Bachelorarbeit

Departement Bau, Umwelt und Geomatik

Untersuchung von «Deep Learning» - Modellen in ArcGIS Pro zur Extraktion von Merkmalen aus historischen Karten: Objektdetektion und Bildübersetzung

Zürich, 10. Juni 2022

Frühlingssemester 2022

Autorin:
Elina Scheiring
selina@student.ethz.ch

Leiter:
Prof. Dr. Lorenz Hurni

Betreuung:
Dr. Magnus Heitzler
Chenjing Jiao
Xue Xia

ETH Zürich

IKG
Institut für Kartografie
und Geoinformation

Zusammenfassung

Historische Karten bilden eine Quelle für Geodaten aus der Vergangenheit. Besonders für die Identifikation von zeitlichen Entwicklungen, werden diese in der Forschung eingesetzt. Eine Methode zur Analyse dieser Karten stellen Deep Learning Modelle dar, welche es erlauben einzelne Objekte zu detektieren.
Bei der vorliegenden Arbeit werden Deep Learning Modelle in ArcGIS Pro analysiert. Ziel ist es, Feuchtgebiete aus der Landeskarte der Schweiz zu extrahieren. Dabei werden die Modelle zur Objektdetektion und zur Bildübersetzung näher betrachtet.
Es wird der allgemeine Workflow von Deep Learning in ArcGIS Pro aufgezeigt. Das Objektdetektionsmodell Mask R-CNN erweist sich als besonders geeignet für die Extraktion von Feuchtgebieten. Hierbei wird zur Verwendung des Backbone Modells ResNet50 geraten, wobei die vortrainierten Netzwerkgewichte nicht eingefroren werden sollen. Bildübersetzungsmodelle hingegen sind nicht für die Aufgabenstellung anwendbar.
Zusätzlich stellt sich die Genauigkeitsanalyse als Schwierigkeit heraus, wodurch generierte Zahlenwerte mittels visueller Analyse überprüft werden müssen. Generell wird die Wichtigkeit der Trainingsdatenqualität identifiziert.

Inhaltsverzeichnis

Abbildungsverzeichnis **iv**

Tabellenverzeichnis **iv**

1 Einführung **1**
1.1 Historische Karten in der Wissenschaft . . . 1
1.2 Zielsetzung . . . 2

2 Grundlagen **3**
2.1 Theoretische Grundlagen . . . 3
2.1.1 Deep Learning . . . 3
2.1.2 Deep Learning in ArcGIS Pro . . . 3
2.2 Daten . . . 4
2.2.1 Historische Karte . . . 4
2.2.2 Feuchtgebiete . . . 4
2.3 Modelle . . . 5
2.3.1 Objektdetektion . . . 5
2.3.2 Bildübersetzung . . . 9

3 Methodik **13**
3.1 Interface vs. Notebook . . . 13
3.2 Allgemeiner Workflow . . . 13
3.2.1 Vorbearbeitung der Daten . . . 14
3.2.2 Erstellung der Trainingsdaten . . . 14
3.2.3 Training der Modelle . . . 16
3.2.4 Anwendung der Modelle . . . 16
3.2.5 Nachbearbeitung der Modellausgabe . . . 17
3.2.6 Evaluierung der Modelle . . . 17
3.3 Konkret angewandter Workflow . . . 19
3.3.1 Vergleich aller Objektdetektionsmodelle . . . 19
3.3.2 Fokus Mask R-CNN . . . 20
3.3.3 Bildübersetzung . . . 21

4 Resultate **22**
4.1 Vergleich aller Objektdetektionsmodelle . . . 22
4.1.1 Training mit 1'600 Trainingsdaten . . . 22
4.1.2 Einfluss der Anzahl Trainingsdaten . . . 22
4.1.3 Einfluss der Backbone Modell Wahl . . . 23
4.2 Fokus Mask R-CNN . . . 25
4.3 Bildübersetzung . . . 27

5 Diskussion **28**
5.1 Generelle Probleme und Analysen . . . 28
5.2 Diskussion der Modellergebnisse . . . 28
5.2.1 Objektdetektion . . . 28
5.2.2 Bildübersetzung . . . 31
5.3 Diskussion des allgemeinen Workflows . . . 31
5.4 Grenzen der Auswertung . . . 31

5.5 Eignung der Objektklasse 'Feuchtgebiet' für Deep Learning Modelle 32

6 Zusammenfassung und Empfehlung 34

7 Ausblick 34

Literaturverzeichnis I

Abbildungsverzeichnis

1 Kartenausschnitt der Landeskarte mit Feuchtgebieten 4
2 Single Shot Detektion Architektur (Quelle: Esri (2022h)) 5
3 RetinaNet Modelarchitektur (Quelle: Esri (2022g)) 6
4 YOLOv3 Architektur (Quelle: Esri (2022l)) 7
5 Faster R-CNN Architektur (Quelle: Esri (2022b)) 8
6 Mask R-CNN Architektur (Quelle: Esri (2022e)) 9
7 Pix2Pix Architektur (Quelle: Esri (2022f)) 10
8 CycleGAN Architektur (Quelle: Esri (2022d)) 10
9 Super Resolution Architektur (Quelle: Esri (2022i)) 11
10 Image Captioner Architektur (Quelle: Esri (2022j)) 12
11 Grundfunktionen / Workflow Deep Learnig in ArcGIS Pro 13
12 Workflow der Datenaufbereitung bei Bildübersetzungsmodellen 14
13 Beispielhafte Trainingsdaten von Objektdetektion 15
14 Beispielhafte Trainingsdaten von Bildübersetzung 16
15 Workflow der Datennachbearbeitung bei Bildübersetzungsmodellen 17
16 Vergleich der Modelle mit unterschiedlicher Anzahl Trainingsdaten 23
17 Precision der Objektdetektionsmodelle mit verschiedenen Backbone Modellen 23
18 Accuracy von YOLOv3 und RetinaNet bei verschiedenen Backbone Modellen 24
19 Objektdetektionsmodelle im visuellen Vergleich 25
20 Visueller Vergleich der Modelleinstellungen 26
21 Resultate Pix2Pix auf Rasterbild der Grösse 256x256 Pixel (a): Modellausgabe; (b): Modellausgabe mit Ground Truth Polygonen überlagert; (c): Modelleingabe 27
22 Resultate Pix2Pix auf Rasterbild der Grösse 1000x1000 Pixel (a): Modellausgabe; (b): Modellausgabe mit Ground Truth Polygonen überlagert; (c): Modelleingabe 27

Tabellenverzeichnis

1 Metadatenformat für unterschiedliche Modelle (Quelle: Esri (2022k)) 14
2 Grundgrössen für die Modellevaluation 18
3 Unterschiedliche Mask R-CNN Modelleinstellungen 21
4 Genauigkeiten während des Trainingsprozesses mit 1'600 Trainingsdaten . . 22
5 Genauigkeiten bei der Anwendung der Objektdetektionsmodelle 22
6 Genauigkeiten der Mask R-CNN Variationen während des Trainingsprozesses 25
7 Genauigkeiten Mask R-CNN Modellvariationen bei Anwendung 26

1 Einführung

Wenn man die Entwicklungen der Vergangenheit versteht, kann man lehrreiche Schlüsse für die Zukunft ziehen sowie deren Entwicklung auf eine positive Art beeinflussen. Diese Erkenntnis, welche auf unterschiedlichste Forschungsfelder zutrifft, ist nicht neu und beschäftigt unzählige Forschungsteams. Unter anderem auch die Forschung rund um Feuchtgebiete und deren Entwicklung.
Historische Karten bieten in diesem Bezug analoge Abbildungen der räumlichen Vergangenheit und sind oftmals auch die einzige Quelle von historischen Geodaten. Neue Technologische Möglichkeiten bieten eine Chance historische Karten noch genauer untersuchen, vergleichen und analysieren zu können.
Vielversprechend und vielseitig einsetzbar ist Deep Learning. Der Computer lernt dabei grosse Datenmengen nach Mustern und Trends zu untersuchen und mithilfe von neuronalen Netzen innere Strukturen zu erkennen. Softwares, wie ArcGIS Pro integrieren neuartige Deep Learning Modelle in ihr Programm, welche im Zuge dieser Arbeit auf ihre Anwendbarkeit untersucht werden.

1.1 Historische Karten in der Wissenschaft

Historische Karten stellen eine Quelle für geographische, aber auch politische Informationen dar, welche oftmals auch die einzig überbrachten Daten mit Raumbezug sind. Bereits in den 1980er Jahren wurden historische Karten in der Wissenschaft verwendet. Vor allem die Verbreitung von Scans analoger Karten führten zu neuen Möglichkeiten und einem wachsenden Interesse an historischen Karten. (Weiweiiduann et al. (2020))

Es finden sich Anwendungsbereiche in unterschiedlichsten Forschungsgebieten, von Natur-, Gesundheits- und Sozialwissenschaften, bis hin zu Urbanisierung und Landschaftsökologie. Viele Forschungsansätze erfordern interdisziplinäre Zusammenarbeit.
Ein Beispiel ist die Forschung zur Biodiversität. Auf historischen Karten abgebildete Landschaftstypen und generelle räumlichen Begebenheiten können Hinweise über die beheimateten Lebewesen zu einem bestimmten Zeitpunkt geben. (Chiang et al. (2014))

Um historische Karten für weitere Analysen verwenden zu können, bedarf es Methoden, welche den Inhalt der Karten erkennen und extrahieren. Hierbei gibt es starke Anstrengungen in der Informatik sowie der Geoinformationswissenschaft, um dies möglich zu machen.

Grundsätzlich können laut Weiweiiduann et al. (2020) mehrere Herausforderungen bei der Prozessierung von historischen Karten identifiziert werden: das Finden von möglichst effizienten und effektiven Methoden um die Karteninhalte in ein maschinenlesbares Format zu konvertieren, wobei hier auf mögliche Unsicherheiten der Daten sowie Ungenauigkeiten während der Digitalisierung geachtet werden muss; die Verknüpfung der historischen Daten mit andern Datenquellen; und zuletzt der Daten- und Erkenntnisaustausch zwischen Forschenden.
Chiang et al. (2014) betont weiter vor allem die Notwendigkeit einer besseren Kommunikation und mehr Austausch zwischen Forschungsteams. Ein generelles Framework für Kartenprozessierungsmethoden könnte hier helfen alle Möglichkeiten aufzuzeigen und die zur Aufgabenstellung passende Methode zu wählen.

Während die Prozessierung von historischen Karten früher manuell durchgeführt wurde, werden heutzutage meist Machine Learning Modelle verwendet, vor allem seit dem Aufkommen von Convolutional Neural Networks (CNNs). Auch wenn der Fokus bei CNNs auf

dem Prozessieren von Bildern des täglichen Lebens liegt, und somit nicht speziell auf historische Karten ausgerichtet ist, stellen diese Netzwerke eine Chance für historische Karten in der Wissenschaft dar. (Weiweiiduann et al. (2020))

1.2 Zielsetzung

Ziel dieser Arbeit ist es die vorhandenen Objektdetektions- und Bildübersetzungsmodelle in ArcGIS Pro kennenzulernen und ihre Anwendbarkeit auf die Extraktion von Feuchtgebieten zu testen. Folgende Forschungsfragen sollen dabei beantwortet werden:

> Was ist der generelle Workflow von Deep Learning Modellen in ArcGIS Pro?
> Welche Modelle funktionieren am besten für die Extraktion von Feuchtgebieten aus historischen Karten?
> Welche Modelleinstellungen und Parameterwahlen liefert die besten Ergebnisse?

Die Fragen widerspiegeln eine explorative Arbeitsweise. Zuerst wird mit dem Workflow das generelle Vorgehen bei Deep Learning Modellen in ArcGIS Pro bestimmt, worauf eine genauere Betrachtung der einzelnen Modelle folgt. Bei erfolgsversprechenden Modellen werden durch Modelleinstellungs- und Parametervariation weitere Untersuchungen getätigt.

Die untersuchten Objektdetektionsmodelle sind:

- RetinaNet
- YOLOv3
- Single Shot Detektion
- Faster R-CNN
- Mask R-CNN

Weiter werden folgende Bildübersetzungsmodelle betrachtet:

- Pix2Pix
- CycleGAN
- Super Resolution
- Image Captioner

2 Grundlagen

2.1 Theoretische Grundlagen

Deep Learning und die generelle Prozessierung von Bildern und Karten sind Themen, welches sich in den letzten Jahren stark weiterentwickelt haben und sich auch zukünftig noch verändern und verbessern werden. In den nächsten Kapiteln sind für diese Arbeit relevante Grundlagen aufgelistet, welche jedoch nur einen Teil des vielfältigen und detailreichen Gebietes von Deep Learning abdecken.

2.1.1 Deep Learning

Deep Learning ist ein Ansatz von Artificial Intelligence (AI). Genauer gesagt eine Unterkategorie von Machine Learning, welche auf Neuronalen Netzen basiert. Es wird versucht abstrakte Darstellungen in Form von weniger abstrakten Konzepten zu betrachten, wobei eine hierarchische Verschachtelung der Konzepte mithilfe von Layern (Schichten) entsteht. (Goodfellow et al. (2016))

Besonders in den letzten Jahren sind Deep Learning Modelle nützlich geworden, da die verfügbaren Datenmengen stark gestiegen sind. Auch die Verbesserung der Computerinfrastruktur, mittels immer leistungsstärker Rechner, trägt zur steigenden Popularität von Deep Learning bei. Es werden immer bessere Genauigkeiten erreicht, und neue Anwendungsbereiche, in denen Deep Learning nützlich sein kann, gefunden. (Goodfellow et al. (2016))

Convolutional Neural Networks

Unter Convolutional Neural Networks, versteht man Netzwerke für die Datenverarbeitung, welche spezielle Lineare Operationen, namentlich Convolutionen, beinhalten. Im Jahr 2012 wurde AlexNet als erstes CNN vorgestellt, was eine deutliche Qualitätssteigerung von Neural Networks mit sich brachte. Ein bekanntes und weit verbreitetes CNN ist ResNet, welches von Kaiming He am Microsoft Research Institut entwickelt wurde. (Shang & Song (2020))
ResNet Modelle stehen mit unterschiedlichen Layertiefen zur Verfügung, wobei die Anzahl Layer an den Namen ResNet angehängt wird, um dies deutlich zu machen. Trainiert werden ResNet Modelle mit einem Bildset, welches über eine Millionen Bilder beinhaltet (Shang & Song (2020)). Im Rahmen dieser Arbeit stehen ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152 zur Verfügung, welche in ArcGIS Pro integriert sind.

In ArcGIS Pro können CNNs als Backbone Modelle gewählt werden. Dies bedeutet, dass schon vortrainierte Gewichte in den Trainingsprozess miteinbezogen werden.

2.1.2 Deep Learning in ArcGIS Pro

ArcGIS Pro bietet Deep Learning Funktionalitäten für Aufgaben, wie beispielsweise Merkmalextraktion oder Pixelklassifizierung, an. Hierzu ist die Verwendung von unterschiedlichen Frameworks möglich. TensorFlow, Keras sowie Pytorch sind integriert. Zu Beginn ist die Installation eines Evironments notwendig, welches insgesamt 99 Packages beinhaltet. Zusätzlich wird die Esri Erweiterung 'Image Analyst' benötigt. (Esri (2022a))

Bei den Modellen im Rahmen dieser Arbeit wurde Pytorch als Framework genommen, was einer Voreinstellung von ArcGIS Pro entspricht. Auch wurde mit der Version 2.9.0 von

ArcGIS Pro gearbeitet.

ArcGIS stellt weiter die Möglichkeit zur Verfügung mittels eines Notebooks Funktionalitäten mit Python auszuführen. Hierbei bietet das arcgis.learn Modul dieselben Funktionalitäten wie das Interface von ArcGIS Pro.

2.2 Daten

2.2.1 Historische Karte

Als Rastergrundlage dienen 100 Kartenblätter der Landeskarte, welche zwischen 1980 und 1990 publiziert wurden. Diese bilden das Mittelland der Schweiz ab. Die Karten wurden mit einem Masstab von 1:25'000 publiziert, wobei die Genauigkeit bei 1.25 m/Pixel liegt. Einzelne Kartenblätter sind 14000 Pixel breit und 9600 Pixel hoch. Das Koordinatensystem der vorliegenden Blätter ist das CH1903 LV03.

Abbildung 1 zeigt einen Kartenausschnitt der Landeskarte. Die Symbolisierung der Feuchtgebiete sind blaue Striche, wobei die Anordnung dieser keiner erkennbaren Regelmässigkeit folgt. Auch entstehen dadurch keine klaren Grenzen der Feuchtgebietsflächen.

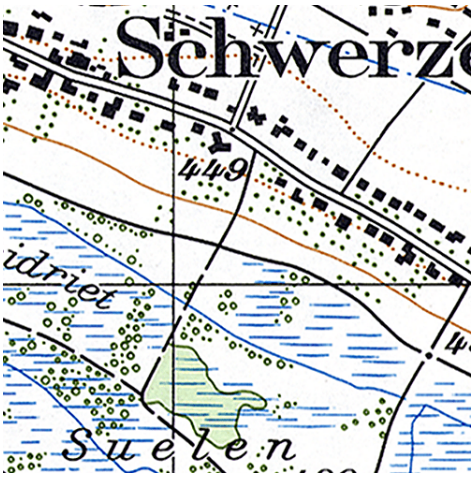

Abbildung 1: Kartenausschnitt der Landeskarte mit Feuchtgebieten

2.2.2 Feuchtgebiete

Als weitere Datengrundlage dient ein Datensatz, der alle gekennzeichneten Feuchtgebiete der Landeskarte als Polygone beinhaltet. Diese konnten von einem schon bestehenden Projekt übernommen werden, bei welchem eine manuelle Vektorisierung stattgefunden hat. Dieses Projekt hat mithilfe von historischen Karten die Entwicklung von Feuchtgebieten im Laufe der Zeit betrachtet (Stuber Martin & Bürgi Matthias (2018)). Die Genauigkeit der manuell erzeugten Feuchtgebietspolygone kann als gut bezeichnet werden, wobei es besonders an den Feuchtgebietsrändern oftmals zu kleinen Unstimmigkeiten kommt und Symbole nicht als Feuchtgebiet erfasst wurden.

Drei Kartenblätter wurden von einer studentischen Hilfskraft noch manuell nachbearbeitet, um die Genauigkeit der Polygone zu erhöhen. Hierbei wurden die Polygone genauer der Landeskarte angepasst, sodass auch kleine Details berücksichtigt wurden. Diese Karten-

blätter waren für das Training und die Evaluierung der Modelle aufgrund der Genauigkeit von besonderer Bedeutung.

2.3 Modelle

2.3.1 Objektdetektion

Unter Objektdetektionsmodellen versteht man Modelle, welche als Modellausgabe eine Bounding Box, also ein rechteckiges Polygon, generieren. Man möchte hier den Ort von Objekten erkennen und ihnen gleichzeitig eine Klasse zuordnen.

Es kann eine Einteilung in one-stage und two-stage Modelle gemacht werden. Der Unterschied ist hierbei, dass bei one-stage Modellen die Locationssuche sowie die Klassifizierung von nur einem Netzwerk übernommen wird. Hier wird meist ein Convolutional Neural Network verwendet, bei welchem ein Durchgang über ein Bild ausreicht, um den Ort und die Klasse eines Objektes zu bestimmen. Bei two-stage Modellen hingegen, übernehmen zwei unterschiedliche Netzwerke dies und trennen somit die Aufgaben voneinander. Generell kann gesagt werden, dass one-stage Modelle schneller sind, da nur ein Netzwerk durchlaufen werden muss. Dies führt dazu, dass diese Modelle oft für Echtzeit Objekterkennungen verwendet werden. Die Verwendung von mehreren Netzwerken bei two-stage Modellen führt hingegen zu einer höheren Genauigkeit. Es kommt somit immer auf die Aufgabenstellung darauf an, welche Modellklasse sich besser eignet.

Nachfolgend werden drei one-stage Modelle, namentlich Single Shot Detektor, RetinaNet und Yolov3 sowie zwei two-stage Modelle, Faster R-CNN und Mask R-CNN genauer vorgestellt.

Single Shot Detektion (SSD)

Abbildung 2 zeigt die Architektur eines Single Shot Detektors. Die ersten Layer, welche hier mit weissen Boxen dargestellt sind, bilden das Backbone Modell ab. Die SSD Layer am Schluss des Ablaufs sind die modelleigenen convolutional Layer, welche auch als SSD head bezeichnet werden. (Esri (2022h))

Das SSD Modell legt über das Bild ein Gitter, wobei in jeder entstandenen Zelle nach Objekten gesucht wird. Dadurch wird auch die Position mitbestimmt. Unterschiedliche Anchor boxes erlauben es, mehrere Objekte, oder Objekte mit verschiedenen Grössen innerhalb einer Gitterzelle zu bestimmen. Während des Trainings werden die Anchor Boxen den Objekten angepasst, damit diese die Objekte bestmöglich treffen. (Esri (2022h))

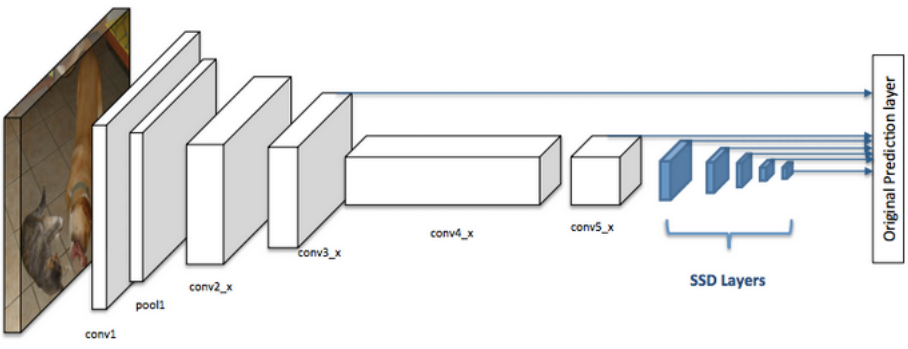

Abbildung 2: Single Shot Detektion Architektur (Quelle: Esri (2022h))

RetinaNet

Das einstufige Objektdetektions Model RetinaNet wird oft für die Erkennung von kleinen Objekten auf Satellitenbildern verwendet. Anhand eines Feature Pyramid Networks (FPN) hebt sich dieses Modell von anderen one-stage Modellen ab. Die Pyramidenstruktur führt dazu, dass man Bilder in kleinere und weniger gut aufgelöste Bilder unterteilt und auf jeder Ebene Objekte extrahiert. (Esri (2022g))

Die Modellarchitektur von RetinaNet kann in vier Teile geteilt werden. In einem ersten Teil (vgl. Teil (a) in Abbildung 3) werden mittels einem Backbone Modell unterschiedliche grosse Merkmalkarten erstellt. Im Teil (b) werden mittels des Top-Down Pfades räumlich gröbere Merkmalkarten aus höheren Pyramidenebenen zusammengeführt. Weiter werden mit den lateralen Verbindungen die Schichten mit derselben Grösse zusammengebracht. Beim Klassifizierung - Unternetzwerk wird im Teil (c) die Wahrscheinlichkeit eines Objekts in einer bestimmten Region ermittelt. Teil (d) regressiert den Offset für jedes Ground Truth Objekt. Weitere Informationen über den Modellaufbau können der Dokumentation von ArcGIS Pro entnommen werden. (Esri (2022g))

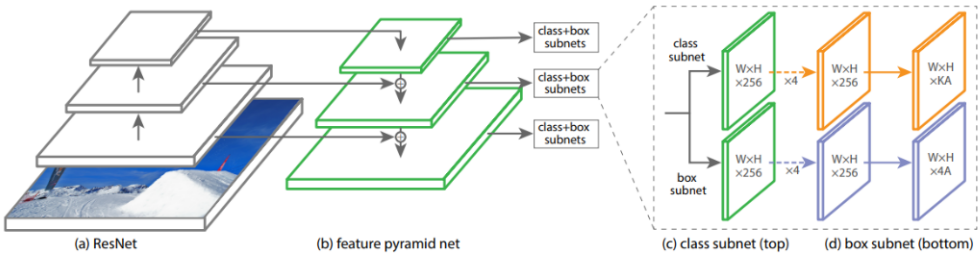

Abbildung 3: RetinaNet Modelarchitektur (Quelle: Esri (2022g))

YOLOv3

YOLO, was für 'You Only Look Once' steht, ist ein populäres Objektdetektionsmodell, welches Anwendung mit Echtzeitdaten findet. Das v3 signalisiert die dritte Version des Modells. Die einstufige Modellstruktur erlaubt es schnelle Klassifizierungen zu erzeugen. Es kann ausserdem auf vortrainierte Gewichte zugegriffen werden, welche 80 Objektklassen erkennen. Dabei handelt es sich um Klassen wie Autos oder Personen und ist dadurch nicht relevant für die Extraktion von Feuchtgebieten aus historischen Karten.
Als Backbone Modell verwendet YOLOv3 DarkNet53 und somit nicht eines der üblichen ResNet Familie. Als Unterschied wird hierbei die Geschwindigkeit genannt. DarkNet soll mit der gleichen Anzahl Layer wie ResNet bis zu zweimal schneller sein. (Esri (2022l))
Die Architektur von YOLOv3 ist in der Abbildung 4 ersichtlich, wobei hier nicht genauer auf die einzelnen Schritte eingegangen wird. Details zur Modellarchitektur werden im Bericht von Ayoosh (Ayoosh (2018)) genauer betrachtet, wobei zusätzlich ein Vergleich zu den Vorgängerversionen von YOLO stattfindet.

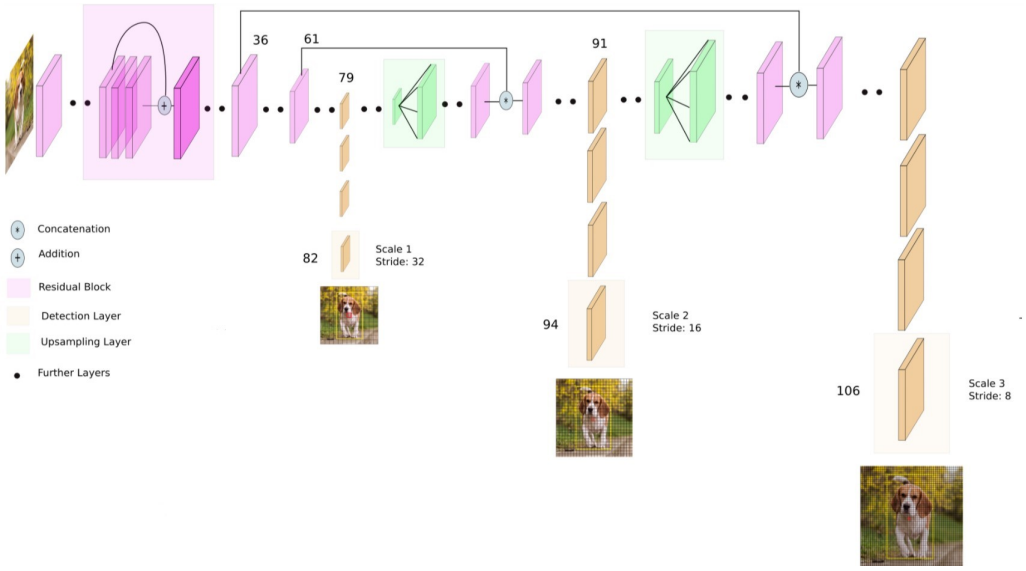

Abbildung 4: YOLOv3 Architektur (Quelle: Esri (2022l))

Faster R-CNN

Der Faster R-CNN Objekt Detektor gehört zur Familie der two-stage R-CNN Modelle. R-CNN steht hierbei für "Regions with Convolutional Neural Networks". Zuerst gab es R-CNN, was in einem ersten Schritt zu Fast R-CNN und in einem zweiten Schritt, mit weiteren Verbesserungen, zu Faster R-CNN weiterentwickelt wurde. Es fanden stets Fortentwicklungen in der Genauigkeit sowie in der Geschwindigkeit der Modelle statt.

Two-stage Modelle waren besonders vor der Entwicklung von one-stage Modellen populär, wobei Faster R-CNN noch immer viel Verwendung findet. Auch wenn Faster R-CNN von der Geschwindigkeit nicht mit one-stage Modellen mithalten kann, ist dieses Modell eines der genausten Objektdetektionsmodelle, die heute existieren.

Als erstes wird bei diesem Modell das Bild in ein CNN eingeführt, wodurch eine Merkmalkarte erstellt wird. Mittels einem Region Proposal Network (RPN) wird ein Regionsvorschlag ermittelt, aus welchem die gefundenen Merkmale durch eine Pooling-Schicht in der Grösse verändert werden. Eine weitere Schicht mit zwei Köpfen, ein Softmax-Klassifikator und ein Bounding-Box-Regressor folgen anschliessend. (Esri (2022b))
Dieses Vorgehen ist auch in der Abbildung 5 ersichtlich, wobei mehr technische Details in einem Bericht von Shilpa (2019) nachzulesen sind.

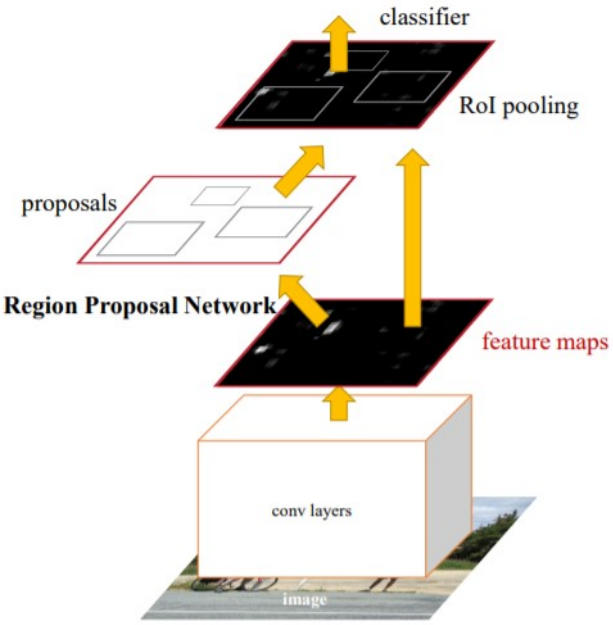

Abbildung 5: Faster R-CNN Architektur (Quelle: Esri (2022b))

Mask R-CNN

Das Mask R-CNN Modell ist kein klassisches Objektdetektionsmodell, bei welchem eine Bounding Box als Modellausgabe generiert wird. Vielmehr ist dieses Modell eine Kombination von Objekterkennung und Pixel Klassifizierung. Dies erlaubt es, die Vorteile der beiden Modellkategorien zu vereinen. In einem ersten Schritt werden Objekte in einer Bounding Box gefunden und in einem zweiten Schritt werden die Pixel innerhalb der gefundenen Boxen klassifiziert. Die generierte Modellausgabe sind somit Polygone, welche die Form der Objekte aufweisen. (Esri (2022e))

Als Modell für die Objektdetektion wird Faster R-CNN verwendet, welches mit zusätzlichen Erweiterungen zur Pixel Klassifizierung ausgestattet wird. Beim Faster R-CNN Modell wird der Rol-Pool (vgl. Abbildung 5) durch RolALign ersetzt, wodurch räumliche Informationen erhalten bleiben. Es wird binäre Interpolation verwendet, um eine Merkmalkarte zu erstellen. Die Ausgabe der RolAlign Schicht wird dann dem Mask Head übergeben, welcher aus zwei Faltungsschichten besteht (vgl. Abbildung 6). Hier werden Pixel für Pixel einem Objekt zugewiesen. (Esri (2022e))

Da es bei Pixelklassifizierungsmodellen oftmals zum Weichzeichnen der Grenzen kommt, was je nach dem nicht gewünscht ist, wurde die Möglichkeit ein weiteres Netzwerk einzubinden erschaffen. Hierbei handelt es sich um ein punktebasiertes Netzwerk, welches Pointrend genannt wird. Dieses verhindert, dass die Grenzen zu sehr geglättet werden. (Esri (2022e))

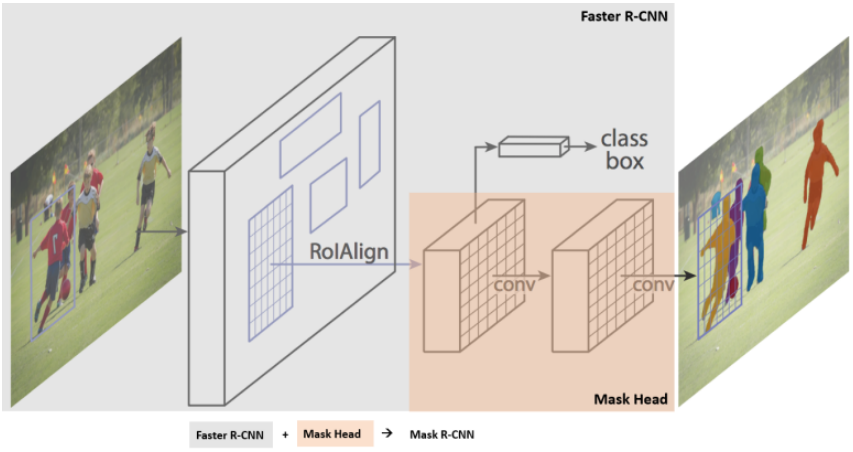

Abbildung 6: Mask R-CNN Architektur (Quelle: Esri (2022e))

2.3.2 Bildübersetzung

Bei der zweiten untersuchten Modellart handelt es sich um Bildübersetzungsmodelle. Ziel ist es ein Rasterbild in ein anderes Rasterbild zu transformieren. Hierbei findet keine konkrete Klassifizierung oder Objektidentifikation statt. Um Feuchtgebiete, im Sinne der Aufgabenstellung dieser Arbeit, identifizieren zu können, bedarf es Vor- und Nachbearbeitungen der Daten, was im Kapitel 3.2.2 genauer erklärt wird.

Pix2Pix

Das Modell Pix2Pix ist eines der bekanntesten und meist verwendeten Bildübersetzungsmodelle. Anwendungsbeispiel sind unter anderem das Umwandeln eines Satellitenbilds in eine abstrakte Karte oder das Umwandeln von RGB Bildern zu Multispektralbildern.
Pix2Pix basiert auf einer GAN-Architektur, bei welcher ein Generator und ein Diskriminator gleichzeitig trainiert werden. Der Generator beschäftigt sich damit Daten zu erzeugen, während der Diskriminator sich mit der Unterscheidung der erzeugten Daten von den echten Daten beschäftigt. (Esri (2022f))
Wie in Abbildung 7 ersichtlich, wird zuerst mit einem Encoder das Input Rasterbild zerlegt, worauf dann die GAN Elemente folgen. Genauere Informationen zu Funktionsweise können der ArcGIS Pro Webseite entnommen werden, wobei auch die Arbeit von Henry et al. (2021) ein gutes Verständnis des Modells vermittelt.

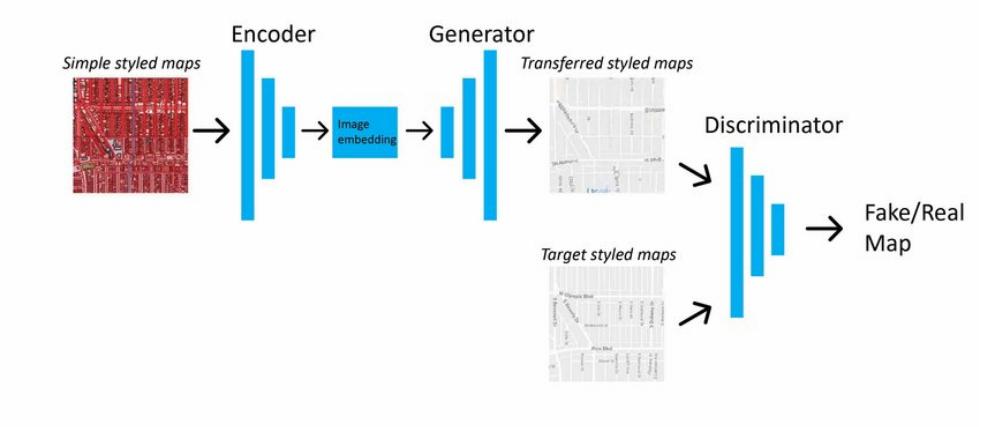

Abbildung 7: Pix2Pix Architektur (Quelle: Esri (2022f))

CycleGAN

Beim Modell CycleGAN handelt es sich um eine Weiterentwicklung des Pix2Pix Modells. Bei Pix2Pix werden gepaarte Rasterbilder benötigt, bei welchen eine klare Einteilung in Modelleingabe und Modellausgabe geschieht. Bei CycleGAN ist dies nicht notwendig. Mithilfe von zwei Generatoren und zwei Diskriminatoren ist eine Konvertierung in beide Richtungen möglich, also muss nicht klar definiert sein was die Eingabe und die Ausgabe ist. Dies ist vor allem vorteilhaft für Aufgabenstellungen, bei welchen beidseitige Konvertierungen gebraucht werden. Die Anwendungsbereiche überschneiden sich mit diesen des Pix2Pix Models. (Esri (2022d))
In Abbildung 8 ist die Ähnlichkeit von CycleGAN zu Pix2Pix erkennbar, wobei eine Erweiterung stattgefunden hat.

Da CycleGAN eine Erweiterung von Pix2Pix darstellt, welche wechselseitige Konvertierungen ermöglicht, welche für die Detektion von Feuchtgebieten jedoch nicht benötigt werden, wurde dieses Modell nicht weiter betrachtet. Weiter stellt diese zusätzliche Funktion eine Verlängerung der Laufzeit dar, wobei nicht von einer Verbesserung der Ergebnisse im Vergleich zu Pix2Pix, bei einer einfachen Konvertierung, ausgegangen werden kann. Dieses Modell wurde bei den Untersuchungen im Rahmen dieser Arbeit nicht weiter betrachtet.

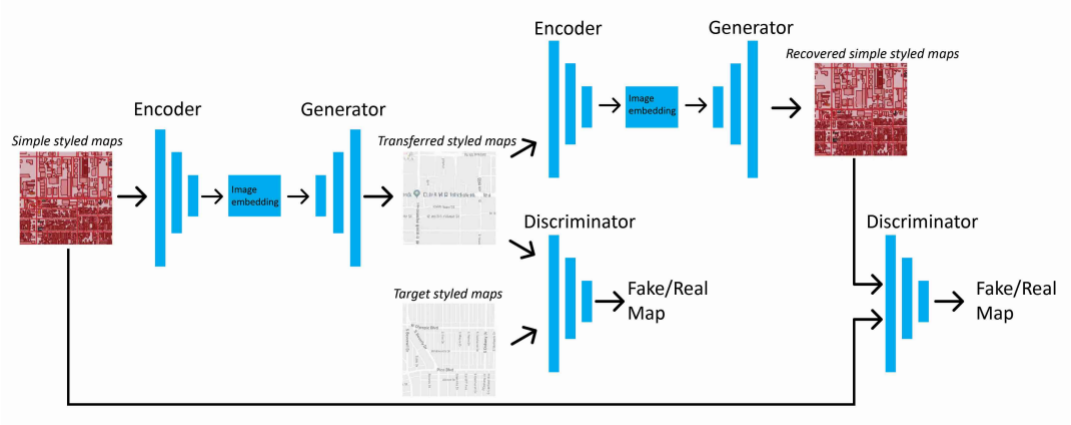

Abbildung 8: CycleGAN Architektur (Quelle: Esri (2022d))

Super Resolution

Das Modell Super-Resolution wird in der Regel dazu verwendet die Bildauflösung eines Rasterbildes zu erhöhen. In einem ersten Schritt wird jeder Pixel eines Rasterbildes mithilfe des 'Image Transformation Networks' so modifiziert, dass das dazugehörige Modellausgabebild entsteht (vgl. Abbildung 9). Das Modell möchte hier lernen, welche Modifikationsschritte notwendig sind, um das Rasterbild zu transformieren. (Esri (2022i))

Da dieses Modell den Fokus auf kleine Veränderungen und Bildauflösungsverbesserungen legt, erscheint eine Anwendung im Rahmen dieser Arbeit als nicht sinnvoll. Es wurden keine Versuche unternommen Feuchtgebiete mit diesem Modell zu detektieren.

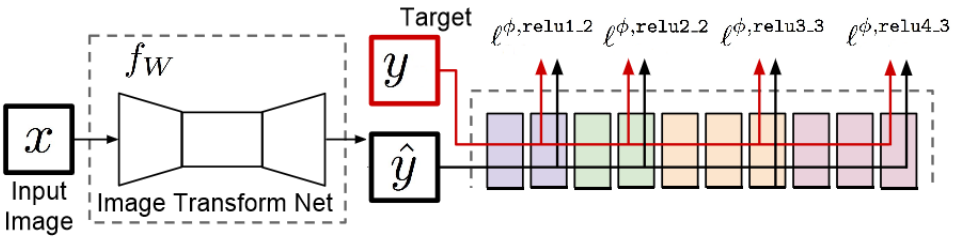

Abbildung 9: Super Resolution Architektur (Quelle: Esri (2022i))

Image Captioner

Ein Image Captioner Modell beschäftigt sich damit Bilder textlich zu beschreiben. Das Modell ist aus einem Encoder und einem Decoder aufgebaut. Der Encoder extrahiert hierbei Objekte von dem Bild und generiert gleichzeitig eine Bildbeschreibung. Wie in Abbildung 10 ersichtlich, werden die Bildobjekte mithilfe Convolutional Layer extrahiert. Ein Rekurrentes neuronales Netz (RNN) sorgt für die Interpretation der extrahierten Objekte. Es kommt ein 'Attention Mechanism' zum Einsatz, welcher dem Modell hilft zu entscheiden, wo nach einem nächsten Wort gesucht wird. In einem nächsten Schritt werden diese zwei Modellausgaben in den Decoder gegeben, bei welchem es sich um ein Sprachmodell handelt. Dieses setzt Wort für Wort eine Aussage zusammen. (Esri (2022j))

Aufgrund des Fokus dieses Modells auf eine textliche Beschreibung des Betrachteten, erscheint eine Anwendung für die Extrahierung von Feuchtgebieten als nicht geeignet. Aus diesem Grund wurden dieses Modell nicht weiter berücksichtigt und getestet.

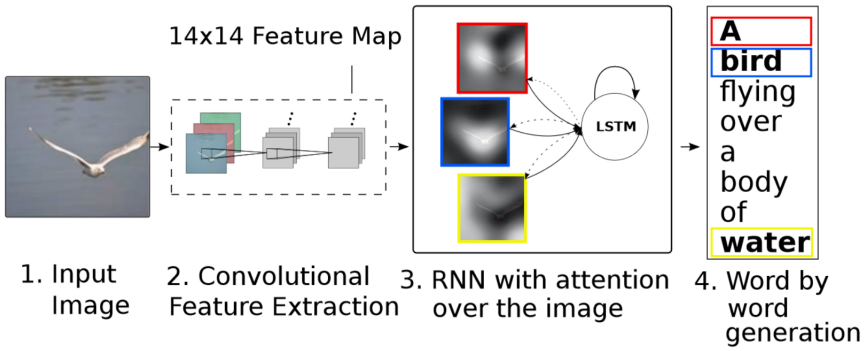

Abbildung 10: Image Captioner Architektur (Quelle: Esri (2022j))

3 Methodik

3.1 Interface vs. Notebook

In ArcGIS Pro 2.9 ist es möglich Funktionen über das Interface direkt, oder auch über die Einbindungen eines Notebooks auszuführen. Bei den Notebooks handelt es sich um Jupyter Notebooks, welche direkt in ArcGIS Pro geöffnet werden. Im Rahmen dieser Arbeit wurden zu Beginn beide Möglichkeiten der Funktionsausführung angewandt und getestet.

Ein Vorteil des Notebooks ist, dass einzelne Schritte sowie eingestellte Parameter direkt ersichtlich sind. Ganze Handlungsabfolgen zum Erfüllen einer Aufgabe können einfach mit einem Notebook weitergegeben und nachvollzogen werden. Als Schwachpunkt kann hier eine fehlende Funktions- und Parametererklärung gesehen werden. Es ist nicht direkt ersichtlich, welche Funktionen verfügbar sind, welche Parameter man einstellen kann und was die Standardeinstellung dieser Parameter ist. Auch wenn mit einer Help Funktion weitere Informationen angezeigt werden, bietet das ArcGIS Pro Interface deutlich mehr Hilfestellung.

Aus diesem Grund erwies sich das Interfaces als intuitiver und einfacher zu bedienen, besonders beim ersten Kennenlernen des Programms. Suchleisten im Interface erleichtern die Suche nach passenden Funktionen. Beim Ausfüllen der benötigten Parameter wird über ein Punktsymbol mit einem kleinen i ein Informationsfenster aufgerufen, was mögliche Parameter auflistet sowie oftmals Wertvorschläge bietet. Auch eine direkte Kontrolle der eingegebenen Parameter findet statt, welche bei einem gefundenen Fehler ein rotes Kreuz neben dem Eingabefeld aufzeigt und die Art des Fehlers spezifiziert.

Es kann gesagt werden, dass sich das Interface zum Einsteigen in ArcGIS Pro gut eignet, weswegen in dieser Bachelorarbeit schlussendlich mit dem Interface gearbeitet wurde.

3.2 Allgemeiner Workflow

Abbildung 11 zeigt den generellen Workflow von Deep Learning Modellen in ArcGIS Pro. Die Namen der einzelnen Blöcke entsprechen hierbei den Funktionsnamen in ArcGIS Pro. Während die ersten zwei Schritte für beide Modelltypen, Objektdetektion und Bildübersetzung, gleich sind, muss beim dritten Schritt unterschieden werden. Dabei handelt es sich um die Anwendung der trainierten Modelle. Bei Bildübersetzungsmodellen müssen zusätzliche Bearbeitungen der Daten erfolgen, was in den Kapiteln 3.2.1 und 3.2.5 beleuchtet wird. Es ist weiter ersichtlich, dass es sich beim Workflow um einen iterativen Prozess handelt, bei welchem stets Optimierungen stattfinden.

Die einzelnen Schritte des Workflows werden in den folgenden Kapiteln näher erläutert.

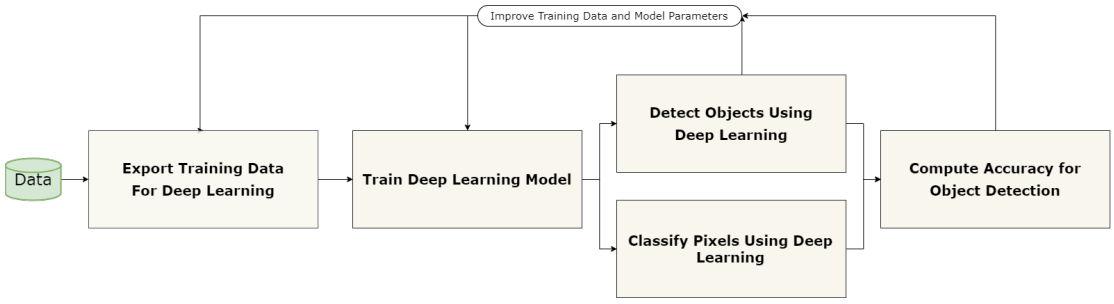

Abbildung 11: Grundfunktionen / Workflow Deep Learnig in ArcGIS Pro

3.2.1 Vorbearbeitung der Daten

Je nach Datengrundlage und Modellanwendungen sind Vorverarbeitungsschritte der Daten notwendig, um diese im gebrauchten Format zu haben.

Bei Bildübersetzungsmodellen werden zwei Rasterbilder benötigt, die das Gleiche abbilden. Da bei dieser Arbeit die Objekte als Feature Layer zur Verfügung standen, müssen zuerst Umwandlungen stattfinden. Die einzelnen Schritte sind in der Abbildung 12 ersichtlich. Mit der Funktion 'Feature To Raster' werden die Polygone in ein Rasterformat umgewandelt, wobei hier die Zellgrösse 1,25 gewählt wird, was der Zellgrösser der Rasterbilder entspricht. In einem zweiten Schritt wird der erstellte Rasterlayer mit der Funktion 'Reclassify' bearbeitet. Hier wird der Wert für ein Feuchtgebiet mit 0 definiert und für kein Feuchtgebiet mit 255. Da der Raster bis jetzt keine Farbinformation besitzt, werden mit der Rasterfunktion Colormap Properties Farbeigenschaften erstellt. Ein letzter Schritt wandelt die Colormap zu 3 Bändern und RGB Farben um, welche für das weitere Vorgehen benötigt werden. Daraus resultiert ein binäres Rasterbild, bei welchem Feuchtgebiete schwarz sind (vgl. Abbildung 14).

Bei Objektdetektionsmodellen ist keine Vorbearbeitung der Daten notwendig, da diese schon in den gebrauchten Formaten vorliegen.

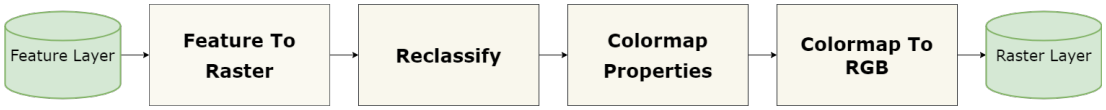

Abbildung 12: Workflow der Datenaufbereitung bei Bildübersetzungsmodellen

3.2.2 Erstellung der Trainingsdaten

Um Trainingsdaten zu erzeugen, wird die Funktion 'Export Training Data For Deep Learning' verwendet. Der entscheidende Parameter ist das Metadataformat, denn hier wird schon entschieden, für welche Modellart die Trainingsdaten später verwendet werden. Tabelle 1 zeigt die möglichen Metadatenfomate sowie die dazugehörigen Modelle.

KITTI_rectangles	Faster RCNN, RetinaNet, SingleShotDetector, YOLOv3
PASCAL_VOC_rectangles	Faster RCNN, RetinaNet, SingleShotDetector, YOLOv3
RCNN_Masks	MaskRCNN
Labeled_Tiles	Image classification, FeatureClassifier model
Multi-labeled Tiles	FeatureClassifier model
Export Tiles	ChangeDetector, CycleGAN, Pix2Pix and SuperResolution models
Classified_Tiles	BDCNEdgeDetector, DeepLab, HEDEdgeDetector, MultiTaskRoadExtractor, PSPNetClassifier and UnetClassifier models

Tabelle 1: Metadatenformat für unterschiedliche Modelle (Quelle: Esri (2022k))

Für die untersuchten Objektdetektionsmodelle wurden PASCAL_VOC_rectangles als Metadatenformat ausgewählt. Das Modell Mask R-CNN benötigt ein eigenes Metadatenformat, da hier auch die Modellausgabe anders aussieht, wie in Kapitel 2.3 genauer beschrieben. Für die Bildübersetzungsmodelle wurde das Metadatenformat Export Tiles verwendet.

Weitere Parameter, welche eingestellt werden können, sind die Grösse der einzelnen Trainingsdaten sowie ein Stride Wert, welcher einen Schrittwert definiert, mit welchem man sich über das Bild, beim Suchen von Objekten, bewegt.

Objektdetektion

Für Objektdetektionsmodelle wird weiter ein Rasterbild sowie ein Feature Layer als Input benötigt. Der Feature Layer beinhaltet bereits definierte Objekte und deren Klassen. Beim Exportieren von Trainingsdaten für Objektdetektionsmodelle kann zusätzlich gesagt werden, dass nur Tiles hinzugefügt werden, welche Features beinhalten. Dadurch wird Speicherkapazität gespart und das spätere Training wird effizienter.

Die exportierten Trainingsdaten werden in einem Ordner abgelegt, welcher zusätzlich ein Textfile mit Statistiken zu den Daten enthält. Bei Trainingsdaten mit dem Matedatenformat PASCAL_VOC_rectangles wird ein Ordner mit den Bildausschnitten und ein Ordner mit den dazugehörigen HTML labels erstellt. Abbildung 13 zeigt Beispiele der generierten Trainingsdaten.

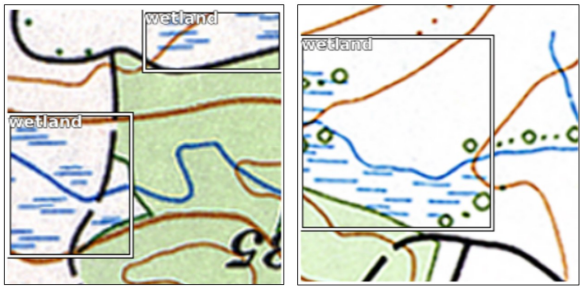

Abbildung 13: Beispielhafte Trainingsdaten von Objektdetektion

Bildübersetzung

Nach der Datenvorbereitung, wie im Kapitel 3.2.1 genauer beschrieben, sind die Daten im richtigen Format um die Funktion 'Export Training Data For Deep Learning' anzuwenden. Es muss hierbei noch ein Mask Layer, welcher der Grösse und Form des Rasterbildes entspricht, der Funktion mitgegeben werden. Dieser sorgt dafür, dass nur Rastertiles innerhalb dieses Maskpolygons erstellt werden. Wenn dieser Mask Layer fehlt, werden Rastertiles auch an den Rändern des Input Rasters generiert, wobei alles ausserhalb als schwarze Fläche markiert wird. Da bei diesen Modellen die Farbe Schwarz als Feuchtgebiet definiert wurde (vgl. Abbildung 14), würde dies zu falschen Detektionen führen.

Beim Generieren von Trainingsdaten mit dem Metadatenfomat Export Tiles werden zwei Ordner mit denselben Bildausschnitten erstellt. Einer davon beinhaltet die historischen Karten und der andere die binären Bilder. Abbildung 14 zeigt zwei Beispiele der Trainingsdaten. Das linke Bild ist jeweils ein Ausschnitt aus dem Orginalraster und das rechte Bild stellt den dazugehörigen transformierten Bildausschnitt dar.

Bei den Trainingsdaten für Bildübersetzungsmodelle ist es nicht möglich automatisch nur Tiles zu erhalten, welche Objekte beinhalten. Der Grund hierfür ist, dass das Rasterbild nicht klassifiziert ist und somit keine Informationen über die Objekte enthält. Dies führt dazu, dass die Anzahl der Trainingsdaten sehr schnell sehr hoch wird und viele Rastertiles nur weiss sind. Abhilfe könnte hier die Verwendung von weiteren Masks schaffen, welche den Ort für das Erstellen von Tiles eingrenzen. Dies wurde bei den vorhandenen Trainingsdaten

nicht verwendet, da die Feuchtgebiete verstreut auf den Rasterbildern sind und Masken mit grossem Aufwand manuell erzeugt werden müssten.

Abbildung 14: Beispielhafte Trainingsdaten von Bildübersetzung

3.2.3 Training der Modelle

Nach dem Erstellen der Trainingsdaten kommt das Trainieren der Modelle. Die Funktion 'Train Deep Learning Model' übernimmt dies. Neben der Eingabe der Trainingsdaten und der Auswahl des Modell Typs, können noch viele weitere Parameter bestimmt werden. Je nach gewähltem Modell stehen modellspezifische Parameter, wie chip_size oder scales zur Verfügung, welche verändert werden können.
Auch ein Backbone Modell oder pre-trained Models können beigeführt werden, wobei ResNet34 hier das standard Backbone Modell ist. Ein weiterer Parameter, der eingestellt werden kann, ist die Anzahl Epochen, wobei man hier ein Maximum definieren kann. Mit Auswahl der Funktion 'Stop when model stops improving' wird ein frühzeitiger Trainingsabbruch erzwungen, wenn keine weitere Verbesserung des Modells stattfindet.
Eine Learning Rate des Modells kann manuell eingegeben werden. Wenn keine spezielle Learning Rate angegeben ist, wird während des Trainingsprozesses eine optimale Learning Rate generiert. In diesem Schritt wird auch bestimmt, wie viel Prozent der Daten für die Validierung verwendet werden. Des Weiteren kann die Batch Size definiert werden. Diese bestimmt wie viel Trainingsdaten gleichzeitig untersucht werden. Dieser Wert hat vor allem einen Einfluss auf die Laufzeit des Modelltrainings.

Während des Trainings wird mithilfe der Validierungsdaten bereits die Performance des Modells evaluiert. Je nach Modell wird hierbei die Accury oder der Average Precision Score ausgegeben. Auch wird der Verlauf der Verlustsfunktionen mit einem Plot ausgegeben.

3.2.4 Anwendung der Modelle

Die trainierten Modelle können auf neue Rasterbilder angewendet werden. Objektdetektionsmodelle verwenden die Funktion 'Detect Objects Using Deep Learning', wohingegen Bildübersetzungsmodelle die Funktion 'Classify Pixel Using Deep Learning' verwenden.

Die Modelle werden auf ein Rastersheet angewendet, für welches die Feuchtgebietpolygone vom Hilfsassistenten verbessert wurden. Dies soll einen späteren Vergleich mit möglichst wahren Werten ermöglichen.
Auch bei den Anwendungsfunktionen können bestimmte Parameter gewählt werden, wobei hier die Grundeinstellungen belassen werden.

3.2.5 Nachbearbeitung der Modellausgabe

Bei Bildübersetzungsmodellen ist eine Nachbearbeitung der Resultate notwendig, um spätere Genauigkeitsevaluationen durchführen zu können. Genauer gesagt müssen die generierten Raster Layer wieder zu Feature Layer umgewandelt werden, damit diese mit den Ground Truth layern der Feuchtigkeitsgebiete, welche als Feature Layer vorliegen, verglichen werden können.

Hierzu sind einige Schritte notwendig, welche in Abbildung 15 gezeigt werden. Die finale Umwandlungsfunktion 'Raster to Polygon' nimmt nur integer Werte an, wodurch Zwischenschritte notwendig sind. Durch 'Raster Calculator' werden die Werte des Rasters, welche zwischen 0 und 1 liegen, mit 1000 multipliziert. Dies ist notwendig für die nächste Funktion 'Int', welche die Rasterwerte auf die nächsttiefere Ganzzahl rundet. Ohne den vorherigen Schritt würden alle Werte auf 0 gerundet werden.
Da das Rasterbild in RGB Farben vorliegt, wird mit der Funktion 'Grayscale' dieses auf ein Band reduziert. Die nachfolgende Funktion 'Remap' weisst den Regionen ohne Feuchtgebiete den Wert NoData zu, wodurch im letzten Schritt der finalen Umwandlung nur Feuchtgebiete zu Polygonen konvertiert werden. Vereinzelt kann die Anwendung der Funktion 'Dissolve Boundaries' noch hilfreich sein, um kleine, sich überschneidende Polygone zusammenzuführen.

Diese Schritte sind bei Objektdetektionsmodellen nicht notwendig, da die Modellausgabe schon einen Feature Layer darstellt. Beim Modell Mask R-CNN ist ein Nachbearbeitungsschritt notwendig, welcher überlagernde Polygone zusammenfasst. Dies wird mit der Funktion 'Dissolve Boundaries' erreicht.

Abbildung 15: Workflow der Datennachbearbeitung bei Bildübersetzungsmodellen

3.2.6 Evaluierung der Modelle

Eine erste Evaluierung der Modelle findet während des Trainingsprozesses statt (vgl. Kapitel 3.2.3). Hierbei wird die Genauigkeit anhand der Modellauswertung auf den Validierungsdaten, welche 10% der Daten betragen, ermittelt. Dabei wird, je nach Modell, die Accuracy oder der Average Precision Score generiert.
Bei Bildübersetzungsmodellen werden hier nur die Verlustwerte (loss) ausgegeben.

Eine nächste Methode zur Genauigkeitsbewertung bietet die Funktion 'Compute Accuracy For Object Detection'. Diese vergleicht die Ergebnisse der Detektionsmodelle mit den wahren Objektpolygonen und ermittelt die True Positiv, False Positiv und False Negativ Werte. Hierbei werden jeweils die Bounding Boxes verglichen, wodurch die Funktion selbst noch Bounding Boxes um die Ground Truth Objekte für den Vergleich bildet.

Ein Parameter, welcher eingestellt werden kann, ist 'Intersection over Union (IoU)'. Dieser gibt die minimale Überschneidung der Bounding Box mit den Ground Truth Werten an, damit die Objekte als gefunden bezeichet werden.

Verwendete Vergleichsgrössen

Da für die Auswertung der unterschiedlichen Modelle statistische Vergleichsgrössen genutzt werden, stellt folgendes Kapitel die Grössen kurz vor. Der Fokus liegt auf der Verwendung der statistischen Grundlagen bei Modellevaluationen.
Die Tabelle 2 zeigt die Definitionen der Grundgrössen für statistische Berechnungen. Diese werden bei den Genauigkeitsevaluationen ermittelt und bieten die Basis für die Bildung von Vergleichsgrössen.

Tabelle 2: Grundgrössen für die Modellevaluation

	Tatsächlich ein Feuchtgebiet	Tatsächlich KEIN Feuchtgebiet
Modell bestimmt ein Feuchtgebiet	True Positiv	False Positiv
Modell bestimmt KEIN Feuchtgebiet	False Negativ	True Negative

Die Accuracy zeigt, wie oft man etwas richtig bestimmt hat. Also wie oft man ein Objekt richtig identifiziert hat und bei Orten ohne ein Objekt, dieses auch nicht erkannt hat. Accuracy wird stark davon beeinflusst, wenn man viele True Negatives hat, also wenn man in Regionen mit keinen Objekten auch keine detektiert hat. (Esri (2022c))

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + TrueNegative + FalsePositive + FalseNegative} \tag{1}$$

Der Recall zeigt, wie viel der Objekte erkannt wurden. Man möchten diesen Wert optimieren, um sicherzugehen, dass keine Objekte nicht detektiert werden. (Esri (2022c))

$$Recall = \frac{TruePositive}{TruePositive + FalseNegativ} \tag{2}$$

Die Precision gibt an, wie gut ein Modell Objekte richtig erkennen kann. Es wird gezeigt wie viel der als richtig deklarierten Objekte tatsächlich richtig sind, also das Verhältnis der Anzahlt von True Positives zur Gesamtanzahl der Vorhersagen.
Zusätzlich wird bei Analysen der Average Precisions Score angegeben. Dabei handelt es sich um die durchschnittliche Precision über alle Recall Werte zwischen 0 und 1. (Esri (2022c))

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{3}$$

F1 wird verwendet, um eine Balance zwischen Precision und Recall zu erhalten. Im Vergleich zur Accuracy liegt der Fokus auf False Negativ und False Positive, was für eine Modellevaluation bedeutende Grössen sind. (Esri (2022c))

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

Abschliessend kann gesagt werden, dass vor allem der Recall für diese Aufgabenstellung wichtig ist. Dieser zeigt, ob man Feuchtgebiete nicht erkannt hat. Die anderen Vergleichsgrössen sind jedoch auch zu betrachten.

3.3 Konkret angewandter Workflow

Generell widerspiegelt der konkret angewandte Workflow dem vollzogenen Lernprozess. Dies bedeutet, dass Modelle oder Modelleinstellungen mit guten Ergebnissen weiter untersucht wurden.

3.3.1 Vergleich aller Objektdetektionsmodelle

Training mit 1'600 Trainingsdaten

Für ein erstes Kennenlernen der Objektdetektionsmodelle werden Trainingsdaten auf der Basis von zwei Rastersheets erstellt, bei welchen die verbesserten Feuchtgebietpolygone zur Verfügung stehen. Das dritte Rastersheet wird bewusst für Evaluationen zurückgehalten. Hierbei entstehen circa 1'600 Rasterbilder der Grösse 256 x 256, welche mehr als 2'200 Objekte beinhalten. Als Backbone Modell wird die Standardeinstellung von ArcGIS Pro, ResNet34, belassen. Auch bei weiteren Parametern und Einstellungen wird der vorgeschlagene Wert verwendet, wobei maximal 100 Epochen gewählt werden. Die Trainingsfunktion generiert automatisch die Accuracy, beziehungsweise den Average Precision Score als ersten Evaluierunsgwert der Modelle.
Die trainierten Objektdetektionsmodelle werden anschliessend auf einem weiteren Rasterbild angewendet und mittels der Funktion 'Compute Accuracy For Deep Learning' evaluiert. Die dabei erhaltenen Werte werden mit einem 'Minimum Intersect' von 0.5 berechnet, was bedeutet, dass wenn ein gefundenes Objekt zu mindestens 50% mit einem wirklichen Objekt übereinstimmt, es als richtig bestimmt gilt.

Einfluss der Anzahl Trainingsdaten

Bei den ersten Anwendungs- und Evaluationsversuchen der Modelle auf einem weiteren Rastersheet war ersichtlich, dass die Aussagekräftigkeit der zur Verfügung stehenden Funktion nicht sehr gross ist. Aus diesem Grund wird auch das dritte Rasterbild mit den bearbeiteten Feuchtgebieten für Trainingszwecke verwendet, um die Anzahl Trainingsdaten zu erhöhen. Daraus resultieren 2780 Rastertiles der Grösse 256 x 256 Pixel mit 4180 erkannten Objekten, womit sich diese fast verdoppelt haben.
Dies ermöglicht es den Einfluss der Trainingsdatenmenge auf die Modelle zu beobachten.

Einfluss der Backbone Modell Wahl

In einem gleichen Schritt wird der Einfluss von unterschiedliche Backbone Modelle analysiert. Es werden hierbei die 2780 Trainingsdaten für das Modelltraining verwendet.
Wie in Kapitel 2.1.1 erwähnt kann bei Deep Learning Modellen ein vortrainiertes Backbone Modell miteinbezogen werden. Um die Auswirkungen dieser Wahl auf die Genauigkeit der Modelle zu betrachten, werden die Objektdetektionsmodelle mit unterschiedlichen Backbone Modellen trainiert. Bei ResNet43 handelt es sich um ein residuales Netzwerk, welches auf einem Bildset mit über einer Millionen Bildern trainiert worden ist. Wobei dies mit 34

layern Tiefe geschieht.
Weitere verwendete Backbone Modelle sind ResNet50, ResNet101 und ResNet152, wobei die Zahl jeweils die Anzahl Layer angibt. Es ist zu beachten, dass beim Modell YOLOv3 nur DarkNet53 als Backbone Modell zur Verfügung steht und hier keine weiteren Untersuchungen unternommen werden. Bei DarkNet53 handelt es sich um ein Convolutional Neural Network, welches mit mehr als einer Millionen Bildern und 53 layern trainiert wurde.
Des Weiteren gibt es noch andere Backbone Modellarten wie DenseNet, diese können jedoch in ArcGIS Pro nicht ausgewählt werden. Nur beim Modell Single Shot Detector lässt das Interface die Wahl anderer Backbone Modellarten zu. Wegen mangelhaftem Vergleich werden hierzu keine Untersuchungen unternommen.
Bei diesen Modellen findet nur eine Bewertung der Genauigkeit während des Trainingsprozesses mit den Validierungsdaten statt.

3.3.2 Fokus Mask R-CNN

Für genauere Untersuchungen wird das Mask R-CNN Modell ausgewählt, welches in den vorgegangenen Untersuchungen gute Ergebnisse aufzeigt. Für das Training werden neue Trainingsdaten erstellt, welche aus 11'359 Raster Tiles mit 19'179 erkannten Objekten bestehen. Diese beinhalten zwei der drei manuell nachbearbeiteten Feuchtgebietsdaten. Das Dritte wird für die spätere Evaluierung zurückbehalten und nicht in den Trainingsprozess integriert.

Tabelle 3 zeigt die unternommenen Modelleinstellungen. Es wird ein Basismodell mit den Standardeinstellungen von ArcGIS Pro generiert, um die Wirksamkeit der Parameteränderungen zeigen zu können.
Bei allen Modellvarianten wird das Training mit einer maximalen Epoche von 200 durchgeführt, wobei das Training stets vorher schon beendet wurde. Dies passierte, da keine weiteren Verbesserungen mehr stattfinden. Weiter werden alle Varianten mit einer Batch Grösse von 16, und einer automatisch optimierten Learning Rate generiert.

Tabelle 3: Unterschiedliche Mask R-CNN Modelleinstellungen

Basis Modell	Bei dem als Basis definiertem Modell handelt es sich um ein Mask R-CNN Modell, welches mit den Standarteinstellungen von ArcGIS Pro trainiert wird. Somit ist das Backbone Modell ResNet34.
ResNet152	Da bei vorherigen Tests das Backbone Modell ResNet152 die höchste Genauigkeit aufgewiesen hat, wird dies nun auch mit mehr Trainingsdaten getestet.
Pointred	Dieses Modell basiert auf dem Basis Modell, jedoch wird das punktbasierte Netzwerk 'Pointrend' integriert (vgl. Kapitel 2.3). Als Backbone wurde ResNet152 gewählt.
Unfreeze Backbone Model	Während bei den Standarteinstellungen die Backbone layer und die vortrainierten Gewichte nicht während dem Trainingsprozess angepasst werden, ändert dies die Einstellung 'Unfreeze Backbone Model'. Bei diesem Modell werden die Backbone Modellparameter den Traingsdaten angepasst, wobei ResNet152 als Backbone Modell dient.
Mehr Trainingsdaten	Um den Einfluss von mehr Trainingsdaten zu betrachten, wird ein Modell mit mehr Traingsdaten trainiert. Hierzu werden 4000 zusätzliche Trainingsbilder generiert, was total zu 15'362 Rasterbildern mit 25'310 detektierten Feuchtgebieten führt. Auch hier wird ResNet152 als Backbone Modell gewählt.

Ein Vergleich der unterschiedlichen Modelleinstellungen findet mittels Auswertungen während des Trainingsprozesses statt, wobei der Average Precision Score generiert wird. Des Weiteren erfolgt eine Anwendung der Modelle auf einem Rasterbild. Hierbei werden die gefundenen Objekte mit dem Ground Truth verglichen, was durch die Funktion 'Compute Accuracy For Deep Learning' passiert. Es wird ein 'Minimum Intersect' von 0.5 für die Auswertungen gewählt.

3.3.3 Bildübersetzung

Die Trainingsdaten der Bildübersetzung basieren auf drei Rasterbildern, bei welchen die manuell bearbeiteten Feuchtgebiete zur Verfügung stehen. Dabei entstehen 21'085 Rastertiles der Grösse 256 x 256 Pixel. Wie im Kapitel 2.3.2 begründet, wird nur das Bildübersetzungsmodell Pix2Pix ausprobiert.

Das Modell wird zuerst auf einem Rasterbild der Grösse 256 x 256 angewendet und später auf einem Rasterausschnitt von 1000 x 1000 Pixel. Genauigkeitsvergleiche während des Trainings finden nicht statt, da ArcGIS Pro hier nicht automatisch Werte generiert. Es findet eine Einschränkung auf visuelle Analysen statt.

4 Resultate

4.1 Vergleich aller Objektdetektionsmodelle

4.1.1 Training mit 1'600 Trainingsdaten

Tabelle 4 zeigt erste Genauigkeitsvergleiche der Objektdetektionsmodellen, welche auf einem Training mit circa 1'600 Trainingsdaten beruhen und während des Trainings generiert wurden.

Es ist ersichtlich, dass YOLOv3 die beste Genauigkeit liefert. Auch Faster R-CNN und Mask R-CNN scheinen mit 47% respektive 40% gute Ergebnisse zu generieren. Hingegen schneidet der Single Shot Detektor mit nur 23% am schlechtesten ab. Hierbei muss jedoch beachtet werden, dass ein direkter Vergleich von Accuracy und dem Average Precision Score schwierig ist.

Tabelle 4: Genauigkeiten während des Trainingsprozesses mit 1'600 Trainingsdaten

	Accuracy	Average Precision Score
RetinaNet	0.38	-
YOLOv3	0.65	-
SSD	-	0.23
Faster R-CNN	-	0.47
Mask R-CNN	-	0.39

Bei der Anwendung auf ein neues Rasterbild der Objektdetektionsmodelle ergeben sich die Ergebnisse der Tabelle 5. TP, FP und FN stehen für True Positiv, False Positiv und False Negative, wobei genauere Erklärungen dem Kapitel 3.2.6 zu entnehmen sind.

Verglichen mit den aus dem Training generierten Werten, sind diese meist deutlich tiefer. Besonders beim SingleShotDetektor ist dies der Fall, wobei hier viele Objekte fälschlicherweise als Feuchtgebiet klassifiziert worden sind, was sich mittels der False Positiv Werte zeigt.
YOLOv3, welches während des Trainings am besten abgeschnitten hat, erreicht hier keine Genauigkeitswerte über 20%. Am besten schneidet bei dieser Vergleichsmethode Mask R-CNN ab, wobei dieses Modell deutlich weniger Fals Positiv Werte aufweist als die anderen Modelle.

Tabelle 5: Genauigkeiten bei der Anwendung der Objektdetektionsmodelle

	Precision	Recall	F1 Score	TP	FP	FN
RetinaNet	0.28	0.30	0.29	60	151	142
YOLOv3	0.16	0.15	0.16	31	157	171
SSD	0.01	0.06	0.02	13	1050	189
Faster R-CNN	0.18	0.31	0.23	63	288	139
Mask R-CNN	0.59	0.45	0.44	68	47	132

4.1.2 Einfluss der Anzahl Trainingsdaten

Der Vergleich der Modellperformance bei unterschiedlicher Anzahl Trainingsdaten wird in der Abbildung 16 gezeigt, wobei hier ResNet34 als Backbone Modell dient. Die Daten über

die Genauigkeit mit 1'600 Trainingsdaten wurden aus dem Kapitel 4.1.1 übernommen. Bis auf YOLOv3 erzielt das Training mit mehr Trainingsdaten bessere Ergebnisse. Besonders bei RetinaNet gibt es eine deutliche Steigerung der Accuracy. Bei Mask R-CNN hingegen gibt es nur eine kleine Verbesserung.

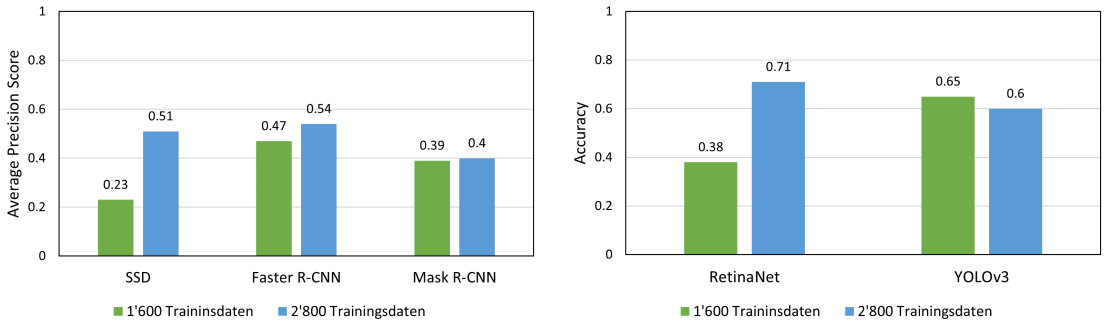

Abbildung 16: Vergleich der Modelle mit unterschiedlicher Anzahl Trainingsdaten

4.1.3 Einfluss der Backbone Modell Wahl

Der Einfluss von unterschiedlichen Backbone Modellen wird in der Abbildung 17 sowie in Abbildung 18 ersichtlich, wobei hier die Genauigkeiten während des Trainingsprozesses gezeigt werden.

Besonders beim Modell Mark R-CNN fällt ein grosser Unterschied zwischen ResNet34 und ResNet50 auf. Während mit dem Standardmodell, ResNet34, nur ein Average Precision Score von 40% erreicht wird, führt die Integration von ResNet50 zu Werten von 89%, was mehr als einer Verdopplung entspricht. Mit zusätzlicher Tiefe des Backbone Modells kann mit Mask R-CNN ein maximaler Average Precision Score von 92% während des Trainings erreicht werden.

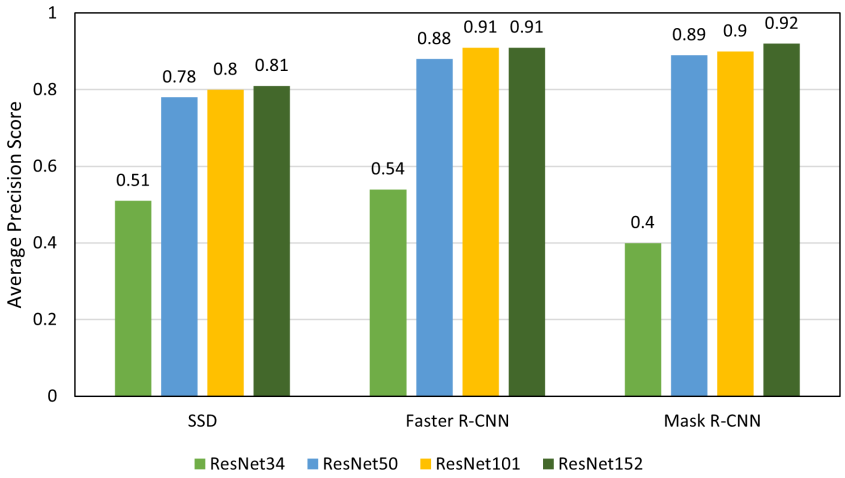

Abbildung 17: Precision der Objektdetektionsmodelle mit verschiedenen Backbone Modellen

In der Abbildung 18 werden die zwei weiteren Objektdetektionsmodelle RetinaNet und YOLOv3 abgebildet. Hierbei wird die Accuracy dargestellt. Die Anwendung der unterschiedlichen Backbone Modelle bei RetinaNet verhält sich ähnlich wie bei den Vergleichen in Abbildung 17. Nur bei ResNet101 gibt es eine Abweichung, und die Accuracy sinkt.
YOLOv3, bei welchem nur DarkNet53 als Backbone zur Verfügung steht, erreicht eine Accuracy von 60%.
ResNet152 erreicht bei allen Modellen die höchsten Genauigkeiten. Generell erfolgt zwischen ResNet34 und ResNet50 bei den meisten Modellen die grösste Steigerung der Genauigkeit, wobei RetinaNet hier eine Ausnahme zu sein scheint.

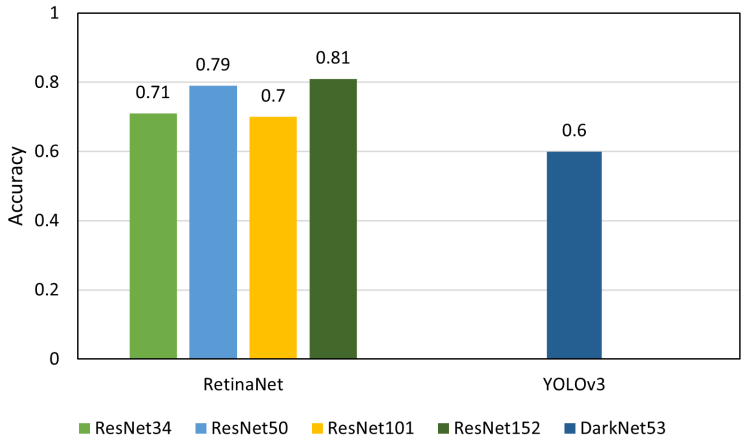

Abbildung 18: Accuracy von YOLOv3 und RetinaNet bei verschiedenen Backbone Modellen

Visuelle Vergleiche sind mittels der Abbildung 19 möglich. Hierbei sind die Modellausgaben, mit ResNet152 als Backbone, bei Anwendung auf einem kleinen Rasterbild ersichtlich. Der Ground Truth stellt hierbei die wahren Feuchtgebietsgrenzen dar.
Auffallend ist, dass RetinaNet die Objekte nicht richtig detektiert. Lediglich ein kleines Objekt in der linken Ecke wird gefunden. Der SingleShotDetektor, Faster R-CNN und Mask R-CNN detektieren aus visueller Sicht die Feuchtgebiete. Ersichtlich ist zusätzlich, wie sich die generelle Modellausgabe von Mask R-CNN, von den Ausgaben der restlichen Objektdetektionsmodellen unterscheidet. Es wird die wahre Form der Objekte gefunden und nicht nur Bounding Boxes erstellt.

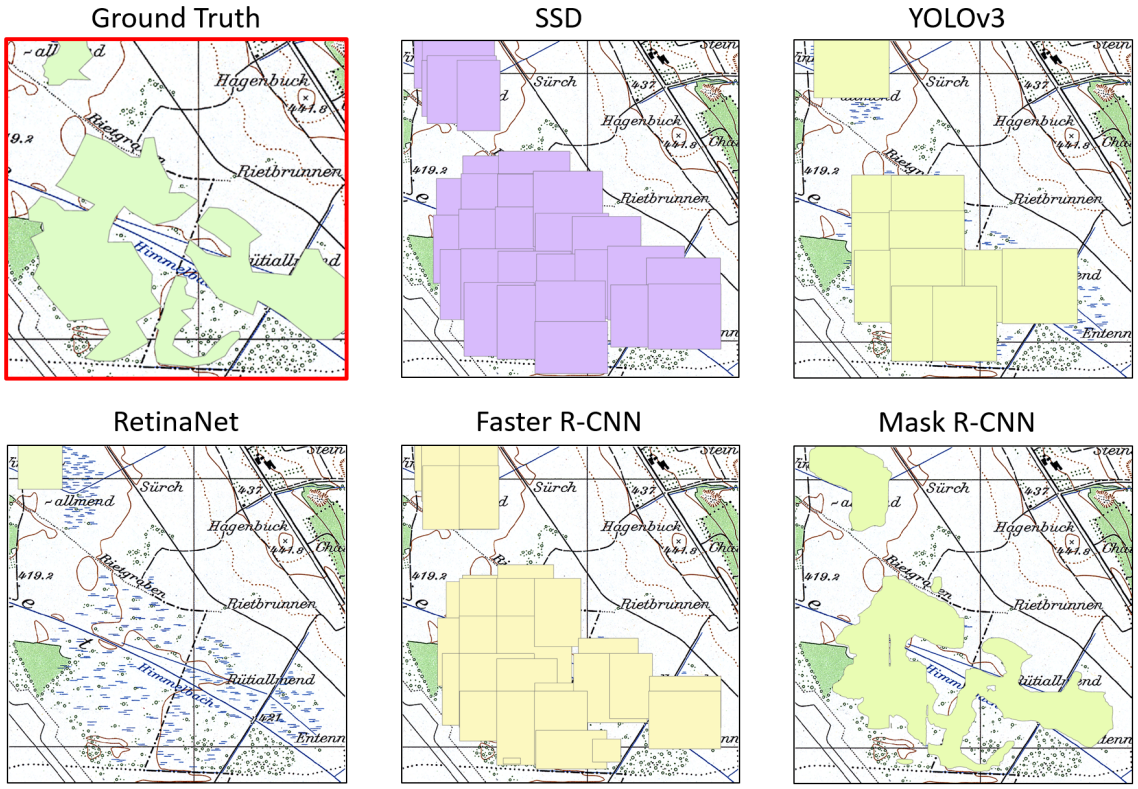

Abbildung 19: Objektdetektionsmodelle im visuellen Vergleich

4.2 Fokus Mask R-CNN

Die erhaltenen Genauigkeiten während des Trainingsprozesses der Modelle werden in der Tabelle 6 gezeigt, wobei hier stets der Average Precision Score angegeben wird. Ersichtlich ist, dass besonders das Default Modell mit 29% Precision schlecht abschneidet. Der Einbezug des Netzwerkes 'Pointrend' generiert bei den Modelleinstellungen die besten Genauigkeiten mit 86%. Auch mit 'unfreeze Backbone Model' kann im Vergleich zu ResNet152 eine Genauigkeitssteigerung erzielt werden.

Tabelle 6: Genauigkeiten der Mask R-CNN Variationen während des Trainingsprozesses

	Average Precision Score
Default Modell	0.29
ResNet152	0.57
Pointred	0.86
Unfreeze Backbone Model	0.65
Mehr Trainingsdaten	0.49

Tabelle 7 basiert auf der Anwendung der Modelle auf ein Rasterblatt. Das Default Modell erreicht mit 87% die höchste Precision aller Modelleinstellungen. Beim Recall hingegen schneidet 'Unfreeze Backbone Modell' am besten ab. Pointrend, welches bei Auswertungen während des Trainingsprozesses die besten Resultate lieferte, erreicht im Vergleich zu den anderen Modelleinstellungen nur mässige Genauigkeiten. Es werden mit Pointrend überdurchschnittlich viel False Positiv Objekte erfasst.

Mit mehr Trainingsdaten werden Objekte teilweise nicht erkannt, was sich in einem hohen False Negative Wert widerspiegelt. Dies führt weiter zu einem geringen Recall Wert von 42%.

Tabelle 7: Genauigkeiten Mask R-CNN Modellvariationen bei Anwendung

	Precision	Recall	F1 Score	TP	FP	FN
Default Modell	0.87	0.68	0.76	137	21	65
ResNet152	0.79	0.60	0.68	121	33	81
Pointred	0.58	0.64	0.61	130	95	72
Unfreeze Backbone Model	0.78	0.74	0.76	149	41	53
Mehr Trainingsdaten	0.71	0.42	0.53	85	34	117

Ein visueller Vergleich der angewendeten Mask R-CNN Modellvariationen wird in Abbildung 20 gezeigt. Besonders bei der Variante mit mehr Trainingsdaten ist klar ersichtlich, dass vereinzelte Feuchtgebiete vom Modell nicht erfasst werden. Dies ist vor allem im unteren Teil des Rasterbildes der Fall. Die restlichen Modelleinstellungen scheinen die Objekte korrekt zu erfassen. Es gibt Unterschiede, wie nahe die Grenze der Objekte an den Symbolen der Feuchtgebiete sind. Bei Pointrends und ResNet152 werden die Objekte mit sehr genauen Grenzen belegt, wohingegen bei den anderen Modellen die Objekte grosszügiger detektiert werden.

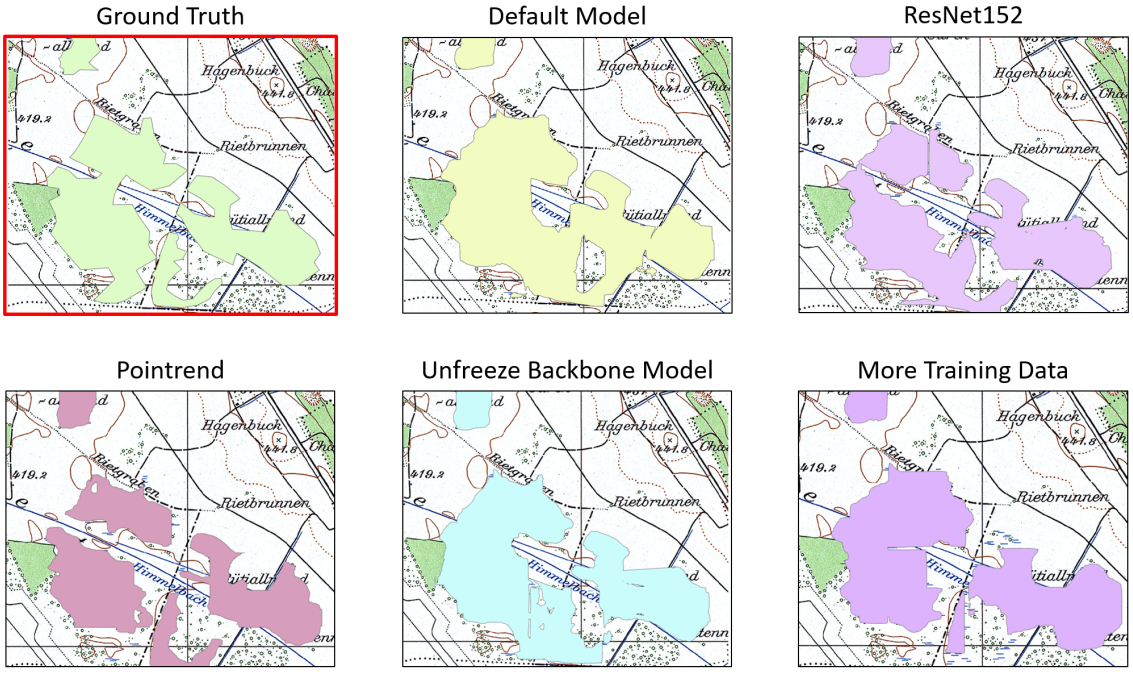

Abbildung 20: Visueller Vergleich der Modelleinstellungen

4.3 Bildübersetzung

Abbildung 21 zeigt die Anwendung des Modells Pix2Pix auf ein Bild der Grösse 256 x 256 Pixel. Es ist zu erkennen, dass die Feuchtgebiete richtig detektiert wurden, also dass bei der Modellausgabe die Objekte schwarz sind. In der rechten oberen Ecke sind leichte Fehler ersichtlich, bei welchen das Modell kleine schwarze Pixel ausgibt, ohne dass sich dort ein Feuchtgebiet befindet.

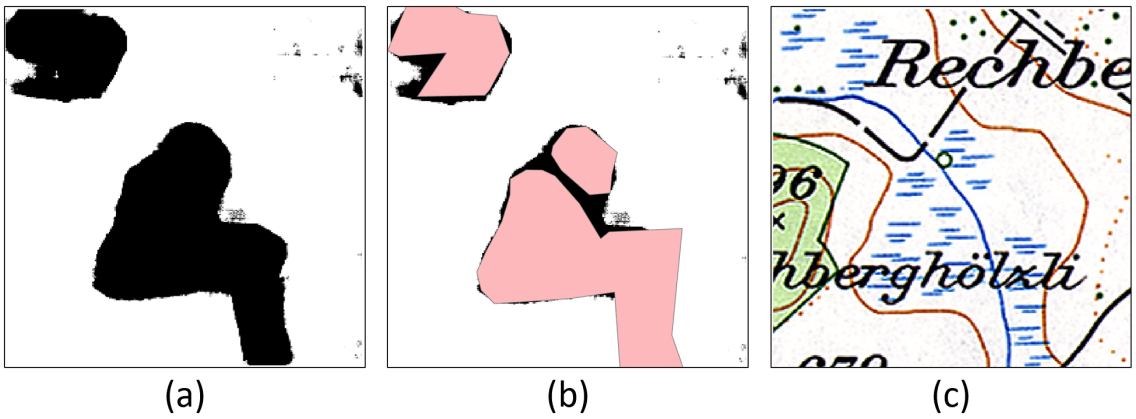

Abbildung 21: Resultate Pix2Pix auf Rasterbild der Grösse 256x256 Pixel
(a): Modellausgabe; (b): Modellausgabe mit Ground Truth Polygonen überlagert; (c): Modelleingabe

Bei einem Anwendungsversuch auf ein grösseres Rasterbild wurde die Ausgabe in Abbildung 22 (a) generiert. Ersichtlich ist, dass hier das Modell nicht zu funktionieren scheint. Die Feuchtgebiete lassen sich mit genauer Betrachtung erahnen, wobei schwarze zusammenhängende Objekte erkennbar sind. In Regionen ohne Objekte gibt das Modell jedoch auch schwarze Rasterpixel zurück.

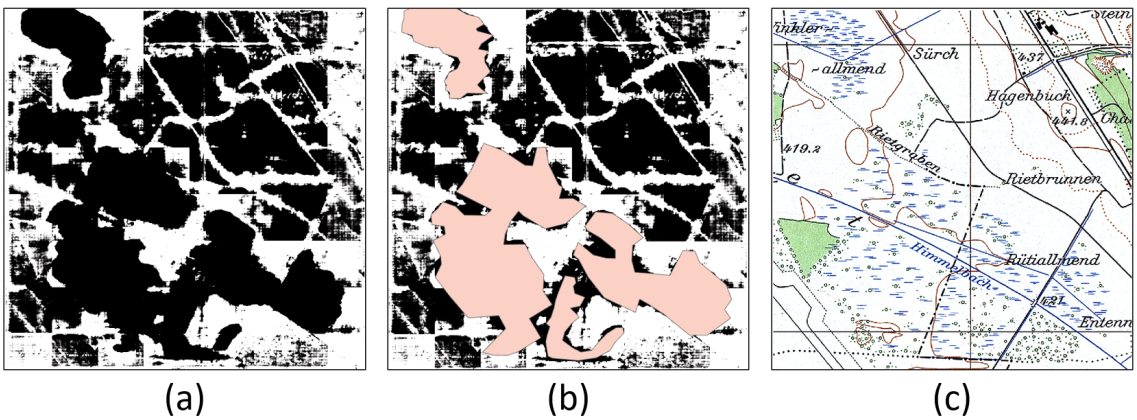

Abbildung 22: Resultate Pix2Pix auf Rasterbild der Grösse 1000x1000 Pixel
(a): Modellausgabe; (b): Modellausgabe mit Ground Truth Polygonen überlagert; (c): Modelleingabe

5 Diskussion

5.1 Generelle Probleme und Analysen

Als grundsätzliches Problem kann die Dokumentation der Deep Learning Modelle betrachtet werden. Der ArcGIS Developer bietet grundlegende Hilfestellungen und Erklärungen zu den Modellen, wobei Detailfragen hier nicht geklärt werden. Das Esri Community Portal, welches Fragen der User beantwortet, bietet eine Plattform für spezifische Errors oder genauere Fragen. Auch hier ist jedoch zu erkennen, dass nur eine kleine Zahl der User mit den Funktionalitäten vertraut sind, und das Portal nur selten genutzt wird.

Nur vereinzelt fanden wissenschaftliche Studien mit den Deep Learning Funktionalitäten in ArcGIS Pro statt. Dies signalisiert, dass die Deep Learning Funktionalität mehr auf einfache Anwendungen ausgerichtet sind. Der Forschungsaspekt steht weniger im Vordergrund. Die Zielgruppe sind daher Personen ohne grosse Vorkenntnisse.

Funktionen in ArcGIS Pro stellen generell eine Black Box dar. Die einzelnen Schritte innerhalb, oder wie etwas verarbeitet wird, ist nicht klar. Aufgrund dieser Eigenschaft ist es oft schwierig ein Problem zu verstehen und zu lösen. Dies ist besonders bei der Funktion 'Compute Accuracy For Object Detection' aufgefallen, worauf im Kapitel 5.4 näher eingegangen wird.

5.2 Diskussion der Modellergebnisse

5.2.1 Objektdetektion

Training mit 1'600 Trainingsdaten

Die ersten Trainingsversuche der Modelle zeigen, dass keine überragenden Genauigkeiten erreicht werden. YOLOv3 erreicht hierbei eine maximale Accuracy von 65% und schneidet am besten ab. Dieser Wert unterliegt noch starkem Verbesserungspotential.
Weiter haben die Resultate gezeigt, dass eine Auswertung mittels der Funktion 'Compute Accuracy For Deep Learning' fragwürdig ist. Hierbei liegen die generierten Werte deutlich unter den Genauigkeiten während des Trainingsprozesses. Nur bei Mask R-CNN konnten realistische Werte generiert werden. Dies kann darauf hinweisen, dass es ein Problem mit den Bounding Boxen der Modellausgaben gibt, denn Mask R-CNN verfügt nicht über Bounding Boxes. Auf mögliche Ursachen und Verbesserungsversuche dieser Auswertungsfunktion wird im Kapitel 5.4 genauer eingegangen.

Einfluss der Anzahl Trainingsdaten

Der Vergleich der Modelle mit einer unterschiedlichen Anzahl Trainingsdaten zeigt, dass der Einfluss stark modellabhängig ist. Von starken bis zu leichten Verbesserungen, sowie geringen Verschlechterungen, durch mehr Trainingsdaten ist alles zu beobachten.
Da die grössten Verbesserungen beim SingleShotDetektor und bei RetinaNet erreicht werden, könnte ein Zusammenhang mit dem generellen Modelltyp bestehen. Bei diesen Modellen handelt es sich um one-stage Modelle. Also die Lokalisierung und die Klassifizierung findet nur mittels eines Netzwerks statt. Dies könnte eine starke Abhängigkeit von der Anzahl Trainingsdaten generieren. Es kann jedoch argumentiert werden, dass diese Modelle dadurch wenig zusätzliche Daten benötigen um bessere Ergebnisse zu liefern.
Die Abnahme der Genauigkeit mit zusätzlichen Trainingsdaten von YOLOv3 ist schwierig

begründbar und könnte auf modellspezifische Eigenschaften zurückzuführen sein.

Two-stage Modelle, wie Faster R-CNN und Mask R-CNN, scheinen resistenter gegen die Anzahl Trainingsdaten zu sein. Dies könnte mit einer komplexeren Netzwerkstruktur zusammenhängen, welche Veränderungen in der Trainingsdatenanzahl weniger stark beeinflusst. Dies bedeutet jedoch auch, dass für Genauigkeitssteigerungen sehr viele zusätzliche Trainingsdaten benötigt werden.

Haldar (2015) betont in einem Bericht aus dem Jahr 2015 die Wichtigkeit der Trainingsdaten für die Leistung der Modelle. Als Faktoren, welche eine Rolle für die Anzahl notwendiger Trainingsdaten spielen, identifiziert er: die generelle Aufgabenstellung, die Genauigkeit, welche man erreichen möchte, die Ungenauigkeit der Trainingsdaten sowie die Komplexität der Modelle, wobei noch andere Einflussfaktoren existieren. Weiter definiert er 'the rule of 10', welche besagt, dass man zehn mal so viel Trainingsdaten benötigt, wie das Modell Parameter hat. Dabei sind die Parameter in den eingebetteten Layern des Modells gemeint. Diese Erkenntnisse unterstreichen die These, dass für komplexere two-stage Modelle mehr Trainingsdaten notwendig sind.

Einfluss der Backbone Modell Wahl

Die Resultate zeigen, dass zu Beginn zusätzliche Backbone Layer eine deutliche Genauigkeitssteigerung mit sich bringen. Die zusätzliche gewonnene Genauigkeit pro Layer nimmt jedoch ab. Ab einer gewissen Tiefe scheinen zusätzliche Layer nur noch minimale Verbesserungen zu erreichen.

Ein Punkt der hier berücksichtigt werden muss, ist die Laufzeit des Trainingsprozesses. Je tiefer die Backbone Netzwerke werden, desto länger dauerte das Training. Beispielsweise benötigte ein Training mit ResNet152 jeweils mehr als 24 Stunden, wohingegen Modelle mit ResNet34 in nur zwei Stunden trainiert wurden. Es muss hier ein Abwägen zwischen Laufzeit und Genauigkeit stattfinden, was je nach Aufgabenstellung anders bewertet werden kann.
ArcGIS Pro wählt ResNet34 als Standardmodell und setzt somit mehr auf die Schnelligkeit des Trainings. Die Resultate zeigen hingegen, dass die Wahl von ResNet50 einen guten Kompromiss darstellt. Dies vor allem aufgrund der erhöhten Genauigkeit im Vergleich zu ResNet34. Des Weiteren werden mit noch tieferen Backbone Modellen wie ResNet50 nur noch leichte Verbesserungen erzielt, wobei die Laufzeit deutlich ansteigt, was diese unattraktiv macht.

Hennig (2021), welche sich mit der Baumdetektion mittels ArcGIS Pro beschäftigt, kommt auch zum Schluss, dass ResNet50 eine gute Balance zwischen Genauigkeit, Laufzeit und auch den verfügbaren Hardware Ressourcen ist. Dabei stützt sie sich neben eigenen Forschungsergebnissen auch auf weiteren Literaturrecherchen.

Rongyu Zhang et al. (2020) unterstreicht die gute Performance von ResNet Modellen bei der Extraktion von kleinen Objekten, wobei das Backbone Modell Xception, bei seinen Untersuchungen, über alle Aufgaben am besten abschnitt. Dies zeigt, dass andere Backbone Modelle noch weitere Verbesserungen erzielen könnten. Hierzu müssten diese erst von ArcGIS Pro zur Verfügung gestellt werden.

Fokus Mask R-CNN

Die Resultate des Fokus auf Mask R-CNN unterstreichen die Wichtigkeit der Qualität der Trainingsdaten. Auch wenn bei diesen Modelleinstellungen viermal so viel Trainingsdaten wie beim Backbone Vergleich mit Mask R-CNN verwendet werden, erreichen die Modelle während des Trainings schlechtere Average Precision Scores. Während ResNet152 mit wenig Trainingsdaten einen Score von 92% erreicht, ist dieser mit viermal so viel Trainingsdaten nur bei 57%, was einer deutlichen Abnahme entspricht. Auch bei ResNet34 verschlechtert sich die Genauigkeit mit zusätzlichen Trainingsdaten um 11%.
Grund dafür ist, dass bei den zusätzlichen Trainingsdaten auch nicht von einer studentischen Hilfskraft bearbeitet Feuchtgebietsdaten verwendet werden. Es kann also davon ausgegangen werden, dass die durchschnittliche Genauigkeit der Trainingsdaten abgenommen hat, wodurch sich die damit trainierten Modelle verschlechtern.

Generell ist bei den Untersuchungen von Mask R-CNN zu erkennen, dass unterschiedliche Methoden zur Genauigkeitsanalyse verschiedene Ergebnisse liefern. Während die Einstellung mit Pointrend während des Trainings, mit einem Average Precision Score von 86%, die besten Genauigkeiten liefert, fällt das Ergebnis bei der Anwendung schlechter aus. Nur die Modellvariante mit nochmals mehr Trainingsdaten schneidet hier noch schlechter ab. Auch beim Default Modell sind grosse Unterschiede zwischen den Evaluierungsmethoden zu erkennen, wobei dieses bei der Anwendung besser abschneidet.

Um realistische Aussagen zur Genauigkeit der Modelle zu machen, müssen alle drei Evaluierungsmöglichkeiten berücksichtigt werden. Also die Werte während des Trainings, bei der Anwendung auf ein neues Rasterbild mit der Funktion 'Compute Accuracy For Deep Learning' sowie ein visueller Vergleich der Ergebnisse.

Zusätzlich sind die Genauigkeitswerte aus der Funktion 'Compute Accuracy For Deep Learning' stärker zu hinterfragen. Diese scheint zwar für Mask-RCNN realistischere Werte als für die anderen Objektdetektionsmodelle zu generieren, jedoch wird auch hier nicht klar, wie die Funktion genau vorgeht. Der Recall kann als aussagekräftigster Vergleichswert gesehen werden, da hier ermittelt wird, wie viele Feuchtgebiete detektiert wurden im Verhältnis zu allen existierenden Gebieten. Auch ist der Recall unabhängig von den False Positiv Werten.

Was jedoch über alle Genauigkeitsanalysen gesagt werden kann ist, dass nochmals mehr Trainingsdaten die Modelle verschlechtern, wenn diese von schlechter Qualität sind.
Auch zeigt der visuelle Vergleich, dass alle Modelle den Grossteil der Feuchtgebiete finden. Die Unterschiede finden sich beim Ziehen der Grenzen der Objekte. Pointrend und ResNet152 definieren hier die Grenzen sehr nahe an den Symbolen, wohingegen die anderen Modelleinstellungen die Grenzen etwas grober ziehen.

Um ein mögliches Optimum der Modelleinstellungen zu erreichen, kann eine Kombination der Modellvariationen vorgeschlagen werden. Pointrend und auch Unfreeze Backbone Modell erreichen gemäss Trainingsanalyse und visuellem Vergleich Verbesserungen gegenüber dem Standardmodell. Daraus resultiert eine Empfehlung Mask R-CNN mit dem Netzwerk Pointrend sowie der Funktion Unfreeze Backbone Modell zu trainieren. Bei der Wahl des Backbone Modells könnte über ResNet50 nachgedacht werden, da dieses aus vorherigen Analysen eine gute Balance zwischen Laufzeit und Genauigkeit bietet.

5.2.2 Bildübersetzung

Das angewandte Modell zur Bildübersetzung hat schon bei den ersten Schritten der Trainingsdatenerstellung Probleme bereitet. Da es sich bei der Datengrundlage um vektorisierte Feuchtgebiete handelt, musste eine Umwandlung in das benötigte Rasterformat stattfinden. Die Funktion zum Exportieren von Trainingsdaten verlangt präzise Eigenschaften, wodurch viele unterschiedliche Umwandlungen notwendig waren (vgl. Kapitel 3.2.1). Dies erschwert den Workflow, die Übersicht und die Nachvollziehbarkeit der vollzogenen Arbeitsschritte.
Auch benötigen Bildübersetzungsmodelle bei der Aufgabenstellung im Rahmen dieser Arbeit eine Nachbearbeitung des Modelloutputs.

Die Resultate haben gezeigt, dass Bildübersetzungsmodelle in kleinen Gebieten zwar funktionieren, für die Aufgabenstellung dieser Arbeit jedoch nicht geeignet sind. Das Modell war nur erfolgreich bei Rasterbildern, welche die gleiche Grösse wie die Trainingsdaten haben. Bei grösseren Rastern hat das Pix2Pix Modell versucht in Regionen ohne Feuchtgebiete, solche doch zu finden. Dieses Verhalten ist nicht erklärbar und kann wahrscheinlich mit besseren Parametereinstellungen oder mehr Trainingsdaten nicht geändert werden. Abgesehen davon, wurde für das Training des Pix2Pix Modells bereits alle drei manuell bearbeiteten Feuchtgebietsdaten verwendet, wodurch eine Erklärung durch eine schlechte Datengrundlage nicht plausibel ist.
Auch ist anzumerken, dass die Trainingsdaten zu einem grossen Teil nur weisse Rasterbiler beinhalten, da das binäre Rasterbild in kleine Ausschnitte zerschnitten wurde, was im Kapitel 3.2.2 genauer erklärt wird. Dies macht es umso schwerer nachzuvollziehen, wieso in der Modellausgabe so viele Flächen schwarz sind.

Es kann argumentiert werden, dass die Aufgabenstellung nicht für Bildübersetzungsmodelle geeignet ist.

5.3 Diskussion des allgemeinen Workflows

Generell sind die einzelnen Arbeitsschritte in ArcGIS Pro zur Anwendung von Deep Learning Modellen einfach und klar verständlich. Es treten jedoch Schwierigkeiten auf, sobald die Daten nicht im benötigten Format vorliegen. Dabei muss auf unterschiedlichste Zwischenschritte zurückgegriffen werden, welche selbst zu identifizieren sind.

5.4 Grenzen der Auswertung

Bei den Trainingsprozessen der Modelle werden automatisch Genauigkeitsangaben mit den Validierungsdaten erstellt, wobei bei manchen Modellen die Accuracy und bei anderen der Average Precision Score generiert wird. Es wurde keine Möglichkeit gefunden hier etwas zu verändern und zu vereinheitlichen, was für den Vergleich der Modelle nicht vorteilhaft ist. Weiter ist hier die Berechnung der Accuracy fragwürdig, bei welcher die True Negative Werte benötigt werden (vgl. Kapitel 3.2.6). True Negativ bedeutet, dass man kein Objekt detektiert hat und sich auch tatsächlich keines dort befindet. Es wird nicht klar, wie die Funktion hier Zahlenwerte generiert, denn auf einer Fläche zu bestimmen wie viele Objekte man richtig nicht erkannt hat, ist in absoluten Zahlen schwierig.

Die Resultate haben gezeigt, dass die Auswertung mittels der Funktion 'Compute Accuracy For Object Detection' nicht aussagekräftig ist. Das Problem hierbei ist, dass grosse

Feuchtgebiete mittels vielen kleinen Bounding Boxes gefunden werden, aber die Ground Truth Bounding Box des Feuchtgebiets deutlich grösser wäre. Die Funktion vergleicht die Überlappung dieser beiden Bounding Boxes, welche mittels einem gewählten IoU (Minimum Intersection over Union) 50% betragen sollte. Diese 50% werden mit den Grössenunterschieden nicht erreicht, wodurch viele False Positiv Werte beobachtet werden können. Dies, obwohl die gefundenen Bounding Boxes voll innerhalb des wahren Objekts liegen. Somit kann der Recall als jener Wert betrachtet werden, welcher die grösste Aussagekraft hat. Denn bei diesem fliessen die False Positiv Werte nicht in die Berechnung ein.
Die visuelle Betrachtung bestätigt die Nichtaussagekraft der Genauigkeitsangaben durch diese Funktion.

Es wurden Versuche unternommen um die Funktion 'Compute Accuracy For Deep Learning' aussagekräftig zu machen. Einerseits wurden die vielen kleinen, sich schneidenden, Bounding Boxen mittels der Funktion 'Dissolve Boundaries' zu grösseren Boxen konvertiert. Dies hat keine besseren Ergebnisse geliefert, wahrscheinlich aus dem Grund, dass die Funktion die Polygone dann nicht mehr als Bounding Boxen erkennt.
Weiter wurde die Funktion 'Non Maximum Supression' vor der Genauigkeitsfunktion angewandt. Diese erkennt sich überlappende Polygone und behält nur die mit der grössten Confidence, also der grössten Wahrscheinlichkeit, dass dies ein Feuchtgebiet ist. Dadurch bleiben zwar deutlich weniger Bounding Boxes, aber das Problem des Grössenunterschieds bleibt bestehen und die Aussagen der Funktion sind nicht genauer.
Um dem Modell zu lernen grössere Bounding Boxes zu generieren, wurden ein Versuch mit grösseren Trainingsrastern unternommen. Der Hintergedanke war, dass die Feuchtgebiete eventuell grösser als die vorhandenen Trainingsraster sind, und somit das Modell nicht lernt, wie das Objekt gesamthaft aussieht. Diese Versuche haben jedoch zu keinen besseren Ergebnissen geführt und die Bounding Boxes der Modellausgabe haben sich nicht verändert. Eventuell könnte ein tieferer IoU Wert helfen, wobei dies noch untersucht werden müsste.

Generell kann weiter gesagt werden, dass die Funktion 'Compute Accuracy For Deep Learning' einen Vergleich mit den Ground Truth Werten erstellt. Problematisch wird es hier, wenn die Genauigkeit dieser wahren Objekte unzureichend ist. Dieses potentielle Problem konnte bestmöglich vermieden werden, indem die Modelle nur auf Rasterbildern angewendet wurden, bei welchen bearbeitete Feuchtgebiete zur Verfügung standen.

Beobachtet wurde weiter, dass teilweise die visuellen Analysen von den generellen Zahlenwerten abweichen. Besonders auffallend war dies beim Modell RetinaNet. Hier wurde während des Trainings eine Genauigkeit von 81% mit ResNet152 als Backbone ermittelt. Bei der Anwendung dieses Modells wurden jedoch grosse Ungenauigkeiten beobachtet. Das Modell hat nur vereinzelt Feuchtgebiete erkannt.
Dies führt dazu, dass eine reine wertebasierte Genauigkeitsanalyse hinterfragt werden muss. Vielmehr braucht es eine Kombination aller Möglichkeiten zur Genauigkeitsüberprüfung.

5.5 Eignung der Objektklasse 'Feuchtgebiet' für Deep Learning Modelle

Im Laufe der Arbeit konnte festgestellt werden, dass die Objektgruppe Feuchtgebiete sich eher schlechter für die Detektion mittels Deep Learning eignet. Hierfür gibt es mehrere Gründe.
Feuchtgebiete werden auf der Landeskarte mittels eines zusammengesetzten Signaturtyps,

blauen Strichen, dargestellt. Die Abstände zwischen diesen Strichen variieren und weisen keine Regelmässigkeit auf. Weiter ist es schwierig klare Grenzen eines Feuchtgebiets zu ziehen, da diese nicht durch eine Linie umrandet werden. Selbst die manuelle Abgrenzung ist teilweise unklar. Als zusätzliche Schwierigkeit zeigt sich die Variabilität der Feuchtgebiete in den Formen, von länglichen, bis zu rund angeordneten Flächen. Jedes Objekt unterscheidet sich, was es für die Modelle schwierig macht Regelmässigkeiten zu lernen. Nicht nur die Form der Feuchtgebiete variiert, sondern auch die Grösse. Während manche Feuchtgebiete nur aus drei Strichen bestehen, können sich andere über ein ganzes Gebiet ausbreiten und Striche im dreistelligen Bereich beinhalten.

Eine Bachelorarbeit aus dem Jahr 2020 hat bereits die Schwierigkeit von Feuchtgebietsextraktion beleuchtet. Es ging in der Arbeit um Trainingsdatenerzeugung für Segmentierungsmodelle, wobei unterschiedliche Objektklassen betrachtet wurden. Auch Feuchtgebiete waren Teil der Untersuchung, wobei diese verglichen mit den anderen Klassen, wie Flüsse oder Seen, schlechter extrahierbar waren und einen maximalen F1-Score von 53% erreichten. Begründet wurde dies mit unklaren Grenzen eines Feuchtgebiets. (Bender (2020))

6 Zusammenfassung und Empfehlung

Zusammenfassend kann die Wichtigkeit der Datenqualität betont werden. Hier gilt der Leitsatz 'Qualität über Quantität'. Die Daten haben einen grossen Einfluss auf die Genauigkeit der Modelle. Auch Modelleinstellungen und Parameter können Verbesserungen respektive auch Verschlechterungen erzeugen, jedoch liegt grössere Wichtigkeit auf den Daten selbst.

Es hat sich herausgestellt, dass sich besonders das Objektdetektionsmodell Mask R-CNN für die Extraktion von Feuchtgebieten aus historischen Karten eignet. Der Vorteil gegenüber anderen Modellen ist, dass die Form der Feuchtgebiete richtig wiedergegeben wird. Auch kann die Integration des Netzwerks Pointrend und die Anpassung der Backbone Gewichten zu Genauigkeitssteigerungen führen. Bei der Backbone Wahl muss eine Abwägung zwischen Laufzeit und Genauigkeit stattfinden, wobei ResNet50 hier eine gute Balance bietet.

Von einer Verwendung von Bildübersetzungsmodellen für die Extraktion von Objekten kann abgeraten werden. Es werden zu viele Zwischenschritte benötigt und eine erfolgreiche Extraktion war nur auf bestimmten Rastergrössen möglich. Dieser Modelltyp eignet sich nicht für die Aufgabenstellung.

Bei der generellen Genauigkeitsanalyse der Modelle wird empfohlen vielseitige Ansätze zu wählen, um Aussagen zu überprüfen. Dies besonders, da nicht klar ist wie ArcGIS Pro die Werte generiert.

7 Ausblick

Um ein finales Modell für die Extraktion von Feuchtgebieten zu erstellen, müssen noch Untersuchungen zu den Kombinationen von Modelleinstellungen getätigt werden. Weiter können noch Feinabstimmungen, wie eine Anpassung des paddings oder der thresholds, bei der Modellanwendung zu verbesserten Ergebnissen führen.

Um die Datenqualität zu erhöhen, müssten noch weiter Feuchtigkeitsgebiete manuell nachbearbeitet werden. Dadurch würde sich die Anzahl guter Trainingsdaten erhöhen, was Genauigkeitssteigerungen verspricht.

Weitergehend kann untersucht werden, ob die Objektdetektionsmodelle so trainiert werden können, sodass diese grössere Bounding Boxen generieren. Dadurch wären potentiell Genauigkeitsvergleiche mit der Funktion 'Compute Accuracy For Deep Learning' möglich.

Insgesamt gibt es noch viele weitere Möglichkeiten Deep Learning Modelle in ArcGIS Pro für die Extraktion von Feuchtgebieten zu untersuchen.

Literaturverzeichnis

Ayoosh, K. (2018). *What's new in YOLO v3? .* Zugriff auf https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b

Bender, J. (2020). *Samplingstrategien zur Trainingsdatenerzeugung für tiefe Segmentierungsmodelle - Bachelor-Arbeit* (Bericht).

Chiang, Y. Y., Leyk, S. & Knoblock, C. A. (2014). *A survey of digital map processing techniques* (Bd. 47) (Nr. 1). Association for Computing Machinery. doi: 10.1145/2557423

Esri. (2022a). *Deep Learning in der Erweiterung ArcGIS Image Analyst—ArcGIS Pro | Dokumentation.* Zugriff auf https://pro.arcgis.com/de/pro-app/latest/help/analysis/image-analyst/deep-learning-in-arcgis-pro.htm

Esri. (2022b). *Faster R-CNN Object Detector | ArcGIS Developer.* Zugriff auf https://developers.arcgis.com/python/guide/faster-rcnn-object-detector/?rsource=https%3A%2F%2Flinks.esri.com%2FDevHelp_HowFasterRCNNWorks

Esri. (2022c). *Funktionsweise des Werkzeugs "Genauigkeit für die Objekterkennung berechnen"—ArcGIS Pro | Dokumentation.* Zugriff auf https://pro.arcgis.com/de/pro-app/latest/tool-reference/image-analyst/how-compute-accuracy-for-object-detection-works.htm

Esri. (2022d). *How CycleGAN Works? | ArcGIS Developer.* Zugriff auf https://developers.arcgis.com/python/guide/how-cyclegan-works/?rsource=https%3A%2F%2Flinks.esri.com%2FDevHelp_HowCycleGANWorks

Esri. (2022e). *How_MaskRCNN_works | ArcGIS Developer.* Zugriff auf https://developers.arcgis.com/python/guide/how-maskrcnn-works/?rsource=https%3A%2F%2Flinks.esri.com%2FDevHelp_HowMaskRCNNWorks

Esri. (2022f). *How Pix2Pix works ? | ArcGIS Developer.* Zugriff auf https://developers.arcgis.com/python/guide/how-pix2pix-works/?rsource=https%3A%2F%2Flinks.esri.com%2FDevHelp_HowPix2PixWorks

Esri. (2022g). *How RetinaNet works? | ArcGIS Developer.* Zugriff auf https://developers.arcgis.com/python/guide/how-retinanet-works/?rsource=https%3A%2F%2Flinks.esri.com%2FDevHelp_HowRetinaNetWorks

Esri. (2022h). *How single-shot detector (SSD) works? | ArcGIS Developer.* Zugriff auf https://developers.arcgis.com/python/guide/how-ssd-works/?rsource=https%3A%2F%2Flinks.esri.com%2FDevHelp_HowSSDWorks

Esri. (2022i). *How SuperResolution works? | ArcGIS Developer.* Zugriff auf https://developers.arcgis.com/python/guide/how-superresolution-works/?rsource=https%3A%2F%2Flinks.esri.com%2FDevHelp_HowSuperResolutionWorks

Esri. (2022j). *Image Captioning | ArcGIS Developer.* Zugriff auf https://developers.arcgis.com/python/guide/how-image-captioning-works/?rsource=https%3A%2F%2Flinks.esri.com%2FDevHelp_HowImageCationingWorks

Esri. (2022k). *Trainingsdaten für Deep Learning exportieren .* Zugriff auf https://pro.arcgis.com/de/pro-app/latest/tool-reference/image-analyst/export-training-data-for-deep-learning.htm

Esri. (2022l). *YOLOv3 Object Detector | ArcGIS Developer.* Zugriff auf https://developers.arcgis.com/python/guide/yolov3-object-detector/?rsource=https%3A%2F%2Flinks.esri.com%2FHowYOLOv3Works

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning.* MIT Press. Zugriff auf http://www.deeplearningbook.org

Haldar, M. (2015). *How much training data do you need?* (Bericht). Zugriff auf https://malay-haldar.medium.com/how-much-training-data-do-you-need-da8ec091e956

Hennig, S. (2021). Orchard Meadow Trees: Tree Detection Using Deep Learning in ArcGIS Pro. *GI_Forum*, *9* (2), 82–93. doi: 10.1553/giscience2021{_}02{_}s82

Henry, J., Natalie, T. & Madsen, D. (2021, 5). *Pix2Pix GAN for Image-to-Image Translation.* doi: 10.13140/RG.2.2.32286.66887

Rongyu Zhang, Lixuan Du, Qi Xiao & Jiaming Liu. (2020). Comparison of Backbones for Semantic Segmentation Network. *Journal of Physics: Conference Series, 1544 012196*. doi: 10.1088/1742-6596/1544/1/012196

Shang, T. & Song, Y. (2020, 7). Analysis of the Topological Structure of the Convolution Neural Network Model RESNET. In *Journal of physics: Conference series* (Bd. 1575). Institute of Physics Publishing. doi: 10.1088/1742-6596/1575/1/012136

Shilpa, A. (2019). *Faster R-CNN for object detection - A techical paper summary.* Zugriff auf https://towardsdatascience.com/faster-r-cnn-for-object-detection-a-technical-summary-474c5b857b46

Stuber Martin & Bürgi Matthias. (2018). *Vom ëroberten Landßum Renaturierungsprojekt - Geschichte der Feuchtgebiete in der Schweiz seit 1700* (Bericht). Bristol-Stiftung.

Weiweiiduann, Y.-Y., Johannes, S., Uhll, H. & Knoblock, C. (2020). *Using Historical Maps in Scientific Studies: Applications, Challenges, and Best Practices.* SpringerBriefs in Geography. Zugriff auf http://www.springer.com/series/10050

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Eigenständigkeitserklärung

Die unterzeichnete Eigenständigkeitserklärung ist Bestandteil jeder während des Studiums verfassten Semester-, Bachelor- und Master-Arbeit oder anderen Abschlussarbeit (auch der jeweils elektronischen Version).

Die Dozentinnen und Dozenten können auch für andere bei ihnen verfasste schriftliche Arbeiten eine Eigenständigkeitserklärung verlangen.

Ich bestätige, die vorliegende Arbeit selbständig und in eigenen Worten verfasst zu haben. Davon ausgenommen sind sprachliche und inhaltliche Korrekturvorschläge durch die Betreuer und Betreuerinnen der Arbeit.

Titel der Arbeit (in Druckschrift):

Untersuchung von Deep Learning Modellen in ArcGIS Pro zur Extraktion von Merkmalen aus historischen Karten: Objektdetektion und Bildübersetzung

Verfasst von (in Druckschrift):

Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich.

Name(n):	**Vorname(n):**
Scheirring	Elena

Ich bestätige mit meiner Unterschrift:

- Ich habe keine im Merkblatt „Zitier-Knigge“ beschriebene Form des Plagiats begangen.
- Ich habe alle Methoden, Daten und Arbeitsabläufe wahrheitsgetreu dokumentiert.
- Ich habe keine Daten manipuliert.
- Ich habe alle Personen erwähnt, welche die Arbeit wesentlich unterstützt haben.

Ich nehme zur Kenntnis, dass die Arbeit mit elektronischen Hilfsmitteln auf Plagiate überprüft werden kann.

Ort, Datum	**Unterschrift(en)**
Zürich, 07.06.2022	Scheirring Elena

Bei Gruppenarbeiten sind die Namen aller Verfasserinnen und Verfasser erforderlich. Durch die Unterschriften bürgen sie gemeinsam für den gesamten Inhalt dieser schriftlichen Arbeit.