Projektbericht zum Seminar: Maschinelles Lernen für Studierende der Physik

Segmentierung von Gewässern auf Satellitenbildern

Samuel Haefs

samuel.haefs@tu-dortmund.de

Projektpartner: Nico Guth

nico.guth@tu.dortmund.de

Abgabe: 15. August 2021

TU Dortmund – Fakultät Physik

Inhaltsverzeichnis

Maschinelles Lernen wird zunehmend relevanter und hat bereits heute große Anwendungsgebiete. Anfänglich wurde künstliche Intelligenz (KI) in der Wissenschaft entwickelt, revolutioniert und angewendet. Heutzutage ist KI kaum aus dem Alltag wegzudenken und ein wichtiger Bestandteil der Datenanalyse geworden.
Zum Beispiel basieren vorgeschlagene Anzeigen/Videos/Produkte auf etlichen „großen“ Internetseiten, auf Vorhersagen von KI. Die KI bekommt dabei jegliche Nutzerinformationen als Eingabe, wie die zuletzt besuchten Seiten / geschaute Videos / gekaufte Produkte und generiert ein Interessensprofil auf dessen Grundlage die Vorhersage getroffen wird. Auch Anwendungen wie Spracherkennung, Autokorrektur oder Bilderkennung wurden mithilfe von KI revolutioniert und wären ohne diese kaum realisierbar.
Das Projekt befasst sich mit dem sog. überwachten Lernen. Hierbei erhält das Netzwerk zu den Eingangsdaten, bereits die Ergebnisse bzw. Ausgangsdaten und bildet beim Training die Gesetzmäßigkeiten nach.

1 Problemstellung

Bereits seit 1957 befinden sich die ersten Satelliten auf einer Umlaufbahn um die Erde. Derzeit befinden sich 4084 (30.04.2021) Satelliten im Betrieb und kreisen im Orbit um die Erde.[1]
Einige der Satelliten, sog. Erdbeobachtungssatelliten nehmen dabei Bilder der Erde auf, um unter anderem basierend auf diesen Karten zu erstellen oder Veränderungen der Erdoberfläche festzustellen. Beispielweise können Klimaforschende so die Entstehung neuer Gewässer bzw. ausgetrocknete Gewässer erkennen und analysieren.
In diesem Projekt soll es genau darum gehen, die Erkennung und Segmentierung von Gewässern auf Satellitenbildern.

Wo befinden sich auf einem gegebenen Satellitenbild innerhalb Europas größere Wasserflächen?

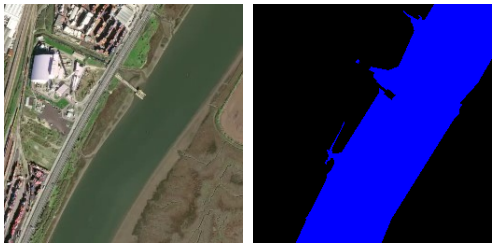
Um allein die Fläche Europas vollständig abzudecken, wären Millionen Satellitenbilder erforderlich. Das liegt daran, dass nur durch einen hinreichend kleinen Zoom-Faktor mittelgroße Gewässer erkennbar sind. Aufgrund der Kapazität, wird sich zum einen auf das europäische Festland begrenzt. Zum anderen sollen nur größere Gewässer und keine Bäche, Teiche oder Ähnliches erkannt werden.
Die Auswertung von Satellitenbildern in großer Zahl ist per Hand mühsam und kostet viel Zeit. Für solch große Datenmengen bietet sich eine Automatisierung mithilfe einer

KI an. Hier können in wenigen Sekunden tausende Satellitenbilder ausgewertet werden.

2 Datensatz

2.1 Erzeugung

Der Datensatz besteht aus einem Satellitenbild und einer Maske. Die Maske steht dabei für eine Karte, in der ausschließlich Gewässer eingezeichnet sind.
Zur Verdeutlichung ist in Abbildung 1 ein Satellitenbild und die zugehörige Maske eines Flusses in Litauen dargestellt.



(a) Satellitenbild **(b)** Maske

Abbildung 1: Beispiel aus dem Datensatz für ein Satellitenbild aus Littauen mit zugehöriger Maske. ©Mapbox, ©OpenStreetMap

Der Datensatz wurde mit Python über die API von Mapbox[3] generiert. Europa hat einen Wasserflächenanteil von wenigen Prozent. Um nun nicht größtenteils gewässerfreie Satellitenaufnahmen zu erhalten, muss geprüft werden ob sich in den gezogenen Koordinaten ein Fluss, See oder Meer befindet. Die Erzeugung des Datensatzes kann in folgende Schritte unterteilt werden:

- generiere gleichverteilte Längen- und Breitengrade auf dem europäischen Festland (Ländergrenzen aus Natural Earth Datensatz [2])
- prüfe ob Gewässer innerhalb eines 'Tiles'[1] vorkommen über die Mapbox API[3]
- Download des Satelliten- und Maskenbildes über die Mapbox API[3]
  per Link: `https://api.mapbox.com/v4/mapbox.satellite/{zoom}/{lon}/{lat}.mvt?access_token={token}`
- skaliere Bilder von (256×256)px zu (128×128)px

[1] Die Karte wird Abhängig von der Zoomstufe Z in 4^Z Quadrate unterteilt, sog. 'Tiles'.

Die zufällig erzeugten Längen- und Breitengrade werden im Bereich

$$-10{,}49° < \text{Längengrad} < 40{,}27°$$
$$34{,}51° < \text{Breitengrad} < 71{,}20°$$

gezogen, somit wird Russland und Island im Vorhinein ausgeschlossen. In dem vorliegenden Bild, welches einem Tile entspricht muss Wasser vorhanden sein. Zusätzlich darf das Satellitenbild nicht ausschließlich Wasser enthalten, um so redundante Bilder zu vermeiden. Die Längen- und Breitengrade werden akzeptiert, wenn dies zutrifft und die Koordinaten nicht bereits verwendet wurden. Das auf der Aufnahme abgebildete Land wird mittels der Koordinaten ermittelt.
Zum Herunterladen der Bilder müssen die zuvor generierten Längen- und Breitengrade und zum anderen der Zoom-Faktor angegeben werden. Umso größer der Zoom-Faktor, desto kleinere Gewässer können erkannt werden, doch desto größer ist die Wahrscheinlichkeit Satellitenbilder auschließlich mit Wasser zu erhalten. Mit dem gewählten Zoom-Faktor $Z = 15$ liegt der Fokus auf Flüsse, mittelgroße/große Seen und Küstenabschnitte.
Um die Laufzeit zu verkürzen wird die Anzahl der Pixel pro Bild um 75 % reduziert. Die Farbwerte des Satellitenbilds werden auf Werte zwischen 0 und 1 normiert. Während die Pixel des Maskenbilds mit Werten im Bereich $[0,1]$, per Schwellenwert (0.5) in binäre Werte umgerechnet werden.

2.2 Eigenschaften

Der Datensatz beinhaltet 57 931 Einträge mit jeweils folgenden Informationen:
Längengrad, Breitengrad, Name des Landes, Satellitenbild, Maskenbild

Dabei erhält das Modell das Satellitenbild als Eingabe in der Form $(128, 128, 3)$ und die Maske als Ziel in der Form $(128, 128, 1)$. Die anderen Attribute geben den Ort der Aufnahme an, sind aber nicht für die eigentliche Problemstellung, sondern für die Reproduzierbarkeit notwendig.
Das Satellitenbild liegt im JPEG-Format vor, beinhaltet also die drei Farbkanäle *RGB* mit Werten von 0 bis 255, bzw. nach der Normie-

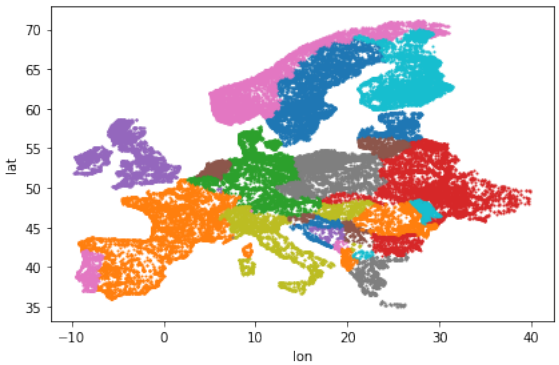


Abbildung 2: Orte der Satellitenbilder, dargestellt als Karte.

rung von 0 bis 1.
Hingegen wird für das Maskenbild das 'PNG'-
Format verwendet. Es handelt sich um Graustufenbilder mit einem Farbkanal, welcher hier den Wert 1 für Wasser bzw. 0 für kein Wasser annehmen kann.
Wie bereits erwähnt, beträgt die Auflösung für Masken- und Satellitenbild $128\,\text{px} \times 128\,\text{px}$.

3 Deep Convolutional Neural Network: U-Net

Zur Verarbeitung von Bildern haben sich in der Vergangenheit Convolutional Neural Networks (CNNs) bewiesen. Grundsätzlich besteht ein CNN aus mehreren Convolutional Ebenen, gefolgt von einer Pooling Ebene. Wird von einem tiefen CNN gesprochen, so ist damit ein Netzwerk aus mehreren CNN-Einheiten gemeint.
Die hier verwendete CNN-Struktur wird als U-Net bezeichnet und gehört zu den tiefen CNNs. Das U-Net gewann 2015 zwei Challanges, zum einen für automatische Computerdetektion von Karies in Bissflügel-Röntgenaufnahmen (ISBI 2015) und zum anderen die Erkennung von Zellen (ISBI 2015).[4] Bis heute sind Modelle dieser Art eine beliebte Wahl bei Segmentierungen.

3.1 Architektur

Die verwendete Netzwerkstruktur ist in Tabelle 3 aufgeführt und in Abbildung 3 visuell dargestellt. Das U-Net besteht dabei aus einem absteigenden und einem aufsteigenden Ast, ähnlich wie bei einem Auto-Encoder. Zusätzlich sind Lagen aus dem Encoder direkt mit Lagen aus dem Decoder verknüpft (siehe Pfeile in gen. Abb.). Die Daten werden hier lediglich kopiert und über die Verkettungs-Lage mit den vorherigen Daten zusammengefügt.
Ein wie in der genannten Abbildung dargestellter Block (oder CNN-Einheit) im aufsteigendem Ast besteht aus einer Convolutional Lage, gefolgt von einem Dropout, einer weiteren Convolutional Lage und einer Max Pooling Schicht. In der Convolutional Lage wird, wie der Name schon sagt die diskrete Faltung berechnet. Mithilfe dieser Lage können bei der Bildverarbeitung Bereiche indentifiziert werden. Durch die Einstellung 'padding = same' ist sichergestellt, dass die Bildgröße unverändert bleibt.
Die Max Pooling Ebene bewegt sich dabei über die Eingabe und speichert den maximalen Wert. Dabei werden überflüssige Informationen verworfen und die Dimension des Bildes verkleinert.
Im absteigenden Ast besteht eine CNN-Einheit aus einer Convolutional Lage, gefolgt von

einem Dropout, einer weiter Convolutional Lage, einer Transposed Convolutional Lage und endet mit einer Verkettungsebene. Hier wird die Bildgröße wieder erhöht, daher wird statt der MaxPooling eine Transposed Convolutional Lage verwendet.
Um das Übertrainieren des Netzes zu verhindern, wird in jedem CNN-Block ein Dropout verwendet. Der Dropout steigt um 0.1 alle zwei CNN-Einheiten für den aufsteigenden Ast (mit der Tiefe des Netzes). Der maximale Dropout wird bei der mittleren CNN-Einheit erreicht. Bei dem aufsteigendem Ast wird der Dropout wieder um 0.1 je zwei CNN-Einheiten reduziert.
Analog dazu wird die Filtergröße der Convolutional Lage mit der tiefe pro CNN-Einheit verdoppelt bzw. beim aufsteigenden Teil halbiert. Die Filtergrößen der einzelnen Convolutional Lagen sind zur Verdeutlichung in Abbildung 3 eingezeichnet. In der letzten Lage wird die Sigmoid-Funktion und in den restlichen Convolutional Lagen die Rectifier Aktivierungsfunktion (ReLU) verwendet.
Zudem wird Adam als Optimierer festgelegt. Als Verlustfunktion wird die binäre Kreuzentropie eingesetzt, da diese eine gute Wahl für Klassifizierungsprobleme mit zwei Klassen (0/Land oder 1/Wasser) ist. Das Modell wird über die Genauigkeit bewertet, d.h. der Anteil der übereinstimmenden Pixel der vorhergesagten Bilder mit der Maske.
Die Wahl der Hyperparameter wird im folgenden Unterabschnitt 3.2 erläutert.

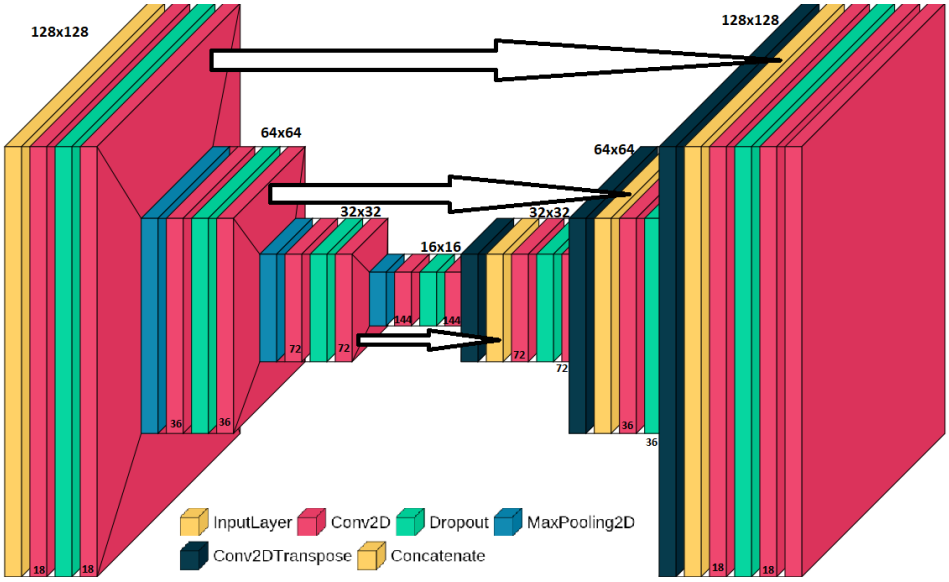


Abbildung 3: Schematische Darstellung der verwendeten Netzwerkstruktur: U-net. (VisualKeras)

3.2 Optimierung der Hyperparameter

Zur Bestimmung der optimalen Hyperparameter wurde eine zufällige Gittersuche durchgeführt. D.h. es werden nach dem Zufallsprinzip Parameter gewählt und mit diesen das Modell trainiert und evaluiert. Auf eine vollständige Gittersuche wurde aufgrund der hohen Rechenzeit verzichtet.
Es wurden 3000 Daten zum Trainieren und 3000 zum Validieren verwendet. Insgesamt wurden 100 Modelle mit einer Batchsize von 128 und Epochenanzahl von 20 trainiert und ausgewertet. Die Ergebnisse der einzelnen Parameter sind in Abbildung 4 graphisch dargestellt.

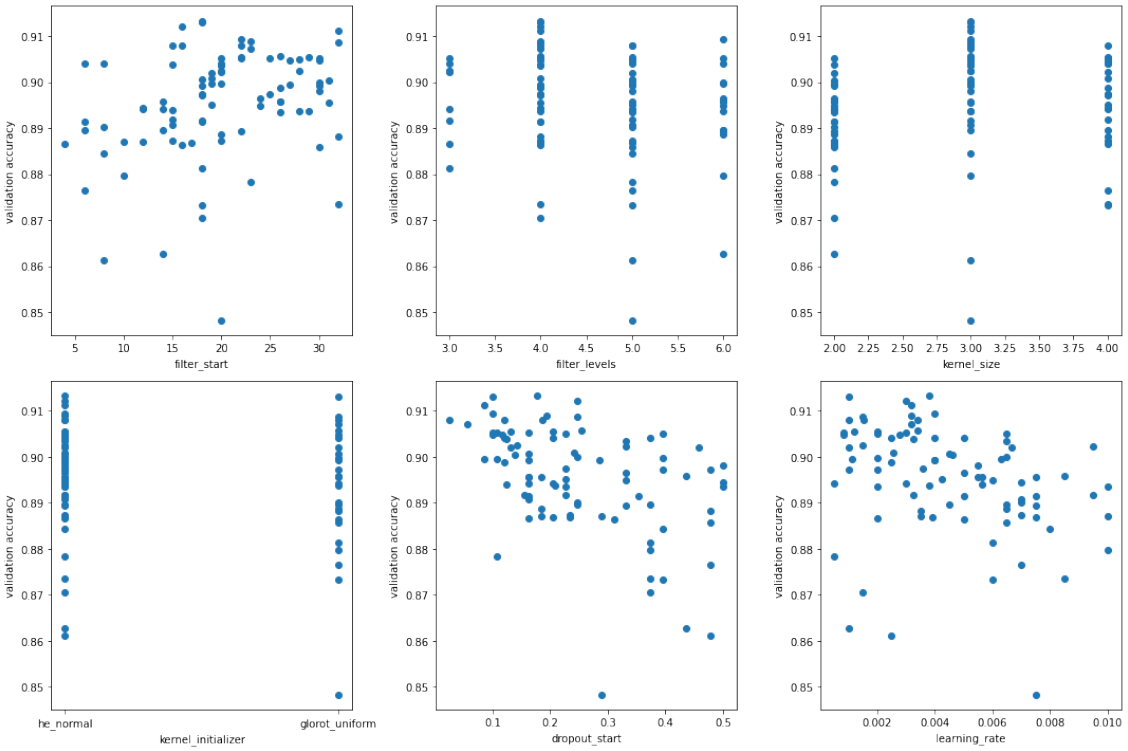


Abbildung 4: Die Genauigkeit zu den sechs variierten Parametern der zufälligen Gittersuche.

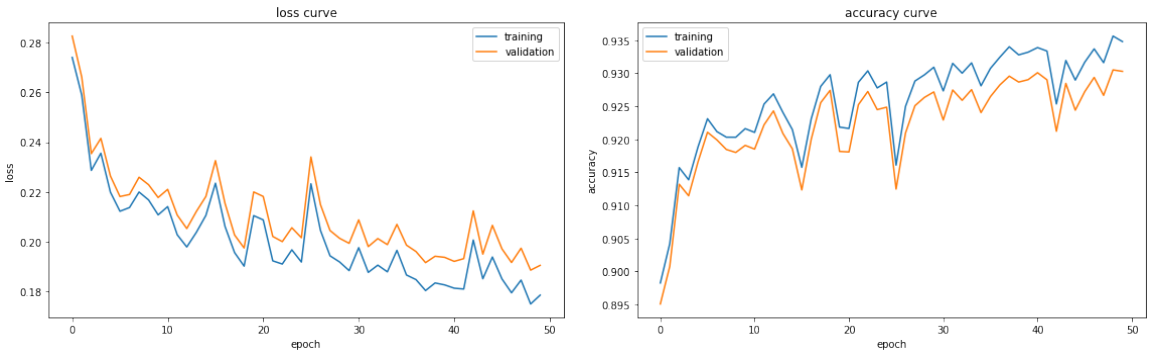
Die Wahl des 'Kernel Initialisierer' scheint kaum Einfluss auf die Genauigkeit zu haben. Hingegen ist bei der Lernrate und beim Startwert der Dropoutrate bzw. der Filteranzahl ein Trend zu erkennen. Eine geringe Dropout- und Lernrate bzw. eine größere Filteranzahl deutet auf eine höhere Genauigkeit hin. Ebenso sollte eine 3×3 Kernelgröße der Convolutional Ebene und eine U-Net Tiefe von 4 gewählt werden.
Das Modell mit der höchsten Genauigkeit auf den Validierungsdaten, wurde anschließend auf allen Daten ($\sim 58\,000$) trainiert.

Tabelle 1: Die fünf besten Modelle, absteigend nach der höchsten Genauigkeit auf den Validierungsdaten sortiert.

Filter	Tiefe	Kernelgröße	Kernel Init.	Dropout	Lernrate	Val. Genauigkeit
18	4	3	He Normal	0,177 895	0,003 795	0,913 355
18	4	3	Glorot Uniform	0,100 000	0,001 000	0,913 100
16	4	3	He Normal	0,247 368	0,003 000	0,912 037
32	4	3	He Normal	0,086 316	0,003 179	0,911 288
22	6	3	He Normal	0,100 000	0,004 000	0,909 450

3.3 Training und Schwellenwert

Das Modell, welches die besten Ergebnisse (Validierungs Genauigkeit = 91,34 %) in der Gittersuche erzielte (siehe erster Eintrag in Tabelle 1), wurde dann auf dem ganzen Datensatz trainiert und evaluiert. Es hat sich gezeigt, dass eine hohe Batchgröße eine postive Auswirkung auf die Genauigkeit hat. Aufgrund des limitierten Arbeitsspeichers wurde die maximal mögliche Batchgröße von 128 gewählt und das Modell über 50 Epochen trainiert. Aus der Epoche mit der höchsten Validierungs Genauigkeit wird das Modell gespeichert. Der Verlauf der **Genauigkeit** und der **Verlustfunktion** des Modells auf dem gesamten Datensatz, kann in Abbildung 5 betrachtet werden.



(a) Verlustfunktion **(b)** Genauigkeit

Abbildung 5: Verlauf der Genauigkeit und der Verlustfunktion über die Epochen.

Der Verlust und die Genauigkeit für Validierungs- und Trainingsdaten unterscheiden sich minimal. Es gibt keine Anzeichen für ein übertrainiertes Modell. Grund dafür ist zum einen der verwendete Dropout, aber auch die Größe des Datensatzes. Das Neuronale Netz mit $\sim 600\,000$ trainierbaren Parameter hat nicht die Möglichkeit sich an der großen Anzahl an Daten anzupassen bzw. die Bilder auswendig zu lernen.
Aufgrund der in der finalen Ebene verwendeten Aktivierungsfunktion 'Sigmoid', werden

als Ausgabe Pixelwerte zwischen 0 und 1 generiert. In Abbildung 6 ist die Verteilung der Pixelwerte für Wasser bzw. kein Wasser dargestellt.
Es handelt sich weiterhin um ein binäres Klassifizierungsproblem, daher wird ein **Schwellenwert** gesetzt. Liegt der Wert eines Pixels über bzw. unter dem Schwellenwert, so wird dieser als Wasser bzw. kein Wasser klassifiziert.
Um den optimalen Schwellenwert zu finden, wird die Genauigkeit auf den Validierungsdaten in Abhängigkeit des Schwellenwerts betrachtet. (siehe Abbildung 7)

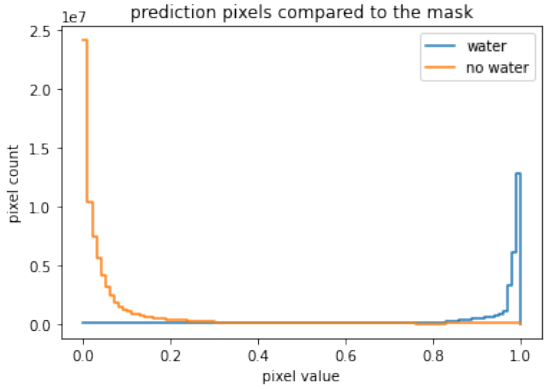


Abbildung 6
Darstellung der Häufigkeit der vorhergesagten Pixelwerte.

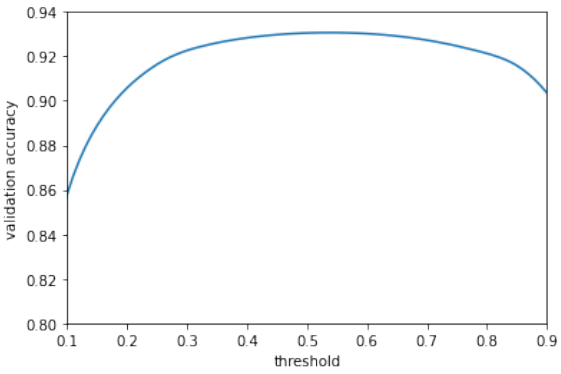


Abbildung 7
Validierungs Genauigkeit in Abhängigkeit des Schwellenwerts.

Die maximale Validierungs Genauigkeit 93,07 % wird beim Schwellenwert $S = 0{,}54$ erreicht.

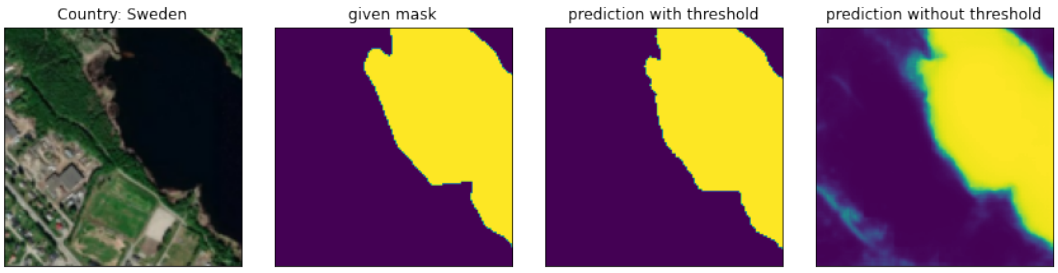


Abbildung 8: Typ der Bilder von links nach rechts: Satellitenbild, Maske, Vorhersage mit Schwellenwert, unbearbeitete Vorhersage. ©Mapbox, ©OpenStreetMap

Abbildung 8 stellt die Aufgabe des Schwellenwerts anhand eines Beispiels dar. Der Pixelwert ist ein Maß für die Wahrscheinlichkeit, ob sich an dieser Stelle Wasser befindet. Wie in dem Beispielbild zu sehen, wird der blaue Bereich links unten nach setzten des Schwellenwerts nicht mehr als Wasser eingestuft, was auch mit der Maske übereinstimmt.

Weitere Ergebnisse können in Abschnitt 5: „CNN und Random Forest im Vergleich“ betrachtet werden.

4 Alternative Methode: Random Forest

Das vorliegende binäre Klassifizierungsproblem wurde über eine weitere Methode, den Random Forest gelöst. Dieser kombiniert Ergebnisse mehrerer Entscheidungsbäume. Die Entscheidungsbäume selbst werden zufällig und unkorreliert erstellt. Jeder Baum trifft einzelne Entscheidungen, auf dessen Grundlage eine Finale Enscheidung gefällt wird.
Folgende Attribute erhält der Random Forest als Eingabe:

- 3 Farbkanäle und Graustufen des Satellitenbildes
- Gradient für die 3 Farbkanäle und Graustufen des Gradienten

Ein Gradient gibt die Steigung bzw. Ableitung in einem Punkt an. Hier beschreibt der

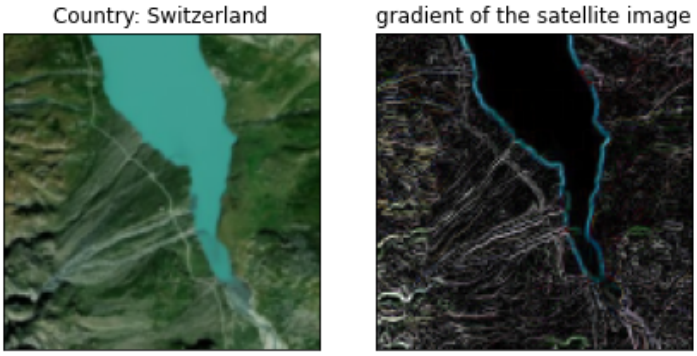


Abbildung 9: Beispielhafte Darstellung des Gradienten der Farbkanäle. ©Mapbox, ©OpenStreetMap

Gradient die Stärke der Farbänderung eines Pixels zu den benachbarten Pixeln. Wie in Abbildung 9 zu sehen, können so Kanten detektiert werden.
Aufgrund der hohen Auslastung des Arbeitsspeichers, wurde der Random Forest auf 1 000 000 Pixel trainiert. Die Pixel wurden zufällig den Bildern des Trainingsdatensatzes entnommen. Es ist nicht notwendig alle Pixel eines Bildes zu verwenden, da die Entscheidung Wasser/kein Wasser für jeden Pixel separat getroffen wird.
Als maximale Tiefe der einzelnen Bäume hat sich 32 und als Gesamtanzahl der Bäume 60 durchgesetzt. Die Ergebnisse werden auf der Metrik Genauigkeit evaluiert. Obwohl der Random Forest mit nur 1 000 000 Pixeln trainiert wurde, ist auch hier kein Anzeichen für Übertraining zu sehen. Die Entscheidung ob in dem vorliegenden Pixel Wasser vorhanden ist, wird auschließlich über die Farbwerte und Gradienten bestimmt.

5 Ergebnisse: CNN und Random Forest im Vergleich

Die in Abbildung 11 aufgeführten Bilder, können die Ergebnisse nur beschränkt repräsentieren. Es gibt viele unterschiedliche Satellitenbilder und Probleme auf die die Algorithmen stoßen. Häufig stellt die Maske nicht die wahren Wasserbereiche dar. Zum Beispiel werden Ränder von Gewässern nur sehr grob wiedergegeben, wie im letzten Bild in der genannten Abbildung zu sehen. Für dieses Satellitenbild hat das CNN und der Random Forest eine Vorhersage getroffen, die besser als die Maske selbst ist.

Tabelle 2: Genauigkeit beider Modelle auf Trainings-, Validierungs- und Testdaten.

Modell	Training	Validierung	Test
CNN	93,58 %	93,07 %	93,32 %
Random Forest	89,40 %	89,09 %	89,37 %

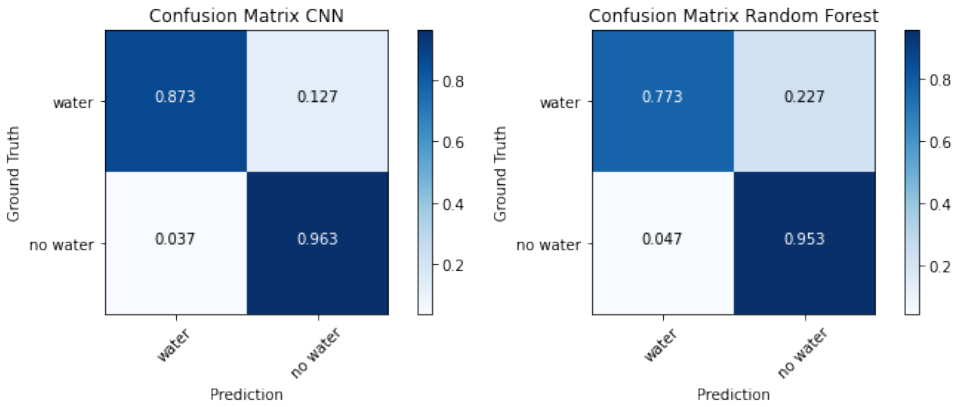
Zur Berechnung der Genauigkeit in Tabelle 2 wird die Maske als Wahrheit angenommen. Dies trifft bei einigen Masken nicht zu, weshalb eine Genauigkeit von 100 % zum einen nicht möglich ist und zum anderen nicht einer perfekten Detektion entspricht. Dennoch sind die Masken in den meisten Fällen korrekt und die Genauigkeit behält ihre Gültigkeit. Insgesamt schneidet das CNN in der Genauigkeit etwa 4 % besser als der Random Forest ab.

Zu beachten ist, dass der Random Forest aufgrund der hohen Auslastung des Arbeitsspeichers auf einem deutlich kleineren Datensatz trainiert wurde. Die Ergebnisse sind daher nur begrenzt vergleichbar.

Dennoch fällt auf, dass das CNN ganze Gebiete als Wasser klassifiziert. Hingegen betrachtet der Random Forest jeden Pixel einzeln und entscheidet auf der Grundlage der Farbwerte und Gradienten ob es sich um Wasser handelt. Die hier vorhergesagten Bilder besitzen daher häufig ein Rauschen, welches durch Pixel mit Farbwerten ähnlich zu den von Wasser verursacht wird. Dieses Problem kann auch durch einen größeren Datensatz nicht gelöst werden.

Der Fokus liegt wie anfangs erwähnt auf größeren Gewässern. D.h. Bäche und Teiche sollen nicht erkannt werden, weil diese auch nicht in der Maske enthalten sind. Dennoch können kleinere Flussarme die von einem größeren Fluss ausgehen, in der Größe eines Bachs vorkommen.(siehe gen. Abb.) Das CNN klassifiziert diese dann irrtümlich als 'kein Wasser', da beim Training Bäche und andere kleine Gewässer nicht als Wasser gewertet wurden.

Die Konfusionsmatrix in Abbildung 10 zeigt, dass Wasser zu 12,7 % (CNN) bzw. 22,7 %



(a) CNN **(b)** Random Forest

Abbildung 10: Die Konfusionsmatrix für das Convolutional Neural Network (CNN) und für den Random Forest.

(Random Forest) fälschlicher Weise als 'kein Wasser' klassifiziert wird. Hingegen wird 'kein Wasser' gerade mal zu 3,7 % für das tiefe Neuronale Netz und zu 4,7 % für den Random Forest als Wasser erkannt. Das liegt unter anderem an der ungleichen Verteilung von Wasser und 'kein Wasser'. Es sind ca. 33 % Wasserflächen auf den Satelliten- und Maskenbildern enthalten.

Insgesamt schneidet das Convolutional Neural Network sehr gut ab. Zu beachten ist, dass die Satellitenbilder in einer geringen Auflösung vorliegen und dadurch wenige Details über die Gewässer bekannt sind. Trotzdem konnte mit einer Genauigkeit von über 93 % Gewässer segmentiert werden.

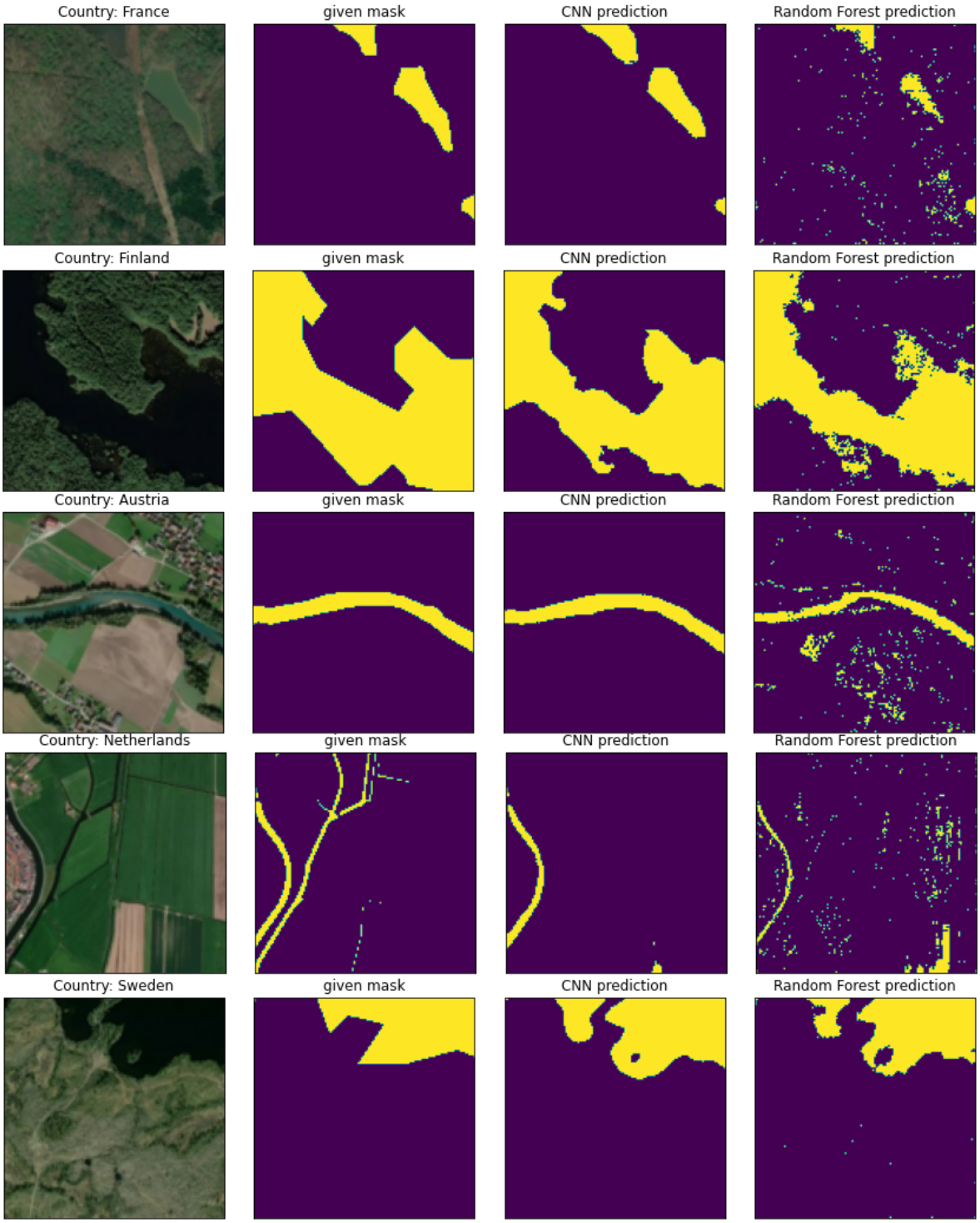


Abbildung 11: Zufällige gezogene Ergebnisse des CNNs und Random Forest auf den Testdaten. ©Mapbox, ©OpenStreetMap

6 Zusammenfassung und Fazit

In dem vorliegenden Projekt ist das Ziel die Segmentierung von Gewässern auf Satellitenbildern. Vorerst musste der Datensatz, bestehend aus etwa 58 000 Satelliten- und Maskenbildern von Gewässern erzeugt und verarbeitet werden. Das Maskenbild beinhaltet nur Gewässer, also genau das was der Algorithmus segmentieren soll.
Das Problem wird über zwei Wege angegangen. Zum einen wird ein tiefes neuronales Netz mit Convolutional Ebenen (CNN) verwendet. Durch eine besondere Anordnung der Ebenen (U-Net), konnte ein effizientes Netz mit einer geringen Parameteranzahl zur Segmentierung erzeugt werden. Die wichtigsten Hyperparameter wurden mit einer zufälligen Gittersuche optimiert.
Eine andere Methode, der sog. Random Forest segmentiert Satellitenbilder anhand von Farbwerten und Gradienten der Pixel. Das Training ist sehr ineffizient und konnte deshalb nur auf einem Teil des Datensatzes durchgeführt werden.

Rückblickend kann man sagen, dass der Random Forest trotz der erzielten Genauigkeit von über 89 % für diese Problemstellung nicht geeignet ist. Dieser verarbeitet die Pixel separat und kann somit keine zusammenhängende Gebiete erkennen. Diese hier notwendigen Verbindungen zwischen den Pixeln, kann ein tiefes neuronales Netz, wie das CNN herstellen.
Aufs Ganze gesehen hat das CNN mit einer Genauigkeit von über 93 % das vorliegende Problem erfolgreich lösen können. Die Ergebnisse übertrafen unsere (aufgrund des teils fehlerhaften Datensatzes) niedrige Erwartungshaltung um ein Vielfaches.

Literatur

[1] Union of Concerned Scientists. *UCS Satellite Database*. 2021. URL: https://www.ucsusa.org/resources/satellite-database.

[2] Natural Earth. *vector and raster map data*. URL: https://www.naturalearthdata.com.

[3] Mapbox. *Maps and location for developers*. URL: https://www.mapbox.com/.

[4] O. Ronneberger, P.Fischer und T. Brox. „U-Net: Convolutional Networks for Biomedical Image Segmentation“. In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Bd. 9351. LNCS. (available on arXiv:1505.04597 [cs.CV]). Springer, 2015, S. 234–241. URL: http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a.

Anhang

Abbildung 12: Alternative Darstellung eines U-Net zur Verdeutlich der U-förmigen Struktur. Das Level der Tiefe dieses U-Net beträgt 5. [4]

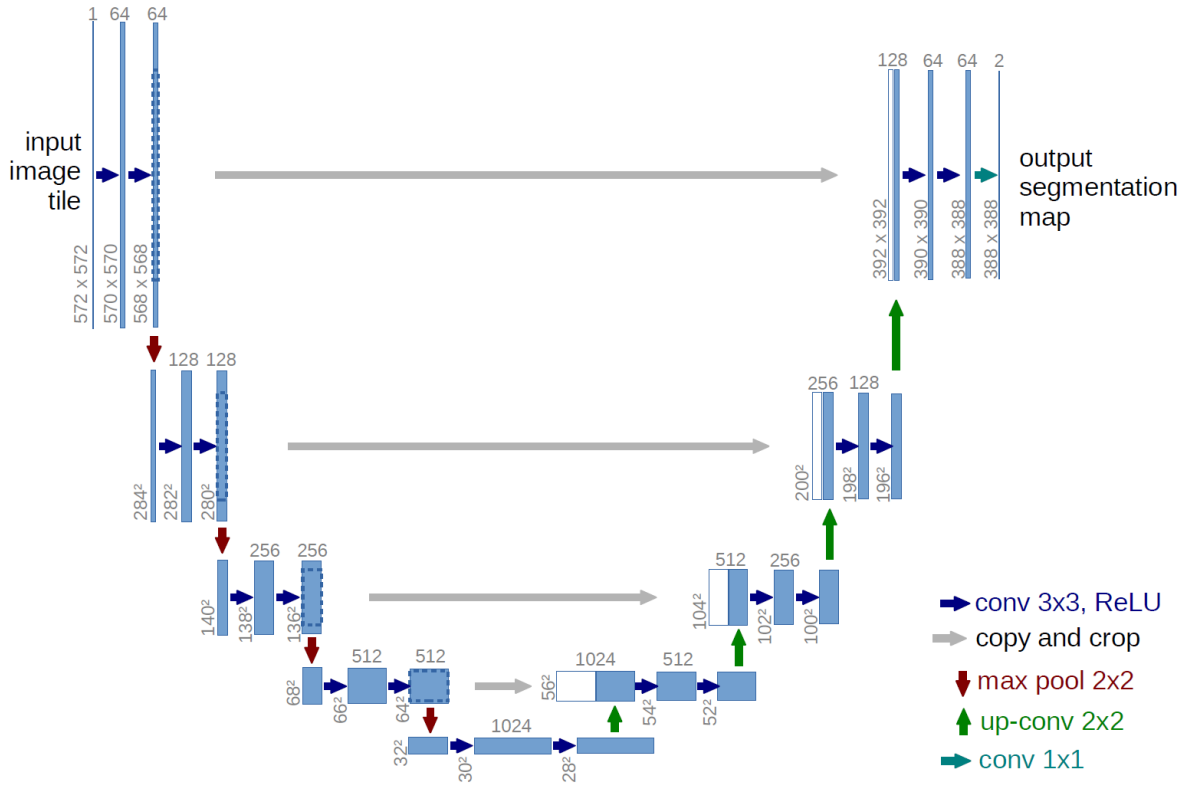


Tabelle 3: Netzwerk Struktur: U-Net

Layers	Output Dim.	Parameter
Conv 2D	$(128, 128, 18)$	504
Dropout	$(128, 128, 18)$	0
Conv 2D	$(128, 128, 18)$	2934
Max Pooling 2D	$(64, 64, 18)$	0
Conv 2D	$(64, 64, 36)$	5868
Dropout	$(64, 64, 36)$	0
Conv 2D	$(64, 64, 36)$	11700
Max Pooling 2D	$(32, 32, 36)$	0
Conv2D	$(32, 32, 72)$	23400
Dropout	$(32, 32, 72)$	0
Conv 2D	$(32, 32, 72)$	46728
Max Pooling 2D	$(16, 16, 72)$	0
Conv 2D	$(16, 16, 144)$	93456
Dropout	$(16, 16, 144)$	0
Conv 2D	$(16, 16, 144)$	186768
Conv 2D Transpose	$(32, 32, 72)$	41544
Concatenate	$(32, 32, 144)$	0
Conv 2D	$(32, 32, 72)$	93384
Dropout	$(32, 32, 72)$	0
Conv 2D	$(32, 32, 72)$	46728
Conv 2D Transpose	$(64, 64, 36)$	10404
Concatenate	$(64, 64, 72)$	0
Conv 2D	$(64, 64, 36)$	23364
Dropout	$(64, 64, 36)$	0
Conv 2D	$(64, 64, 36)$	11700
Conv 2D Transpose	$(128, 128, 18)$	2610
Concatenate	$(128, 128, 36)$	0
Conv 2D	$(128, 128, 18)$	5850
Dropout	$(128, 128, 18)$	0
Conv 2D	$(128, 128, 18)$	2934
Conv 2D	$(128, 128, 1)$	19
Gesamtanzahl:		609 895

Abbildung 13: Beispielbilder für **gute** Vorhersagen des CNNs. ©Mapbox, ©OpenStreetMap

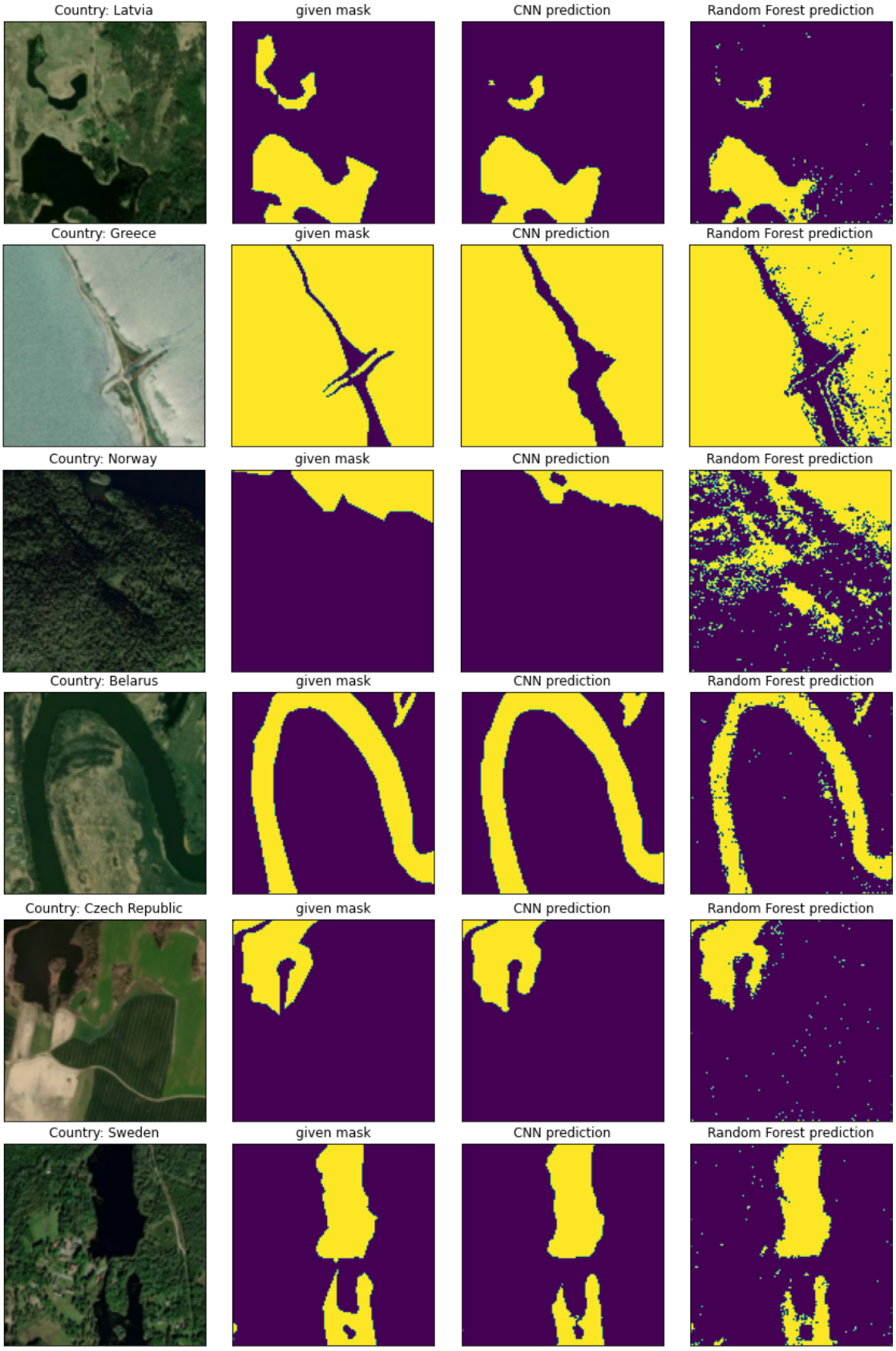


Abbildung 14: Beispielbilder für **schlechte** Vorhersagen des CNNs. ©Mapbox, ©OpenStreetMap

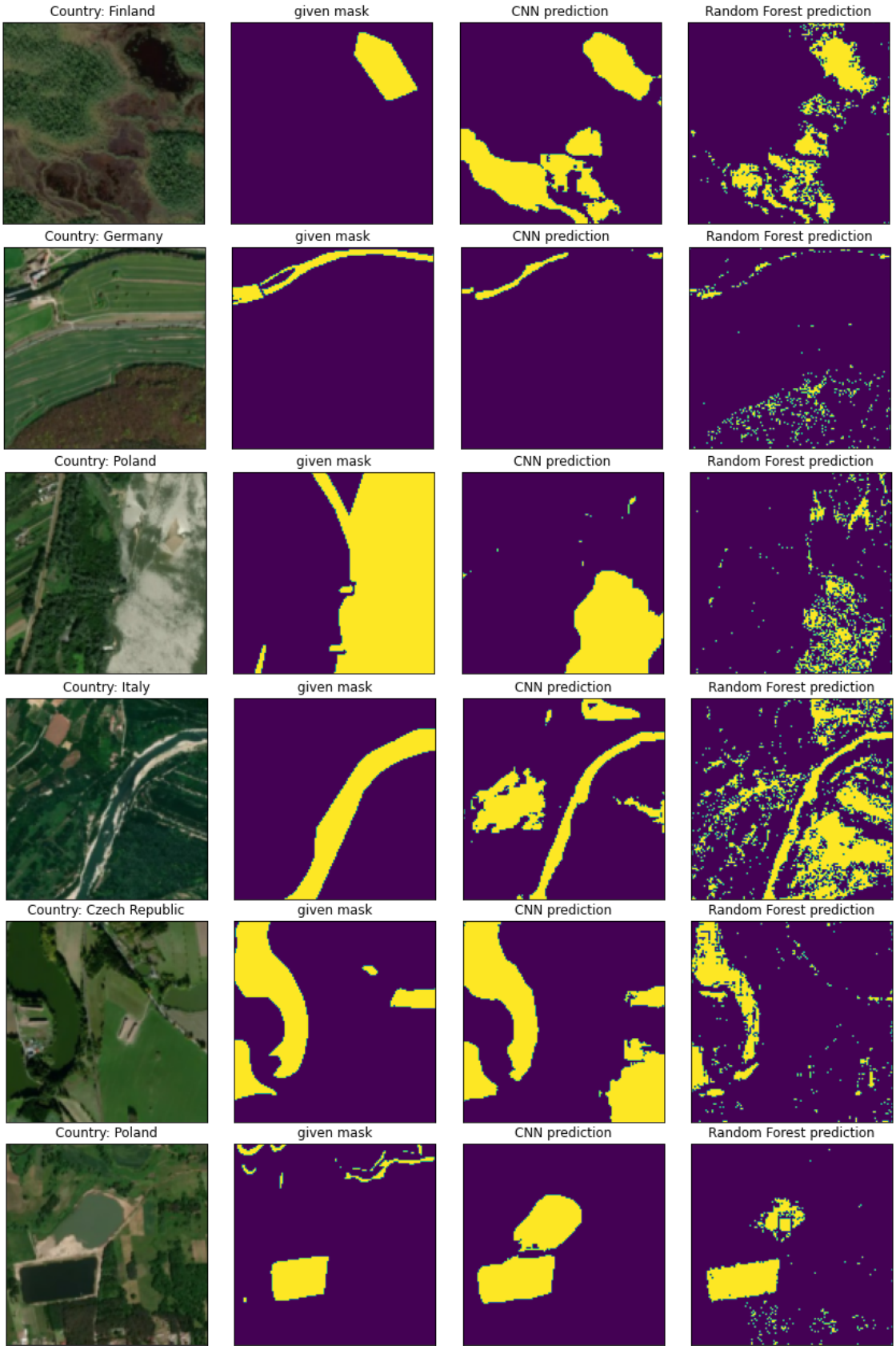


Abbildung 15: Beispielbilder zur Darstellung des Gradienten und des Schwellenwerts. ©Mapbox, ©OpenStreetMap