nein	Ja	Ja	Line, Point	Manuell	CSV
make-sense	https://github.com/SkalskiP	Ja	31.08.2021	Nein	Ja	Ja	Point, Line, Label	Manuell, semi-automatisiert	CSV, YOLO, VOC-XML, VGG JSON, COCO JSON
MedTagger	https://github.com/medtagg	Ja	06.01.2021	Nein	Ja	Ja	Point	Manuell	unklar
OpenLabeling	https://github.com/Cartucho	Ja	07.05.2020	Ja	Ja	Nein		Manuell, semi-automatisiert (ABER nur Edgedetector)	VOC, YOLO
OpenLabeler	https://github.com/kinhong/	Ja	07.03.2021	Nein	Ja	Ja		Manuell, Annotationstipps	VOC, COCO
pixelAnnotationTool	https://github.com/abreheret	Ja	02.06.2021	Nein	Nein	Nein	Brush + OpenCV watershed marked	Manuell, semi-automatisiert	PNG mit Maske
VIA	https://www.robots.ox.ac.uk	Ja	04.03.2021	Ja	Ja	Ja	Circle, Point	Manuell	COCO, JSON, CSV
VoTT	https://github.com/microsof	Ja	13.04.2021	Ja	Ja	Ja		Manuell	VOC, CSV, JSON, CNTK und weitere
Pixie	https://github.com/buni-roc	Nein	10.01.2019						
Sloth	https://github.com/cvhciKIT	Nein	02.06.2017						
turktool	https://github.com/jasonyu	Nein	21.01.2018						
Web Annotation	https://github.com/mpizenb	Nein	15.08.2018						
yolo_mark	https://github.com/AlexeyAl	Nein	24.4.2019						
Labelbox	KOSTENPFLICHTIG								

Vorhersagen der Test-Münzen

File	Class	Bbox-Prediction	Polygon-Prediction
11008.jpg	Griffin	Griffin	Griffin
11013.jpg	Griffin	Griffin	Griffin
21159.jpg	Griffin	Griffin	Griffin
21161.jpg	Griffin	Griffin	Griffin
28649.jpg	Griffin	Griffin	Griffin, Lion
28770.jpg	Griffin	Griffin, Lion	Griffin
5905.jpg	Griffin	Griffin	Griffin
5906.jpg	Griffin	Griffin	Griffin
5909.jpg	Griffin	Griffin	Griffin
6421.jpg	Griffin	Griffin	Griffin
6423.jpg	Griffin	Griffin	Griffin
6425.jpg	Griffin	Griffin	Griffin
6449.jpg	Griffin	NULL	Lion
6454.jpg	Griffin	Griffin	Griffin
6500.jpg	Griffin	NULL	Griffin, Lion
6501.jpg	Griffin	Griffin	Griffin
6508.jpg	Griffin	NULL	Griffin, Lion
6510.jpg	Griffin	NULL	Griffin
6616.jpg	Griffin	Lion	Lion
6621.jpg	Griffin	Lion	Lion
10036.jpg	Lion	Lion	Lion
10037.jpg	Lion	Lion	Lion
10038.jpg	Lion	Lion	Lion
10045.jpg	Lion	Lion	Lion
10048.jpg	Lion	Lion	Lion
10194.jpg	Lion	Lion	Lion
10198.jpg	Lion	Lion	Lion
10199.jpg	Lion	Lion	Lion
10200.jpg	Lion	Lion	Lion
10874.jpg	Lion	Griffin	Lion
3138.jpg	Lion	Lion	Lion
3143.jpg	Lion	Lion	Lion
3333.jpg	Lion	Lion	Griffin, Lion
39602.jpg	Lion	NULL	Griffin, Lion
39604.jpg	Lion	Griffin	Griffin
39620.jpg	Lion	Griffin	Griffin, Lion
39629.jpg	Lion	Griffin	Griffin
39632.jpg	Lion	Lion	Griffin, Lion
3121.jpg	Lion * 2	Griffin	Griffin
3311.jpg	Lion * 2	Griffin	NULL
18281.jpg	Serpent	NULL	Serpent
18283.jpg	Serpent	Lion	Serpent, Lion
19133.jpg	Serpent	Lion	Lion
19135.jpg	Serpent	Lion	Lion
19136.jpg	Serpent	NULL	Lion
19148.jpg	Serpent	Serpent	Serpent, Serpent
22994.jpg	Serpent	Griffin	Griffin
22998.jpg	Serpent	Griffin	Griffin
23054.jpg	Serpent	Griffin	Griffin
23055.jpg	Serpent	Griffin	Griffin
26573.jpg	Serpent	Lion	NULL
27920.jpg	Serpent	NULL	Serpent, Griffin
28044.jpg	Serpent	NULL	NULL
40259.jpg	Serpent	Griffin	Griffin
40680.jpg	Serpent	NULL	NULL
5114.jpg	Serpent	Lion	Lion
5115.jpg	Serpent	Lion	Serpent, Lion
5127.jpg	Serpent	Lion	Lion
5214.jpg	Serpent	Lion	Lion
5152.jpg	Serpent * 3	NULL	Serpent, Serpent, Lion

Literaturverzeichnis

[CLO20] CLOUDFACTORY. Image Annotation for Computer Vision: A Guide to Labeling Visual Data for Your Machine Learning Project. `https://www.cloudfactory.com/image-annotation-guide`, 2020. Besucht am 10.09.2021.

[CZF+18] Qimin Cheng, Qian Zhang, Peng Fu, Conghuan Tu und Sen Li. A survey and analysis on automatic image annotation. *Pattern Recognition*, 79:242–259, 2018.

[Gru18] Ethan Gruber. Linked Open Data and Hellenistic Numismatics. In *Alexander the Great: A Linked Open World*, S. 17–33. Ausonius Éditions, 2018.

[GTCM20] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias und Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 37(3):362–386, 2020.

[Han08] Allan Hanbury. A survey of methods for image annotation. *Journal of Visual Languages & Computing*, 19(5):617–627, 2008.

[HGDG17] Kaiming He, Georgia Gkioxari, Piotr Dollár und Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*, S. 2961–2969, 2017.

[KAH20] Asharul Islam Khan und Salim Al-Habsi. Machine Learning in Computer Vision. *Procedia Computer Science*, 167:1444–1451, 2020.

[KR18] Monique F Kilkenny und Kerin M Robinson. Data quality: "Garbage in – garbage out", 2018.

[MJTS19] James F Mullen Jr, Franklin R Tanner und Phil A Sallee. Comparing the Effects of Annotation Type on Machine Learning Detection Performance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, S. 0–0, 2019.

[MSG+20] Scott Mayer McKinney, Marcin Sieniek, Varun Godbole, Jonathan Godwin, Natasha Antropova, Hutan Ashrafian, Trevor Back, Mary Chesus, Greg S Corrado, Ara Darzi et al. International evaluation of an AI system for breast cancer screening. *Nature*, 577(7788):89–94, 2020.

[Sha20] Dhruvil Shah. Mask RCNN implementation on a custom dataset! `https://www.towardsdatascience.com/mask-rcnn-implementation-on-a-custom-dataset-fd9a878123d4`, 2020. Besucht am 06.10.2021.

[SJZ21] Christoph Sager, Christian Janiesch und Patrick Zschech. A survey of image labelling for computer vision applications. *Journal of Business Analytics*, S. 1–20, 2021.

[ZKS08] Sebastian Zambanini, Martin Kampel und Mario Schlapke. On the Use of Computer Vision for Numismatic Research. In *Proceedings of the 9th International Conference on Virtual Reality, Archaeology and Cultural Heritage*, S. 17–24, 2008.