Diplomarbeit

Gegenüberstellung der automatischen Risssegmentierung mit Hilfe eines convolutionalen neuronalen Netzwerkes an unterschiedlichen Oberflächenmaterialien auf Basis von Open-Source-Datensets

ausgeführt zum Zwecke der Erlangung des akademischen Grads
Diplom-Ingenieurin
eingereicht an der TU Wien, Fakultät für Architektur und Raumplanung

---

Diploma Thesis

Comparison of automatic crack segmentation with the aid of a convolutional neural network on different surface materials using open-source datasets

submitted in satisfaction of the requirements for the degree
Diplom-Ingenieurin
of the TU Wien, Faculty of Architecture and Planning

Anna Maria Geiger, BSc

Matr.Nr.: 01548281

Betreuung: Associate Prof. Dipl.-Ing. Dr.techn. **Christian Schranz**, M.Sc.
Dipl.-Ing. Dr.techn. **Harald Urban**, BSc
Institut für Baubetrieb und Bauwirtschaft
Forschungsbereich Digitaler Bauprozess
Technische Universität Wien
Karlsplatz 15/235-03, 1040 Wien, Österreich

Wien, im Februar 2023

Kurzfassung

Künstliche Intelligenz findet im Bauwesen immer mehr Anwendung. Insbesondere die Bauwerksinspektion, die in regelmäßigen Intervallen Brücken, Straßenzustand und weitere Infrastrukturbauten kontrolliert und bewertet, bietet viele Möglichkeiten für eine weitreichende Optimierung. Visuelle Kontrollen sind ein wichtiger Bestandteil zur Bewertung des statischen Zustands eines Bauwerkes und zudem ein zeit- und ressourcenaufwendiger Prozess. Das Ziel der vorliegenden Arbeit liegt darin, die automatische Segmentierung auf ausgewählten Oberflächenmaterialien, basierend auf open-source-Datensätzen, gegenüberzustellen und zu bewerten. Darüber hinaus wird auch bewertet, inwiefern die gewählte Netzwerkarchitektur die Ergebnisse der einzelnen Materialien beeinflusst. Hierzu wird folgende Forschungsfrage aufgestellt: Inwiefern beeinflusst das Oberflächenmaterial die Ergebnisse der Bildersegmentierung und wie können die Resultate im Allgemeinen verbessert werden?

Auf Grundlage der theoretischen Aufarbeitung wurden Voruntersuchungen durchgeführt, die die Basis der optimale Netzwerk-Architektur für diese Arbeit bilden. Basierend auf den Erkenntnissen wurde ein angepasstes convolutionales neuronales Netzwerk (CNN) mit einer Architektur, die einem U-NET nachempfunden ist, erstellt, welche das Rückgrat der weiteren Untersuchung bildet. Es wurden teils angepasste Datensätze angelegt, welche drei-kanälige RGB-Bilder enthalten, die die unterschiedlichen Materialien abbilden und Risse aller Art, Form und Dicke abbilden. Diese Sammlungen stützen sich auf öffentlich zugänglichen Projektdatenbanken.

Die verwendete Methode zeigt, dass Materialien, die viele Schattierungen und Bild-basierte Störungen enthalten, beispielsweise Holz, zu einer Minderung der Wahrscheinlichkeit der Korrektheit führen. Materialien, die einen kleinen Farbraum in einem RGB-Histogramm in Anspruch nehmen, beispielsweise Beton, erzielen bereits bei einer kleinen Datenmenge eine hohe Wahrscheinlichkeit der Richtigkeit.

Ein einheitliches System ist für die Segmentierung von Rissen sinnvoll. Daher ist eine Netzwerkarchitektur zu wählen, die auf eine hohe RGB-Streuung optimiert wurde. Zukünftige Forschungen über die Segmentierung von Holzrissen sind zu empfehlen, um die Ergebnisse für dieses Material zu verbessern.

Abstract

Artificial intelligence is finding increasing application in the construction industry. In particular, structural inspection, which checks and evaluates bridges, road conditions and other infrastructure structures at regular intervals, offering many opportunities for far-reaching optimization. Visual inspections are an important part of evaluating the structural condition of a structure and are also a time and resource consuming process. The objective of this work is to contrast and evaluate automatic segmentation on selected surface materials based on open-source datasets. In addition, the extent to which the chosen network architecture affects the results of chosen surface materials (asphalt, concrete, masonry, wood) is also evaluated. Following research question is posed: To what extent does the surface material influence the image segmentation results and how can the results in general be improved?

Based on the theoretical chapters, preliminary investigations were conducted, which form the basis of the optimal network architecture for this work. Based on the findings, an adapted convolutional neural network (CNN) with an architecture mimicking a U-NET was created, which forms the backbone of further investigation. Partially adapted datasets for each surface material were created containing three-channel RGB images representing the different materials and depicting cracks of all types, shapes, and thicknesses. These collections are based on publicly available project databases.

The method used shows that materials containing many shades and image-based disturbances, for example wood, lead to a decrease in the probability of correctness. Materials that occupy a small color space in an RGB histogram, for example concrete, achieve a high probability of correctness even with a small amount of data.

A uniform system is useful for the segmentation of cracks. Therefore, a network architecture optimized for high RGB dispersion should be chosen. Future research on wood crack segmentation is recommended to improve the results for this material.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Bei der Errichtung, dem Betrieb und der Nutzung von Gebäuden entstehen unvermeidbare Risse aller Art [17]. Die Ursachen sind so vielfältig wie Risse an sich, von ungleichen Setzungen im Untergrund, bis hin zu Mängeln bei der Errichtung. Risse können komplett harmlos sein oder ein Indiz für eine schwerwiegende und gravierende Komplikation. Ein frühzeitiges Erkennen von ernstzunehmenden Rissen kann ein zukünftiges Versagen des Bauwerks verhindern. Die Kontrolle und Einschätzung der Mängel sind ein aufwendiger und teurer Prozess, der rund alle sechs Jahre von einer geschulten Fachkraft durchgeführt werden muss [17]. Alleine in Österreich gibt es rund 5000 kontroll-relevante Bauwerke, in Deutschland sind es fast 40.000. Zum einen muss jede Beschädigung verortet und ausgemessen werden, zum anderen muss eine Bewertung des Allgemeinzustandes des Bauwerkes abgegeben werden. Mit der Frage, inwiefern dieser Workflow erleichtert oder gar ersetzt werden kann, beschäftigen sich zahlreiche Start-ups, Unternehmen und Universitäten. Die Aufnahme von Bauwerksdefekten ist je nach Größe des zu kontrollierenden Objektes häufig ein sehr zeit-, personal- und ressourcen-intensiver-Prozess, der durch smarte Systeme unterstützt werden kann. Die Erkennung von Risses auf Bildern würde hierfür bereits viel Zeit einsparen.

Mit der Digitalisierungswelle Ende des 20. Jahrhunderts und dem dadurch ausgelösten Aufstieg der künstlichen Intelligenz (KI) öffneten sich viele Möglichkeiten zur Optimierung und Automatisierung von repetitiven Arbeitsprozessen, zu dem auch die Bauwerksinspektionen gezählt werden. Workflows in allen Berufsfeldern, welche viel Zeit und Genauigkeit in Anspruch nehmen, konnten von intelligenten Systemen unterstützt werden. Somit wurde nicht nur die Effizienz im Allgemeinen gesteigert, sondern auch die Fehleranfälligkeit durch menschliches Versagen minimiert. Visuelle Kontrollen, welche bei der Bauwerksinspektion zum Einsatz kommen, können durch Anwendungen des maschinellen Sehens unterstützt werden. Zu diesen Anwendungen zählt auch das convolutionale neuronale Netzwerk (CNN). Es beschreibt einen Vorgang, in dem durch mehrere zusammenhängende mathematische Prozesse, ausgewählte Merkmale auf Bildern erkannt werden. Für die Erstellung und das Training eines CNN sind viele Bilddaten notwendig.

2014 beschrieben Gates und Matthews [20] in einem umfangreichen Rechercheartikel, dass aufgrund neuer Technologien und Algorithmen eine große Anzahl an Daten gesammelt werden, welche im 21. Jahrhundert als neue Währung gesehen werden. Sie bilden die essenzielle Basis von KI Applikationen. Eine große, diverse und qualitative Datensammlung hat in Bezug auf den Trainingsprozess von intelligenten Algorithmen mehrere Vorteile. Zum einen wird der zeitaufwendigste Prozess der Datenfindung und -erstellung umgangen, zum anderen erhöht sich die Richtigkeit beim Einsatz außerhalb von Idealbedingungen. Diese Arbeit soll zeigen, dass mit frei zur Verfügung stehenden Ressourcen einfache Tools entwickelt werden können, die Prozesse innerhalb eines Lebenszyklus eines Bauwerkes optimieren und Planer:innen unterstützen.

Der Aufstieg von KI führt allerdings auch zu Ängsten und düsteren Zukunftsprophezeiungen für die Arbeiter:innenschaft. „Robots expected to overtake humans in construction by 2025“ [16] lautet eine Headline des Chicago Agent Magazins. Schlagzeilen wie diese sind keine Seltenheit, wirken Angst schürfend und verringern so den Fokus auf das Potential von intelligenten Systemen. Auf

der Website `WillRobotsTakeMyJob.com` [16] kann sogar mit Hilfe einer KI die Wahrscheinlichkeit für die Ersetzbarkeit eines Jobs errechnet werden. Aus Abbildung 1.1 ist entnehmbar, wie hoch das Potenzial für Bauwerksinspektor:innen ist. Demnach besteht das Risiko einer vollständigen Automatisierung dieser Berufsgruppe bei 56 %. Die Glaubhaftigkeit und Resultate dieses KI Systems sind wie bei allen intelligenten Algorithmen kritisch zu hinterfragen.

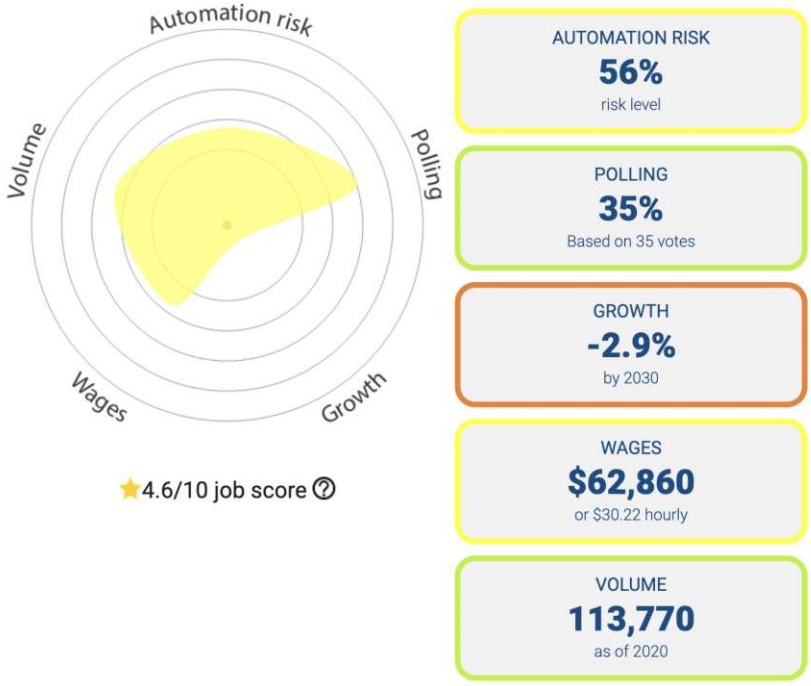


Abb. 1.1: Wahrscheinlichkeit der Berufsgruppe Bauinspektion von einer KI ersetzt zu werden

1.1 Derzeitiger Stand der Forschungen

Zahlreiche Universitäten, Start-ups und Unternehmen beschäftigen sich mit dem Thema der automatischen Risserkennung, -segmentierung und teilweise auch mit der automatischen Verortung. Insbesondere in China und dessen staatlichen Universitäten führen sehr viel Recherchearbeit diesbezüglich durch. Bereits 2006 erforschten Sinha und Fieguth [71] Erkennung und Segmentierung von Rissen, Mängel an Betonrohren. Hintergrund der Forschung war die Überwachung der Abwasserinfrastruktur, da diese insbesondere in Nordamerika häufig zu Problemen führt. Angesichts der Tatsache, dass zu Betonoberflächen viel geforscht wird, wurden hierfür bereits angepasste Algorithmen zu spezifischen Anwendungsfällen entwickelt. Ren et al. [60] führten Untersuchungen zu Betonrissen im Tunnelbau durch, wofür sie einen neuen Algorithmus „CrackSegNet“ entwickelten.

Es ist zu vermerken, dass sich Publikationen fast ausschließlich auf die Oberflächenmaterialien Beton und Asphalt beschränken. Dies ist darauf zurückzuführen, dass zum einen die meisten Ingenieursbauten in Stahlbeton oder Spannbeton erbaut wurden. Aus Abbildung 1.2 kann die Verteilung an österreichischen Autobahnbrücken nach Baumaterial entnommen werden. Daraus erkenntlich ist, dass Beton (3338 Objekte) mit Abstand am häufigsten eingesetzt wird [74]. Zum Anderen sind die RGB-Bilder von Beton und Asphalt Oberflächen vergleichsweise sehr geräuscharm. Das bedeutet, dass das RGB-Histogramm eines Bildes eine geringe Streuung entlang

der x-Achse aufweist. Allgemein beschreiben Histogramme die Abhängigkeit der Anzahl der Pixel (y-Achse) zur Helligkeit (x-Achse). Unterteilt werden die Histogramme in die Farbkanäle: R = rot, G = grün, B = blau. Eine Normalverteilung des Graphen besagt demnach ein gleichmäßig ausgeleuchtetes Bild mit wenig Schattierungen. Beinhalten Bilder eines Oberflächenmaterials wenig Störfaktoren, ist es für die KI vergleichsweise leicht einen Riss zu erkennen. Hier werden bereits bei kleinen Netzwerkarchitekturen aussagekräftige Wahrscheinlichkeiten und akkurate Ergebnisse erzielt, da die Inputs nicht mehrere Durchläufe absolvieren müssen.

Vereinzelt werden Forschungen zu andere Oberflächen durchgeführt, wie die des englischen Forschungsteams rund um Dais et al. [12], welche Risse auf Mauerwerk erforschen. In ihrer Recherche führten sie auch Experimente zur Bildsegmentierung durch, wofür sie als Basis das umfangreich erforschte U-NET verwendeten. Um die Genauigkeit auf 95,3 % zu heben, reicherten sie das verwendete Datenset auf 50 000 Bildern an, um dem Netzwerk möglichst viele unterschiedliche Vorkommen von Rissen beizubringen.

In 2020 führte das chinesische Team rund um Liu et al. [38] zum allerersten Mal in der Geschichte der Risserkennung Untersuchungen zu Rissen in Holz durch. Sie fokussierten sich hierbei auf traditionelle chinesische, meist denkmalgeschützte Gebäude, welche architektonisch wertvoll sind. Hierfür wurde der YOLOv3 Algorithmus verwendet, welcher häufig bei Echtzeitdaten von Überwachungskameras eingesetzt wird. Die Untersuchungen von Liu et al. [38] beschränken sich jedoch nur auf die reine Erkennung von Rissen in Holz. Eine Segmentierung, genauer Unterteilung eines Objektes in Untergruppen, wurde für Holzoberflächen bis dato noch nicht durchgeführt.

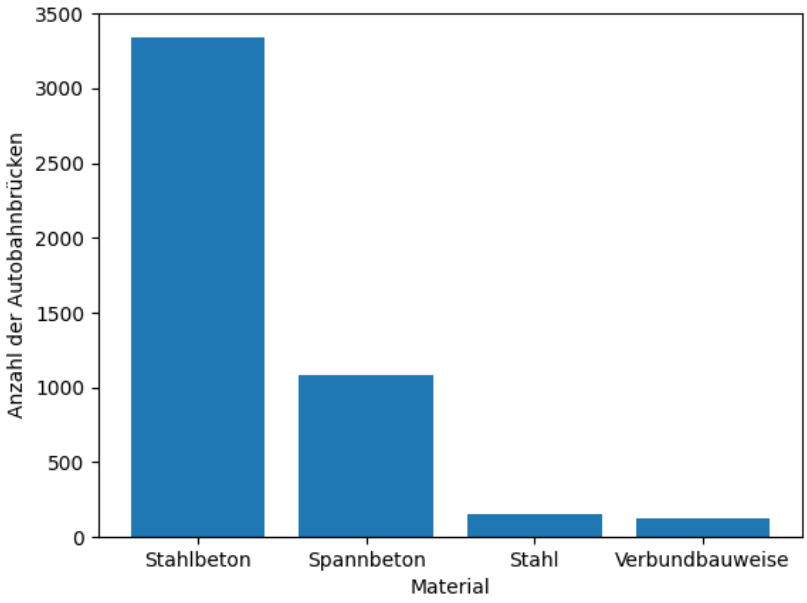


Abb. 1.2: Anzahl der österreichischen Autobahnbrücken unterteilt nach Baumaterial im Jahr 2021 [74]

1.2 Literaturstudie

Dieser Abschnitt schafft einen Überblick über verwandte Projekte und Literatur, welche sich mit der automatischen Segmentierung von Rissen beschäftigen. Mit dem Aufstieg neuer Technologien

rund um KI entwickelte sich in den letzten Jahren ein steigendes Interesse an der Objekt- und Elementerkennung. Forschungen und Anwendungen zur Segmentierung im Bauwesen sind nur wenige bekannt.

1.2.1 Forschungsprojekte zur Risssegmentierung in Bau- und Infrastruktursektor

Das Austrian Institute of Technology (AIT) forscht im Rahmen des bereits abgeschlossenen Projektes „RIBET“ an der automatischen Systematisierung und Bewertung von Rissbildern auf unbewehrten Tunnelinnenschalen [88]. Hierfür wurde ein ausführlicher Katalog erstellt, welcher Rissphänomene zu Mechanismen zuordnet. Die Trainings-, Test- und Validierungsdaten stammen aus regelmäßigen, manuell durchgeführten Messungen. Es wurden jedoch keine RGB-Bilder verwendet, sondern faseroptische Messungen, welche Risse in 95,3 m Entfernung auf 95,3 mm genau erfassen. Die Schäden an der unbewehrten Tunnelinnenschalen können nicht nur einer Ursache zugeordnet werden, sondern auch die Dicke bestimmt werden.

Das Gemeinschaftsunternehmen StrucInspect von VCE, Palfinger und der Angst Gruppe deckt die automatische Segmentierung von Rissen innerhalb einer Bauwerksinspektion ab [75]. Über die allgemeine Vision des Unternehmens wird im Abschnitt 2.4.2 eingegangen. Grundsätzlich funktioniert die Segmentierung von Rissen beim Produkt dieses Unternehmens nur auf Betonoberflächen, da diese Oberflächenmaterial den Großteil der Infrastrukturbauwerke ausmacht. Es wird von Seiten des Unternehmens weder Auskunft über das verwendete Datenset, noch über die erstellte Netzwerkarchitektur gegeben.

Das von Michelin in 2022 gekaufte, in den USA ansässige Unternehmen „RoadBotics“ entwickelte das Produkt „Roadway“, welches zur Bewertung und Verwaltung von Straßennetzen eingesetzt wird [14]. Die Input-Daten werden mit Hilfe einer Kamera (Smartphone, Drohne, GoPro oder 360°-Kamera) aufgenommen und anschließend mit Hilfe eines neuronalen Netzes bewertet und verortet. Hierbei werden die Risse auf den Bildern erkannt, wenn notwendig auch segmentiert, und anschließend in die Zustandsklassen 1 bis 5 eingeteilt. Diese werden farblich hervorgehoben und auf einer Karte lagerichtig markiert.

Jährlich wird ein Workshop der Forschungsgruppe „European Group for Intelligent Computing in Engineering“ abgehalten, in der auch immer wieder Themen zur künstlichen Intelligenz in der Baubranche Platz finden. 2021 stellten Artus et al. [4] ein Projekt vor, in der mit Hilfe eines CNN, BIM Modelles und Graphical User Interfaces (GUI) Abplatzungen auf Putz erkannt, nachmodelliert und verortet werden. Sie nannten den Prozess „Damage Information Modelling“ (DIM). Zur Segmentierung der Abplatzungen wurde das TaurusNet16 verwendet, welches jedoch nicht die gewünschten Ergebnisse erzielte. Die Generierung der Geometrie aus einer photogrammetrischen Punktwolke stellte sich auch als Problem heraus. Zum einen musste ein hochqualitatives Mesh zur Gültigkeitsprüfung erstellt werden, zum anderen musste dieses zur Visualisierung in einem IFC Viewer simplifiziert werden, um Errors zu vermeiden.

Meistens beinhalten und bearbeiten Forschungsprojekte immer nur ein Oberflächenmaterial. Hybridlösungen werden derzeit nicht entwickelt, da die Anforderungen hierfür zu groß und aufwendig sind.

1.2.2 Risssegmentierung mit U-NET

Das in Abschnitt 5.3 beschriebene U-NET wird nicht nur in der medizinischen Diagnostik zur Erkennung von Rupturen und Frakturen verwendet, sondern auch bei in der Baubranche zur Segmentierung von Rissen vor allem auf Beton und Asphalt [96]. Im Vergleich zur Risserkennung wird die Segmentierbarkeit von Rissen auf Bildern derzeit noch weniger erforscht. In der Konferenz IEEE International Intelligent Transortation System Conference wurde 2021 ein

hierzu angepasstes U-NET, das „DAU-Net“ präsentiert, welches eine höhere Genauigkeit als die state-of-art semantischen Segmentierungsmodelle auf Asphalt aufzeigt [27]. Hierzu wurde ein Novel-Channel-Attention Blöcke (CAB) in die Netzwerkarchitektur eingebaut, welche Antworten vom System vereinheitlicht und hervorstechende Eigenschaften betont.

Das ACAU-Net ist ein weiteres angepasstes U-NET zur automatischen Segmentierung von Rissen auf Asphalt. 2022 wurde es in Shijiazhuang, China, bei der International Conference on Computer Engineering and Artificial Intelligence (ICCAI) vorgestellt [33]. Das Team rund um Feng Jun verwendete als Basis ein angepasstes ResNet50 Modell mit U-NET Komponenten. Zudem wurden „Dense Atrous Convolutions Blöcke“ und Aufmerksamkeitsmechanismen eingebaut, welche die Dichte der extrahierten Eigenschaften bestimmen. Das Forschungsteam erzielte mit dem Open-Source-Datenset GAPs384 in allen Ergebniskategorien genauere Ergebnisse als andere Modelle.

El-hariri et al. [25] testeten drei angepasste U-NETs für historische Oberflächen, welche dem der erarbeiteten Netzwerkarchitektur dieser Diplomarbeit ähnlich sind. Die ausgewählten KI-Segmentierungs-Modelle lauten „U2-Net“, „ResU-Net++“ und „Deep ResU-Net“. Es wurden residuale Blöcke eingebettet, die eine akkuratere Vorhersage über Risspixel oder Nichtrisspixel gewährleisten. Die Unterschiede liegen jeweils in der Tiefe der Netzwerke und der Anzahl an Ebenen innerhalb der Modelle. Es ist anzumerken, dass das Deep ResU-Net nicht am gewählten Datenset getestet wurde, da es im Training am schlechtesten abschnitt. Aus den Resultaten ist abzuleiten, dass das U2-Net im Bezug auf die Genauigkeit besser abgeschnitten hat.

1.2.3 Vergleich der Oberflächenmaterialien

Die meisten wissenschaftlichen Arbeiten beschäftigen sich hauptsächlich mit nur einem Oberflächenmaterial und testen hierbei entweder unterschiedliche Modelle daran aus oder versuchen Netzwerkarchitekturen zu optimieren. Aus diesem Grund beinhalten open-source-Datensets oftmals nur ein ausgewähltes Oberflächenmaterial. Im umfangreichsten Datenset von qinnzou [55] sind Asphalt, Beton, Mauerwerk und Putz vorhanden. Die Autoren entwickelten jedoch im Rahmen ihrer Recherchen ein gleichnamiges, hierarchisch aufgebautes und aussagekräftiges CNN. Sie beschäftigten sich nicht mit den möglichen unterschiedlichen Ergebnissen der einzelnen Oberflächen. Gao und Jin [19] und Celik und König [10] verwendeten beim Training der vorgestellten Modelle jeweils das DeepCrack Dataset. Beide Teams wiederum führten nur Untersuchungen zur Netzwerkarchitektur durch und keinen Oberflächenvergleich.

Özgenel [46] analysierten die Performance unterschiedlicher vortrainierter Architekturen anhand eines sehr diversen Datensets. AlexNet, VGG16, VGG19, GoogleNet, ResNet50, ResNet101, ResNet152 wurden auf Beton, Asphalt, Putz und Mauerwerksuntergründen getestet. Aus den Experimenten konnte geschlossen werden, dass die Netzwerkarchitektur von VGG16 mit einem großen Trainingsdatenset von 14 000 Bildern zu aussagekräftigen Ergebnissen in der Risserkennung führt. Es ist auch ein Zusammenhang zwischen Netzwerktiefe, hohe Accuracy (Genauigkeit) und Precicion (Präzision) zu verzeichnen. Die Autoren der Arbeit vergleichen die allgemeine Leistung der ausgewählten Risserkennungsmodelle. Die im angelegten Datenset enthaltenen Materialien werden weder unterschieden noch untereinander verglichen. Die verwendeten Bilder zur Risserkennung wurden eigens für die Recherche aufgenommen und sind nicht öffentlich zugänglich. Es wird der Durchschnitt der Inputdaten erarbeitet und nicht in die einzelnen Materialien unterteilt. Die ausgewählten Netzwerkarchitekturen eignen sich abschließend nur für die Risserkennung und nicht für die Risssegmentierung.

1.2.4 Zusammenfassung

Sowohl Unternehmen als auch Universitäten inkludieren in ihren Forschungsarbeiten grundsätzlich nur wenige Oberflächenmaterialien. Vereinzelt wurden in Recherchen Datensets verwendet, welche nicht nur ein Oberflächenmaterial beinhalten, sondern mehrere. Diese Arbeit bearbeitet vier Materialien: Asphalt, Beton, Mauerwerk und Holz. Für Anwendungen und Arbeiten werden sowohl öffentlich zugängliche, als auch private und für den spezifischen Anwendungsfall manuell angelegte Datensets verwendet. Zu diesem Zeitpunkt waren noch wenige Forschungsarbeiten veröffentlicht, welche sich mit dem Vergleich der Segmentierbarkeit von Rissen auf unterschiedlichen Materialien anhand derselben Netzwerkarchitektur befasst.

1.3 Motivation und Ziel der Arbeit

Aufgrund der genannten Lücken in der Forschung hat diese Arbeit zum Ziel, nicht nur die bereits in der automatischen Risserkennung und Segmentierung erforschten Oberflächenmaterialien, wie Beton und Asphalt, gegenüberzustellen, sondern auch wenig erforschte, wie Holz, Mauerwerk und Fliesen. Die Arbeit ist von allgemeinem Interesse, da Anwendungsbereiche für die automatische Risssegmentierung sich nicht nur in der Bauinspektion von Infrastrukturprojekten finden, sondern auch in der Denkmalpflege, die zahlreiche Gebäude aus Stein und Holz beinhalten. Insofern ist die Entwicklung von material-unabhängigen Netzwerken ein wichtiger und unumgänglicher Prozess. Dabei soll darauf geachtet werden, dass es sich immer um eine Symbiose zwischen intelligenten Systemen und menschlichen Know-how handelt. Prozesse sollen optimiert und für den Menschen vereinfacht, aber nicht ausschließlich von KI übernommen werden.

Die Kernfrage der Arbeit lautet: Inwiefern beeinflusst das Oberflächenmaterial die Ergebnisse der Risssegmentierung und wie können die Resultate im Allgemeinen verbessert werden anhand Open-Source Datenbanken?

Das Ziel der Arbeit liegt darin, die Wahrscheinlichkeit der akkuraten Erkennung von Rissen auf unterschiedlichen Oberflächenmaterialien gegenüberzustellen und basierend auf den Ergebnissen Empfehlungen für die zukünftige Weiterarbeit zu geben.

Kapitel 2

Grundlagen zu Rissen

In diesem Kapitel wird ein Überblick über Risse im Allgemeinen und deren materialspezifischen Entstehung gegeben. Anschließend werden Rissarten durchleuchtet und der derzeitige Workflow der Risserkennung und Rissverortung beschrieben. Im letzten Abschnitt wird auf moderne Technologien in der Mängelerkennung eingegangen.

2.1 Definition Risse

Gemäß Duden [15] ist ein Riss eine Stelle, an der das Material eingerissen, angerissen, zerrissen oder gerissen ist. Es stellt immer eine Form von Materialversagen dar, während oder nach der Errichtungsphase eines Objektes. Risse können laut Zanke [91] die Statik eines Gebäudes verschlechtern und im schlimmsten Fall zu einem Bauwerksversagen führen. Mit rund 42 % der Bauschäden nehmen Risse den ersten Platz ein und werden dementsprechend häufig analysiert und auf mögliche Ursachen überprüft [52].

2.2 Entstehung von Rissen

Die Ursachen und Entstehung von Rissen sind laut Pilny [52] sehr vielfältig und nicht immer genau zuordbar. Die häufigsten Hintergründe lassen sich in folgende Kategorien gemäß Schulz [67] unterteilen:

- baugrundbedingte Risse,
- baustoffbedingte Risse,
- untergrundbedingte Risse und
- konstruktiv bedingte Risse.

Baugrundbedingte Risse

Diese Art von Rissen bedürfen einer genauen Abklärung der Ursache [72]. Sie können infolge einer geologischen Setzung oder aufgrund diverser Erschütterungen (Erdbeben, naheliegender Schwerverkehr) auftreten. Es wird bei dieser Form von Rissen geraten, nicht sofort und dauerhaft zu sanieren, da zuallererst die Ursache des Problems geklärt werden muss. Da sich Risse in den meisten Fällen nur langsam ausbreiten, wird häufig zu einer Beobachtung über einen definierten Zeitraum geraten.

Baustoffbedingte Risse

Zanke [91] erläutert, dass baustoffbedingte Risse immer materialspezifischen Faktoren entstammen. Das bedeutet, dass beispielsweise Holz aufgrund seiner charakteristischen quellenden und schwindenden Eigenschaft, andere Risse erzeugt als Beton. Sie unterscheiden sich in Form, Länge, Dicke und Position im Bauteil. Aufgrund dieser Tatsache sind Risse in unterschiedlichen Materialien anders zu gewichten und zu beurteilen.

Untergrundbedingte Risse

Diese Art von Riss wirkt an der Oberfläche relativ harmlos, verbirgt jedoch häufig ein ernstzunehmendes Problem – Sei es ein Riss im darunterliegenden Mauerwerk, welcher sich im Putz darüber abzeichnet, oder die rostende Armierung, welche die Betonüberdeckung zum Reißen bringt [2]. Die Ursache sollte so schnell wie möglich geklärt werden, um den Schaden einzudämmen.

Konstruktiv bedingte Risse

Laut Thagunna [80] handelt es sich bei konstruktiv bedingten Rissen um sehr ernstzunehmende Mängel, da sie die Statik eines Bauwerkes betreffen. Sie entstammen oftmals bereits entweder schon aus einer inkorrekten Planung oder falschen Ausführung und Montage. Daher ist eine Sanierung bzw. Korrektur der Fehler dringend anzuraten.

Auch bei fachgerechter Planung und Ausführung können Risse nicht komplett verhindert werden. In der Planungsphase kann lediglich die Rissbreite kontrolliert und Sicherheitsvorkehrungen, wie zum Beispiel Bodengutachten angeordnet werden. Oft ist es laut Schulz [67] schwierig, die exakte Ursache für Risse herauszufinden, da häufig mehrere Probleme zusammenhängen. Im Allgemeinen kann nur die Wahrscheinlichkeit für einzelne Ursachen und deren Auswirkungsgrad eruiert werden. Der Auswirkungsgrad kann in den Mängelstufen 1 – 10 eingeteilt werden und beschreibt schematisch die Mängelfreiheit bzw. die Brauchbarkeit der untersuchten Bauwerke oder Bauteile.

2.3 Rissklassifizierung

Grundsätzlich hängt die Art und oftmals auch der Schweregrad des Risses stark vom Material bzw. von dessen Untergrund (beispielsweise bei Putz) ab. Sie können nach verschiedensten Augenpunkten kategorisiert und eingeteilt werden. Gemäß Schulz [67] bilden sich konstruktionsbedingt maßgeblich acht Kategorien für die Klassifizierung:

- Risse, die den Fugenverlauf nachzeichnen,
- Einzelrisse mit auffällig geradlinigem Verlauf,
- Risse mit weitgehender vertikaler oder waagerechter Ausrichtung,
- Vertikalrisse im Eckbereich von Mauerwerken im Abstand der Wanddicke,
- Schubrisse,
- Risse, die Mauerwerksöffnungen miteinander verbinden,
- Kerbrisse und
- Abrisse.

Da es für eine ordnungsgemäße Klassifizierung nicht ausreicht, den Riss pauschal zu analysieren, muss für eine ausführliche Beschreibung eines Risses, dieser auch verortet werden, der Verlauf skizziert und in der Breite, bzw. der Dicke abgemessen werden [67]. Diese Rissdicken können wiederum in Klassen eingeteilt werden, welche in Tabelle 5.2 angeführt sind. Daraus kann auch die zu jeder Rissart zugehörige Beschreibung entnommen werden. Insbesondere bei großen Rissen ist die Angabe der Breite und Tiefe in Millimeter wichtig.

Wichtig anzumerken ist, dass der Schweregrad der Risse stark vom Material abhängt. Die ÖNORMEN B 3346 und B 4710-1 listen die zulässigen Rissbreiten auf Putz und Beton auf. Eine spezifische Rissbreite kann in einem Material ernstzunehmende Folge haben, als in einem

Tab. 2.1: Rissklassifizierung nach Rissbreite in Millimeter mit Beschreibung

Rissart	Beschreibung	Rissbreite in mm
feiner Haarriss	„gerade“ noch sichtbarer Riss	0,05
Haarriss	haarfeiner Riss, der sich deutlich abzeichnet	0,20
feiner Riss	zwischen „Haarriss“ und „mittlerem Riss“	bis ca. 0,25
mittlerer Riss	deutlich sichtbar und breiter	bis ca. 0,50
großer Riss	Angabe erforderlich: Breite: ..... mm Tiefe:..... mm	mehr als 0,50

anderen. Unterdessen werden auch Maßnahmen zur Unterbindung von Rissen geschildert. Darüber hinaus tragen Risslänge, Rissverlauf und eine mögliche Fortbewegung dazu bei, dass der Mangel fachgerecht nach Schweregrad eingeordnet werden kann.

Die Einordnung und Kategorisierung von Rissen bzw. allgemein Bauwerksmängel und Schäden wird im Rahmen einer Bauwerksinspektion durchgeführt. Anzumerken ist jedoch, dass dies nur einen kleinen Bruchteil des ganzen Prozesses ausmacht. Der gesamte Prozess wird im nachfolgenden Abschnitt ausführlich erläutert.

2.4 Bauwerksinspektion

Die Prüfung und Instandhaltung von Bauwerken hat besonders für Brückenbauten eine große Bedeutung [69]. Im Allgemeinen stellt sie einen integralen Bestandteil des Lebenszyklus eines Gebäudes bzw. eines Ingenieurbaus dar. Die Bauwerksinspektion wirkt sich in mehreren Themenfeldern positiv auf die Nutzungsdauer aus. Zum einen wird durch die Früherkennung von Problemen nicht nur den Lebenszyklus eines Objektes verlängert, sondern auch die Wirtschaftlichkeit durch Senkung mehrerer Folgekosten erhöht. Bei rund 6500 Brücken in Österreich, 37 000 in Deutschland und 8000 in der Schweiz können durch die Bauwerksinspektionen und das daraus resultierende verfrühte Erkennen von Problemen, große Geldbeträge eingespart werden [29, 70].

Die DIN 1076 (RI-EBW-Prüf) [8] ist in Deutschland die Grundlage für die Prüfung und Überwachung von bestehenden Ingenieursbauwerken. In Österreich findet sich keine Norm für solche Vorhaben, jedoch bilden die Richtlinien und Vorschriften für den Straßenbau (RVS) die gesetzliche Basis, die wiederum häufig auf die DIN 1076 verweisen. Im Speziellen die RVS 13.71 [64] bietet Grundlagen für die Prüfung, Kontrolle und Überwachung von Ingenieursbauten. Diese Richtlinien zeichnen sich, gleichwertig den ÖNORMEN in Österreich, durch den aktuellen Stand der Technik aus und können durch das Bundesministerium für Wirtschaft und Arbeit (BMWA) für verbindlich erklärt werden.

Die Bauwerksinspektion ist ein essenzieller Bestandteil des Qualitätsmanagements eines Gebäudes oder eines Ingenieurbaus, die von einer geschulten Fachkraft durchgeführt werden muss. Insbesondere während der Nutzungsphase sind regelmäßige Bauwerksprüfungen ein wichtiger Teil der Erhaltungsmaßnahmen,die rund alle sechs Jahre durchgeführt werden muss. Es ist wichtig zu beachten, dass bereits in der Planungsphase an die Bauwerksinspektion gedacht werden sollte. Teile mit hohem Verschleiß z. B. Auflager, Seile und Knotenpunkte müssen auswechselbar sein und somit auch zugänglich geplant werden.

2.4.1 Derzeitiger Workflow der Bauwerksinspektion

Laut Schulz [67] findet sich in Österreich bis heute kein einheitliches Regelwerk für die Minimierung oder Bewertung von Rissen. Trotz unterschiedlicher Begrifflichkeiten kann aus vielen Merkblättern und Regelungen der gleiche Grundgedanke für die Bewertung herausgefiltert werden. Das deutsche

Bundesministerium für Digitales und Verkehr [8] definiert in der DIN 1076 die Bedeutung einer Bauwerksprüfung, Maßnahmen, Bestimmungen und einen empfohlenen Ablauf. Abbildung 2.1 zeigt, dass je nach Architektur des Ingenieursbaues sehr aufwendige Maßnahmen notwendig sein können um alle Mängel zu erfassen. Gerüste, eigens für die Inspektion konstruierte Fahrzeuge und Hubgeräte können zum Einsatz kommen. Während der Kontrolle erkannte Risse werden oftmals direkt am Bauwerk markiert und beschriftet, wie in 2.1 b) zu sehen ist. Dadurch wird die Sichtbarkeit auf Bildern erhöht und eine ergänzende Dokumentation beigefügt.

Abb. 2.1: Ausschnitte der Maßnahmen die bei einer Brückeninspektion notwendig sein können

Schematisch kann der Prozess der Bauwerksinspektion in drei Überkategorien eingeteilt werden: die Vorbereitung, die Durchführung und die Dokumentation und Auswertung.

Vorbereitung

Im ersten Schritt muss die Bauwerksinspektion geplant und eine fachkundige Person beauftragt werden. In Wien bietet das Magistrat für Brückenbau und Grundbau (MA 29) einen Lehrgang zum fachkundigen Brückeninspektor:in an. Dieser ist in einen Basis- und Aufbaulehrgang gegliedert und kann mit einem offiziellen Zertifikat und mit Urkunde abgeschlossen werden. Es wird im Allgemeinen laut DIN 1076 [8] empfohlen, dass die Überwachung und Überprüfung von unterschiedlichen Einheiten, bzw. Organisationen durchgeführt wird, um ein „Mehr-Augen-Prinzip“ zu gewährleisten. Vielerorts werden auch externe Teams mit der Aufgabe beauftragt. Um sich optimal auf die Bauwerksinspektion vorzubereiten, müssen die Dokumente der vorhergegangenen Prüfungen, Bauwerksdaten und -dokumente gesichtet werden. Besondere Anweisungen zur Inspektion sind im Bauwerksbuch vermerkt. Anschließend wird die Prüfung organisiert und alle Vorkehrungen getroffen.

Durchführung

Prinzipiell gibt es zwischen fünf Arten von Bauwerksprüfungen und -überwachungen, die sich in den Intervallen und Inhalt der Prüfung unterscheiden:

- Laufende Beobachtungen,
- Besichtigung,
- Einfache Prüfung,
- Hauptprüfung und
- Sonderprüfung.

Wichtig ist, dass die Prüfungen ergänzend zueinander gesehen werden müssen. Die zentrale Bauwerksinspektion ist die Hauptprüfung, die schon vor der Abnahme der Bauleistung durchgeführt wird. Anschließend wird die zweite Hauptprüfung vor Ablauf der Verjährungsfrist ein weiteres Mal durchgeführt, letztlich wird diese periodisch im Sechs-Jahres-Abstand wiederholt. Hierbei sind alle Bauwerksteile zu kontrollieren, auch solche, die schwer zugänglich sind. Beschädigungen, Mängel und Veränderungen sind in einem Prüfbericht zu dokumentieren und zu kennzeichnen, der bei einfachen Prüfungen kontrolliert wird. Diese werden drei Jahre nach der Hauptprüfung als ergänzende Kontrolle durchgeführt. Eine Sonderprüfung wird nur aufgrund eines besonderen Anlasses beauftragt. Sie ist eine außertourliche und keine in Intervallen durchzuführende Überprüfung, welche beispielsweise aufgrund von einem Unfall oder Hochwasser verursachte Schäden verursacht wurden. Sonderprüfungen werden nur beauftragt, wenn es als notwendig erscheint, und ersetzen weder eine Hauptprüfung, noch eine einfache Prüfung. Besichtigungen werden laut DIN 1076 jährlich ohne größere Hilfsmittel durchgeführt. In den Jahren, in denen eine Hauptprüfung oder eine einfache Prüfung durchgeführt wird, wird keine Besichtigung veranlasst. Nichtsdestotrotz sind Veränderungen, Mängel und Schäden zu protokollieren. Laufende Beobachtungen werden grundsätzlich zweimal jährlich im Rahmen einer Streckenkontrolle absolviert. Aufkommende Beschädigungen, Mängel und Veränderungen müssen ebenfalls protokolliert und vermerkt werden.

Dokumentation und Auswertung

Die Ergebnisse der Bauwerksinspektion werden in einem Prüfbericht dokumentiert, inklusive aller Messdaten, Skizzen, Fotos und zusätzlichen Resultaten von weiteren Untersuchungen. Schäden und Mängel werden anhand drei Kriterien bewertet: Verkehrssicherheit (V), Standsicherheit (S) und Dauerhaftigkeit (D). Anschließend wird basierend auf den Erkenntnissen der Bauwerksprüfung eine Zustandsnote von dem:der Ingenieur:in vergeben. Diese reicht von„sehr guten Zustand“ (bis 1,4) bis „ungenügender Zustand“ (bis 4,0). Abschließend übernimmt die zuständige Person mit ihrer Unterschrift die fachliche Verantwortung, dass sich das Bauwerk derzeit in dem dokumentierten Zustand befindet.

2.4.2 Digitalisierung der Bauinspektion

Eine Studie zur Digitalisierung der Bauindustrie aufgrund von Covid-19 des Consulting Unternehmens PricewaterhouseCoopers [54], kurz PwC, besagt, dass die Pandemie kaum Auswirkungen auf die Digitalisierungslage hatte im Vergleich zu anderen Branchen. 62 % der im Bausektor tätigen, befragten Personen berichten hingegen von einem ausbaufähigen Stadium in der Kategorie „Digitale Lösungen“, welche unter anderem Laserscanning, AR, VR, sowie KI-Anwendungen beinhaltet [54]. Im Bereich der Drohnenüberwachung konnte sogar ein 1 %-iger Rückgang im

Vergleich zu 2019 verzeichnet werden. Das Potenzial zur Optimierung von Arbeitsprozessen wird von 50 % der Befragten als sehr groß angesehen. Daraus kann geschlossen werden, dass schon vor der Pandemie ein Digitalisierungsrückstand im Vergleich zu anderen Industrien bestanden hat. Prozesse, die schon pre-Covid vom analogen auf den heutigen Stand der Technik gehoben wurden, waren von der Pandemie nicht betroffen. In der Studie von PricewaterhouseCoopers [54] wurden die teilnehmenden Personen auch nach der Begründung für das Ausbleiben der Digitalisierungswellen innerhalb der Baubranche gefragt und der Großteil (81 %) berichtete von einem Fachkräftemangel bzw. dem Ausbleiben des fachspezifischen Know-hows. Auch Pan und Zhang [48] können dies bestätigen. Das Interesse an KI unterstützten Anwendungen in der Baubranche ist enorm und kann anhand der immer mehr zunehmenden Zahlen der Recherchearbeiten belegt werden. Insbesondere durch die mittlerweile etablierte Methode des Building Information Modelling, kurz BIM, werden große Mengen an Daten gesammelt, die mit Hilfe von KI-Anwendungen sinnvoll aufgearbeitet und verarbeitet werden können [37, 48].

Sicherheit

Eines der bedeutsamsten Argumente in Hinblick auf den Einsatz für smarte Systeme ist die allgemeine Sicherheit. Laut Manuel Eberhardt [39] liegt die Baubranche mit 17 637 [39] rechtlich gültigen Arbeitsunfällen, 27 davon tödlich, im Jahr 2013 nur knapp auf Platz zwei hinter der Wirtschaftsklasse „Erzeugung von Waren“ – Ungeachtet der Tatsache, dass Unfälle, welche aufgrund von schadhaften, fehlerhaften oder qualitativ unzureichenden Bauwerken oder Bauteilen ausgelöst wurden, in diese Zahl noch nicht einberechnet worden sind. Es ist daher wenig verwunderlich, dass sehr viel Forschungsarbeit in die Richtung Baustellensicherheit stattfindet. Pan und Zhang [48] beschreiben unter anderem mehrere Anwendungsfälle, in denen künstliche Intelligenz bezüglich Sicherheit zum Einsatz kommt. Insbesondere die Computer Vision, welche mit Video- und Bildmaterial arbeitet, kommt zum einen bei der automatischen Erkennung von Helmen auf Baustellen zum Einsatz, zum anderen auch bei der in dieser Arbeit genannten Bauinspektion, um das Bauwerk an sich sicherer zu machen. Zhang [93] spricht von einem großen Potenzial in Bezug auf die Sicherheit. Dem Autor nach könnte die allgemeine Sicherheit durch Computer Vision Anwendungen um bis zu 94 % gesteigert werden.

Erkennung von Rissen

Mit Fujita et al. [18] wurde die automatische Erkennung von Rissen in die Wege geleitet. Trotz der im Vergleich zum heutigen Zeitpunkt schwachen Rechner ist es dem Team gelungen, effektive Ergebnisse auf RGB-Bildern mit hoher RGB-Streuung zu erzielen. Es wurde jedoch nur mit dem Material Beton gearbeitet, da dies das am häufigsten vorkommende Material in Bezug auf Infrastrukturbauten ist. Karaaslan et al. [34] berichten von einer ausgefallenen Mischung aus Virtual Reality (VR) und maschinellem Lernen, welche sie „Smart-Mixed Reality“ nennen, für die optimale Bauwerksüberwachung und -inspektion. Der Hintergedanke dieser Forschungsarbeit liegt darin, dass die Fähigkeiten des Menschen mit Unterstützung von intelligenten Systemen erweitert werden. Risse werden automatisch erkannt und über die VR-Brille wird der Schweregrad bzw. die Benotung des Risses eingeblendet. Das Forschungsprojekt AR-AQ Bau von Urban et al. [81] setzt auf die Verwendung von Augmented Reality (AR). BIM-Modelle werden mit der Realität überlagert und Abweichungen sollen automatisch erkannt werden.

StrucInspect

Das österreichische Start-up und Joint Venture „Structinspect“ versucht mit Bildsegmentierung den Workflow der Bauschädenmarkierung und Bewertung zu automatisieren und optimieren. Laut Fercher [17] ist das Produkt von Strucinspect überdies effektiver als der traditionelle Weg, da das Bauwerk ganzheitlich erfasst wird und nicht nur die Bauschäden an sich. Es werden folglich nicht nur die Risse aufgenommen, sondern zugleich auch ein digitales dreidimensionales Modell

erstellt. 2020 wurde das Unternehmen mit dem ersten Platz der „Digital Awards“ in der Kategorie „Transformation“ ausgezeichnet. Strucinspect deckt den ganzen Prozess von der Erfassung der Risse bis hin zur Digitalisierung und Beurteilung des Bauwerkszustandes ab. Die Daten hierfür werden mit Hilfe einer Drohne aufgenommen. Die Bilder werden sowohl für die Segmentierung verwendet, als auch für die Erstellung eines digitalen Zwillings mittels Photogrammetrie. Um optimale Ergebnisse zu gewährleisten, ist auch immer Fachpersonal vor Ort, das einerseits die Drohne steuert, andererseits zusätzliches Augenmerk auf auffällige Schäden legt.

Kapitel 3

Grundlagen des Machine Learnings

Machine Learning ist bereits im Alltag der meisten ein wichtiger Bestandteil [68] – sei es der Virtual Assistant, z. B. Siri, Cordana und Co, die automatische Berechnung der Fahrzeit und -distanz bei Navigationssystemen wie Google Maps oder der Suchmaschinen-Algorithmus von Google. Machine Learning macht nicht nur unseren Alltag effizienter, sondern bietet auch Unterstützung bei der Verarbeitung von großen Datenmengen. In diesem Kapitel wird ein kurzer Überblick über Machine Learning gegeben und komprimiert die allgemeine Funktionsweise geschildert. Zu Beginn werden die Anfänge dieser Technologie kompakt erläutert und die wichtigsten Grundsteine beschrieben, welche für den heutigen Stand der Technik gelegt wurden. Es werden auch wichtige, verwandte Begriffe angeführt und voneinander abgegrenzt. Dies ist notwendig, da insbesondere in der Informatik häufig „Buzzwörter“ als Synonym verwendet werden, ungeachtet davon, dass sich ihre eigentliche Bedeutung stark voneinander unterscheiden. Im Weiteren werden wichtige Maßnahmen für einen reibungslosen Trainingsprozess beschrieben und Vorgänge erläutert, welche währenddessen stattfinden.

3.1 Geschichte des Machine Learnings

Das Jahr 1956 gilt nach Wennker [85] als Geburtsjahr der Künstlichen Intelligenz. In „A Proposal for the Darthmouth Summer Research Project on Artificial Intelligence“ [31] wurde ein Förderantrag mit folgendem Forschungsziel eingereicht [31, S.2]:

> *„The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves.“*

Die Forschenden beschäftigten sich hauptsächlich mit Natural Language Processing (NLP). Auch in den darauf folgenden Jahren wurde viel Geld und Mühe in die automatische Übersetzung von russischen Texten ins Englische investiert [68]. Der Grundgedanke war, dass jeder Prozess des Lernens oder jede andere Form von „Intelligenz“ so detailliert beschrieben wird, dass sich dies ein Computer anhand einer Anleitung aneignen kann. Hierbei sollen Programme und Skripte von sich selbst lernen, was bis dato einzig und alleine nur für den menschlichen Verstand denkbar gewesen wäre. Da Erfolge nur sehr langsam zu verzeichnen waren, kam es in den frühen 1970er zu einem vorübergehenden Ende der Forschung.

In Russels und Norvigs Informatik Klassikers [63] „Artificial Intelligence, A Modern Approach“ wurde KI ein weiteres Mal Ende des 20. Jahrhundert aufgegriffen. Die beiden Autoren erforschten intelligente Agenten, welche beispielsweise in IBMs Schachroboter “Deep Blue“, der 1997 den Schach-Champion Garry Kasparov besiegte, zum Einsatz kommen. Diese Art von „intelligenten“ Systemen nehmen ihre Umgebung wahr und reagieren auf deren Reize bzw. Inputs. Deep Blue

beobachtet das Schachbrett und jeden Spielzug des Gegners. Mit jeder Veränderung der vorherrschenden Bedingungen führt der Schachroboter Berechnungen mit den neuen Inputs durch, um die eigenen Gewinnchancen zu erhöhen. Durch selbstlernende Algorithmen verbessern diese Systeme ihre Wahrscheinlichkeit auf einen Erfolg, was bedeutet, dass sie mit jedem Training und neuen Inputs besser werden [85]. Dies beeindruckte nicht nur Investor:innen und Programmierer:innen, sondern auch die allgemeine Gesellschaft. Es wurden mehrere Bücher, Filme und Artikel über dieses Ereignis publiziert. Somit war die Faszination, aber auch die damit verbundene Angst vor der künstlichen Intelligenz, dem demnach Unbekannten und Unberechenbaren, geboren.

Aufgrund der immer leistungsstärker und effizienter werdenden GPUs (Graphical Processing Unit oder Grafikkarte) entstand in den letzten zehn Jahren ein regelrechter Boom in der Entwicklung von KIs. Viele Prozesse werden stetig durch intelligente Systeme ersetzt. Nicht nur aus wirtschaftlicher Sicht, durch die Einsparung von Personalkosten, sondern auch aus sicherheitstechnischer Sicht macht der Einsatz von smarten Systemen Sinn. Es werden sowohl sehr eintönige Abläufe ersetzt, als auch solche, die ein erhöhtes Sicherheitsrisiko für den Menschen verbergen. Dennoch muss sich laut Ore und Sposato [45] vor der Erstellung eines ML-Algorithmus die Frage gestellt werden, inwiefern ebendieser diskriminiert (Personengruppen oder im Allgemeinen Datenlabel), beeinflusst und welche Resultate damit erzielt werden. Mit dem genannten Aufstieg der künstlichen Intelligenz entwickelten sich auch immer mehr und komplexere Lern-Algorithmen, für die eine große Rechenleistung aufgebracht werden muss. Leistungsstarke GPUs ermöglichten die Entwicklung von Deep-Learning-Algorithmen, welche durch versteckte Prozesse zum heutigen Stand am meisten Rechenleistung beanspruchen [21].

KI, Deep Learning, Machine Learning und viele weitere verwandte „Buzzwords“ werden häufig als Synonyme verwendet, obwohl sie jeweils unterschiedliche Themenbereiche abdecken. Die Begriffe lassen sich nur sehr schwer voneinander trennen und oftmals finden sich mehr Gemeinsamkeiten als Unterschiede. Jedoch ist es falsch, die einzelnen Fachbegriffe sinngleich zu verwenden. Künstliche Intelligenz ist laut Raschka und Mirjalili [68] der Dach-Begriff für eine Bandbreite an Technologien und intelligenten Methoden. Genauer ausgedrückt handelt es sich hierbei um ein Teilgebiet der Informatik, welches sich mit der Automatisierung von Prozessen mit Hilfe von Machine Learning (ML) beschäftigt. Eine exakte Definition ist schwierig, da sich bereits griechische Philosophen bezüglich der Bezeichnung von „Intelligenz“ nicht einig waren.

Machine Learning hat sich aus KI heraus entwickelt und erleichtert das Verarbeiten, Analysieren, Filtern und Extrahieren von Daten aus großen Datenmengen. Es handelt sich hierbei um eine Art Werkzeug, welches für Problemlösungen zum Einsatz kommt. ML ist demzufolge eine Methode, um künstliche Intelligenz zu erreichen. Deep Learning wiederum ist eine Form des Machine Learnings mit biologischem Ursprung. Sinngemäß verhalten sich Künstliche Intelligenz, ML und Deep Learning zueinander wie Matrjoschkas.

3.2 Arten des Machine Learnings

Im Allgemeinen kann laut Raschka und Mirjalili [68] zwischen drei Arten des maschinellen Lernens unterschieden werden. Sie differenzieren sich fundamental in der Form des Lernens und in der Verarbeitung der Daten, wobei sich im Laufe der Zeit weitere Subkategorien herauskristallisiert haben, welche in Abbildung 3.1 zusammengefasst sind. Daraus wird ersichtlich, dass sich viele Unterarten im Überbegriff “Machine Learning” befinden. Die Wahl des geeigneten Algorithmus hängt vom Anwendungsfall ab und muss individuell entschieden werden.

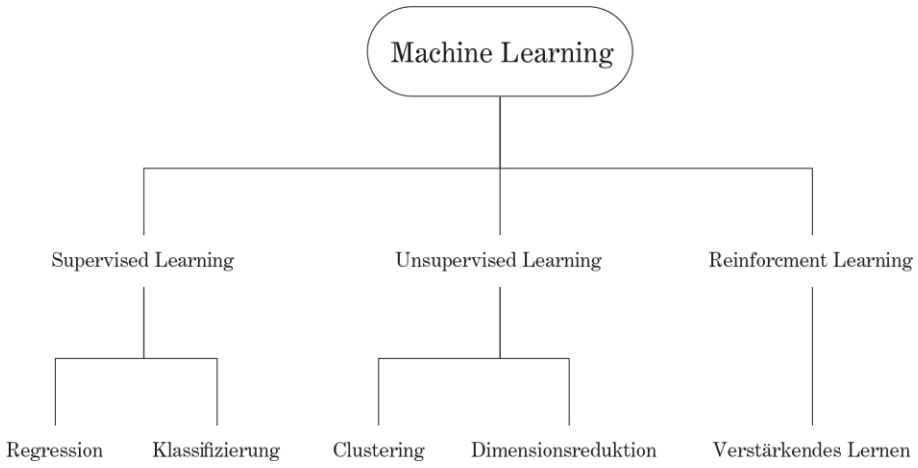


Abb. 3.1: Überblick über die Machine Learning Modelle

Überwachtes Lernen

Beim überwachten Lernen(„Supervised Learning“) werden Daten mit Label, welche auf die erwünschten Ergebnisse hinweisen, verarbeitet. Das bedeutet, dass in einem vorausgehenden Arbeitsschritt die zu erzielenden Resultate an einem begrenzen Datenset vormodelliert werden. Dadurch kann das System den Soll-Zustand erlernen. Solche Modelle können Aussagen über unbekannte, aber verwandte Daten treffen und zukünftige Sachverhalte vorhersagen.

Unüberwachtes Lernen

Die Daten, welche beim unüberwachten Lernen ("Ùnsupervised Learning“) verwendet werden, sind nicht gelabelt und enthalten keine Zielkonfiguration. Das bedeutet, dass diese Modelle hauptsächlich zum Auffinden von Strukturen und Muster innerhalb der Daten verwendet werden.

Reinforcement Learning

Modelle, die zum Reinforcement Learning zählen, bilden immer einen Entscheidungsvorgang ab. Es werden Aktionen erlernt, die durch ein Belohnungssystem trainiert werden. IMBs Deep Blue ist ein Projekt, welches mit Reinforcement Learning erstellt wurde.

3.2.1 Wie lernt ein Computer?

Goodfellow [22, S.108] beschreibt 2018 den Prozess als folgenden:

> *„Ein maschineller Lernalgorithmus lernt über eine Erfahrung, welche über eine Aufgabe und die dabei entstehende Performance definiert wird. Erfahrung und Performance variieren je nach Zyklus, die Aufgabe bleibt gleich und wird im allerersten Schritt definiert“.*

Die Aufgaben werden durch den:die Nutzer:in bestimmt. Basierend darauf wird ein passendes Modell gewählt [31]. Die Art des Lernens wird folglich vom Modell und dem sich darunterliegenden Algorithmus bestimmt. Der Algorithmus beinhaltet alle Befehle, die für das „Lernen“ essenziell sind und kann individuell angepasst werden. Es handelt sich hierbei um eine exakte Anleitung für die benötigten Prozesse. Schon 1956 wurde künstliche Intelligenz sehr ähnlich definiert. Ein Computer erzielt als Ergebnis jedoch immer nur Wahrscheinlichkeiten, zumal sich die Prozesse auf statistische und mathematischen Verfahren stützen. Folglich „lernt“ ein Rechner nicht, sondern er rechnet wie bereits im Namen steckt.

3.2.2 Künstliche Neuronen

Häufig wird beim maschinellen Lernen von Neuronen gesprochen, welche Informationen weitertragen. Die Neuronen im ML sind den biologischen Neuronen im Hirn nachempfunden und funktionieren schematisch sehr ähnlich [68]. Die Nervenzellen im menschlichen Gehirn sind miteinander verknüpft und leiten Signale untereinander weiter. Vergleichbar mit dem Binärcode in der Computertechnik, welcher nur die Werte 0 und 1 annehmen kann, kann eine biologische Nervenzelle, wenn sie einen Schwellenwert übertrifft, „aktiv“ oder „inaktiv“ sein. Ein Neuron braucht sowohl in der Biologie, als auch in der Computertechnik ein Aktivierungssignal.

In der biologischen Nervenzelle wird die Signalweiterverarbeitung über chemische Neurotransmitter gesteuert und beim maschinellem Lernen kommen unterschiedliche Schwellenwertfunktionen zum Einsatz. Hier handelt es sich um eine mathematische Funktion, welche im Abschnitt 3.1 definiert wird. Daraus kann abgelesen werden, dass die Funktion je nach Eingabe „v“ nur die Werte 0 und 1 annehmen kann. Ist die Eingabe „v“ kleiner Null, kommt es zu keiner Übermittlung von Information, bei einer Eingabe größer oder gleich null, wird das künstliche Neuron aktiviert.

$$\varphi^{\text{hlim}}(v) = \begin{cases} 1 & \text{bei } v \geq 0 \\ 0 & \text{bei } v < 0 \end{cases} \tag{3.1}$$

Neuronale Netze sind das Rückgrat des Deep Learnings. Über miteinander verbundene künstliche Nervenzellen wird ein lernfähiges Schichtenkonstrukt geschaffen, welches komplexe Probleme lösen kann. Das vom Rechner lesbare System lernt aus Fehler und verbessert sich ständig selbst. Ähnlich des menschlichen Lernprozesses, können aus den dabei entstandenen Erkenntnissen zukünftige, verwandte Probleme gelöst werden.

3.2.3 Deep Learning

Mit dem Beginn der künstlichen Intelligenz trat für den Menschen aus mathematischer und programmiertechnischer Sicht ein schwierig zu lösendes Problem ein, das für den Computer jedoch leicht zu verstehen war. Es handelte sich hauptsächlich um große Datenmengen, die sortiert, gefiltert oder extrahiert werden mussten. KI wurde folglich für die „Masserverarbeitung“ eingesetzt, um das menschliche Versagen zu eruieren. Komplexe Aufgaben, zum Beispiel Informationen aus Bild- und Audiodateien zu extrahieren, war aus damaliger Sicht aufgrund einiger Faktoren, insbesondere durch die fehlenden technischen Kenntnisse, unmöglich.

Deep Learning beschäftigt sich genau mit diesen Aufgaben, die für den menschlichen Verstand leicht zu lösen sind, aber für Rechner eine große Rechenleistung und Herausforderung darstellen. Bereits Kleinkinder können ohne große Anstrengung Objekte benennen und Rückschlüsse über deren Position, Lage und Zusammenhänge ziehen. Dem Computer muss hingegen im ersten Schritt das zu erkennende Objekt antrainiert werden. Hierbei handelt es sich um einen sehr aufwendigen Prozess, der über mehrere Stunden, teils Tage, verläuft. Sinnbildlich ist Deep Learning eine Art Blackbox, in welcher sich der Rechner selber die Regeln für die Aufgabenlösung erstellt. Im Deeplearning Handbook von Goodfellow [22] werden die Arbeitsschritte eines tiefen neuronalen Netzes ausführlich beschrieben und, soweit möglich, deren interne Zusammenhänge verdeutlicht.

Ein Deep Learning Modell besteht aus vielen unterschiedlichen Ebenen, die jeweils verschiedene Arbeitsschritte erledigen. Das Alleinstellungsmerkmal sind die „Hidden Layers“ in welchen die Prozesse uneinsichtig und in den meisten Fällen auch für den Menschen nicht mehr nachvollziehbar sind. Diese Verfahren sind dementsprechend auch sehr GPU-lastig, da massenweise Informationen über komplexe mathematische Funktionen verarbeitet werden.

Anwendungsfälle, welche Deep Learning Modelle einsetzen, sind beispielsweise die Gesichts- und Spracherkennung, die heutzutage in jedem Smartphone zu finden sind. Die Firma Tesla verwendet Deep-Learning-Modelle für die Erstellung des Autopiloten. Bei der Autofahrt werden Daten gesammelt, durch die das System weiter lernt und folglich die Vorhersagen effektiver und genauer werden.

3.3 Trainingsprozess

In diesem Abschnitt wird der Training-Prozess eines convolutionalen neuralen Netzwerk (CNN) mit besonderem Fokus auf die Bildsegmentierung in simplen Schritten theoretisch aufgearbeitet. Dieser Prozess ist der maßgebende Arbeitsschritt in der Entwicklung eines neuronalen Netzes. Wichtige Begriffe werden in diesem Kapitel aufgegriffen und deren Bedeutung erläutert. Dieser Abschnitt stellt die Basis für die Methodologie der Durchführung dar.

Das Training eines neuronalen Netzwerkes beschreibt einen Prozess, in welchem Parameter optimiert werden. Die gewichteten Parameter W und Biases werden festgelegt, um die Ergebnisse der unbekannten Funktion f zu optimieren. Hierfür wird eine Verlustfunktion, Backpropagation (auch Fehlerrückführung genannt), Kostenfunktion und optimierte Gewichte benötigt. In den folgenden Abschnitten werden zudem auch die Prozesse beschrieben, die zu den genannten Funktionen führen.

3.3.1 Erstellung der Datensets

Für die Erstellung eines neuronalen Netzes und dessen Validierung und Testung sind drei unterschiedliche Datensets notwendig:

- Trainingset,
- Validierungsset und
- Testset.

Alle Datensets enthalten Daten, die ein oder mehrere signifikante Merkmale aufweisen, welche bei der Aufbereitung der Daten in die benötigte Form extrahiert werden. Dieses Verfahren wird als Merkmalsextraktion oder Preprocessing bezeichnet. Zheng und Casari[95] definieren den Begriff „Merkmal“ als eine numerische Darstellung der Rohdaten. Da nicht alle Produkte einer Untersuchung, Messung oder Testung numerisch dargestellt werden können, beschränken sich „Merkmale“ in der angewandten Informatik auf Bilder und Sprach- und Musikdaten. Es existiert jedoch keine universelle Methode Merkmale zu extrahieren, da jede Dateiendung, wie z. B. „.jpg“, Merkmale auf eine andere Art und Weise abspeichert, die wiederum auf den Dateityp angepasst extrahiert werden müssen.

Zheng und Casari [95] prägen auch den Begriff der „Merkmalskonstruktion“. Dieser bezeichnet einen Vorgang, welcher die Merkmale konstruiert, die sich, stützend auf den Daten und das gewählte Modell, am besten zur Lösung der Aufgabe eignen. Merkmale werden in der Informationstheorie benötigt, um eine Mustererkennung zu erleichtern. Basierend auf zuvor erlernten Merkmalen ist es möglich in verwandten Daten ebendiese wiederzuerkennen.

Der gängigste Datensplit ist das klassische „7-2-1-Splitting“ [22]. Hierbei werden 70 % der Daten zum Training, 20 % zum Testen und 10 % zum Validieren des neuronalen Netzes verwendet. Häufig wird jedoch aufgrund der Informationslage auf individualisierte Aufteilungen zurückgegriffen. Bei einer sehr kleinen Datenmenge kann das Kreuzvalidierungsverfahren angewendet werden [87]. 70 % – 80 % der Inputs werden für das Training, der Rest für die Kreuzvalidierung verwendet.

Anschließend werden k Durchläufe definiert, welche die Wiederholungen des Test-Training-Verfahrens angibt. $k = 10$ bedeutet dementsprechend, dass der Prozess 10 Mal durchgeführt wird. Im Anschluss wird zur Beurteilung der Modellleistung der Durchschnitt der Durchläufe berechnet.

3.3.2 Aktivierungsfunktionen

Dem Vorbild von biologischen Gehirnneuronen nachempfunden, braucht ein künstliches Neuron in einem CNN ein Aktivierungssignal, damit die Information der Input-Ebene in die versteckten Ebenen Convolutionale und Pooling-Ebene gelangen kann [78]. Im einfachsten Fall werden die Inputinformationen mit einem Gewicht, welches die Wichtigkeit darstellt, multipliziert und anschließend eine Bias angefügt. Die Bias beschreibt die „Fairness“ eines abgegrenzten Prozesses. Sie kann herbeiführen, dass ein Neuron aktiviert wird, obwohl der Wert einer Funktion gleich null ist, um alle Neuronen in einem Netzwerk miteinzubeziehen. Abschließend wird die Aktivierungsfunktion auf das Resultat angewendet und das Ergebnis, wenn es die definierte Bedingung erfüllt, an die nächste Ebene weitergegeben. Diese Prozesse sind uneinsichtig und werden daher auch Hidden Layer oder Hidden Processes genannt. Dieser Prozess wird so oft wiederholt bis die letzte Ebene, auch Output Layer genannt, erreicht wird. Die gebräuchlichsten Aktivierungsfunktionen namengebend für die Ebenen sind: ReLU, tanh, SoftMax und Sigmoid. In der Regel gibt es innerhalb einer Netzwerkarchitektur mehrere Aktivierungsebenen, die jeweils unterschiedliche Aktivierungsfunktionen beinhalten. Nur Aktivierungsebenen können die Aktivierungsfunktion enthalten. Somit können innerhalb eines neuronalen Netzwerkes alle genannten Funktionen vorkommen.

Sigmoid ist die älteste Aktivierungsfunktion und kann grafisch aus Abbildung 3.2 entnommen werden. Aus dem Graphen ist ablesbar, dass es sich bei der Sigmoidfunktion um eine verschobene Tangens-Hyperbolicus-Funktion handelt. Aus Formel 3.2 lässt sich erkennen, dass Sigmoid differenzierbar ist, wodurch sie den Einsatz von Backpropagation ermöglicht.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.2}$$

ReLU [41] wird am häufigsten in neuronalen Netzen eingesetzt, da es in seiner Funktionsweise sehr simpel ist. Die Abbildung 3.3 zeigt, dass es sich hierbei um eine einfache lineare Funktion handelt, die positive Werte weitergibt, und negative Werte auf null verändert. Folgend kann ReLU mathematisch definiert werden:

$$R(z) = \max(0, z) \tag{3.3}$$

3.3.3 Kostenfunktion

Gu et al. [23] beschreiben mit einer Reihe an Experimenten die Wichtigkeit der Optimierung von Gewichten und Biases in einem neuronalen Netz, um die Genauigkeit der Ergebnisse eines Modells zu verbessern. Die Kostenfunktion ist ein Maß der Qualität des Modells, welches eine Art Belohnungssystem zur Folge hat. Das bedeutet, dass gute Ergebnisse in Form von hohen Wahrscheinlichkeiten belohnt werden, indem die angenommenen Gewichte und Biases beibehalten werden. Die Kostenfunktion ist eine Funktion, die den Fortschritt eines Optimierungsalgorithmus misst. Sie wird verwendet, um das optimale Ergebnis einer Optimierung zu finden. Sie ergibt sich aus der Kreuzentropie zwischen dem Trainingsinput und der Vorhersage eines Modells. Die Kreuzentropie stammt von der Maximum-Likelihood-Methode ab, welche am häufigsten in

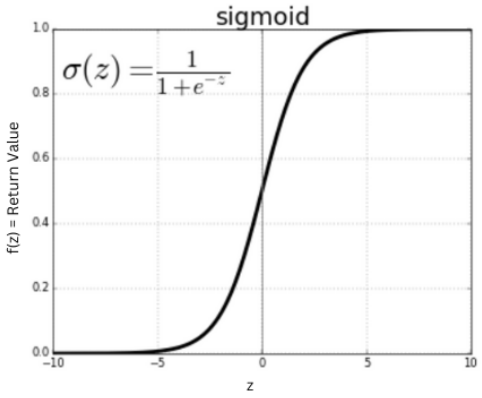


Abb. 3.2: Graph einer sigmoid Aktivierungsfunktion [11]

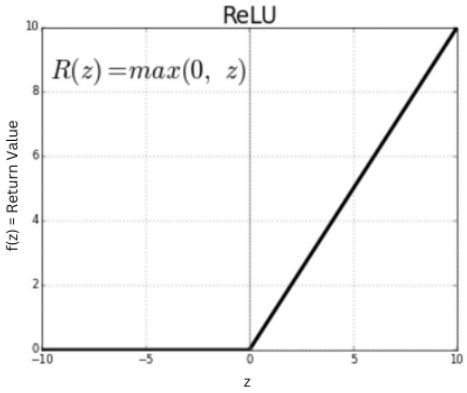


Abb. 3.3: Graph einer ReLU Aktivierungsfunktion [11]

neuronalen Netzen eingesetzt wird [23]. Im Allgemeinen wird das Gradientenverfahren verwendet, um die Kostenfunktion zu minimieren, indem es an ihr entlang absteigt.

Maximum-Likelihood-Methode

Rubinstein und Kroese [61] beschreiben die Maximum-Likelihood-Methode als parametrisches Schätzverfahren, welches aus Stichproben die Hilfsvariablen der Grundgesamtheit ermittelt. Es werden die Parameter ausgewählt, welche am wahrscheinlichsten zur maximalen Chance, sprich zur höchsten Wahrscheinlichkeit, führen. Mathematisch bedeutet das, dass n Werte $X = \{x_1, x_2, \ldots, x_n\}$ ausgewählt werden, welche wahr sind, jedoch eine unbekannte Wahrscheinlichkeitsverteilung $p_{data}(x)$ aufweisen. Die parametrische Menge, welche die Wahrscheinlichkeitsverteilung von θ klassifiziert, wird folgend beschrieben: $p_{model}(X;\theta)$. θ stellt die Gewichte und Biases des Modells dar. Basierend darauf kann die maximale Wahrscheinlichkeit von θ mathematisch ausgedrückt werden:

$$\theta_{ML} = \underset{\theta}{\operatorname{argmax}}\, p_{\text{model}}\,(X;\theta) \tag{3.4}$$

Formel (3.4) kann mit Hilfe von $p_{data}(x)$ in Gleichung (3.5) umgeformt werden.

$$\theta_{ML} = \underset{\theta}{\operatorname{argmax}} \mathbb{E}_{x\sim\hat{p}_{\text{data}}}\ [\log p_{\text{model}}(x;\theta)] \tag{3.5}$$

Kreuzentropie

Schematisch beschreibt eine Kreuzentropie die Ähnlichkeit zweier Wahrscheinlichkeitsverteilungen $P(x)$ und $Q(x)$, welche aus der Gleichung (3.6) entnommen werden kann. Der Begriff stammt aus der Statistik, wie auch Informatik und beschreibt ein Maß für die Genauigkeit einer Wahrscheinlichkeitsvorhersage. Dies haben bereits Rubinstein und Kroese [61] sinngemäß ident hergeleitet.

$$H(P,Q) = -\mathbb{E}_{x\sim P} \log Q(x) \tag{3.6}$$

Da die Kreuzentropie von der Maximum-Likelihood-Methode abstammt und im Grunde genommen eine negative Logarithmus Wahrscheinlichkeitsfunktion ist, kann Formel (3.6) mit der Formel (3.5) kombiniert werden:

$$J(\theta) = -\mathbb{E}_{x,y\sim\hat{p}_{\text{data}}}\ [\log p_{\text{model}}\,(y \mid x)] \tag{3.7}$$

3.3.4 Gradientenverfahren

In der Numerik und auch bei der Erstellung von neuronalen Netzen wird das Gradientenverfahren eingesetzt, um Optimierungsprobleme zu überwinden [89]. Die Parameter werden im Trainingsprozess optimiert und versuchen hierbei das globale Minimum der Funktion zu erreichen. Hierfür wird ein Algorithmus eingesetzt, welcher von einem zuvor definierten Startpunkt aus, meist θ, das globale Minimum zu finden versucht. Bei diesem Vorgang werden in absteigender Richtung schrittweise Berechnungen durchgeführt, bis keine Optimierung mehr festgestellt werden kann. Dieser Vorgang wird schematisch in Abbildung 3.4 dargestellt. Das Ziel hierbei ist, von θ aus einen neuen Punkt θ' zu generieren, welcher möglichst nahe am globalen Minimum liegt. Dadurch werden die Gewichte in Bezug auf die Kostenfunktion optimiert.

Abb. 3.4: Schematische Darstellung des Gradientenverfahren

Da in der Realität jedoch weitaus komplexere Funktionen zum Einsatz kommen (siehe z. B. Abbildung 3.5), kann nicht garantiert werden, dass das globale Minimum gefunden wird. Die meisten Algorithmen, die bereits als Befehle in externen Bibliotheken angelegt sind, können jedoch Aussagen über das Erreichen eines anderen kritischen Punktes treffen, z. B. ein lokales Minimum.

Backpropagation

Bei der Backpropagation (dt. Fehlerrückführung) handelt es sich um eine Methode der Fehlerrückführung [22]. In Abbildung 3.6 wird die Funktionsweise schematisch dargestellt. Es ist

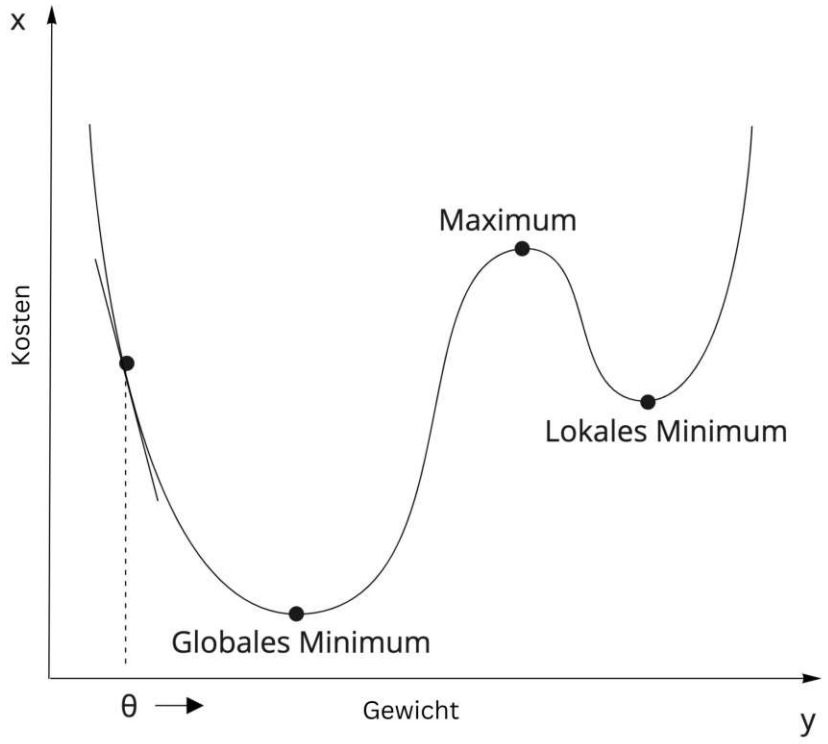


Abb. 3.5: Schematische Darstellung der kritischen Punkte während des Gradientenverfahren

eine Verfahrensweise des überwachten Lernens. Hierfür wird eine exakte Definition des Fehlers benötigt, welcher über die Verlustfunktion, oder auch Fehlerfunktion, errechnet wird (siehe Abschnitt 3.3.5). Mathematisch kann die Fehlerrückführung als eine Unterscheidung zwischen Realwerten und Vorhersage gesehen werden:

$$e_O = O_{\text{real}} - O_{\text{Vorhersage}} \tag{3.8}$$

In jeder Epoche eines Trainings werden die Gewichte basierend auf der Prognose der vorigen Epoche, bzw. der Wahrscheinlichkeit der Richtigkeit der Ergebnisse, angepasst [22]. Eine Epoche ist ein Durchlauf durch das gesamte Trainingstdataset in der maschinellen Lernverarbeitung. Die Vorhersage ist ein vorwärsgerichteter Prozess (foreward), die Gewichtsanpassung ein rückwärtsgerichteter (backwards). In der allerersten Epoche werden die Parameter wahllos gesetzt, da im nächsten Schritt das Gradientenverfahren angewendet wird.

Um die Parameter zwischen Input und Output Ebene angemessen anpassen zu können, muss der Fehleranteil berechnet werden. Neuronale Netze beinhalten mehrere versteckte Ebenen, durch die sich der Prozess als sehr aufwendig gestalten kann – Je mehr Ebenen, desto komplizierter und komplexer der Prozess. Generalisiert kann der Fehleranteil mathematisch aus Gleichung (3.9) entnommen werden. Es ist ersichtlich, dass es sich um eine Kettenfunktion mit w_n Gewichten und e_n Fehlern handelt.

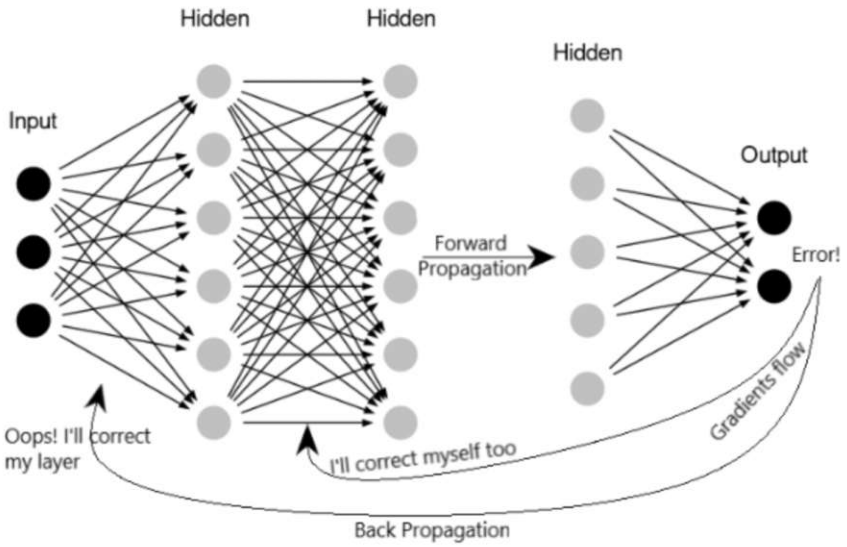


Abb. 3.6: Schematische Darstellung der Funktionsweise von Backpropagation [57]

$$e_N = \begin{pmatrix} w_{1,1} & w_{1,n} \\ w_{2,1} & w_{2,n} \end{pmatrix} \cdot \begin{pmatrix} e_1 \\ e_n \end{pmatrix} \tag{3.9}$$

Die Kurzform der Formel (3.9) lautet: $e_N = w^T \cdot e_o$. Bei w^T handelt es sich um die Gewichte potenziert mit der Anzahl der Layerebenen. e_o beschreibt den errechneten Fehler, ergo den Unterschied zwischen Realwerten und Vorhersage. Beim Scripting eines neuronalen Netzes übernimmt der gewählte Optimizer die Funktion der Backpropagation.

3.3.5 Verlustfunktionen

Bei einem Trainingsprozess muss am Ende der abgeschlossenen Epoche der Unterschied zwischen dem erzielten Ergebnis und dem Original berechnet werden. Dies wird mittels einer Verlustfunktion erreicht. Laut Jadon [32] hängt die Wahl der richtigen Verlustfunktion stark mit dem Anwendungsfall und der damit verbundenen Art der Klassifizierung oder Segmentierung zusammen. Überdies hat die Entscheidung der richtigen Verlustfunktion unmittelbare Auswirkungen auf die Ergebnisse. Die Art des Anwendungsfalles und der dazugehörigen Verlustfunktionen kann aus der Tabelle 3.1 entnommen werden. Da es sich in dieser Arbeit um eine Bildsegmentierung handelt, wird mit einem „Region-based Loss“ gearbeitet. Diese wird sehr häufig auch in der Medizin bei Segmentierung von Bild-basierten Befunden verwendet, beispielsweise die Erkennung und Verortung von Tumorzellen,. Regionalbasierte Verlustfunktionen versuchen die vorhergesagte und vom neuronalen Netz generierte Segmentierung basierend auf dem originalen Input zu maximieren. Die Herleitung und Wahl der finalen Verlustfunktion kann im Kapitel 6 nachgelesen werden.

Irrtümlicherweise werden Verlust-, Aktivierungs- und Kostenfunktion häufig miteinander vertauscht. In Abbildung 3.7 werden die Zusammenhänge im Kontext des Trainingsprozesses dargestellt. Durch die Aktivierungsfunktion wird festgelegt, ob ein Neuron aktiviert wird und ein Prozess aktiviert wird. Die Verlustfunktion berechnet die Fehler für einen einzigen Trai-

Tab. 3.1: Verlustfunktionen kategorisiert nach Art der Anwendung

Art	Verlustfunktionen
Compounded Loss	Combo Loss
	Exponential Logarithmic Loss
Region-based Loss	Dice Loss
	Sensitivity-Specificity Loss
	Tversky Loss
	Focal Tversky Loss
	Log-Cosh Dice Loss
Boundary-based Loss	Hausdorff Distance loss
	Shape aware loss
Distribution-based Loss	Binary Cross-Entropy
	Weighted Cross-Entropy
	Balanced Cross-Entropy
	Focal Loss
	Distance map derived loss penalty term

ningsdurchlauf. Die Kostenfunktion ermittelt den Durchschnitt aller Verlustfunktionen in einem ganzen Trainingsprozess.

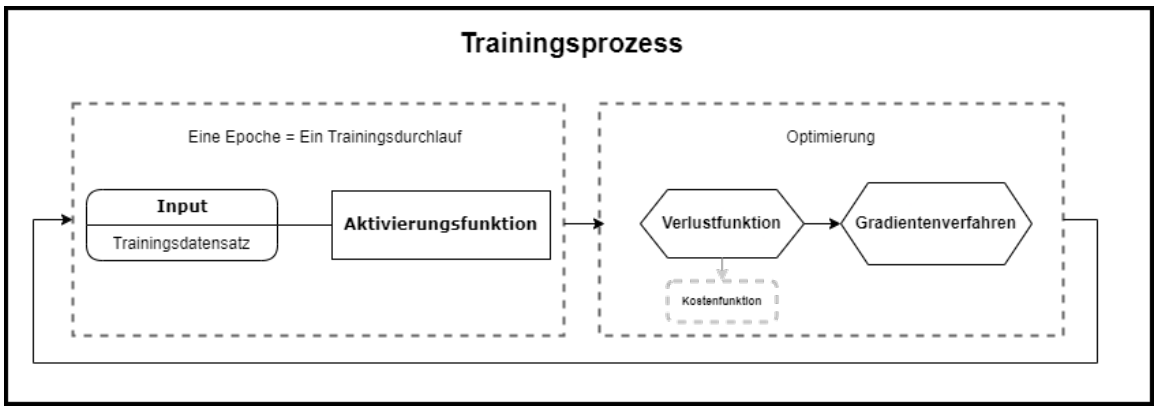


Abb. 3.7: Schemahaftes Ablaufdiagramm eines Trainingsprozesses

Sowohl Verlust- als auch Aktivierungsfunktionen werden über die Keras Tensorflow API abgedeckt [78]. Über Befehle können diese in den Algorithmus eingebaut und bei Bedarf ausgetauscht werden. Somit müssen die einzelnen Prozesse nicht eigens algorithmisch beschrieben werden, da sie bereits vordefiniert sind.

Kapitel 4

Convolutionales Neurales Netzwerk

Ein CNN [22] ist ein neuronales Netzwerk, welches mindestens eine convolutionale Ebene enthält. Die genannte Layer (Ebene) verwendet eine Faltung (Convolution), welche über mathematische Funktionen erreicht wird, um Merkmale aus den Eingabedaten zu extrahieren. Es ist besonders gut für die Verarbeitung von Bilddateien geeignet, da diese rasterartig aufgebaut sind. Auch andere Daten, die in einer 3D-Matrix angeordnet sind, z. B. Audiodateien, Sprachaufnahmen und Audiosignale, können mit Hilfe eines CNN verarbeitet werden [47].

Bereits 1987 wurde im Rahmen des „Neural Information Processing Workshops“ in Denver, Colorado [5] an Spracherkennung mit Hilfe eines Neocognitron gearbeitet. Es handelt sich hier um einen Vorläufer des heute bekannten CNN und ist ein hierarchisch aufgebautes, mehrschichtiges, künstliches neuronales Netz. LeCun gilt offiziell als der Begründer und Erfinder der Computer Vision und dem sich daraus entwickelte CNN. Mit einem System, welches automatisch die Postleitzahl aus Bildern auslesen kann, bildete der französische Informatiker in 1989 eine wichtige Basis für weitere Forschungen [13]. Im Zuge seiner Ermittlungen wurde 1998 laut Lecun et al. [36] das LeNet-5, ein simples sieben-Ebenen CNN, geboren und es wurde lange Zeit zur Erkennung von handgeschriebenen Zahlen verwendet.

Nach vielen Jahren der Stille in der Forschung gewann 2012 Krizhevksy et al. [35] mit AlexNet die ImageNet Challenge. Erst in 2015 konnte das AlexNet vom effektiveren „Very Deep CNN“ von Microsoft, welches komplexer aufgebaut ist, in der Treffsicherheit übertrumpft werden.

In diesem Kapitel werden die Grundzüge des CNN erklärt. Zu Beginn wird das Konzept der Convolution erläutert und im weiteren Schritt die charakteristische Architektur beschrieben. Anschließend werden die Funktionsweisen der einzelnen Ebenen aufgefasst und erklärt. Darüber hinaus werden Strategien zur Regularisierung angeführt, welche die Prozesse optimieren und in der Verbesserung der Ergebnisse maßgeblich unterstützen. Abschließend wird die Bildsegmentierung mittels CNN erläutert, welche die grundlegende Technologie dieser Arbeit ist.

4.1 Convolution

Eine Convolution ist ein mathematischer Vorgang, welcher aus einem Integral zweier Funktionen f und g besteht. Jede Convolution ist eine kleine Matrix von Gewichten, die über das Eingabebild bewegt wird, um bestimmte Merkmale wie Kanten, Ecken, Farbverläufe, Texturen und andere relevante Eigenschaften zu erkennen. Ian Goodfellow et. al. [22] beschreiben den Vorgang einer Convolution im „Deeplearning Book“ anschaulich. Die nachfolgenden Formeln basieren auf den wegweisenden Anleitungen von Goodfellow. Eine der beiden Funktionen f und g ist verschoben und umgekehrt. Das bedeutet, dass eine der genannten Funktionen gespiegelt und entlang der y-Achse verschoben wurde. Eine Convolution wird als $f * g$ ausgeschrieben und folgend definiert:

$$s(t) = (f * g)(t) = \int f(a)g(t-a)da \tag{4.1}$$

Für die GLeichung (4.1) müssen mehrere Faktoren gegeben sein. Zum einen muss die Funktion g eine gültige Wahrscheinlichkeitsdichtefunktion sein, da sie sonst keinen gewichteten Durchschnitt ergibt. Zum Anderen muss gegeben sein, dass für alle negativen Inputs $g = 0$ ist. Eine Wahrscheinlichkeitsdichtefunktion beschreibt eine Verteilung um einen beliebigen Wert und gibt somit die Dichte um einen Punkt an.

Daten werden im Computer abgetastet und diskretisiert. Das bedeutet, dass Funktionen sich einer Menge an Zahlen bedienen, welche endlich ist oder die Menge N der natürlichen Zahlen ist. Das Ergebnis der Funktion kann aus diesem Grund nur einen bestimmten Wert annehmen. Die Funktion (4.1) kann folglich in eine diskrete Form gebracht werden, welche in Formel (4.2) ablesbar ist.

$$s(t) = (f * g)(t) = \sum_{a=-\infty}^{\infty} f(a)g(t-a) \tag{4.2}$$

In einem CNN ist f der Input und g eine gewichtete Funktion, der Kernel. Gemeinsam ergeben sie den Output, welcher auch „Feature Map“ genannt wird.

In einer Convolution bzw. Faltung können mehrere Achsen zum Prozess beitragend sein. Da diese Arbeit 2D-Bilder bearbeitet, nimmt der Kernel K 2 Dimensionen an. Der Input B ist das 2D-Bild. Daraus kann folgende Formel abgeleitet werden:

$$S(i,j) = (B * K)(i,j) = \sum_m \sum_n B(m,n)K(i-m, j-n) \tag{4.3}$$

Eine Convolution ist ein kommunikativer Vorgang, daher ist es möglich Formel (4.3) in Gleichung (4.4) zu adaptieren. Die Verwendung ist kongruent und je nach Anwendungsfall und Datenlage zu entscheiden.

$$S(i,j) = (K * B)(i,j) = \sum_m \sum_n B(i-m, j-n)K(m,n) \tag{4.4}$$

Häufig wird in einem CNN keine Convolution, sondern eine Kreuzkorrelation verwendet, welche mit der Faltung eng verwandt ist. Der Unterschied liegt darin, dass der Kernel nicht umgedreht wird, welches den kommunikativen Charakter der Funktion ausmacht. Convolutionen werden in der Mathematik häufig für Beweise verwendet, da der kommunikative Part aufrechterhalten wird. Dieser ist jedoch in einem neuronalen Netzwerk nur nebensächlich und kann vernachlässigt werden. Eine Umformung zu einer Kreuzkorrelation ergibt folgende Formel:

$$S(i,j) = (K * B)(i,j) = \sum_m \sum_n B(i+m, j+n)K(m,n) \tag{4.5}$$

4.2 Grundidee eines CNN

Vergleichend mit anderen neuronalen Netzen, findet auch das CNN sein Vorbild in der Natur. Hubel und Wiesel [28] gewannen 1980 den Nobelreis in der Kategorie „Physiologie oder Medizin“ für ein Experiment an Katzen. Durch das Experiment konnte nachgewiesen werden, dass bei visueller Stimulation einzelne Neuronen in der Großhirnrinde aufleuchten. Diese Neuronen wurden „Rezeptive Felder“ genannt und es konnte auch beobachtet werden, dass angrenzende Neuronen ähnliche Verhaltensweisen aufweisen. Es ist ihnen auch gelungen nachzuweisen, dass simplere Bilder, z. B. einfache Punkte oder Linien, im frühen visuellen Cortex nachgebildet werden. Je komplexer das Bild, desto länger der visuelle Weg im Hirn. Mazzoni et al. [40] beschreiben, dass dieser Prozess auch auf ein CNN in Grundzügen umlegbar ist. Je komplexer der Input,

desto komplexer und „tiefer“ das neuronale Netz. Das bedeutet, dass eine Erkennung von Objekten auf Bildern ein weniger tiefes neuronale Netz benötigt, als eine technisch aufwendigeren Bildsegmentierung.

4.3 Charakteristische Architektur eines CNN

Der Aufbau eines CNN ist, schematisch dargestellt, ein simpler. Der Input umfasst Bilder, die in eine Pixelmatrix aufgebrochen werden. Die Matrix hat üblicherweise drei Dimensionen, die sich in zwei Graustufenebenen und eine Farbebene aufteilen. Anschließend werden die Informationen über mehrere spezifische Ebenen (engl. Layer)(siehe Abbildung 4.1) hinweg und über verschiedene Prozesse analysiert und aufgearbeitet.

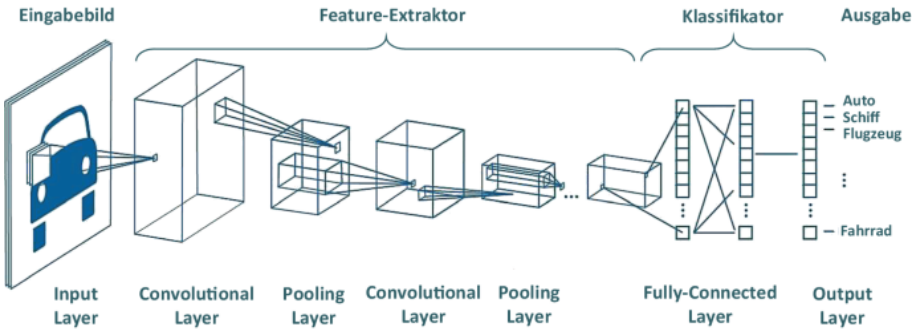


Abb. 4.1: Charakteristischer Aufbau eines CNN [97]

Die Namensgebung der Ebenen ist dem darin durchlaufenden Prozess nachempfunden und lauten folglich:

- Input Ebene,
- Convolutions Ebene,
- Pooling Ebene,
- Vollständig verbundene Ebene und
- Output Ebene.

Einfache neuronale Netzwerke sind komplett verbunden. Das bedeutet, dass alle linearen Einheiten einer Ebene mit allen linearen Einheiten der nächsten Ebene verbunden sind. Bei einem CNN ist dies nicht der Fall, daher werden sie auch teil- bzw. partiell verbundene neuronale Netzwerke genannt. Es handelt sich hierbei um ein sehr wichtiges Charakteristikum eines CNN. Eine lineare Einheit gibt deren Information in diesen Modellen nur an manche Teile der nächsten Ebene weiter. In Abbildung 4.2 ist die Funktionsweise eines partiell verbundenen Netzwerkes dargestellt und in Abbildung 4.3 die eines komplett verbundenen. Im CNN ist der Zusammenhang zwischen den Werten untereinander wichtiger, als eine ganzheitliche Vernetztheit. Darüber hinaus würde es aufgrund der vielen Ebenen und tiefen Netzwerkarchitektur zu sehr langen Trainingsintervallen kommen.

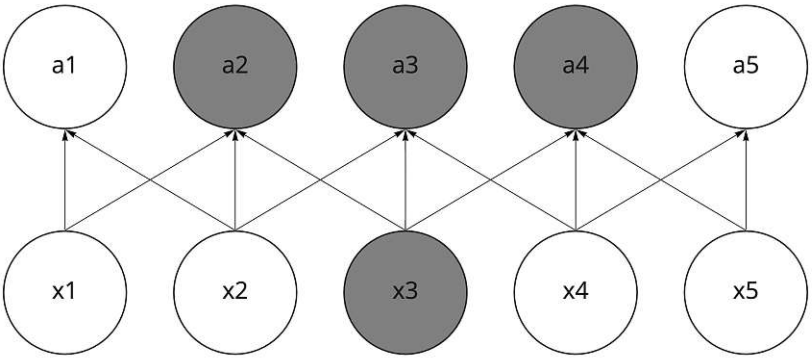


Abb. 4.2: Partiell verbundene neuronale Netze

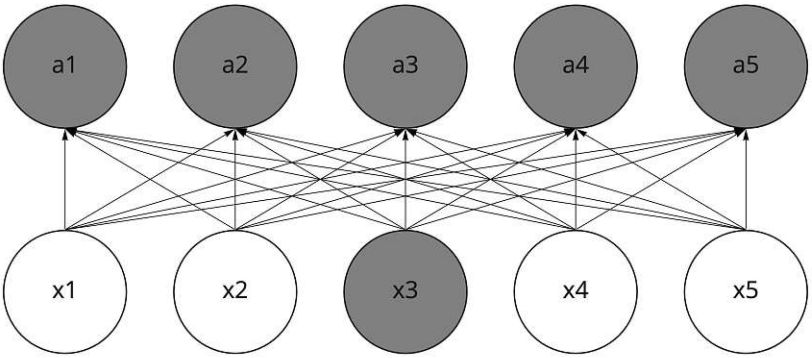


Abb. 4.3: voll verbundene neuronale Netze

4.3.1 Input und Output Ebene

Die Input-Ebene enthält alle notwendigen Eingaben für das neuronale Netzwerk. Oftmals wird auch von „Input and Output Shapes“ gesprochen, welche die Eingabedaten grafisch umschreiben sollen. In Abbildung 4.4 ist diese grafische Umschreibung dargestellt. In den meisten Fällen werden zur Erstellung einer Input-Layer die Bildbreite und -höhe, die Anzahl der Bilder (Batch) und die Anzahl der Farbkanäle benötigt. Die Anzahl der Kanäle kann über das Farbmodell, z. B. RGB = 3, CYMK = 4, herausgefunden werden.

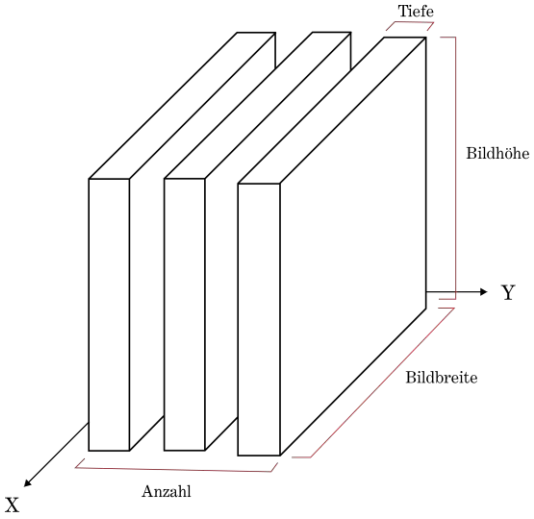


Abb. 4.4: Schematische Darstellung der Input-Ebene

4.3.2 Convolutionale Ebene

Die convolutionale Ebene ist das Kennzeichen und Unterscheidungsmerkmal eines CNN gegenüber anderen künstlichen Netzwerken. Sie ist auch der Namensgeber der Netzwerkarchitektur. Grundlegend werden in dieser Ebene die charakteristischen Merkmale extrahiert und ausgelesen. In diesem Prozess werden die Bilder komprimiert und über mehrere Filter neue Bilder erzeugt. Gesteuert wird dieses Verfahren von einer mathematischen Funktion, welche im Abschnitt 4.5 erklärt wird. Diese Filter führen zum Ergebnis eines Skalarprodukts, welches die Inputs der vorherigen Ebene komprimiert.

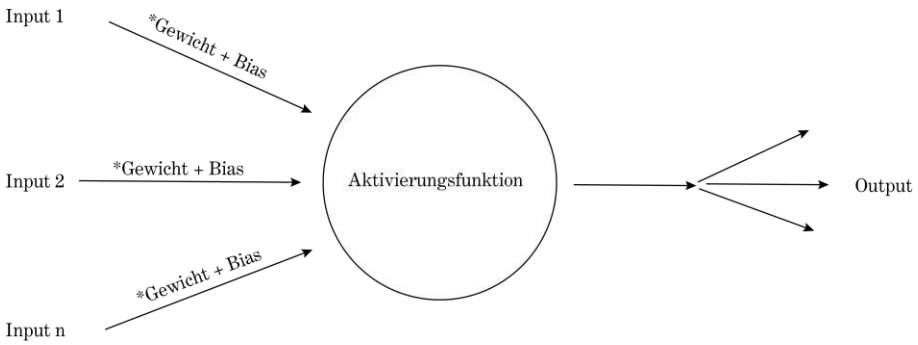


Abb. 4.5: Schematischer Aufbau eines künstlichen Neurons in der Aktivierungsebene [76]

4.3.3 Pooling Ebenen

In einem Auszug eines Datensatzes werden nicht immer alle Daten benötigt. Durch das Pooling können überflüssige Informationen ausgesiebt werden, um somit die Datengröße gering zu halten. Der Begriff bedeutet „zusammenlegen“ oder „vereinigen“ und beschreibt den grundlegenden Arbeitsprozess. Die Größe von Merkmalen kann durch das Pooling signifikant verringert werden, ohne dabei das Endergebnis zu beeinflussen. Hierdurch wird der Arbeitsprozess effizienter. Es wird jedoch empfohlen, Merkmale zu generieren, die nicht disponiert auf Störfaktoren in den Input-Daten sind, da durch ein breites RGB-Histogramm bzw. viele unterschiedliche Pixel ohne Zusammenhang die Resultate beeinflusst werden können.

In einem CNN können grundsätzlich zwei Arten von Pooling unterschieden werden: das Max-Pooling und das Mean-Pooling, welches auch Average-Pooling genannt wird. Typischerweise wird nur einer der beiden Typen in einem CNN verwendet.

Unterschied von Max-Pooling und Average-Pooling
Die Vereinigungsart ist bei beiden Varianten eine sehr unterschiedliche. Beim Max-Pooling wird der größte Wert behalten, was bedeutet, dass ein scharfes Bild erzeugt wird, welche nur die Maximalwerte der umliegenden Pixel enthält. Dieser Prozess kann in Abbildung 4.6 gesehen werden. Beim Average-Pooling wird der Durchschnittswert der umliegenden Werte bzw. Pixel errechnet und verwendet. Folglich entsteht ein sanftes Bild ohne große Extremwerte. Der Prozess kann grafisch anhand der Abbildung 4.7 erklärt werden. Die Ergebnisse der beiden Abbildungen 4.6 und 4.7 zeigen sehr gut den grundlegenden Unterschied der beiden Algorithmen auf.

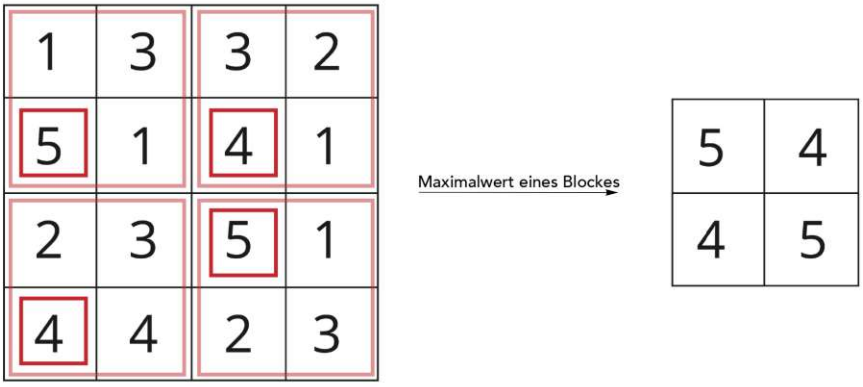


Abb. 4.6: Schematische Darstellung von Max-Pooling

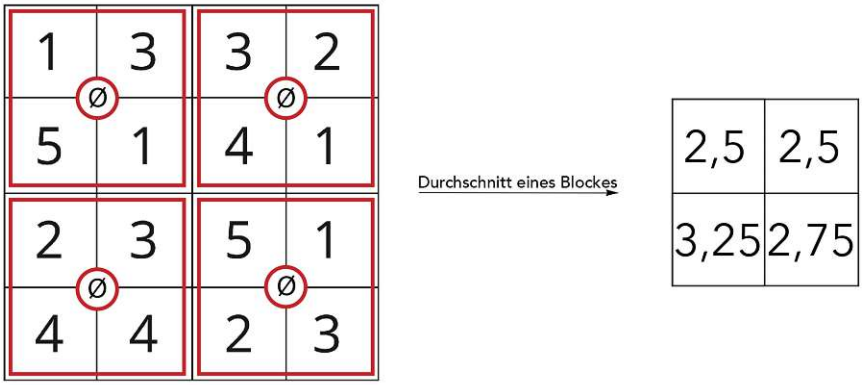


Abb. 4.7: Schematische Darstellung von Average-Pooling

4.3.4 Vollständig verbundene Ebene

Trotz der partiellen Verbundenheit der Ebenen innerhalb eines neuronalen Netzwerkes, ist es notwendig, alle Informationen noch vor der Output Ebene eines Algorithmus zu sammeln und zusammenzuführen. Dieser Vorgang ist erforderlich, um Vorhersagen treffen zu können. In der vollständig verbundenen Ebene werden die Informationen der vorhergehenden Ebene gesammelt und anschließend an die Output-Ebene weitergegeben.

4.4 Strategien der Regularisierung

In diesem Unterkapitel werden Methoden beschrieben, welche die Regularisierung eines neuralen Netzes unterstützen. Laut Raschka und Mirjalili [68] ist es eine enorme Herausforderung, die Größe eines Netzes und die Gewichtung aller Ebenen richtig zu bestimmen. Bei einem einfachen Netz ohne einer versteckten Schicht kann eine lineare Entscheidungsgrenze eingesetzt werden. Bei einem mehrschichtigen Netz, z. B. bei einem CNN, ist dies nicht ausreichend.

Eine Regularisierung des Netzes verhindert zum einen die Unteranpassung, welche zu einer geringeren Leistung und schlechten Ergebnissen führt, als auch die Überanpassung, die wiederum ebenfalls fehlerhafte Endergebnisse erbringt. Es gibt unterschiedliche Methoden ein Netz zu regularisieren. Die in den letzten Jahren jedoch am häufigsten eingesetzte ist das Dropout-Verfahren.

Regularisierung mit Dropout

Srivastava et al. [73] beschreiben in ihrem Paper, dass das Dropout-Verfahren die Generalisierungsfähigkeit verbessert, indem es während der Trainingsphase bei jeder Wiederholung, bzw. Iteration einen Teil der verborgenen Neuronen zufällig entfernt. Die Wahrscheinlichkeit entfernt zu werden, wird als p_{drop} und nicht entfernt zu werden als $p_{keep} = 1 - p_{drop}$ definiert. p_{drop} bzw. p_{keep} muss vom der:die Ersteller:in definiert werden. Die Gewichte der verbleibenden Neuronen werden nach dem Entfernen des Anteils neu berechnet. Infolgedessen erlernt das Netz aus robusten Mustern zu lernen und nicht basierend auf obsoleten Daten. Dies bezieht sich letztlich nur auf den Trainingsprozess, bei der Auswertung und Vorhersage werden alle Daten miteinbezogen.

Regularisierung mit verfrühtem Stopp

Der verfrühte Stopp (Englisch „Early Stopping“) ist laut Prechelt [53] eine häufig eingesetzte Methode zur Regularisierung, insbesondere wegen des zu vermeidenden Overfitting. Mit jeder Trainingsepoche sind Generalisierungserror rückläufig zu vermerken, nicht jedoch bei dem Validierungsset. Nach einer gewissen Anzahl an Epochen beginnen Error wieder häufiger aufzutreten. Dieses Phänomen kann aus Abschnitt 6.2.5 entnommen werden. Dies ist ein guter Indikator, dass das Modell an die Trainingsdaten überangepasst ist, dies wird als Overfitting bezeichnet.

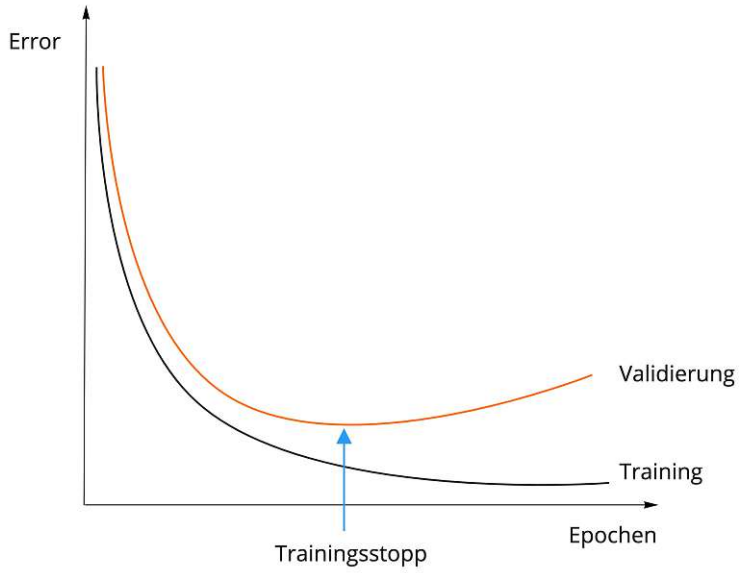


Abb. 4.8: Schematische Darstellung eines Generalisierungserror-Epochendiagramm

Die Grundidee liegt darin, bevor eine Überanpassung einsetzt, das Training frühzeitig abzubrechen. Dadurch wird auch das Ergebnis des Generalisierungserror optimiert. Um dies zu ermöglichen, wird gestoppt, wenn der niedrigste Wert des Validierungserror erreicht ist.

Datensatzerweiterung

Eine simple Möglichkeit, um eine Überanpassung zu verhindern, ist das Datenset zu erweitern [23]. Hierzu werden nicht neue Bilder benötigt, da bereits kleine Anpassungen, z. B. Rotation, Spiegelung, Translation von Pixeln und Skalierung der Bilder neue Daten darstellen. Dazugehörigen Labels werde nicht ausgetauscht. Die Daten innerhalb eines Datensatzes müssen auch repräsentativ für den gewählten Anwendungsfall sein. Exemplarisch kann ein großes Datenset mit Rissen zu schlechten Ergebnissen führen, wenn die enthaltenen Daten Verfälschungen darstellen, z. B. Schatten oder Insekten.

4.5 Bildsegmentierung mit CNN

Werner [86] beschreibt, dass sich ein convolutionales neurales Netzwerk bildliche Information, ähnlich wie der menschliche Verstand, aufgrund von Erfahrung und externem Input neue Eindrücke aneignet. Im Machine Learning wird hierzu das Convolutionale Neurale Netzwerk verwendet. Bilder bestehen aus einer definierten Anzahl an Pixel, die rasterartig angeordnet sind. Diese Pixel enthalten wiederum Subpixel, welche die Informationen für die Farbgebung in sich tragen. Dabei wird auch von Farbmodellen oder Farbraum gesprochen. Die bekanntesten und gängigen Vertreter sind RGB und CYMK, die den Subpixel numerische Werte verleihen.

Nach der Definition Aggarwal [1] arbeitet ein CNN mit Gitter-artigen Inputs, die je nach Merkmal fokussierte Abhängigkeiten in bestimmten Regionen haben. Bei einem Bild mit Riss sind Pixel mit Riss im Fokus. Diese Verbindung wird mit Hilfe einer Maske hergestellt, einer Schwarz-Weiß-Darstellung mit dem gewünschtem Objekt im Fokus. Sie stellen eine Art „Anleitung“ für das Training dar, womit das erstellte Netz Vorhersagen immer mit dem Original vergleichen kann. Der Vorteil an diesem Modell ist die Anordnungsunabhängigkeit aufgrund der Beziehung der Pixel untereinander. Das bedeutet, dass ein Riss erkannt wird, egal an welcher Stelle sich dieser im Bild befindet.

Aus Abbildung 4.9 wird deutlich, dass sich die Herangehensweise zwischen Objekterkennung und Bildsegmentierung kaum unterscheidet. Die Bildsegmentierung kann als weiterer Schritt zur Objekterkennung gesehen werden, welche hauptsächlich sogenannte „Bounding Boxes“ um die zu erkennenden Objekte legt. Darüber hinaus werden die erwähnten Rahmen mit den Wahrscheinlichkeiten versehen, dass es sich um das besagte Element handelt.

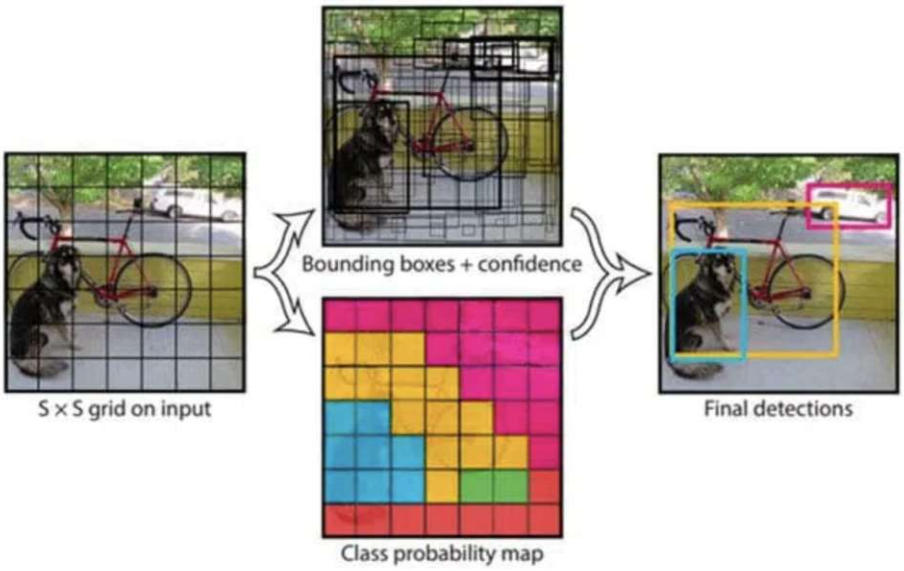


Abb. 4.9: Funktionsweise der Bildererkennung und Bildersegmentierung [1]

Kapitel 5

Auswahl der CNN-Architektur

In diesem Kapitel werden Experimente durchgeführt, welche fundamental den Entscheidungsprozess der im Kapitel der Methodologie verwendeten Systeme unterstützen. Die Herangehensweise ist wichtig, um die bestmöglichen Voraussetzungen der gewählten CNN Architektur und die daraus folgenden treffsichereren Ergebnisse zu gewährleisten. Zu Beginn dieses Kapitels wird das verwendete Datenset erwähnt und die Kennwerte hierzu genannt. Anschließend wird der Prozess zur Findung der geeigneten CNN-Architektur geschildert und das in Verwendung kommende Netzwerk beschrieben. Abschließend werden die Resultate der Architektur zusammengefasst und eine geeignete Verlustfunktion, basierend auf den Erkenntnissen gesucht, welche im Theorieteil geklärt wurden.

5.1 Verwendetes Datenset

Diese Arbeit stützt sich auf Open-Source-Datensätze. Zu Beginn der Recherche für die geeigneten Input-Daten der vorangehenden Experimente wurde Tabelle 5.2 angelegt. Sie beinhaltet den Namen, die enthaltenen Oberflächenmaterialien und die Größe des Datensets. Für die vorangehenden Experimente ist es wichtig, dass möglichst viele der ausgewählten Oberflächenmaterialien in der Bildersammlung enthalten sind und sich die Größe auf maximal 1000 Bilder beschränkt, da die Resultate der Experimente aussagekräftig und nicht perfektioniert werden sollen. Als Datenset für die vorangehenden Experimente der geeigneten CNN-Architektur werden dreikanälige RGB-Bilder des DeepCrack Datensets verwendet, abgeleitet aus Tabelle 5.2.

In Abbildung 5.1 sind fünf zufällig gewählte Bilder und die dazugehörige Maske zu sehen. In Abbildung 5.2 sind augmentierte Bilder, welche rotiert, gespiegelt etc. wurden, zu sehen. Dies wurde durchgeführt, um das Datenset zu vergrößern und somit genauere Ergebnisse zu erzielen (siehe Abschnitt 4.4). Insgesamt befinden sich 300 Bilder in dem gewählten Datenset, die im nächsten Schritt in Training (70 % Testset (20 %) und Validierung (10 %) gesplittet werden. Hierbei handelt es sich um das klassische "7-2-1-Splitting". Das DeepCrack-Datenset eignet sich für die pre-methodische Phase besonders gut, da es Asphalt-, Beton- und Putzrisse bereits abdeckt. Es ist also nicht von einem homogenen Datenset die Rede, welches sich nur einer Anwendung bedient. Darüber hinaus wird es laut Website PaperswithCode.com [50] häufig für Recherchearbeiten eingesetzt, wodurch eine Unterrepräsentanz der Datenlage für den Anwendungsfall ausgeschlossen werden kann.

5.2 CNN-Architektur

Bianco et al. [7] führten einige Untersuchungen zu Akkuratheit von Deep Learning Architekturen durch. Aus Abbildung 5.3 lässt sich schließen, dass es sehr viele unterschiedliche Netzwerkarchitekturen gibt. Diese unterscheiden sich stark in der Komplexität und Anzahl an Parameter, der Genauigkeit von Resultaten und der Leistungsfähigkeit pro Sekunde. Schlussfolgernd ist nicht jedes der angeführten Netzwerke für jeden Anwendungsfall geeignet.

Tab. 5.2: Auflistung von Open-Source-Datensets sortiert nach Datensatzgröße und enthaltenen Oberflächenmaterialien

	enthaltene Oberflächenmaterialien	**Datensatzgröße**
SDNET2018	Beton	56 000
Wood Defect Dataset	Holz	43 000
Crack Detection and Segmentation Dataset	Asphalt, Beton	11 298
CrackSeg9K	Asphalt, Beton, Glas	9000
Historical Crack 18-19	Beton, Mauerwerk, Putz	3886
LCW Dataset	Asphalt	3817
GAP V2	Asphalt	2469
GAP V1	Asphalt	1969
DeepCrack	Asphalt, Beton, Putz, Mauerwerk	537
Crack500	Asphalt	500
GAPs384	Asphalt	384
Crack Forest Dataset	Asphalt	329
Masonry Crack Dataset Leeds University	Mauerwerk	224
Masonry Crack Dataset	Mauerwerk	89
NHA12D	Asphalt	80
AigleRN	Asphalt	38
GAP 10m	Asphalt	20

FLOP, kurz für Floating Points Operation per Second, ist eine Maßeinheit für die Leistungsfähigkeit eines Rechners. Sie bezeichnet die Gleitkommazahl Operationen, sprich Addition und Multiplikation, die pro Sekunde möglich sind. Angesichts der Tatsache, dass nicht alle Deep-Learning-Architekturen für Bildsegmentierung geeignet sind, wurden einige dementsprechend nicht in die Auswahl miteinbezogen. Eine Anpassung der Architektur von reiner Objekterkennung zu einer Bildsegmentierung ist für diese Arbeit als zu aufwendig einzustufen, da dies sehr viel Zeit und Aufwand in Anspruch nehmen würde.

Pantofaru und Hebert [49], Wei et al. [84], Yu et al. [90] und Zhou et al. [96] vergleichen allesamt unterschiedlichste Bildsegmentierungsalgorithmen. Insbesondere Akosman et al. [3] unterscheidet mehrere Netzwerkarchitekturen anhand von Risssegmentierung auf Marmor. Basierend auf den Erkenntnissen aller Untersuchungen wurden das U-NET, SegNet und ResNet für die Auswertung der geeigneten Netzwerkarchitektur ausgewählt. Als Vergleichsfaktor für die Netzwerkfindung wurde ein für die Bildsegmentierung angepasstes AlexNET gewählt, da es einer der ältesten und einfachsten CNN-Architekturen ist.

Voraussetzungen

Es ist zu beachten, dass alle Netzwerkarchitekturen mit den gleichen Voraussetzungen trainiert wurden. Es wird jeweils das gleiche Trainings-, Test- und Validierungsset verwendet, um die gleichen Bedingungen zu schaffen. Die Algorithmen werden auch nicht angepasst oder verändert. Es wurden nur 40 Epochen für die Trainingsdauer verwendet, da die Ergebnisse für die pre-methodische Phase nicht perfekt sein müssen. Das Ziel der Recherche ist, eine geeignete Architektur für die Segmentierung von Rissen auf unterschiedlichen Oberflächenmaterialien zu finden.

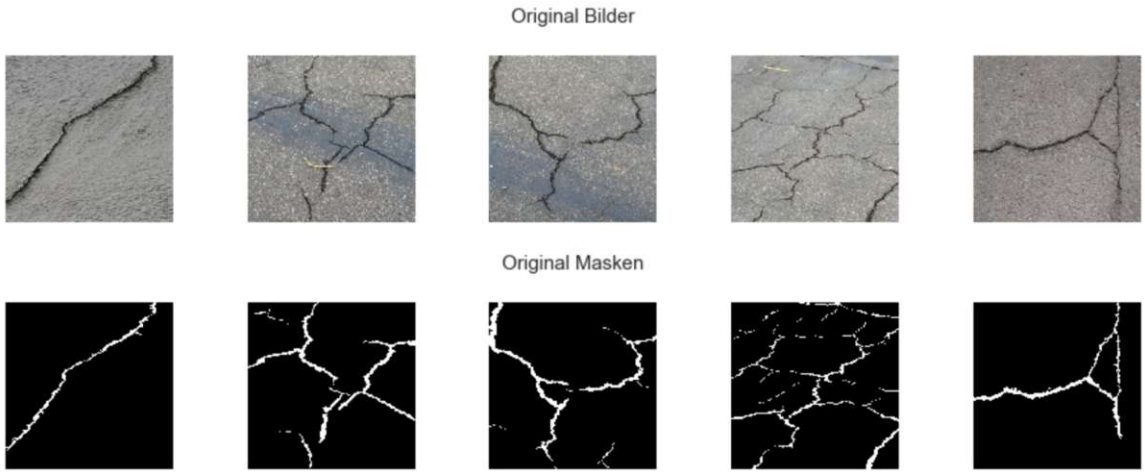


Abb. 5.1: Bilder und Masken aus dem DeepCrack Testdatenset

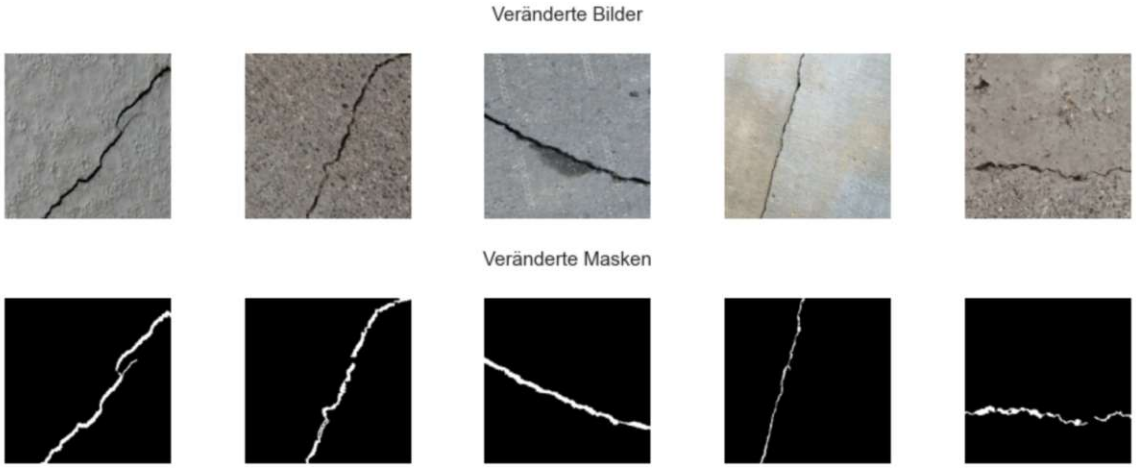


Abb. 5.2: Veränderte Bilder und Masken aus dem DeepCrack Testdatenset zur Datenvolumenerhöhung

Tab. 5.4: Ergebnisskennzahlen der gewählten Netzwerkarchitekturen

Architektur	Accuracy %	Precision %	Recall %	F1-Score %
AlexNet	0,57	0,61	0,54	0,60
U-NET	0,85	0,80	0,79	0,89
SegNet	0,85	0,83	0,89	0,89
ResNet	0,83	0,81	0,81	0,89

5.3 Resultate

Die Resultate der pre-methodischen Analyse können aus Tabelle 5.4 entnommen werden. AlexNet war den Erwartungen und Resultaten entsprechend eine zu ungenaue Architektur. Beim Einsatz dieser Netzwerkarchitektur würden, bei den für diese Arbeit verwendeten Input-Daten, keine aussagekräftigen Bildsegmentierungswahrscheinlichkeiten erzielt werden. Bei AlexNet handelt es sich auch um das älteste Netzwerk, welches nur als Vergleichsfaktor verwendet wurde, dementsprechend sind U-NET, SegNet und ResNet mehr am Stand der aktuellen Technik.

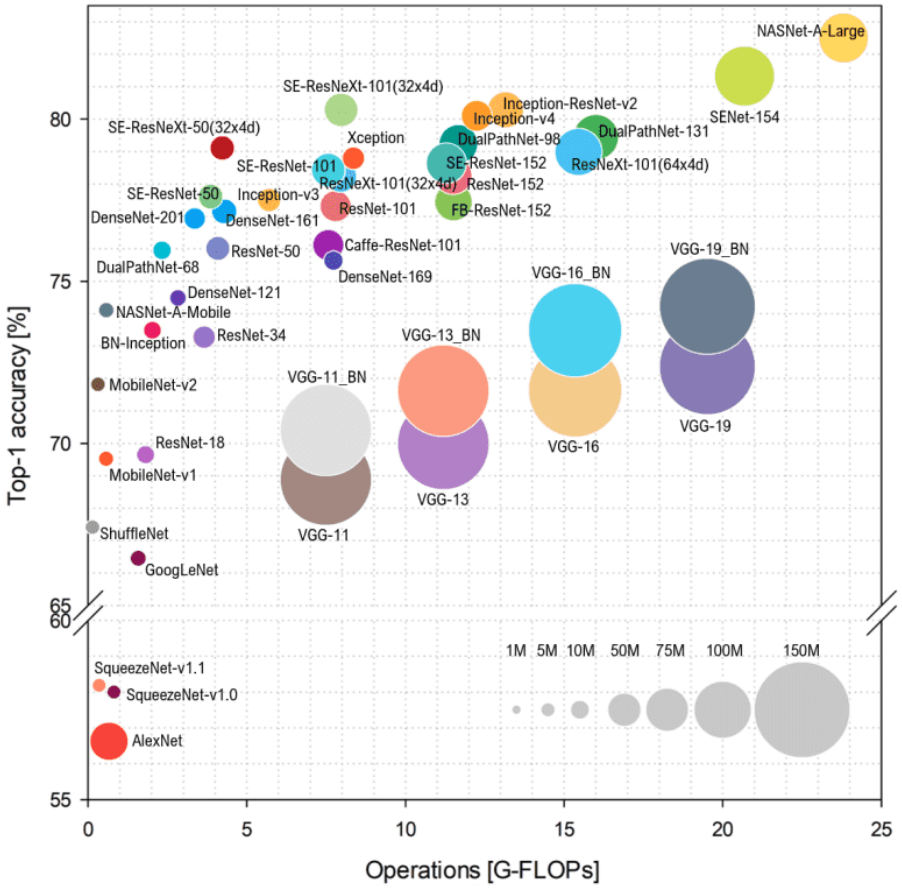
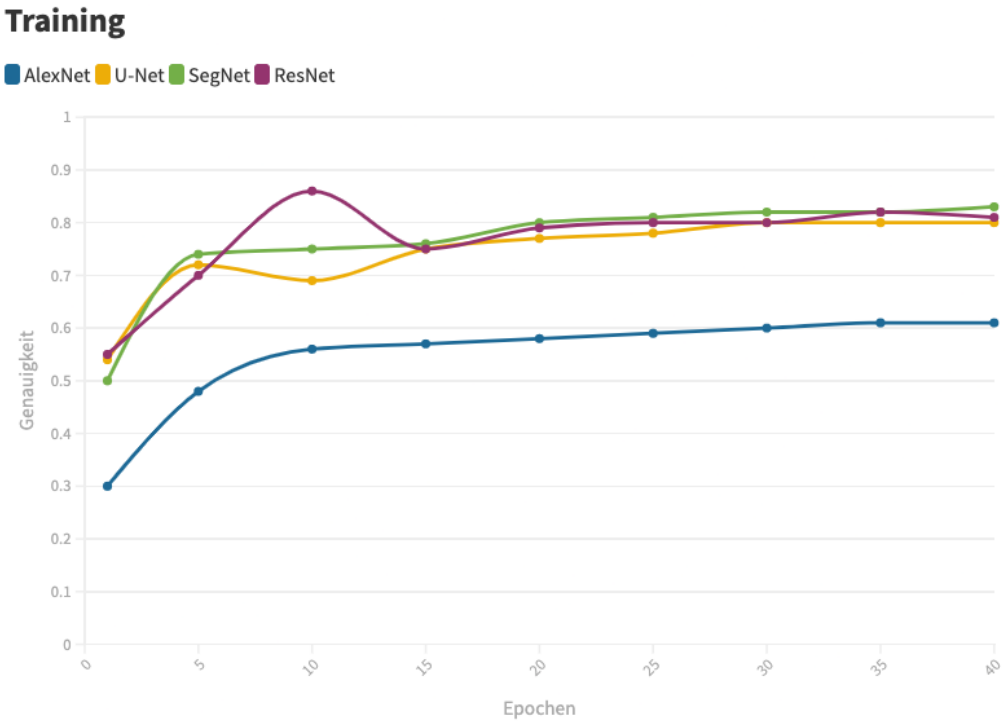


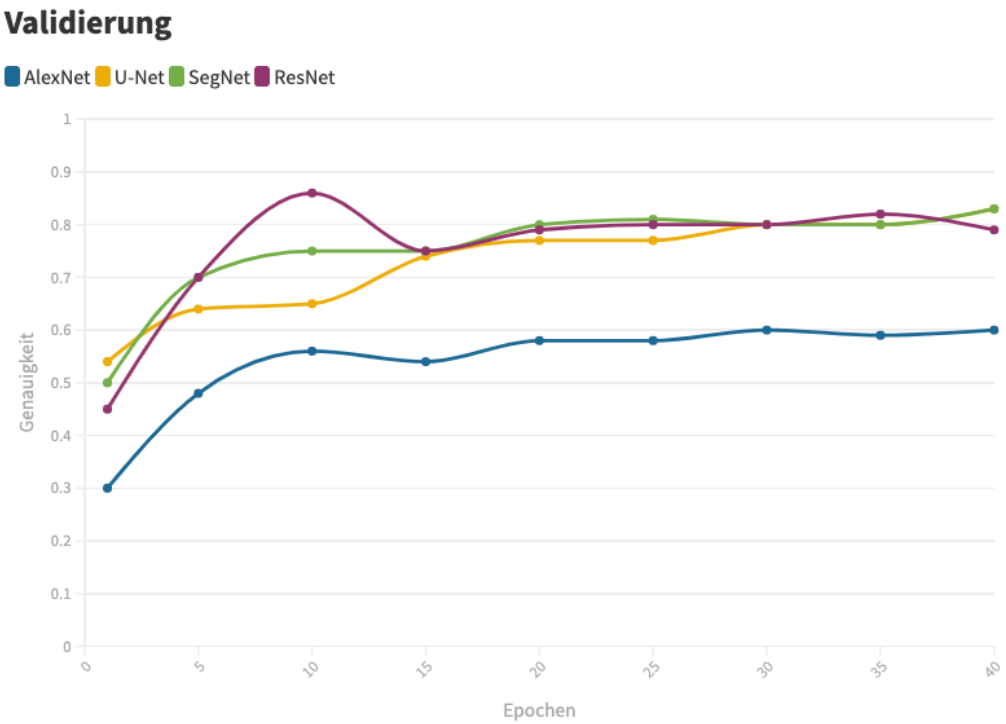
Abb. 5.3: Darstellung von Deep-Learning-Architekturen basierend auf Komplexität, Genauigkeit und Leistungsfähigkeit (FLOP) [7]

Aus den Ergebnissen der Experimente, welche in Tabelle 5.4 angeführt sind, konnte sowohl das ResNet, als auch das U-NET in allen vier Kategorien (Richtigkeit, Genauigkeit, Recall und F1-Score) überzeugen. Obwohl ResNet von den Ergebnissen her minimal prozentual höher abgeschnitten hat, wurde das U-NET als Grundarchitektur verwendet, da die meisten Forschungsarbeiten, welche sich mit Bildsegmentierung beschäftigen, U-NET verwenden. Unterdessen ist das U-NET kleiner und dadurch schneller in Trainingsdurchläufen.

In Abbildung 5.4 ist deutlich sichtbar, dass auch der Verlauf über mehrere Epochen hinweg nichts an dem Endresultat verändert. Je länger der Trainings- oder Validierungsprozess läuft, desto mehr nähern sich auch die Ergebnisse der einzelnen ausgewählten Netzwerkarchitekturen aneinander an. Es stellt sich stützend auf dem DeepCrack-Datenset eine Art Plateau ein. Abschließend zu erwähnen ist, dass zu U-NET die meisten Recherchearbeiten veröffentlicht werden, da es häufig in der Medizintechnik zum Einsatz kommt [50]. Basierend darauf kann geschlossen werden, dass die Netzwerkarchitektur für sehr viele unterschiedliche Anwendungsfälle geeignet ist. Daraus kann geschlossen werden, dass das U-NET für RGB-Bilder von unterschiedlichen Oberflächenmaterialien nutzbar ist.



(a) Genauigkeitskurve der Netzwerkarchitekturen



(b) Validierungskurve der Netzwerkarchitekturen

Abb. 5.4: Ergebnisse anhand eines Liniengraphs

Es ist abschließend zu bekunden, dass in der Zwischenzeit Nachfolger des U-NETs auf dem Markt sind, welche, je nach Anwendung, treffsichere Endergebnisse erzielen. Diese wurden jedoch für die

Recherche und die vorangehenden Experimente ausgeschlossen, da sich die Netzwerkarchitektur nur geringfügig von der originalen unterscheiden [96]. Der Grundsatz der vorliegenden Arbeit liegt im Vergleich der Ergebnisse der Oberflächenmaterialien und nicht das bestmögliche Endresultat zu erzeugen.

U-NET

Das Institut für Informatik der Universität Freiburg hat 2015 eine neuronale Netzwerkarchitektur erstellt, welche die Bildsegmentierung von biomedizinischen Befunden erleichtert [62]. Es baut auf einem FCN-Netzwerk auf, wurde aber soweit adaptiert, dass es mit weniger Trainingsdaten treffsicherere Ergebnisse erzielen konnte. Klassisch für das U-NET ist eine komprimierende Phase zu Beginn der Architektur und eine expandierende Phase am Ende. Aus Abbildung 5.5 kann die namengebende U-Form, welche von der Kontraktion und Expansion abstammt, abgelesen werden. Dieses Kennzeichen entstammt der optimierenden Natur der einzelnen Funktionsebenen.

In einem U-NET wird abwechselnd eine 3×3-Convolution (siehe Abschnitt 4.1) und Maxpooling (siehe Abschnitt 4.3.3) durchgeführt. Da diese Netzwerkarchitektur mit relativ wenig Inputs arbeitet, wird eine Datenaugmentierung durchgeführt. Das bedeutet, dass die vorhandenen Inputs leicht verändert werden, um die Datensatzgröße zu erhöhen.

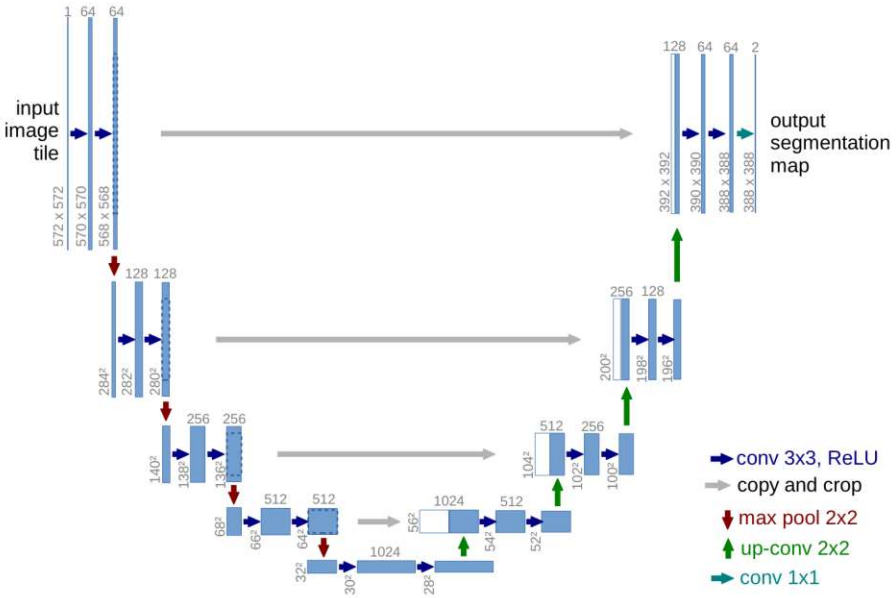


Abb. 5.5: Klassische Form eines U-NETs [62]

5.4 Verlustfunktion

Die Ergebnisse der Verlustfunktion können in Graphen der sogenannten „Verlustkurve“ dargestellt werden. Diese sind visuelle Indikatoren für Over- und Underfitting Problematiken, welche in Abschnitt 6.2.5 diskutiert wurden. Durch Experimente mit der Netzwerkarchitektur hat sich herausgestellt, dass es kategorisch am sinnvollsten ist, in der Output-Ebene eine Sigmoid-Funktion (siehe Abschnitt 3.3.2) in Anbetracht zu ziehen. Dadurch bekommt jedes Pixel einen Wert zwischen null und eins. Hierbei kann abgeleitet werden, ob es sich um ein „Risspixel“ oder „Nicht-Risspixel“ handelt. Aufgrund der Datenlage ist die Wahrscheinlichkeit höher, dass ein Pixel ein „Nicht-Risspixel“ ist, da Risse nur einen kleinen Bruchteil des Bildes einnehmen. Somit würde es zu

einer Klasseninbalance führen, welche die Resultate in ihrer Wahrscheinlichkeit verschlechtern würde.

Da sich eine Risssegmentierung auf Oberflächenmaterialien faktisch kaum von einer Bruchsegmentierung in der Medizin unterscheidet, wurden für dieses Unterkapitel medizinische Anwendungen zum Vorbild genommen. Die am häufigsten verwendete Verlustfunktionskategorie in der Medizin ist der in Abschnitt 3.3.5 erläuterte region-based loss (regionalbasierter Verlust). Basierend auf Recherchen von Zhao et al. [94] und Jadon [32] kann angenommen werden, dass zwar der Focal Dice Loss in der Theorie bessere Ergebnisse erzielen kann, jedoch sich bei tiefen neuronalen Netzwerken ein Overfitting-Problem einstellen kann. Demnach wurde der Sørensen–Dice-Koeffizient gewählt, welcher auch zur Segmentierung von Brüchen und Tumorzellen in der Medizin eingesetzt wird. Die ausgewählte Verlustfunktion kann mathematisch folgend definiert werden:

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|} \tag{5.1}$$

Aus der Formel ist ersichtlich, dass die Ähnlichkeit zweier Funktionen X und Y ermittelt wird [9]. Daraus lässt sich schlussfolgern, dass dieser Algorithmus besonders gut für Anwendungsfälle geeignet ist, in denen mit Masken, sprich mit bereichsabhängigen Verarbeitungsprozessen, gearbeitet wird. Dies ist in dieser Diplomarbeit der Fall, da Risse auf RGB-Bildern immer nur einen Bruchteil des Gesamtbildes einnehmen.

5.5 Zusammenfassung

In diesem Kapitel wurde ermittelt, welche Netzwerk-Architektur sich besonders gut für die Bildsegmentierung basierend auf dem DeepCrack-Datenset, eignet. Die genannte Datensammlung wurde aufgrund der bereits beinhalteten unterschiedlichen Materialien ausgewählt. Mit der Austestung unterschiedlicher Netzwerkarchitekturen kristallisierte sich das U-NET als passend heraus, welches kombiniert mit dem Sørensen–Dice-Koeffizient als Verlustfunktion die Grundlage für die weitergehenden Untersuchungen bilden. Im weiteren Schritt wurde die vorliegende Diplomarbeit von verwandten Projekten abgegrenzt.

Kapitel 6

Durchführung und Entwicklung des CNN

In diesem Kapitel wird die Durchführung und Entwicklung des CNN beschrieben. Dieser Abschnitt baut auf den Erkenntnissen der vorherigen Kapitel auf und setzt diese praktisch um. Zuallererst werden die zu untersuchenden Materialien definiert und in Training-, Validierungs- und Testdatensätzen aufgeteilt. Hierzu wird der Prozess dokumentiert und die benötigten Hilfsmittel aufgelistet. Die verwendete CNN-Architektur wird anschließend erklärt und die feinen Anpassungen argumentiert. Folgend wird der Pre-Trainingsprozess und der Trainingsprozess per definitionem geschildert und veranschaulicht. Ziel der Untersuchung ist, die Genauigkeit, präziser, die Wahrscheinlichkeit einer richtigen Erkennung und Segmentierung eines Risses auf unterschiedlichen Oberflächenmaterialien festzustellen.

6.1 Datensätze

Da es kein einheitliches Datenset mit Rissen auf mehreren Oberflächenmaterialien gibt, wurde für jedes Material ein eigenes Datenset erstellt und angelegt. Die gewählten Materialien wurden aufgrund ihrer Häufigkeit im Bauwesen ausgewählt. Für den Trainingsprozess wurden 200 Bilder je Datenset ausgewählt. Es wurde nach dem „7-2-1-Splitting“ unterteilt: 100 für das Training, 80 zum Testen und für den Validierungsprozess 20 Bilder. Für die Erstellung und allgemeine Optimierung des neuronalen Netzes wird nur das Trainingsset herangezogen. Die Daten überschneiden sich innerhalb der materialspezifischen Unterordner nicht, wodurch das neuronale Netz seine Fähigkeiten an neuen Inputs austesten muss. Folgende Werkstoffe wurden selektiert:

- Beton /Stahlbeton
- Mauerwerke
- Holz
- Asphalt

Beton

Für das Material Beton wurde das Concrete-Crack-Image-Datenset ausgewählt. Die Datenzusammenfassung von Özgenel [46] beinhaltet 600 RGB-Bilder, welche eine Größe von 227×227 Pixel aufweisen. Zu jedem Originalbild findet sich auch die dazugehörige Maske.

Asphalt

Das Crack500-Datenset von Zhang et al. [92] wurde für den Werkstoff Asphalt ausgewählt. Es beinhaltet 500 RGB-Bilder und die dazugehörigen Masken. Die Bilder sind 2000×1500 Pixel groß und müssen in die richtige Form gebracht werden.

Holz

Die Generierung des Holzdatensets bedarf eines größeren Aufwandes. Die Basis bildet das „Wood Defect“ Dataset von Pavel et al. [51]. Der Datensatz beinhaltet optimal ausgeleuchtete Idealbilder einer Holzart. Insgesamt beinhaltet dieses Datenset 43 000 Bilder, welche jedoch nicht nur Risse im Holz abdecken, sondern auch andere Defekte, die im Holz auftreten können, beispielsweise Astlöcher diverser Art. Es wurden manuell 200 RGB-Bilder ausgewählt, welche unterschiedliche Formen von Rissen und unterschiedliche Holzbereiche eines Querschnittes abdecken. Weiters wurde mit Hilfe von Label Studio Masken erstellt, da diese auch nicht im Datenset vorhanden waren.

Mauerwerk

Für das Mauerwerk wurde das „Masonry Crack Dataset“ von Dais et al. [12] ausgewählt. Es sind darin 240 RGB-Bilder mit den dazugehörigen Masken enthalten. Der Vorteil an diesem Datenset ist, dass die Bilder bereits eine Größe von 224×224 haben, welches das Training erheblich beschleunigt. Weiters sind Risse aller Art und Form vertreten, womit das neuronale Netz mehrere Rissarten kennenlernt. Die Qualität der Masken ist teilweise mangelhaft und es wurde vereinzelt mit GIMP nachgebessert.

6.2 Schritte des Preprocessing

In diesem Abschnitt werden die einzelnen Arbeitsschritte erklärt und begründet. Zu Beginn werden die wichtigsten Pythonmodule in ihrer Funktionalität erklärt und die verwendeten Versionen aufgelistet. Im Allgemeinen werden alle Untersuchungen und Netzwerke anhand einer Intel Iris Plus Graphics 655 1536 MB Grafikkarte durchgeführt. Die Schritte des Preprocessings bilden die Basis der gesamten Untersuchungen.

6.2.1 Wichtige Module und Umgebungen

In diesem Abschnitt werden die wichtigsten Kernpunkte aufgelistet und begründet, die für die Erstellung eines neuronalen Netzes und hinzugefügte externen Bibliotheken notwendig sind. Es wird die verwendete Programmiersprache schematisch beschrieben und die darüber hinausgehende Programmierumgebung angeführt.

Programmiersprache und Umgebung

Die Programmiersprache Python wurde gewählt. Die Programmierung wurde in der aktuellsten Version „3.9.“ erstellt. Python ist eine leicht zu erlernende und objektorientierte Programmiersprache, welche laut Raschka und Mirjalili [68] am häufigsten in Verbindung mit künstlicher Intelligenz zum Einsatz kommt. Überdies sind sehr viele externe, importierbare Bibliotheken auf dem Markt, welche Python und Prozesse im maschinellen Lernen unterstützen.

Als Umgebung wurde „Jupyter-Notebook“ von Project Jupyter gewählt. Es handelt sich hierbei um eine Open-Source-Entwicklungsumgebung, welche es ermöglicht, den Code anschaulich und übersichtlich zu dokumentieren und auszuführen. Ein Jupyter-Notebook wird in JSON-Format abgespeichert und hat eine „.ipynb“ Dateiendung. Um ein Jupyter-Notebook zu erstellen, wird entweder auf einen Editor zurückgegriffen, z. B. Visual Studio Code, oder auf das JupyterLab, welches im Rahmen dieser Arbeit eingesetzt wurde. Der Vorteil der gewählten Umgebung liegt zum einen daran, dass mehrere Kernel miteinbezogen werden, welche eine schnellere Berechnung garantieren. Zum Anderen können Visualisierungen und Darstellungen schnell entwickelt und exportiert werden.

Machine-Learning-Ökosystem

Als Machine-Learning-Framework wurde in der vorliegenden Diplomarbeit Tensorflow und Keras verwendet. Keras ist seit der Veröffentlichung von Tensorflow 1.4 fixer Bestandteil der Machine-Learning-Bibliothek. Durch die übersichtliche Gestaltung der Lernressourcen und Ausgestaltung von Tutorials ist die Erstellung eines neuronalen Netzes vergleichsweise gut zu handhaben. Tensorflow ist auch das am häufigsten zum Einsatz kommende Schema, da die Bibliothek für viele Anwendungsfälle Lösungen bietet [79]. In der vorliegenden Arbeit wurde die zu diesem Zeitpunkt aktuelle Version von Tensorflow (Version 2.9.0.), welche im Mai 2022 veröffentlicht wurde, zur Erstellung des neuronalen Netzes verwendet.

Optimizer

Als Hyperparameter-Optimierer wurde ADAM verwendet. Reddi et al. [58] beschrieben in ihrem Researchpaper, dass ursprünglich der gleitende Durchschnitt in den Berechnungen zu fehlerhaften Ergebnissen geführt hat. Der neueste Optimizer ADAM and Beyond verwendet einen gleitenden Durchschnitt namens „AMSGRAD“. Dieser verringert nicht nur auftretende Probleme in den Berechnungen, sondern auch die allgemeine Performance der komplexen Kalkulierungen, welche in den einzelnen Ebenen stattfinden. Weiters wurde die Bibliothek von Scikit-learn, insbesondere Scikit-Image verwendet, um die Verarbeitung von Bildern zu optimieren und den Prozess zu vereinfachen. Walt et al. [83] beschreibt, dass Scikit-Image die beste Python Bibliothek ist, um vergleichsweise leicht neuronale Netze, welche mit Bildern arbeiten, zu erstellen. Obendrein ist sie auch noch frei von Restriktionen und kostenlos verwendbar und anpassbar.

Visualisierung

Zwei verschiedene Bibliotheken wurden für Visualisierungen verwendet. Zum einen wurde „PILLOW“ als externe Bibliothek importiert, welche sich gut für die Darstellung der Netzwerkarchitektur eignet – Zum anderen Matplotlib, welche zur Visualisierung von Diagrammen zum Einsatz kommt. Beide ermöglichen Einsicht in die komplexen Vorgänge der CNN-Architektur und veranschaulichen interne Prozesse.

Datenstruktur

Da der Umgang mit mehrdimensionalen Arrays und Vektoren ausschließlich mit Python schwierig ist, wurde die Bibliothek „Numpy“ verwendet. Es handelt sich hierbei um eine Befehlssammlung, welche die Handhabung mit numerischen Operationen erleichtert.

6.2.2 Netzwerk-Architektur

Durch die durchgeführten Voruntersuchungen in Kapitel 5 wurde ersichtlich, dass das U-NET für den vorliegenden Anwendungsfall am geeignetsten ist. Demnach wird die genannte Architektur für diese Diplomarbeit eingesetzt. Da sich jedoch mit dem Austesten unterschiedlicher Materialien immer ungenügender werdende Ergebnisse herauskristallisierten, wurden Elemente von einem ResNet miteingebracht. Charakteristisch für die genannte Netzwerkarchitektur sind die ResNetblöcke, welche Verbindungen überspringen, um wieder Verbindung zum Originalpixel aufzunehmen. Diese Blöcke wurden eingebaut um die Genauigkeit zu erhöhen und hiermit auch die allgemeine Richtigkeit der Ergebnisse zu erhöhen.

Aus Abbildung 6.1 kann der klassische Aufbau eines ResNetblockes entnommen werden, dieser wurde auch in der vorliegenden Arbeit eingebaut. Das Grundkonzept liegt darin, dass die Information nicht nur in die nächste Ebene, sondern auch zwei oder mehrere Ebenen weiter eingeleitet wird. Damit werden Informationen mit vorhergehenden Ebenen verglichen und miteingespeist. Veit et al. [82] beschreibt, dass sich residuale neuronale Netzwerke wie ein

Ensemble mit kurzen Distanzen zwischen den Neuronen verhalten, was auf die Architektur zurückzuführen ist.

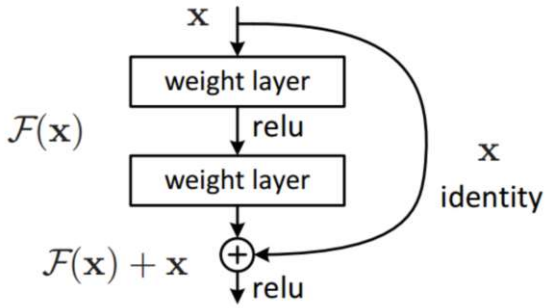


Abb. 6.1: Klassische Architektur eines residualen Blocks [65]

Szegedy et al. [77] stellt 2015 fest, dass es eine Korrelation zwischen der Netzwerktiefe (bzw. der Anzahl der Ebenen), der Qualität der Ergebnisse und dem Overfitting gibt. Demnach werden die Ergebnisse zwar treffsicherer und genauer je mehr Ebenen und Prozesse sie durchlaufen, jedoch steigt die Wahrscheinlichkeit des Overfitting mit jedem Layer an. Durch das Überspringen von einer oder mehreren Ebenen kann dem entgegengewirkt werden. Hierdurch fließt Information von der Ursprungsebene in die aktivierte Ebene. Dank Experimenten von He et al. [26] war es möglich, dem Overfitting in tiefen neuronalen Netzen entgegenzuwirken und aussagekräftigere Ergebnisse als bei nicht tiefen Netzen zu erzielen. Dies war nicht nur ein Durchbruch für das Deep Learning im Allgemeinen, sondern insbesondere für die Computer Vision, welche auf tiefe Netze angewiesen ist. Es ist wenig verwunderlich, dass das daraus resultierende ResNet zu viel Begeisterung in der Forschung rund um Deep Learning und Computer Vision führte.

Aus der Darstellung 6.2 kann die neue adaptierte Netzwerkarchitektur abgelesen werden. Im Gegensatz zur klassischen U-NET Architektur befinden sich in der angepassten Form mehrere residuale Blöcke. An der sanduhrartigen Form kann abgeleitet werden, dass die Bilder über mehrere Ebenen hinweg komprimiert werden. Dabei handelt es sich um die klassische Form eines U-NET, die im Abschnitt 5.2 erklärt wird. Im Abschnitt 4.3.3 wird der Vorgang beschrieben, der der Netzwerkarchitektur die charakteristische Form verleiht. Insgesamt werden die Informationen, welche auf den Bildern enthalten sind, viermal mit Hilfe von Max-Pooling optimiert. Die Ursprungsgröße der Bilder liegt bei 256×256, welche anschließend auf 128×128 komprimiert werden. Diese werden im weiteren Schritt wieder auf 64×64 komprimiert, bis sie abschließend mit 32×32 ihre finale Form erreichen. Gleichzeitig ist zu erkennen, dass mit jeder Komprimierung die besagten Blöcke auch in der Dicke, bzw. Breite zunehmen. Dies hat den Hintergrund, dass es sich um eine „Faltung“ (siehe Abschnitt 4.1) handelt. Sinnbildlich beschrieben wird durch eine Faltung die 2D-Fläche des Bildes kleiner, die Dicke jedoch größer. Es werden die Daten lediglich in eine andere, optimierte Version umgeformt. Stützend auf der Tatsache, dass in der vorliegenden Arbeit mit Bildern, genauer betrachtet zweidimensionalen Inputs, gearbeitet wird, wird als convolutionale Ebene ausnahmslos Conv2D-Layer verwendet.

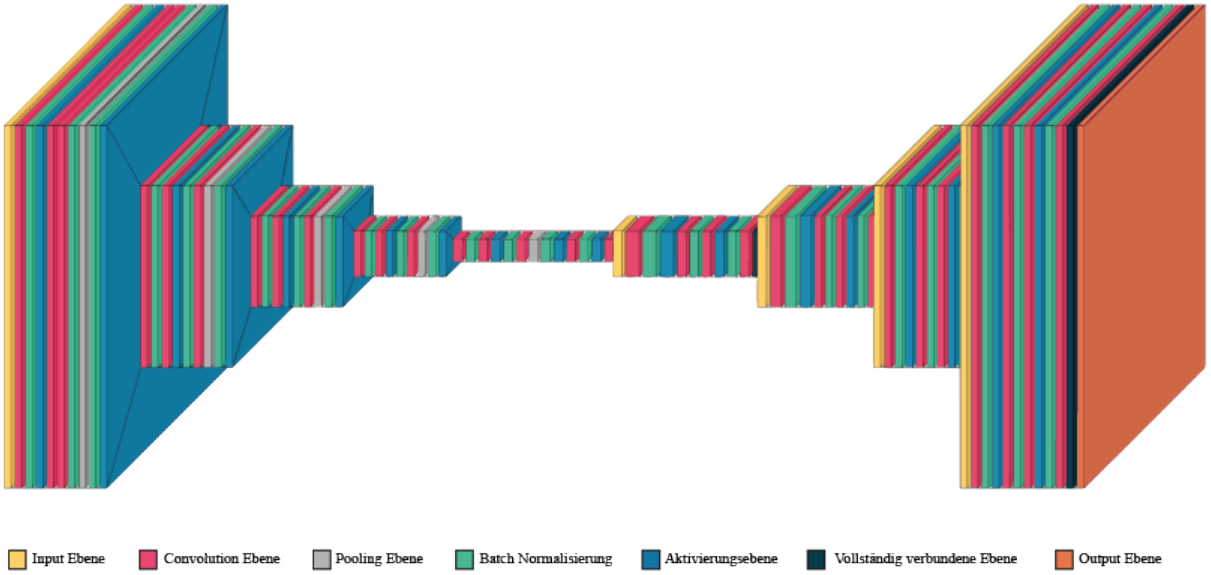


Abb. 6.2: Schematische Darstellung der angepassten Netzwerkarchitektur

Batch-Normalisierung

In Abbildung 6.2 ist zu erkennen, dass reichlich Batch-Normalisierungs-Ebenen in der Netzwerkarchitektur eingebaut worden sind. Oppermann [44] beschreibt, dass diese Algorithmustechnologie, die auch Gruppenstandardisierung genannt wird, immer ein verbindlicher Teil eines tiefen neuronalen Netzwerkes ist. Die Batch-Normalisierung berechnet zunächst den Mittelwert und die Standardabweichung der Eingabe-Daten in einer kleinen abgegrenzten Datenmenge. Anschließend folgt die Normalisierung der Daten anhand einer Teilung mittels Standardabweichung. Folglich werden die Ergebnisse mit einem Faktor multipliziert und mit einem Verschiebefaktor verschoben. Diese Schicht hat Lernparameter in Form von Gewichten und Verschiebungen, die während des Trainings angepasst werden können.

Wenn sich der Output einer Ebene verändert, welcher Vorgang als interner Kovarianzverschiebung bekannt ist, so müssen die darauffolgenden Layer neue Parameter und Verteilung der Ausgabewerte erlernen. Um diesem Problem entgegenzuwirken, wird die Batch-Normalisierung für jedes einzelne Neuron bzw. für jede einzelne Dimension berechnet (Mini-Batches). Das bedeutet, dass mit der Batch-Normalisierung das Ausmaß an Verschiebung der Werte minimiert wird. Dies geschieht noch vor der Aktivierungsfunktion, das auch in der Abbildung zu erkennen ist. Die Gruppennormalisierung gleicht die Eingabewerte in Form eines Mini-Batch der Aktivierungsfunktion somit an.

Santurkar et al. [66] beschreibt, dass die Batch-Normalisierung viele Vorteile in Bezug auf den Trainingsprozess mit sich bringt und unmittelbar auch die Qualität der Ergebnisse beeinflusst. Zum einen werden stabilere Gradienten erzeugt, welche die Findung des globalen Minimums (siehe Abschnitt 3.3.4) während der Anwendung der Verlustfunktion unterstützt. Darüber hinaus wird hier auch der Trainingsprozess und die Lernphase erhöht, zumal die Lernraten durch die Gradientenstabilisierung verkürzt werden. Basierend auf der Tatsache, dass die Gruppenstandardisierung die Kopplung an Gewichten und Biases durch eigene Parameter etwas löst, hat hierdurch auch die korrekte Befüllung der genannten Parameter nicht mehr einen zu hohen Stellenwert. Es wird dem neuronalen Netz hiermit etwas Spielraum ermöglicht.

Zusammenfassend wird die Anwendung der Batch-Normalisierung aufgrund vieler Vorteile begründet. Sie steigert die Leistungsfähigkeit während eines Trainingsprozesses, ohne zu großen Aufwand zu verursachen. Insbesondere durch die Verwendung eines nicht für Deep-Learning-

ausgelegten Rechner sind Optimierungsmaßnahmen eine gute Möglichkeit, um die Qualität der Ergebnisse dennoch zu gewährleisten.

6.2.3 Maskenerstellung

Masken werden benötigt, um dem neuronalen Netz ohne Störfaktoren beizubringen, an welcher Stelle sich der Riss auf dem Originalbild befindet. Es handelt sich hierbei um eine Schwarz-Weiß-Verortung des zu erkennenden Objektes. Somit erlernt das neuronale Netz während des Trainingsprozesses Risse auf den Bildern zu erkennen. Viele Open-Source Bildsegmentierungs-Datenset beinhalten bereits Masken, doch da diese Arbeit sich auf spezielle Datensets stützt, sind nicht immer Masken beigelegt. Die Masken der originalen RGB-Rissbilder sind schwarz-weiße RGB-Bilder, welche mit Label Studio erzeugt wurden. Auf einer separaten Ebene wurde mit Hilfe von „Bucketfill“ der Riss eingefärbt und somit markiert. Anschließend wird die neu erstellte Maske exportiert und in einem neuen Unterordner abgespeichert. Sie dient als Vergleichsfaktor beim Training und Evaluieren der Ergebnisse.

6.2.4 Trainingsprozess

Noch während der Scripting-Phase müssen einige Parameter bezüglich des Trainings festgelegt werden. Es wurden 100 Epochen für das Training festgelegt. Im Allgemeinen gilt, je mehr Feedbackdurchläufe, genauer gesagt Epochen, desto besser trainiert und genauer ist das neuronale Netzwerk. In Abbildung 6.3 ist ein Ausschnitt aus dem Trainingsprozess zu sehen. Es werden die Parameter, welche in der aktuellen Trainingsepoche getestet werden, aufgelistet und mit jedem einzelnen Prozess verbessert. Im Anschluss des Trainingsprozesses können Bilder aus dem Testset getestet werden. Hierbei werden Genauigkeit und weitere Schlüsselergebnisse ermittelt.

```
Epoch 1/100
27/27 [==============================] - 174s 6s/step - loss: 0.4433 - accuracy: 0.9325 - IOU: 0.4336 - dice_coef: 0.
5567
Lernrate:  0.001
Epoch 2/100
27/27 [==============================] - 172s 6s/step - loss: 0.2682 - accuracy: 0.9810 - IOU: 0.5842 - dice_coef: 0.
7318
Lernrate:  0.001
Epoch 3/100
27/27 [==============================] - 187s 7s/step - loss: 0.2461 - accuracy: 0.9835 - IOU: 0.6103 - dice_coef: 0.
7539
Lernrate:  0.001
Epoch 4/100
27/27 [==============================] - 184s 7s/step - loss: 0.2590 - accuracy: 0.9819 - IOU: 0.5941 - dice_coef: 0.
7410
Lernrate:  0.001
Epoch 5/100
27/27 [==============================] - 165s 6s/step - loss: 0.2295 - accuracy: 0.9839 - IOU: 0.6301 - dice_coef: 0.
7705
Lernrate:  0.001
```

Abb. 6.3: Ausschnitt aus dem Trainingsprozesses

6.2.5 Hyperparameteroptimierung

Parameter eines maschinellen Lernsystems, welche noch vor dem Trainingsprozess festgelegt werden müssen, werden Hyperparameter genannt [30]. Durch diese sehr feinen und präzisen Einstellungen kann festgelegt werden, auf welche Weise sich der erstellte Algorithmus verhält. Zu nennen sind die Lernrate, Anzahl der Ebenen, die Datengröße und -Menge. Sie werden während des Trainingsprozesses nicht optimiert. Die Parameter sollten im optimalen Fall auf einem eigens hierzu erstellten Validierungsset angepasst werden, welches weder im Trainingsprozess, noch

den Generalisierungsfehler bewertet. Hyperparameter nehmen Einstellungen für verschiedene Abläufe vor. Beispielsweise können sie die Lernrate eines neuronalen Netzes beeinflussen. Hierbei handelt es sich um einen kontinuierlichen Prozess oder diskrete Einstellungen, z. B. die Anzahl der versteckten Ebenen.

Für diese Diplomarbeit wurde zu Beginn die Standardlernrate von 0,1 gewählt, welche anschließend auf 0,01 verringert wurde. Sie beschreibt, mit welcher Gewichtung die in vorausgehenden Epochen erkannten Fehler miteinbezogen werden sollen. Bei tiefen Netzen kann eine hohe Lernrate zu einer Überanpassung führen, welches den ersten Durchläufen bewahrheitet hat. Aus diesem Grund wurde die Lernrate reduziert und angepasst.

Aus Abbildung 6.4 ist ersichtlich, dass darauf geachtet werden muss, dass es weder zu einem Unterfitting, sprich einer Unteranpassung des Modells, noch zu einem Overfitting, der Überanpassung eines Modells, kommt. Beide Extrema beeinflussen die Resultate und Ergebnisse negativ, da sie keine Vorhersagen in zukünftigen Daten treffen können. Ein gutes Modell veranschaulicht den allgemeinen Verlauf der Daten. Um dies zu verhindern, gibt es unterschiedliche Möglichkeiten, um Hyperparameter zu trainieren. Zum einen sind Suchstrategien, z. B. Zufallssuche (engl. random search) und Rastersuche (engl. grid search), eine Möglichkeit, um passende Parameter zu finden. Bei der Zufallssuche definiert der:die Entwickler:in eine Randverteilung für jeden benötigten Parameter. Anschließend muss eine Anzahl an Durchläufen bestimmt werden, welche die Häufigkeit von wahllos eingefügten und getesteten Werten für die Parameter definiert. Als Ergebnis werden die besten Werte für die Hyperparameter ausgewählt. Bei der Rastersuche wird von Beginn an von dem:der Entwickler:in eine endlose Anzahl an Zahlen festgelegt, welche für die Hyperparameter eingesetzt werden. Anschließend testet der Algorithmus die Werte nacheinander und die aussagekräftigsten Ergebnisse werden als Hyperparameter eingesetzt.

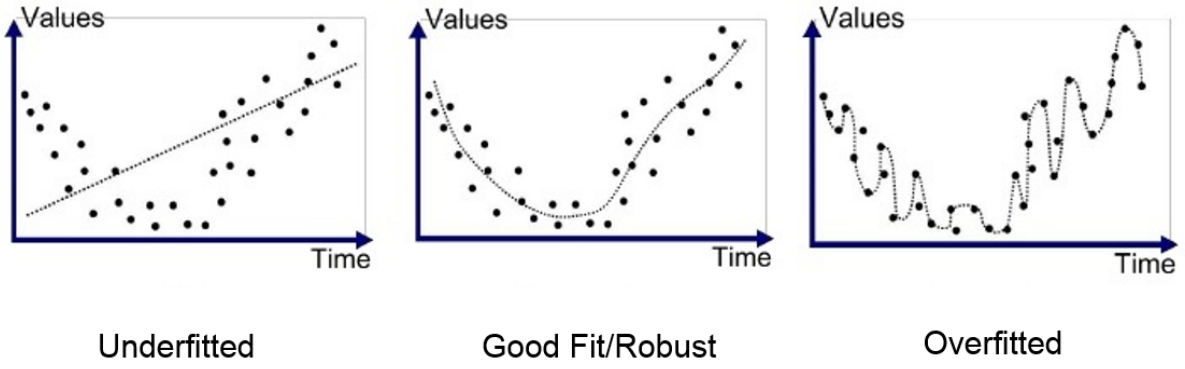


Abb. 6.4: Darstellung von Over-, Right-, und Overfitting [6]

Kapitel 7

Ergebnisse

Zusammenfassend durchlaufen in diesem Kapitel die Ergebnisse, die aus der Entwicklung des CNN erreicht wurden, einer Analyse und vergleichenden Bewertung. Die Resultate sind in zwei Kapitel unterteilt, allgemeine Ergebnissen, die für alle Materialien gelten, und oberflächenmaterialspezifische Resultate. Darüber hinaus werden zu Beginn die wichtigsten Kennziffern und Messzahlen im Zusammenhang mit den Endergebnissen erklärt. Die ausschlaggebenden Faktoren, Herausforderungen und Eigenschaften werden aufgegriffen und deren Zusammenhang mit dem Ergebnissen dargestellt. Hindernisse, welche während des Prozesses aufgetreten sind, werden geschildert und mögliche Lösungsprozesse erläutert. Zusätzlich wird hinzukommend die Bedeutung der Datensatzgröße beschrieben. Abschließend werden die Parameter der Confusion Matrix erklärt und behandelt. In diesem Abschnitt stehen im weiteren Schritt auch mögliche Optimierungsvorschläge im Fokus.

7.1 Begrifflichkeiten

Am Ende eines erfolgreichen Trainings- und Testdurchlaufes werden diverse Kennziffern und Messzahlen des neuronalen Netzwes aufgelistet. Sie beschreiben die Ergebnisse und die Qualität des erstellten Netzwerkes auf unterschiedliche Art und Weise. Werden die Ergebnisse nur mit einer der genannten Kennziffern beschrieben, wäre die Interpretation einseitig. Die Summe der Kennziffern ist notwendig, um das Ergebnis von unterschiedlichen Standpunkten und Sichtweisen zu durchleuchten.

Accuracy

Die Accuracy (Genauigkeit), misst die Wahrscheinlichkeit, dass ein Ergebnis positiv ist. Richtig Positive ist die Anzahl der Male, bei denen das Modell die positive Klasse richtig vorhersagt. Richtig Negative ist die Summe der richtig prognostizierten negativen Klasse. Die Summe der Vorhersagen ist die Gesamtzahl der vom Modell getätigten Vorhersagen. Dies bedeutet, dass umso höher der Prozentsatz, desto genauer sind die Resultate des neuronalen Netzes. In Formel (7.1) ist die Genauigkeit mathematisch dargestellt. Es werden alle positiven Resultate mit der gesamten Anzahl an Ergebnissen dividiert. Es ist wichtig zu beachten, dass die Genauigkeit häufig als allgemeine Maßzahl für die Leistung eines Modells verwendet wird. Bei einer gleichmäßigen Klassenverteilung, ist das kein Problem – Bei einer ungleichen Klassenverteilung, d. h. wenn eine Klasse häufiger vorkommt als andere, kann die Bewertung nur mit der Genauigkeit, irreführend sein. In solchen Fällen können andere Bewertungsmetriken, wie Präzision und Recall, sinnvoller sein, um die Leistung des Modells zu verstehen.

$$Genauigkeit = \frac{richtig\ positiv + richtig\ negativ}{Anzahl\ der\ gesamten\ Ergebnisse} \tag{7.1}$$

Precision

In Gleichung (7.2) wird ersichtlich, dass die Precision (Präzision), die Wahrscheinlichkeit darstellt, dass ein richtiges Ergebnis positiv ist. Es misst dementsprechend die Exaktheit eines Resultates.

$$Precision = \frac{richtig\ positiv}{richtig\ positiv + falsch\ positiv} \tag{7.2}$$

Recall

Recall stellt das Gegenstück zur Präzision dar. Es misst die Qualität der Ergebnisse in Bezug auf die Negativklassifizierung. Aus Formel (7.3) wird ersichtlich, dass ermittelt wird, mit welcher Wahrscheinlichkeit es nicht zu einer falsch negativen Einordnung kommt. Bei einer nicht-binären Anwendung, beispielsweise bei Mehrklassifizierungs-Anwendungen, kommt noch eine zusätzliche Kenngröße zum Einsatz, die Sensitivität. Diese Messzahl stellt die richtige Zuordnung einer Klassifizierung dar. Bei einer binären Anwendung ist das Recall ident zur Sensitivität.

$$Recall = \frac{richtig\ negativ}{richtig\ positiv + falsch\ negativ} \tag{7.3}$$

F1-Score

F1 oder F1-Score stellt den Zusammenhang von Precision und Recall dar. Es handelt sich hierbei um eine sehr wichtige Kenngröße, die die Gesamtgenauigkeit der Klassifizierung darstellt. Der F1-Score ist ident mit dem Dice-Koeffizient, welcher in Abschnitt 5.4 beschrieben wurde. Aus Gleichung (7.4) kann die mathematische Darstellung entnommen werden.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{7.4}$$

Spezifität

Die Spezifität veranschaulicht die Wahrscheinlichkeit, dass ein negatives Ergebnis tatsächlich negativ ist. Formel (7.5) stellt diesen Zusammenhang mathematisch dar. Somit steht sie im direkten Zusammenhang mit der Recall-Kennziffer.

$$Spezifität = \frac{richtig\ negativ}{richtig\ negativ \cdot falsch\ positiv} \tag{7.5}$$

Verlustkurve

Die Verlustkurve ist ein wichtiges Instrument, um die Performance von maschinellen Lernmodellen zu beurteilen. Sie zeigt das Verhältnis von Trainingsverlust (siehe Abschnitt 5.4) zur Anzahl der Epochen. Ein idealer Verlauf der Verlustkurve sieht zunächst steil abfallend aus, da das Modell in den ersten Epochen schnell Fortschritte macht. Mit zunehmender Anzahl von Iterationen über das Trainingsdatenset wird der Verlust langsamer abnehmen und die Kurve wird flacher. Wenn das Modell gut an die Trainingsdaten angepasst ist, wird der Verlust optimalerweise konstant bleiben. Bei einer Überanpassung an den Trainingsdatensatz steigt die Kurve in den letzten Epochen wieder an. Hierdurch wird die Vorhersage auf neuen, unbekannten Daten schlechter.

7.2 Allgemeine Ergebnisse und Erkenntnisse

Grundsätzlich wurde durch das Training von mehreren Netzen rasch sichtbar, dass die Grafikkarte für die benötigte Rechenleistung schnell die Leistungskapazitätsgrenze erreicht. Die Durchführung der Trainingsprozesse der einzelnen Netze beanspruchte mehrere Stunden bis Tage, welches ein rasches Austesten von Experimenten enorm beeinträchtigte. Dies liegt untere anderem auch an der Tiefe des erstellten Netzwerkes, da es über mehrere Epochen hinweg mehrere Berechnungen

Tab. 7.1: Genauigkeitsergebnisse der Oberflächenmaterialien

	Ergebnis % Asphalt	**Ergebnis % Beton**	**Ergebnis % Mauerwerk**	**Ergebnis % Holz**
Accuracy	96,66	96,91	98,65	92,02
Precision	73,39	91,11	93,37	7,12
Recall	79,61	85,57	79,14	60,94
F1-Score	76,37	88,26	85,66	12,75
Specificity	97,90	98,69	99,70	92,32

durchführen muss. Insgesamt wurden rund 4,723 Mio. Parameter je Material trainiert. Hiervon ist der Großteil, und zwar rund 4,716 Mio. trainierbar, das bedeutet, dass sie anpassbar und optimierbar sind. Der Rest (7296 Parameter) sind nicht optimierbar und somit untrainierbar. Diese Zahlen summieren sich aufgrund der vielen Ebenen und den damit folgenden Algorithmen. Auch die Batch-Normalisierung fügt Parameter hinzu, die nicht adaptierbar sind. Auf Basis der vorhergesagten Bilder (siehe enthaltene Abbildungen im nächsten Abschnitt), welche vom neuronalen Netz erstellt wurden, kann festgestellt werden, dass ein Riss immer erkannt wird. Bei den Durchführungen war nie ein rein weißes oder rein schwarzes Bild zu erkennen, welches auf eine Nichterkennung eines Risses schließen würde. Darüber hinaus konnten bis auf die Resultate des Materials Holz gute Ergebnisse erzielt werden.

Aus Abbildung 7.1 kann die Schlussfolgerung getroffen werden, dass alle ausgetesteten Oberflächenmaterialien einen logarithmischen Funktionsverlauf in Bezug auf die Accuracy aufweisen. Zu sehen ist ein Graph, welcher den Genauigkeitsverlauf aller Oberflächenmaterialien im Bezug auf die Epochen aufzeigt. In den ersten zehn Zeiteinheiten ist ein starker Anstieg in der Genauigkeit zu verzeichnen, welcher sich über die letzten 90 Epochen hinweg abflacht – Erzielten Endwerte können aus Tabelle 7.1 entnommen werden. Das bedeutet, dass das neuronale Netz in den letzten Epochen keine neuen Erkenntnisse erzielt. Einzig und allein das Oberflächenmaterial Holz fällt in den letzten zehn Zeiteinheiten wieder leicht ab. Darauf wird im folgenden Abschnitt eingegangen. Im Vergleich zum Genauigkeitsverlauf kann der F1-Score, welcher die Gesamtgenauigkeit darstellt, aus Abbildung 7.3 für alle Oberflächenmaterialien in Abhängigkeit der Epochen, entnommen werden. Die Annahmen der Accuracy können mit Hilfe dieses Graphen bestätigt werden. Beton Asphalt und Mauerwerk verzeichnen in den ersten zehn Epochen einen starken Anstieg und flachen in den folgenden 90 Zeiteinheiten ab. Das Oberflächenmaterial Holz hingegen hat eine geringe, jedoch kontinuierliche Steigung, das jedoch im Bezug auf den F1 Score schlecht ist. Die Endgültigen Resultate, welche in der hundertsten Epoche erzielt wurden können ebenfalls aus Tabelle 7.1 entnommen erden.

Abbildung 7.2 zeigt die Ergebnisse der Verlustfunktion der einzelnen Oberflächenmaterialien im Zusammenhang mit den Trainingsepochen. Asphalt, Beton und Mauerwerk zeigen einen Idealverlauf der Verlustkurve auf. Diese fällt in den ersten Epochen stark ab und flacht in den darauffolgenden Zeiteinheiten ab. Das deutet auf eine rasches Erlernen der Merkmale und erkennen von Fehlern hin. Es ist auch zu sehen, dass Holz im Vergleich zu den anderen genannten Oberflächenmaterialien sehr schlecht abgeschnitten hat. Risse werden demnach nicht bzw. unvollständig angelernt.

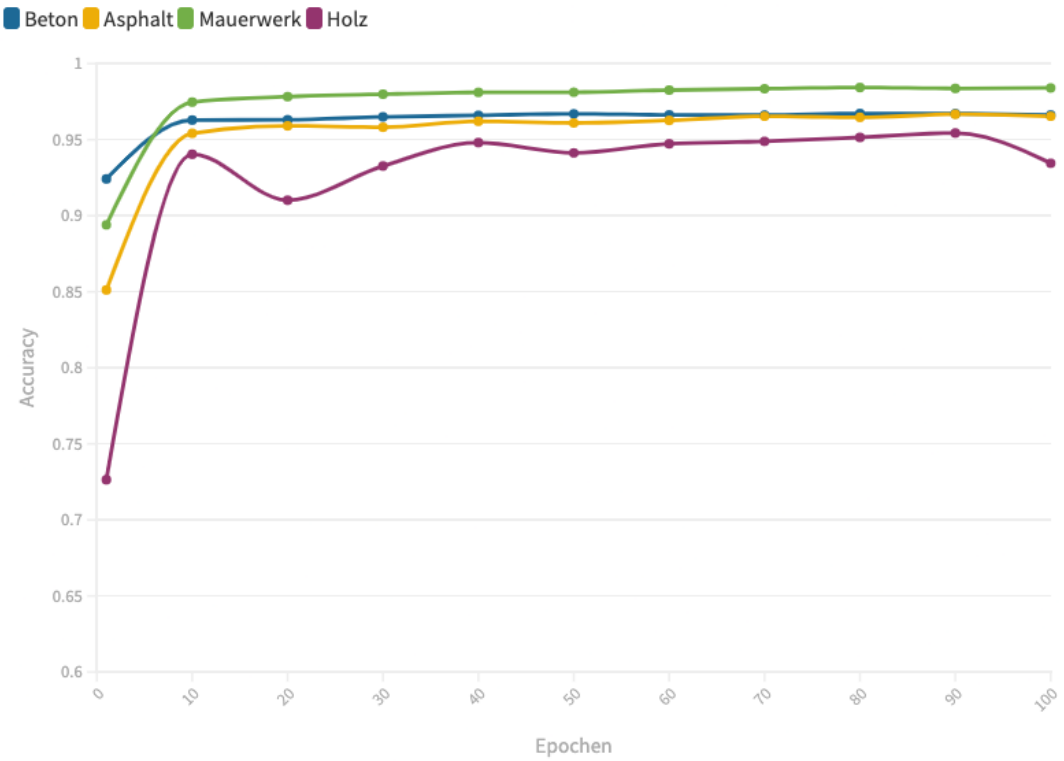


Abb. 7.1: Liniengraph mit der Genauigkeit (Accuracy Graph) aufgesplittet nach Oberflächenmaterial je nach Epoche

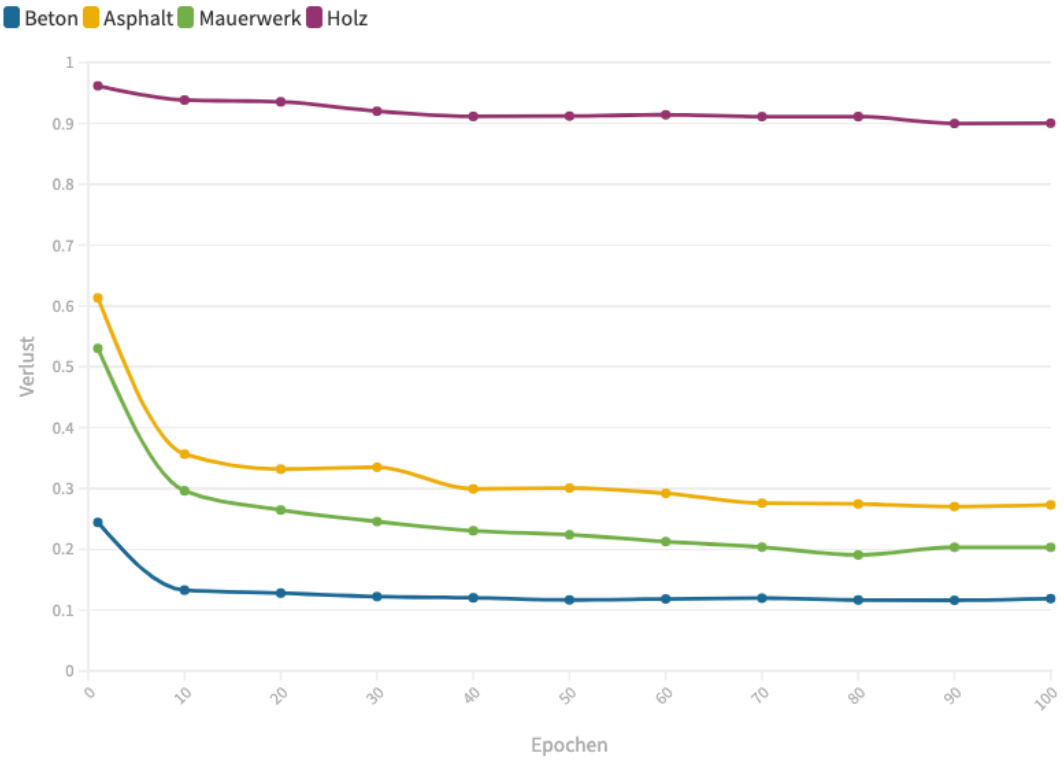


Abb. 7.2: Liniengraph der Verlust-Kurve aufgesplittet nach Oberflächenmaterial je nach Epoche

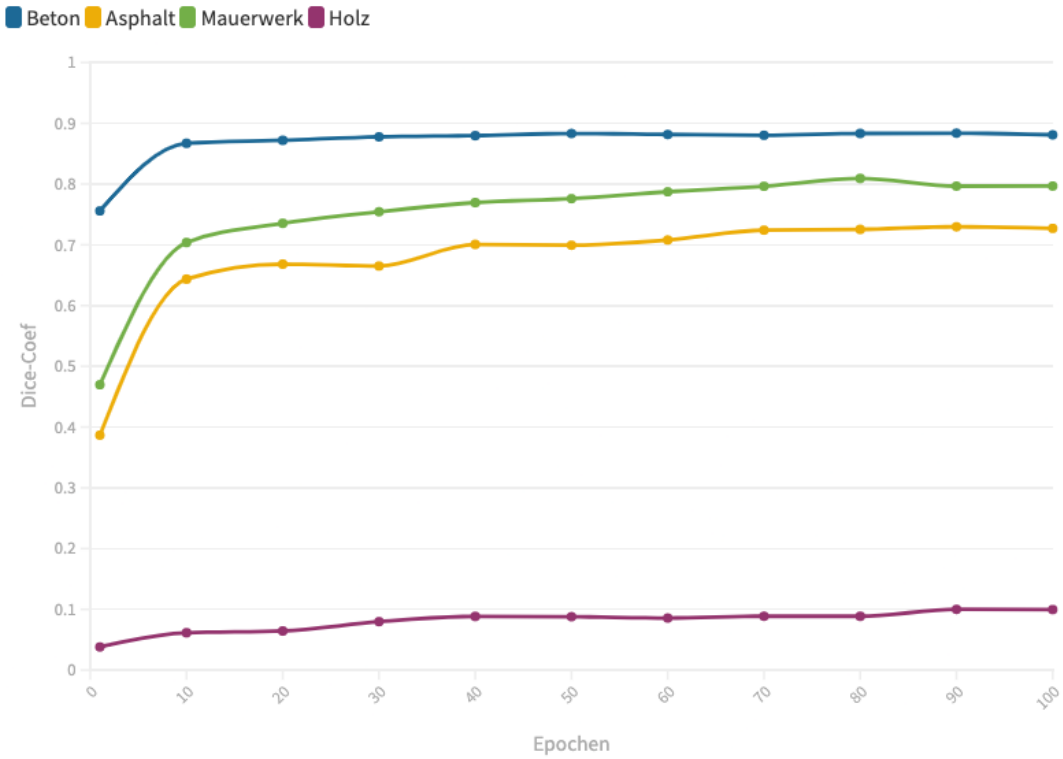


Abb. 7.3: Liniengraph mit der Entwicklung des F1-Score (Dice-Koeffizienten) aufgesplittet nach Oberflächenmaterial je nach Epoche

7.3 Materialspezifische Ergebnisse

In diesem Kapitel wird auf die Ergebnisse, Erkenntnisse und Feststellungen zu den einzelnen Oberflächenmaterialien eingegangen. Zudem werden die Ergebnisse verglichen und mögliche Zusammenhänge ermittelt. Die Resultate werden anschließend interpretiert und entstandene Probleme erörtert.

7.3.1 Asphalt

Im Gegensatz zum Oberflächenmaterial Beton (siehe Abbildung 7.6) enthalten die RGB-Bilder im Datenset Asphalt deutlich mehr Störfaktoren (siehe Abbildung 7.4. Dies liegt zum einen an der physischen Zusammensetzung, zum anderen der daraus resultierenden Schattierungen. Aus Abbildung 7.4 können in der letzten Reihe die Ergebnisse der Segmentierung mit den dazugehörigen Originalbilder entnommen werden. Die Ergebnisse dieser Darstellung belegt, dass das neuronale Netz auch für das Oberflächenmaterial Asphalt den Riss klar erkennt und in den Grundzügen richtig segmentiert. Die Risse sind klar zu registrierten und auch in ihrer Form und Dicke durchwegs richtig. Bei feinen Rissen, z. B. in den ganz rechten Abbildungen zu erkennen, kommt es vereinzelt zu Problemen. Die Differenzierung zwischen Risspixel und Nichtrisspixel verschafft der Architektur des erstellten neuronalen Netzes Schwierigkeiten.

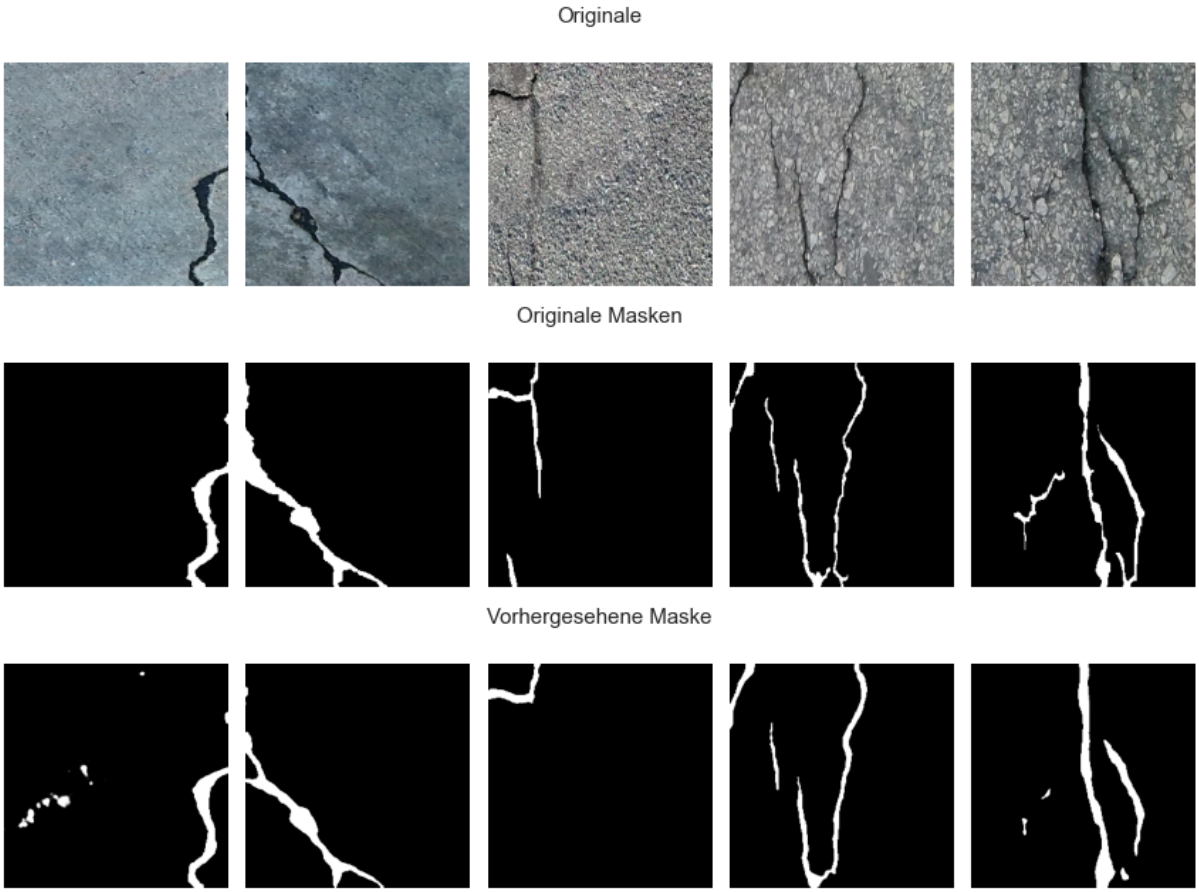


Abb. 7.4: Zufällig ausgewählte Bilder im Datenset Asphalt mit der dazugehörigen Maske und die vom Netzwerk erzeugte Vorhersage

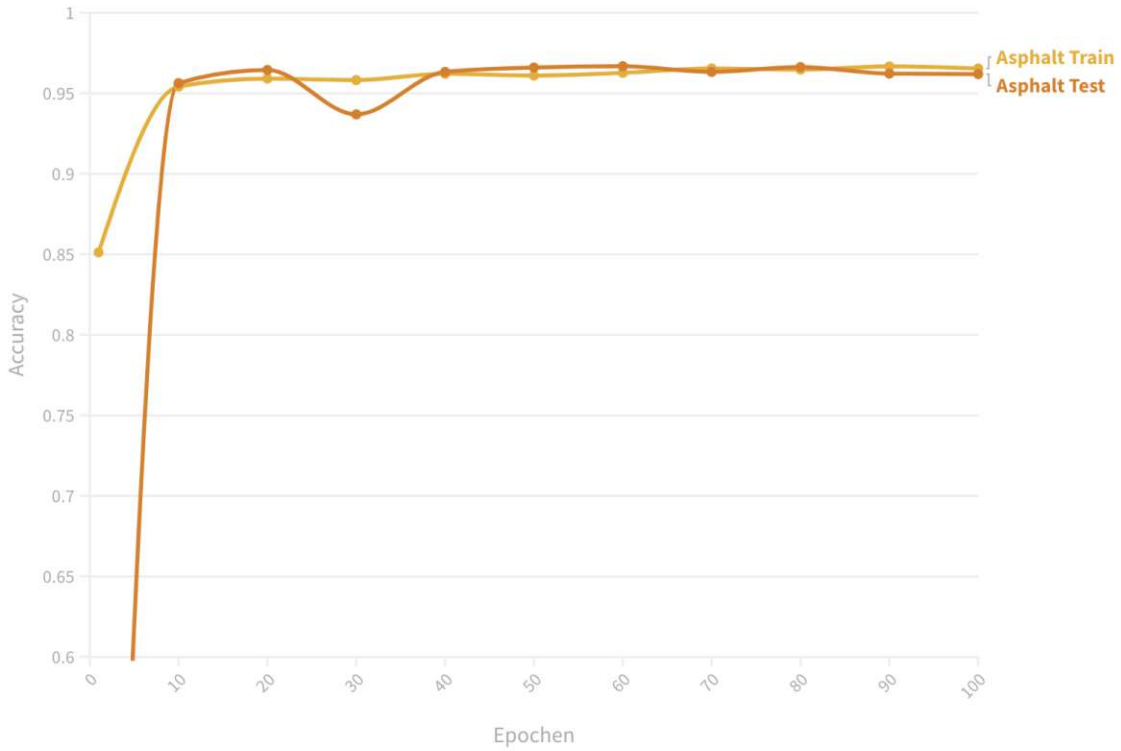
Die Ergebnisse mit dem Datenset Asphalt, welche aus der Tabelle 7.1 entnommen werden können, zeigen im Wesentlichen gute Resultate in Bezug auf die Qualität des neuronalen Netzes auf. In den Resultaten kann jedoch abgelesen werden, dass im Vergleich zum Datenset Beton Störfaktoren, beispielsweise Körnungen und Schattierungen, die Ergebnisse etwas verschlechtern. Die Genauigkeit von 96,66 % ist etwas niedriger als die von Beton – Jedoch ist der F1-Score um mehr als 10 % schlechter. Aus diesem Grund ist die Gesamtgenauigkeit der Erkennung von Rissen auf Asphalt merklich schlechter als die auf Beton. Insbesondere bei einer Präzision von 73,39 % ist das Ergebnis nur als befriedigend einzustufen. Aus diesem Grund kann daraus abgeleitet werden, dass erkannte Risspixel, oftmals nicht Risspixel sind. Demnach ist die Falsch-Positiv-Rate hoch. Schlussfolgernd zeigen die Ergebnisse, dass es Verbesserungspotential gibt.

Aus Tabelle 7.2 können die wichtigsten Maßzahlen mit der Differenzierung zwischen „Richtig“ und „Falsch“ entnommen werden. Die Daten aus dieser Tabelle bestätigen die Vermutung, dass erkannte Risspixel oftmals nicht Risspixel sind. Denn die Richtigkeit ist durchwegs bei allen Kennzahlen nur befriedigend. Pixel, welche in der Tat keine Risspixel sind, werden auch als solche erkannt. Die Richtig-Negativ-Rate liegt daher sehr hoch.

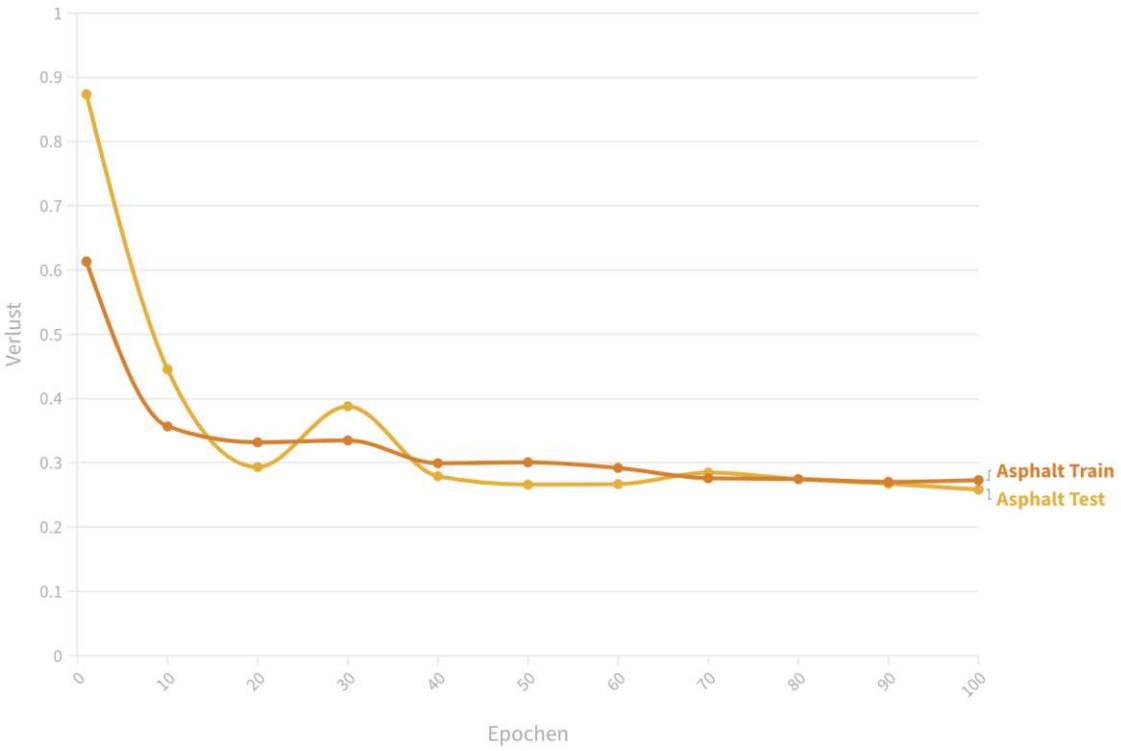
Tab. 7.2: Confusion-Matrix-Resultate des Durchlaufes mit dem Asphaltdatenset

	Precision %	**Recall %**	**F1-Score %**
Falsch	99	98	98
Richtig	73	80	76

In Abbildung 7.5 sind die Genauigkeitskurve und die Verlustkurve abgebildet. Für das Datenset Asphalt ist auffällig, dass die verwendete Netzwerkarchitektur in Bezug auf die Graphendarstellung sehr gute Ergebnisse erzeugt. Insbesondere die sehr aussagekräftige Verlustkurve verläuft ideal. Es ist weder ein Overfitting, noch ein Underfitting abzulesen. Basierend auf diesen Tatsachen lässt sich schließen, dass die verwendete Netzwerkarchitektur sehr gut für Asphalt funktioniert und die Ergebnisse dementsprechend aussagekräftig sind.



(a) Accuracy-Kurve am Datenset Asphalt



(b) Verlust-Kurve am Datenset Asphalt

Abb. 7.5: Graphen aus den Ergebnissen mit dem Datenset Asphalt

7.3.2 Beton

Das Oberflächenmaterial Beton beinhaltet im Vergleich zu den anderen Oberflächenmaterialien kaum Störfaktoren. Kennzeichnend ist die meist einfärbige Erscheinungsweise, welche es für das neuronale Netz leicht macht, ein Risspixel von einem Nicht-Risspixel zu unterscheiden. In Abbildung 7.6 können die bildbasierten Ergebnisse der Bildsegmentierung entnommen werden. Es ist ein deutlicher Riss auf der „Predicted-Maske“ zu verzeichnen, welches auf das Funktionieren der Architektur hinweist. Insbesondere der Vergleich von „Predicted-Maske“ und „Originaler-Maske“ zeigt die guten Ergebnisse der Bildsegmentierung auf. Risse werden sowohl in ihrer Ausrichtung, als auch Dicke richtig eingezeichnet. In der linken Bilderreihe ist zu erkennen, dass auch kleine Einwölbungen und Löcher im Beton als Riss erkannt und dementsprechend segmentiert werden. Auf Basis der erstellten Bilder ist es ersichtlich, dass die Risse auf Beton eindeutig erkannt werden. Das neuronale Netz tendiert eher dazu, dass Störfaktoren hinzugezählt, anstatt Risspixel weggelassen werden.

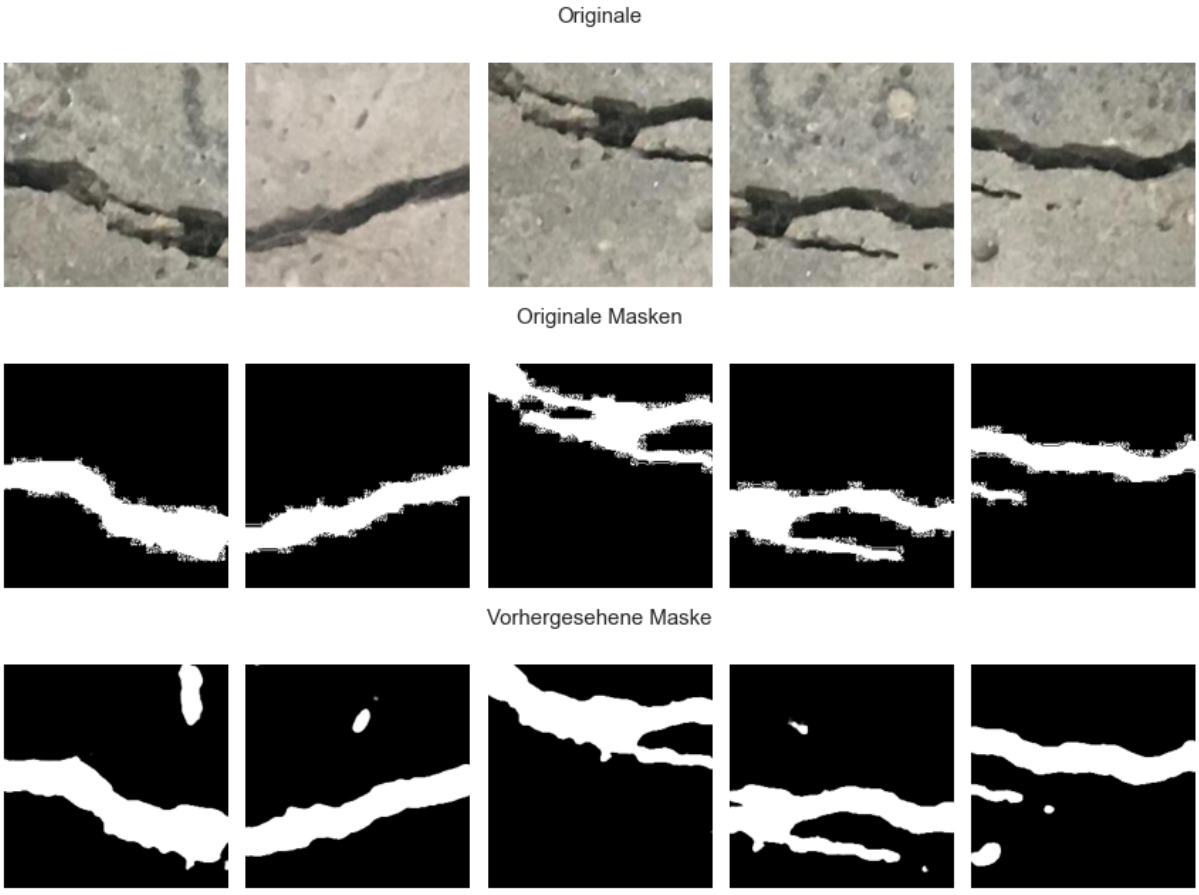


Abb. 7.6: Zufällig ausgewählte Bilder im Datenset Beton mit der dazugehörigen Maske und die vom Netzwerk erzeugte Vorhersage

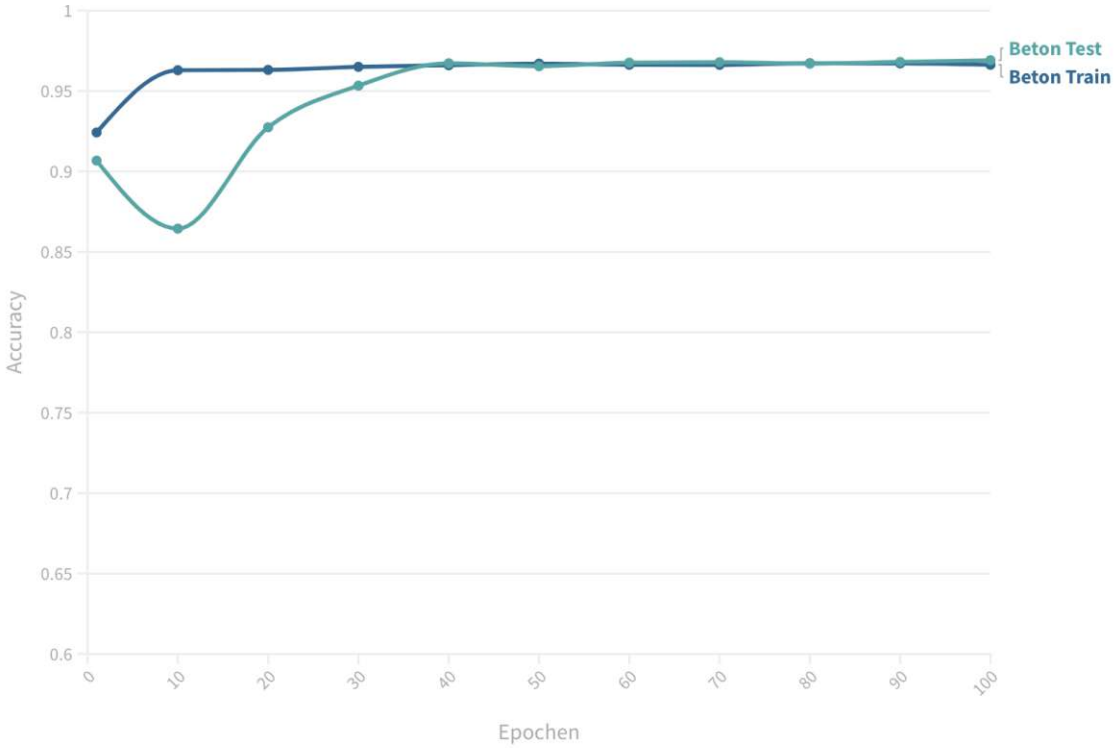
Die Ergebnisse der Tabelle 7.1 zeigen im Wesentlichen gute Resultate in Bezug auf die Qualität des neuronalen Netzes auf. Die allgemeine Genauigkeit liegt bei 96,91 % und der F1-Score bei 88,26 %, die schlussfolgernd als sehr gute Wahrscheinlichkeitsvorhersage gedeutet werden können. Die Möglichkeit, dass es zu keiner negativen Beurteilung kommt, liegt bei 85,57 % und ist somit der niedrigste Wert der Ergebnisskennzahlen.

Aus der Tabelle 7.3 können die wichtigsten Maßzahlen mit der Differenzierung zwischen „Richtig“ und „Falsch“ entnommen werden. In Anlehnung an diese Zahlen kann geschlossen werden, dass durchschnittlich die Wahrscheinlichkeit, dass eine negative Beurteilung korrekt ist, größer ist, als eine richtige Beurteilung. Demnach ist die Wahrscheinlichkeit, dass ein positives Ergebnis falsch ist, sehr hoch. Vergleichbare Resultate zeichneten sich auch beim Datensatz Asphalt ab.

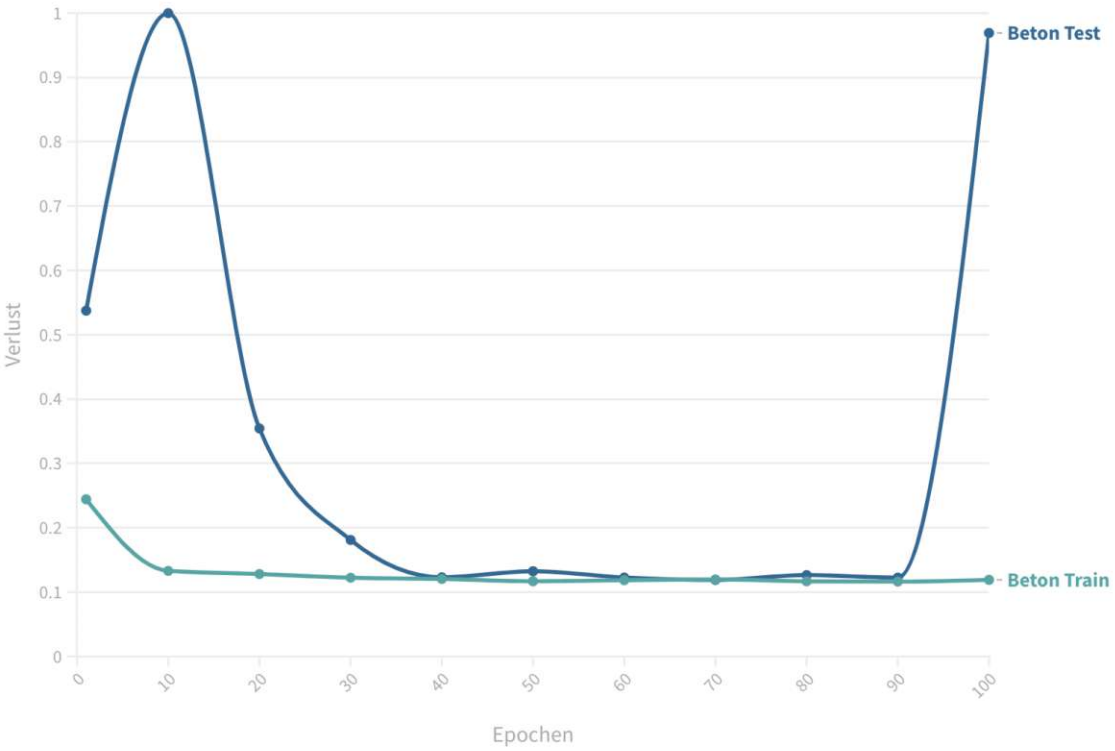
Tab. 7.3: Confusion Matrix Resultate des Durchlaufes mit dem Betondatenset

	Precision %	**Recall %**	**F1-Score %**
Falsch	98	99	98
Richtig	91	86	88

Aus den Abbildungen 7.7 ist zum einen in A die Genauigkeitskurve nach Epochen abgebildet, zum anderen die Verlustkurve in Abbildung 7.7b. Es ist zudem noch zwischen Trainingsset und Testset differenziert. Zu erkennen ist, dass in Bezug auf die Genauigkeit sowohl das Training, als auch die Testung ideal verlaufen. Insbesondere nach 30 Epochen stellt sich bei beiden Kurven ein Equilibrium ein und flachen ab.
In Abbildung 7.7b ist ein Graph mit der Verlustkurve zu sehen. Dieser ist sehr aufschlussreich, da hier klar ein abruptes Overfitting zu erkennen ist. In den letzten Epochen steigt die Testkurve stark an, welches ein klares Indiz für das genannte Problem ist. Das bedeutet, dass sich das Modell den Ergebnissen überangepasst hat. Es repräsentiert ein Auswendiglernen der Daten und wird näher im Kapitel 5.4 und 7.1 beschrieben.



(a) Accuracy-Kurve am Datenset Beton



(b) Verlustkurve am Datenset Beton

Abb. 7.7: Graphen aus den Ergebnissen mit dem Datenset Beton

7.3.3 Mauerwerk

Mauerwerke enthalten viele Störfaktoren, welche es erheblich erschweren können, Risspixel von Nicht-Risspixel zu unterscheiden – Zumal Schattenfugen auch dieselbe Form wie Risse, welche sich in den meisten Fällen entlang der Mörtelfugen entlangspannen haben. Bei den Bildern im Datensatz Mauerwerk handelt es sich immer um heterogene Bilder, da sich der Ziegel vom Mörtel grafisch abhebt. Es sind Daten von Rissen im Mörtel, als auch im Ziegel enthalten. In Anlehnung an diese Fakten kann darauf geschlossen werden, dass das neuronale Netz mit mehreren Störfaktoren umzugehen hat. Aus den Darstellungen in der Abbildung 7.8 können die bildbasierten Ergebnisse der Segmentierung mit den dazugehörigen Testbildern entnommen werden. Es sind erstaunliche Ergebnisse abzulesen, da die Risse sehr gut segmentiert werden konnten. Sowohl die Verortung, als auch die grundlegende Form und Dicke konnten richtig grafisch vorhergesagt werden. Es ist auch zu erkennen, dass Schatten kaum zu einer Beeinträchtigung der Ergebnisse geführt haben. Einzig und alleine sehr feine Risse, z. B. im ersten Bild erkennbar, führen zu Schwierigkeiten.

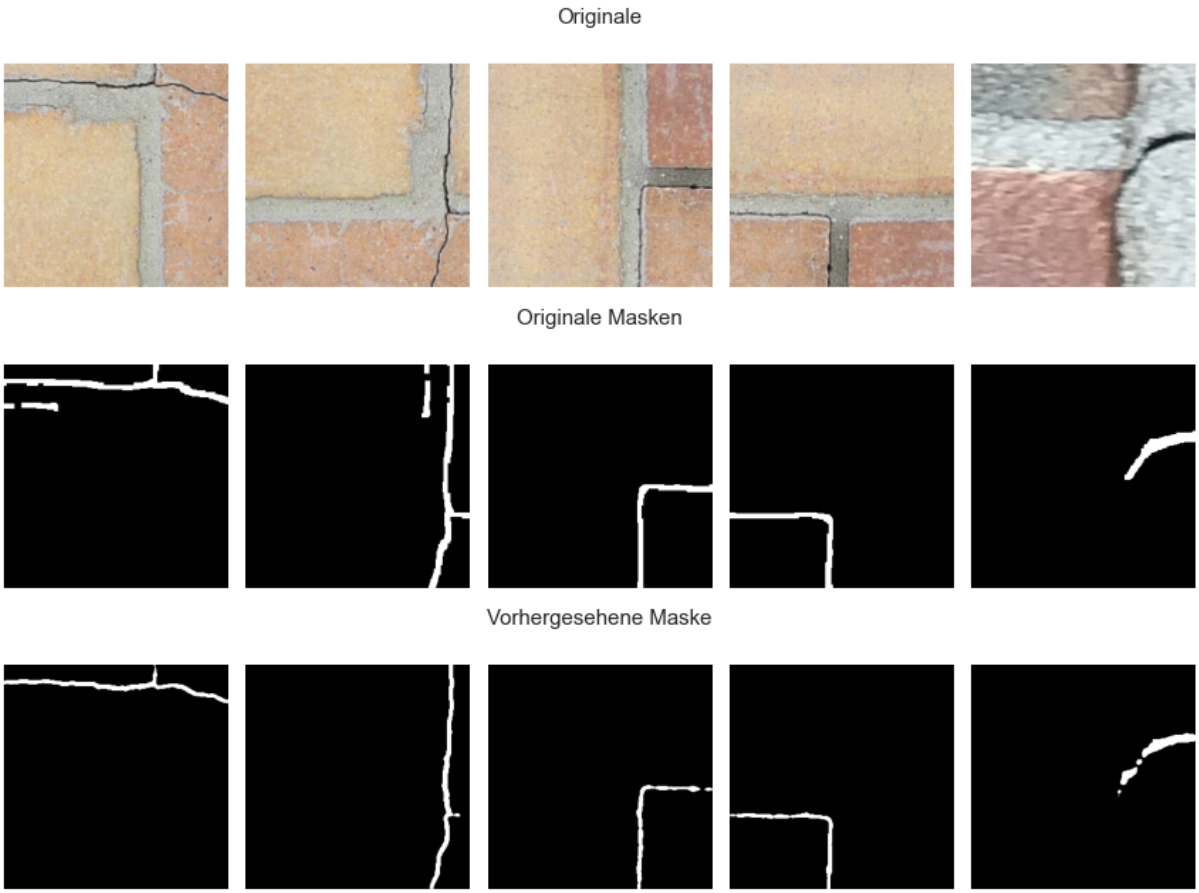


Abb. 7.8: Zufällig ausgewählte Bilder im Datenset Mauerwerk mit der dazugehörigen Maske und die vom Netzwerk erzeugte Vorhersage

Die zahlenbasierten Ergebnisse können aus Tabelle 7.1 entnommen werden. Allgemein können die Kennzahlen mit den bildbasierten Ergebnisse bestätigt werden. Risspixel werden im Wesentlichen richtig erkannt und akkurat verortet. Grundsätzlich wurden auch sichtbar bessere Ergebnisse erzielt, als mit dem Datensatz des Oberflächenmaterials Asphalt. Die Genauigkeit von 98,65 % liegt minimal, dennoch höher als die von Beton, welche 96,91 % beträgt. Dem

ungeachtet liegt der F1-Score bei 85,66 %. Dies belegt, dass die Gesamtgenauigkeit mit dem Datenset Mauerwerk unter der des für Beton liegt. Nichtsdestoweniger sind die Resultate besser als die des Oberflächenmaterials Asphalt und Holz.

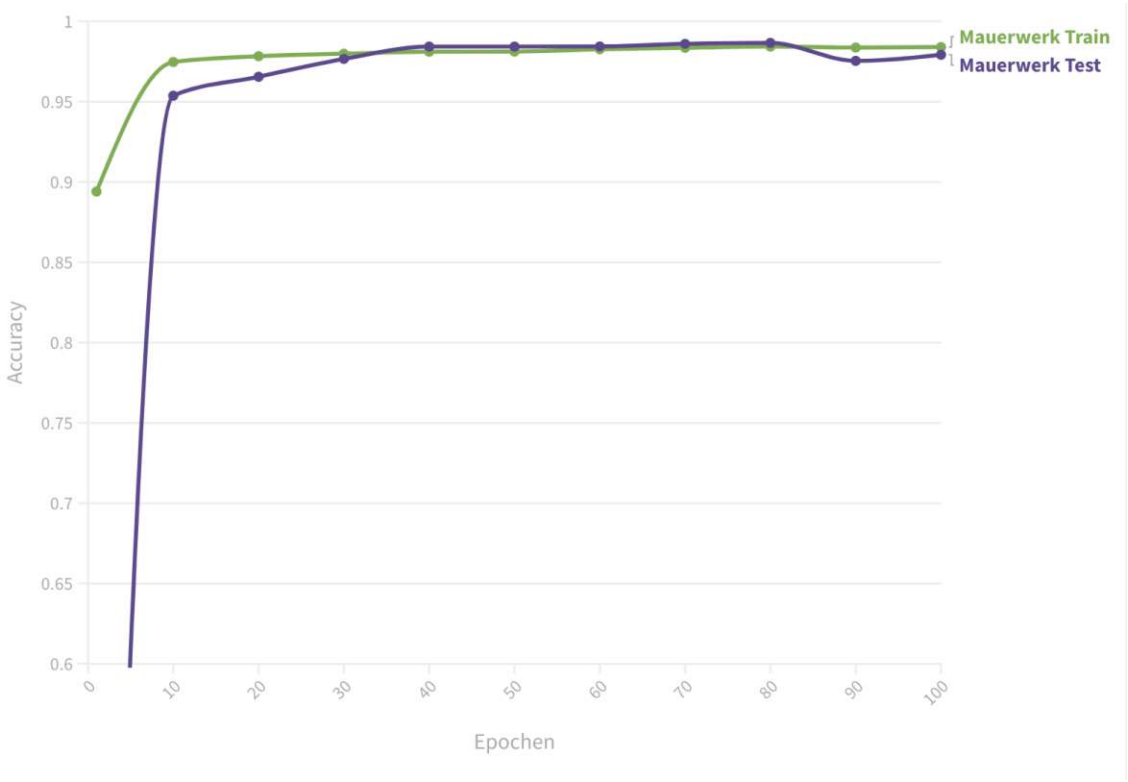
Die Wahrnehmung, dass Schattenfugen zu falschen Ergebnissen führen können, kann aus dem Recall abgelesen werden. Dennoch wurden auch hier mit 79,14 % durchaus gute Genauigkeitsergebnisse erzielt. Daraus lässt sich schließen, dass Schattenfugen nicht zu falsch positiven Ergebnissen führen, welches wiederum im F1-Score verifiziert werden könnte.

Aus Tabelle 7.4 können die wichtigsten Messzahlen mit der Differenzierung zwischen „Richtig“ und „Falsch“ entnommen werden. Aus den Daten ist abzulesen, dass es eher zu keiner Segmentierung des Risses kommt, als dass ein Riss falsch Erkannt wird. Nicht-Risspixel werden als solche sehr gut erkannt und dementsprechend auch nicht segmentiert – Nur Risspixel werden in der Maske eingezeichnet. Es kann darauf geschlossen werden, dass es manchmal zu einer Nicht-Erkennung eines Risspixel kommt. Haarrisse werden somit nicht erkannt und auch nicht als solche eingetragen.

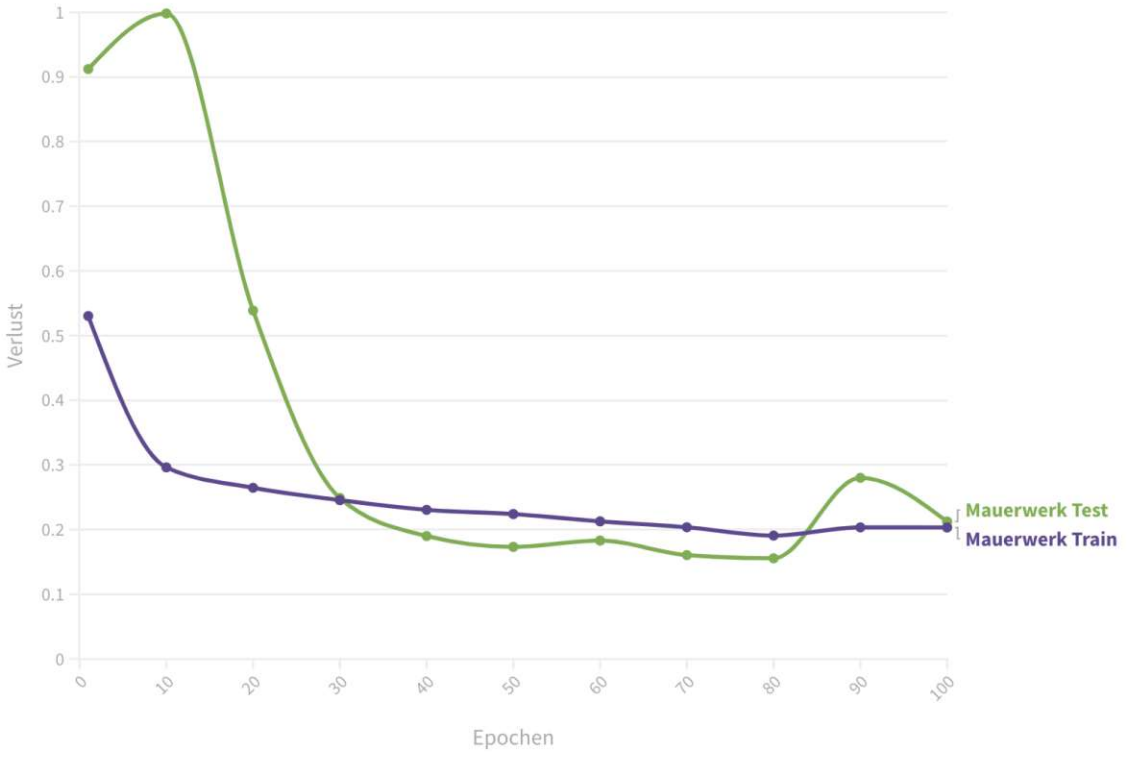
Tab. 7.4: Confusion Matrix Resultate des Durchlaufes mit dem Mauerwerkdatenset

	Precision	**Recall**	**F1-Score**
Falsch	99%	100%	99%
Richtig	93%	79%	86%

Es sind in Abbildung 7.9a die Genauigkeit und in Abbildung 7.9b die Verlustkurve dargestellt. Ähnlich dem Datenset Asphalt sind sehr gute Ergebnisse zu vermerken. In der Verlustkurve ist ein beginnendes Overfitting zu vermerken, welches jedoch mit einer größeren Datenmenge behoben werden könnte. Es lässt sich auch aus den Graphen schließen, dass sowohl das Trainingdatenset als auch das Testdatenset die Informationsanforderungen sehr gut abdecken. Eine unzureichende Daten- und Informationslage wäre aus den Graphen mit einer sehr unruhigen Funktion mit vielen ausschlagenden Punkten ablesbar.



(a) Accuracy-Kurve am Datenset Mauerwerk



(b) Verlustkurve am Datenset Mauerwerk

Abb. 7.9: Graphen aus den Ergebnissen mit dem Datenset Mauerwerk

7.3.4 Holz

Das Oberflächenmaterial Holz bringt viele Schwierigkeiten für das neuronale Netz mit sich. Angesichts der Tatsache, dass es sehr viele unterschiedliche Farben aufgrund der Maserung, Astlöcher und diverser Holzfehler mit sich bringt, können dadurch Risse vereinzelt nur sehr schwierig erkannt werden. Im Weiteren können durch die genannten Fehler auch Schattierungen auftreten, die wiederum die Komplexität der Bilder erhöhen. Anknüpfend an die Tatsache, dass das Material Holz Maserungen aufweist, verlaufen Risse oftmals entlang dieser, da sich an dieser Lage eine geringere Festigkeit verzeichnen lässt [59]. Es ist auch hervorzuheben, dass nur ein kleiner Prozentsatz aller Risse im Holz eine Gefahr für die Statik oder den Allgemeinzustand eines Bauwerkes mit sich zieht.

Aus Abbildung 7.10 kann abgelesen werden, dass Risse zwar in ihrer ungefähren Lage und Ausrichtung richtig segmentiert werden können, jedoch kann weder die Dicke noch die Form richtig erkannt werden. Wird das Original mit der segmentierten vorhergesagten Maske verglichen, kann daraus gedeutet werden, dass die Kernmaserung des Holzes als Riss erkannt und segmentiert wird. Demnach kann die These aufgestellt werden, dass längliche, dunkle Holzmaserungen als Riss erkannt werden.

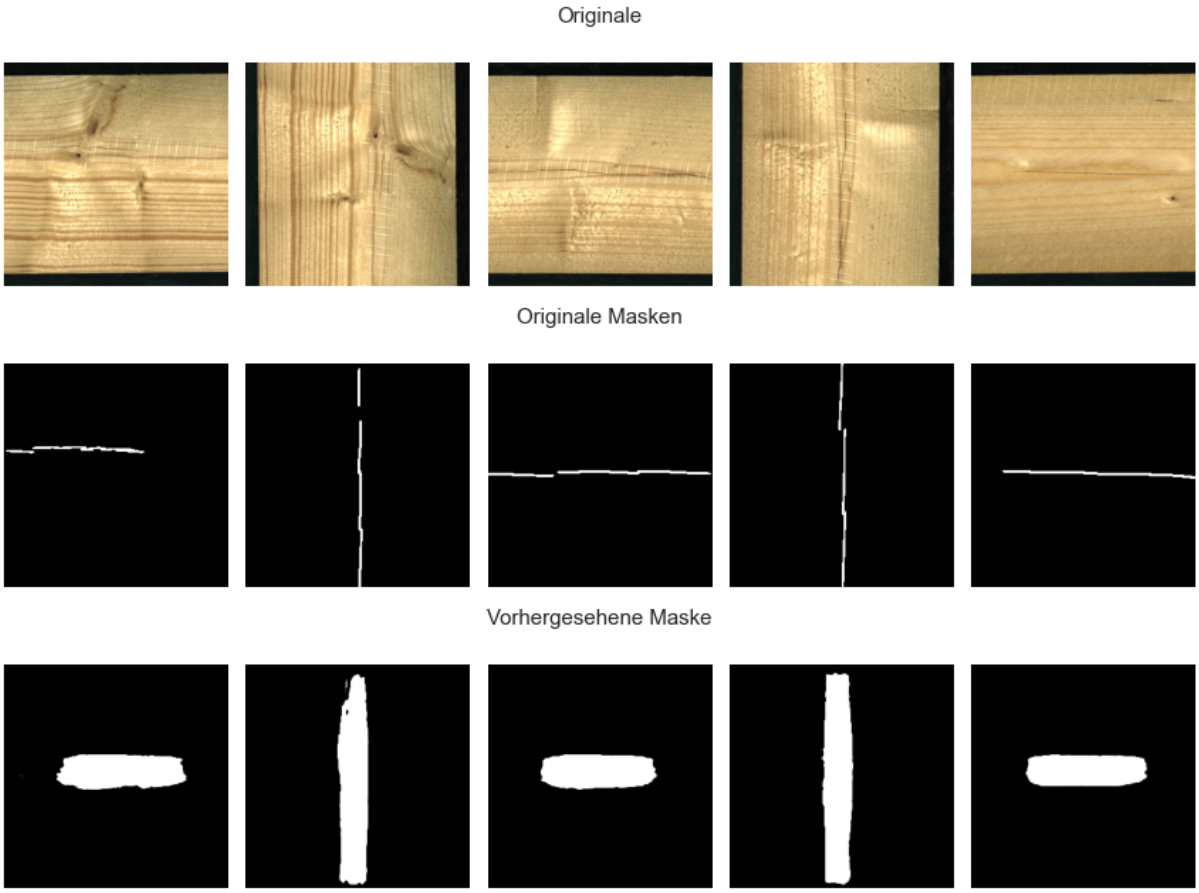


Abb. 7.10: Zufällig ausgewählte Bilder im Datenset Holz mit der dazugehörigen Maske und die vom Netzwerk erzeugte Vorhersage

Das Oberflächenmaterial Holz erzielte im Vergleich zu den anderen ausgetesteten Datenset das mit Abstand schlechteste Allgemeinergebnis. Die Daten können aus Tabelle 7.1 abgelesen werden. Die Genauigkeit liegt bei 10 %92,02, welches als sehr irreführend interpretiert werden

kann. Sinnbildlich dargestellt: Wenn jedes Pixel auf dem RGB-Bild als Risspixel gedeutet werden würde, dann wäre aufgrund der Tatsachen, welche im Abschnitt 7.1 aufgezeigt wurden, die Genauigkeit bei 100 %, da jeder Risspixel erkannt wird. Demnach kann auch die hohe Prozentzahl von 93,32 % bei der Spezifität davon abgeleitet werden, dass ein Nicht-Risspixel tatsächlich ein solches ist. Es kann daher auf eine sehr hohe Falsch-Positiv-Rate geschlossen werden. Die Exaktheit der Genauigkeit, sprich die „Precision“ zeichnet ein klareres Bild ab und bestätigt die Vermutungen der bildbasierten Ergebnisse. Diese liegt bei rund 7 % und muss als sehr niedrig angesehen werden. Es lässt sich daraus schließen, dass ein positives Ergebnis in den meisten Fällen nicht korrekt ist. Ein vermeintliches Risspixel ist somit vielfach als Nicht-Risspixel einzuordnen. Die Kennzahl, welche für die Precision erzielt wurde, wirkt sich auch auf den F1-Score aus, welcher bei 12,75 % ebenfalls sehr niedrig ist. Schlussfolgernd ist die allgemeine Gesamtgenauigkeit in allen Resultaten als sehr schlecht einzustufen.

Aus Tabelle 7.5 können die wichtigsten Maßzahlen mit der Differenzierung zwischen „Richtig“ und „Falsch“ entnommen werden. Schlussfolgernd kann daraus geschlossen werden, dass die richtige Zuordnung eines erkannten Risspixels in vielen Fällen nicht stimmt. Insbesondere die Präsizion der richtigen Ergebnisse ist mit 7 % sehr schlecht. Die treffsicheren Ergebnisse von durchwegs mehr als 90 % in den Kategorien Precision, Recall und F1-Score, der Nicht-Risspixel sind darauf zurückzuführen, dass die Wahrscheinlichkeit, dass Risspixel erkannt werden um einiges höher liegt. Dementsprechend kann daraus abgeleitet werden, wenn ein Nicht-Risspixel erkannt wird, die Wahrscheinlichkeit sehr hoch ist, dass dies auch stimmt.

Tab. 7.5: Confusion Matrix Resultate des Durchlaufes mit dem Holzdatenset

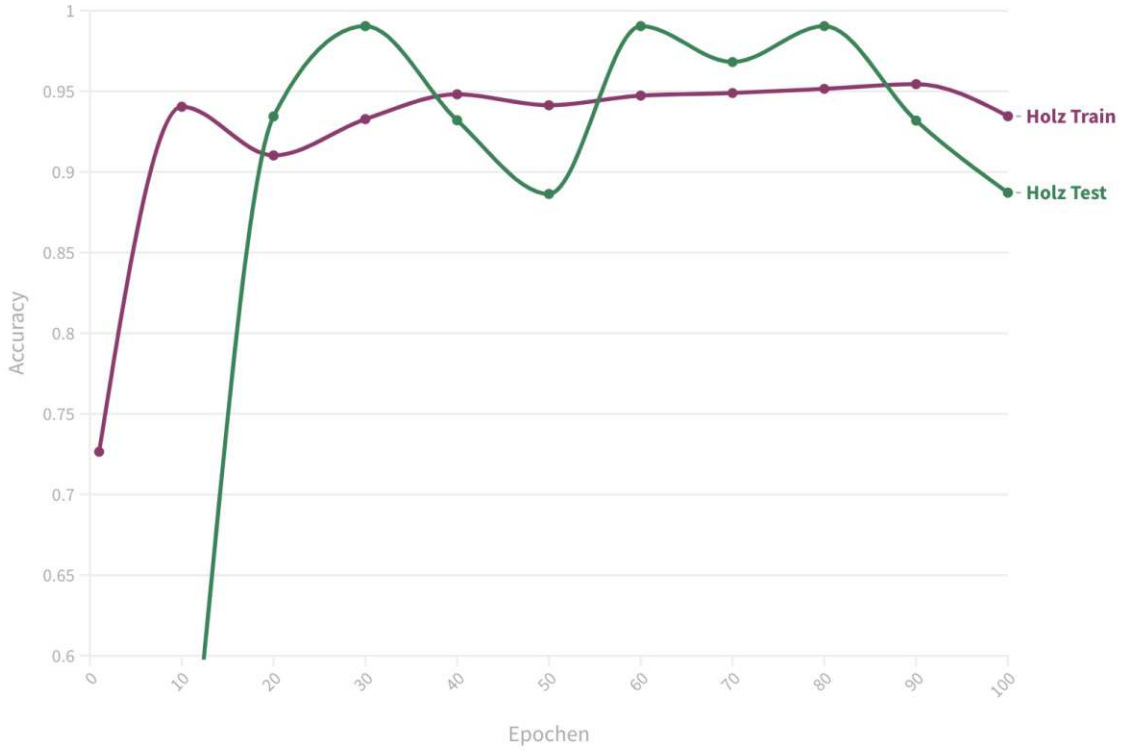
	Precision %	**Recall %**	**F1-Score %**
Falsch	100	92	96
Richtig	7	61	13

Abbildung 7.1 zeigt, dass Holz am schlechtesten abgeschnitten hat. Basierend auf diesen Daten kann gesagt werden, dass sich ein leichtes Overfitting eingestellt hat, welches am Abfallen des Graphen in den letzten Epochen abgelesen werden kann. Trotz Hyperparameteroptimierung konnte dies in dieser Arbeit nicht verhindert werden. Darüber hinaus kann auch abgelesen werden, dass es nach dem starken Anstieg in den ersten zehn Epochen zu einem Abfall in der Genauigkeit gekommen ist. Dies hat den Hintergrund, dass die erstellte neuronale Architektur Schwierigkeiten hatte, die optimalen Hyperparameter zu finden. Dieses Problem konnte jedoch nur für den Datensatz Holz verzeichnet werden.

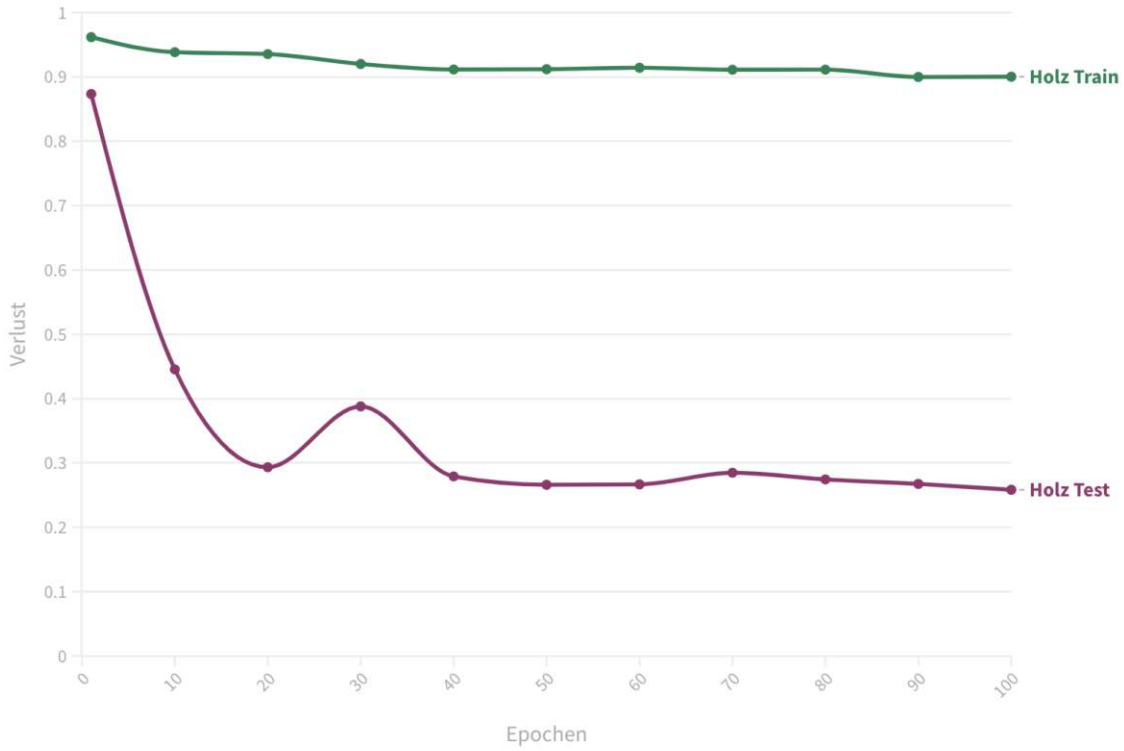
Holz hat im Rahmen dieser Diplomarbeit sehr schlecht in der Verlustkurve abgeschnitten (siehe Abbildung 7.2. Es ist unschwer zu erkennen, dass diese um 80 % höher als die der restlichen Datensätze liegt. Sie verzeichnet auch nicht den klassischen Verlauf, welches mehrere Gründe haben kann. Zum einen kann die verwendete Verlustfunktion ungeeignet für dieses Datenset sein, zum anderen kann die verwendete Netzwerkarchitektur nicht optimal sein.

Ein weiterer Beweis für Untauglichkeit der gewählten Architektur auf dem Material Holz ist in Abbildung 7.11 zu sehen. Der Verlustkurve in 7.11b zeigt, dass bessere Ergebnisse mit dem Testset erzielt werden als im Trainingsset. Das bedeutet, dass es für das Modell leichter ist, Vorhersagen mit dem Testset zu treffen und tritt auf, wenn zu wenig Daten für das zu lösende Problem vorhanden sind. Ein weiterer Punkt ist die sehr hohe und flache Verlustkurve mit dem Trainingset und deutet auf ein Underfitting hin. Das Modell kann mit dem verwendeten Trainingsset nicht lernen und bestätigt die These, dass zu wenig Daten vorhanden sind. Die Genauigkeitskurve in 7.11a ist auch sehr unregelmäßig, das wiederum auf die Untauglichkeit für das Oberflächenmaterial Holz schließen lässt. Die erzielten Ergebnisse sind sehr inkonsistent und

das Modell hat große Schwierigkeiten Vorhersagen zu treffen. Zusammenfassend ist darauf zu schließen, dass die gewählte Architektur für Holzoberflächen nicht geeignet ist.



(a) Accuracy-Kurve am Datenset Holz



(b) Verlustkurve am Datenset Holz

Abb. 7.11: Graphen aus den Ergebnissen mit dem Datenset Holz

Kapitel 8

Zusammenfassung und Diskussion

In diesem Abschnitt wird die aufgestellte Forschungsfrage wieder aufgegriffen:

> *Inwiefern beeinflusst das Oberflächenmaterial die Ergebnisse der Risssegmentierung und wie können die Resultate im Allgemeinen verbessert werden anhand Open-Source-Datenbanken?*

Sie wird mit den Ergebnissen der Untersuchungen beantwortet. Auf den Erkenntnissen aufbauend, werden Optimierungsvorschläge und Empfehlungen angegeben.

8.1 Bildsegmentierung

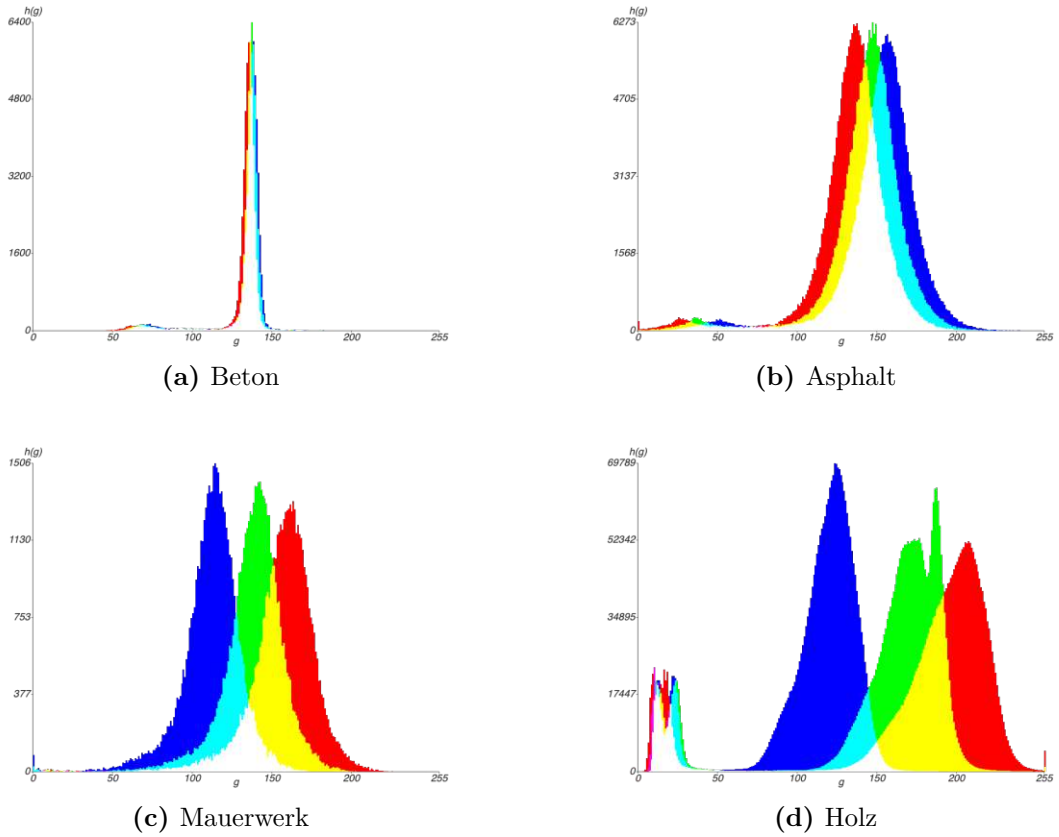
Das Oberflächenmaterial beeinflusst die Bildsegmentierung stark. Im Allgemeinen sind Risse auf Materialien, welche Bilder mit einer starken RGB-Streuung erzeugen, schlecht von einer Segmentierung zu erfassen. In Abbildung 8.1 sind Histogramme stellvertretend für alle Oberflächenmaterialien ersichtlich. Sie geben die Anzahl der Pixel in Abhängigkeit zur Helligkeit an. Unterteilt werden die Histogramme in die Farbkanäle: R = rot, G = grün, B = blau. Holz hat ein weit gestreutes Histogramm und erzeugt somit auch schlechte Genauigkeitsergebnisse bei der Bildsegmentierung. Demnach ist es für den Algorithmus schwierig zu erkennen, ob es sich richtigerweise um einen Risspixel oder Nicht-Risspixel handelt. RGB-streuungsreiche Materialien, z. B. Asphaltoberflächen, sind somit weniger gegenüber Störfaktoren resistent. Dadurch werden exemplarisch Spinnen oder Schatten als Risse erkannt und als solche segmentiert. Überraschenderweise werden Risse auf Mauerwerk besser segmentiert als auf Asphalt. Eine mögliche Erklärung hierfür ist, dass im Datensatz des besser performenden Oberflächenmaterial weniger Störfaktoren enthalten sind. Die Körnung in Asphalt führt zu einer höheren falsch-positiv Rate, die wiederum die allgemeinen Ergebnisse beeinflusst. Zusätzlich verlaufen auf diesem Material Risse in alle Richtungen ohne vorhersehbarer Logik.

8.2 Verbesserungen und Empfehlungen

In diesem Kapitel werden Verbesserungsvorschläge thematisiert und deren mögliche Auswirkungen auf die Kennwerte und Messzahlen erläutert. Es wird analog vorgegangen, das bedeutet, dass mit dem Datensatz begonnen wird. Anschließend wird das Transfer-Learning vorgestellt, welches Prozesse erleichtert und verbessert hätte. Im letzten Schritt werden Verbesserungen in Bezug auf den Trainingsprozess vorgestellt.

8.2.1 Datensatz

Die Datensatzgröße hat unmittelbare Auswirkungen auf die Qualität des Ergebnisses. Große KI Projekte, wie die von OpenAI OpenAI [43], einem Non-Profit-Unternehmen, welches sich mit der Entwicklung von künstlichen Intelligenzen beschäftigt, haben eine Datensatzgröße von mehreren



(a) Beton **(b)** Asphalt

(c) Mauerwerk **(d)** Holz

Abb. 8.1: RGB-Histogramme der unterschiedlichen Oberflächenmaterialien

Millionen RGB-Bilder. Das Projekt „Clip“ von Radford et al. [56] arbeitet mit einem Dataset von 400 Millionen Text-Bilder Pärchen. Dementsprechend sind Ergebnisse zielgenauer als bei kleinen Datensätzen, da das neuronale Netz viele unterschiedliche Varianten, Möglichkeiten und Szenarien erlernen kann. Es ist folglich resistenter für den Alltag. Die verwendeten Open-Source-Datensätze beinhalteten oftmals nicht die benötigten Voraussetzungen für Realbedingungen. Je größer der Datensatz, desto größer auch der Aufwand, um die Daten in das richtige Format zu bringen. Generell kann gesagt werden, je mehr Bilder zum Training verwendet werden, desto besser werden auch die Ergebnisse des neuronalen Netzwerkes (siehe Abschnitt 8.2.1). Ähnlich dem menschlichen Gehirn werden künstliche Neuronen durch viel Input gestärkt und verbessert.Aufgrund der geschilderten Tatsachen werden für die Verarbeitung der Daten jedoch leistungsstarke Computer und sehr viel Speicherplatz benötigt, welche den Rahmen dieser Arbeit gesprengt hätte. Weiters wäre es sinnvoll nicht nur Idealfälle, beispielsweise gut ausgeleuchtete Bilder von Holzlängsschnitten zu verwenden, sondern auch Bilder, die ohne professionellen Equipment aufgenommen wurden. Basierend auf den Fakten, welche im Kapitel 6 genannt wurden, wäre es im Sinne der Alltagstauglichkeit sinnvoll, mehrere Holzarten einzubinden. Beispielsweise ist Kirschholz dunkler als Eichenholz und kann somit zu schlechteren Ergebnissen führen. Infolgedessen könnte das neuronale Netz zuverlässiger im Umgang mit alltäglichen Bildern umgehen. Die Resultate könnten durch eine Datensatzvergrößerung ebenso verbessert werden. Die sehr hoch verlaufende Verlustkurve resultiert auf einen zu kleinen Datensatz. Es sind keine Testungen mit Realaufnahmen durchgeführt worden, da dies nicht die Aufgabenstellung dieser Arbeit war.

Basierend auf den Erkenntnissen in Kapitel 7.2 ist eine Vergrößerung aller Datensätze notwendig, da sich beispielsweise für den Datensatz Beton ein Overfitting abgezeichnet hat. Um dem entgegenzuwirken sind mehr unterschiedliche Inputbilder notwendig, um dem neuronalen Netz mehr Szenarien beizubringen. Daraus ergibt sich ein neuer Mittelwert mit dem das Overfitting verringert werden kann.

8.2.2 Transfer Learning

Der Begriff Transfer Learning kommt aus dem maschinellen Lernen und beschreibt eine Technologie, welche bereits erlerntes Wissen auf eine neue, jedoch verwandte Aufgabe überträgt. Opbroek et al. [42] beschreibt in ihrer Forschungsarbeit zu biomedizinischen Bildsegmentierung, dass Transfer Learning im Bereich des überwachten Lernens unmittelbare positive Auswirkungen auf das Endergebnis der neu zu erlernenden Aufgabe hat. In der Experimentierreihe des niederländischen Teams konnte Transfer Learning das herkömmliches überwachte Lernen übertrumpfen und Klassifizierungsfehler um bis zu 60 % minimieren.

Insbesondere wenn die vorliegende Datensatzgröße des neuen Anwendungsfalles klein ist, kann Transfer Learning eine sinnvolle Möglichkeit sein, um treffsicherere Resultate zu erzielen. Die Sammlung an Bildern je Oberflächenmaterial beschränkt sich auf 200 RGB-Bilder, die signifikate Größe für diese Arbeit, welche jedoch im Vergleich zu größeren Projekten sehr klein ist.

Der Einsatz von Transfer Learning hätte jedoch den Rahmen der vorliegenden Diplomarbeit gesprengt. In Bezug auf weitere Forschungen zu diesem Thema wird aber zu dem Einsatz dieser Algorithmen dringend geraten.

8.2.3 Trainingsprozess

Basierend auf der Tatsache, dass die Erstellung des neuronalen Netzes ein sehr aufwendiger und zeitintensiver Prozess ist, ist der Trainingsprozess am Ende verglichen mit den anfänglichen Problemen reibungslos durchlaufen. Es dennoch sinnvoll die Tiefe des Netzwerkes so gering wie möglich zu halten. Zudem ist auch die Anzahl der benötigten Epochen für den Trainingsprozess genau zu überdenken. Je mehr Epochen trainiert werden, desto aussagekräftiger das Ergebnis, und desto länger dauert das Verfahren. Bei zukünftigen Experimenten wird empfohlen eine möglichst flache Architektur (wenig Ebenen) mit wenig Epochen zu wählen, um schnell Verbesserungen austesten zu können.

Eine Optimierung des Rechners und eine Erhöhung des Speichervolumens würde zu einer Beschleunigung der Durchläufe führen, während die Qualität der Ergebnisse unverändert bleiben. Eine Grafikkarte, welche Deep-Learning-Algorithmen unterstützt und schnelle Rechenarbeiten durchführen kann, ist zu empfehlen. Für Anwendungsfälle, für die eine Risserkennung ausreicht und keine Segmentierung notwendig ist, wird empfohlen, auf die Augmentierung des Datensets zu verzichten und die Anzahl der Trainingsepochen zu reduzieren. Es kann im weiteren Schritt auch an eine Anpassung der Lernrate auf 0,1 angedacht werden.

Ein gängiger Arbeitsschritt ist das Outsourcen der Rechnungsleistung an externe Server [24]. Amazon Web Services bietet eine umfangreiche Leistungsportfolio von der Beratung der benötigten Infrastruktur bis hin zur kompletten Übernahme der Rechenprozesse. Eine Auslagerung der Trainingsprozesse ist zeitsparend und trägt somit auch zur Qualität der Entwicklung von neuronalen Netzwerken bei.

Zudem wird auch eine große interne Festplatte oder zusätzlich externe Datenspeicher benötigt. Nicht nur die Datensätze verbrauchen mehrere Gigabyte an Speicherplatz, sondern auch das abgespeicherte Modell und die dabei entstandenen Daten. Das benötigte freie Datenvolumen

hängt stark von dem Anwendungsfall ab und kann von wenigen Gigabyte bis mehreren Terabyte reichen.

8.3 Fazit

In dieser Arbeit wurde die Bildsegmentierung von Rissen auf unterschiedlichen Oberflächenmaterialien miteinander verglichen. Eine optimale Netzwerk-Architektur wurde mithilfe von Experimenten ermittelt und anschließend an den Anwendungsfall angepasst. Das Ergebnis der Anwendung der angepassten U-NET Architektur auf vier verschiedene Materialien wurde analysiert und interpretiert. Insgesamt zeigt die Arbeit, dass die Bildsegmentierung von Rissen mithilfe von CNN auf unterschiedlichen Oberflächenmaterialien möglich ist, jedoch durch Störfaktoren beeinflusst wird. Allgemein hängt die Qualität der Ergebnisse mit der Streuung des RGB-Histogramms zusammen. Je größer die Streuung, desto schwieriger ist es, eine Segmentierung erfolgreich durchzuführen.

8.4 Weitere Forschungen

Weitgehend kann für vorangegangenen Forschungen dieser Arbeit geschlossen werden, dass für eine Verbesserung der Ergebnisse das Datenset allgemein vergrößert und verbessert werden muss. Eine Sammlung an Bildern, worin Risse auf sehr vielen unterschiedlichen Oberflächen zu sehen sind, wäre auch eine gute Möglichkeit, um die Qualität der Ergebnisse zu verbessern. Zudem werden mehr RGB-Bilder benötigt, in denen reelle Beispiele gezeigt werden und nicht Idealfälle. Im weiteren Schritt wäre es durchaus sinnvoll Transfer-Learning-Methoden einzusetzen, da von einem bereits trainierten und verwandten Trainingsprozess profitiert werden könnte. Überdies ist auch eine weitere Optimierung von Hyperparameter eine gute Möglichkeit, um die Resultate zu steigern. Da die Rechenkapazität des verwendeten Rechners bereits an der maximalen Leistungsgrenze war, wurde darauf weitgehend verzichtet.

Es ist von großem Interesse die verwendete Technologie auch auf Inspektionen in der Denkmalpflege anzuwenden. Architektonisches und kulturelles Erbe wird ähnlich wie Infrastrukturbauten regelmäßig auf Schäden und Gefährdungen kontrolliert und dabei erkannte Mängel dokumentiert. Nichtsdestoweniger ist die Inspektion dieser Bauwerke mit einem CNN schwieriger als jene von Stahlbetonbauten. Wandbilder, Schattierungen von Ornamenten und im Allgemeinen die sehr umfangreichen Störfaktoren erschweren den Einsatz von intelligenten Systemen.
Aufbauend auf den Erkenntnissen dieser Arbeit wäre auch die Ursachenerkennung von Rissen ein sinnvoller nächster Schritt. Dadurch könnte sehr schnell die Ursache und die Ernsthaftigkeit des Problems geklärt werden. Im Weiteren ist eine automatische materialspezifische Klassifizierung der Risse ein interessantes Forschungsthema. Abschließend wäre die Entwicklung einer einheitlichen und materialübergreifenden Schadenserkennung von umfangreichem Interesse, da sich mehrere Disziplinen hiervon bedienen könnten.

Literatur

[1] C. C. Aggarwal. *Neural Networks and Deep Learning: A Textbook*. en. Cham: Springer International Publishing, 2018. ISBN: 978-3-319-94462-3. DOI: 10.1007/978-3-319-94463-0.

[2] D. G. Aggelis, E. Z. Kordatos, M. Strantza, D. V. Soulioti und T. E. Matikas. „NDT approach for characterization of subsurface cracks in concrete“. In: *Construction and Building Materials* 25.7 (Juli 2011), S. 3089–3097. DOI: 10.1016/j.conbuildmat.2010.12.045.

[3] Ş. A. Akosman, M. Öktem, Ö. T. Moral und V. Kılıç. „Deep Learning-based Semantic Segmentation for Crack Detection on Marbles“. In: *2021 29th Signal Processing and Communications Applications Conference (SIU)*. Juni 2021, S. 1–4. DOI: 10.1109/SIU53274.2021.9477867.

[4] M. Artus, M. Alabassy und C. Koch. *IFC based Framework for Generating, Modeling and Visualizing Spalling Defect Geometries*. Juni 2021.

[5] L. Atlas, T. Homma und R. Marks. „An Artificial Neural Network for Spatio-Temporal Bipolar Patterns: Application to Phoneme Classification“. In: *Neural Information Processing Systems*. Maryland: American Institute of Physics, 1987. URL: https://proceedings.neurips.cc/paper/1987/hash/98f13708210194c475687be6106a3b84-Abstract.html (Zugriff am 06. 08. 2022).

[6] A. Bhande. *What is underfitting and overfitting in machine learning and how to deal with it*. en. März 2018. URL: https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76 (Zugriff am 04. 01. 2023).

[7] S. Bianco, R. Cadene, L. Celona und P. Napoletano. „Benchmark Analysis of Representative Deep Neural Network Architectures“. In: *IEEE Access* 6 (2018). arXiv:1810.00736, S. 64270–64277. DOI: 10.1109/ACCESS.2018.2877890.

[8] *BMDV - Bauwerksprüfung und Bauwerksüberwachung*. URL: https://www.bmvi.de/SharedDocs/DE/Artikel/StB/bauwerkspruefung-und-bauwerksueberwachung (Zugriff am 19. 06. 2022).

[9] A. Carass, S. Roy, A. Gherman, J. C. Reinhold, A. Jesson, T. Arbel, O. Maier, H. Handels, M. Ghafoorian, B. Platel, A. Birenbaum, H. Greenspan, D. L. Pham, C. M. Crainiceanu, P. A. Calabresi, J. L. Prince, W. R. G. Roncal, R. T. Shinohara und I. Oguz. „Evaluating White Matter Lesion Segmentations with Refined Sørensen-Dice Analysis“. In: *Scientific Reports* 10 (Mai 2020), S. 8242. DOI: 10.1038/s41598-020-64803-w.

[10] F. Celik und M. König. „A sigmoid-optimized encoder–decoder network for crack segmentation with copy-edit-paste transfer learning“. In: *Computer-Aided Civil and Infrastructure Engineering* (Apr. 2022). DOI: 10.1111/mice.12844.

[11] M. Chaudhary. *Activation Functions: Sigmoid, Tanh, ReLU, Leaky ReLU, Softmax*. en. Aug. 2020. URL: https://medium.com/@cmukesh8688/activation-functions-sigmoid-tanh-relu-leaky-relu-softmax-50d3778dcea5 (Zugriff am 07. 06. 2022).

[12] D. Dais, İ. E. Bal, E. Smyrou und V. Sarhosis. „Automatic crack classification and segmentation on masonry surfaces using convolutional neural networks and transfer learning“. en. In: *Automation in Construction* 125 (Mai 2021), S. 103606. DOI: 10.1016/j.autcon.2021.103606.

[13] J. S. Denker, W. R. Gardner, H. P. Graf, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, H. S. Baird und I. Guyon. „Neural network recognizer for hand-written zip code digits“. In: *In D. Touretzky (Ed.), Proceedings of Neural Information Processing Systems. Morgan-Kaufmann*. Holmdel: AT&T Laboratories, 1991.

[14] N. Doughty. *Here's what's on the road ahead for Pittsburgh-based RoadBotics following its acquisition by Michelin*. Juli 2022. URL: https://www.bizjournals.com/pittsburgh/inno/stories/profiles/2022/07/28/road-ahead-roadbotics-after-michelin-deal.html (Zugriff am 02. 10. 2022).

[15] *Duden | Riss | Rechtschreibung, Bedeutung, Definition, Herkunft*. de. URL: https://www.duden.de/rechtschreibung/Riss (Zugriff am 22. 04. 2022).

[16] Emma Lundmann. *Robots expected to overtake humans in construction by 2025 – Chicago Agent Magazine New Construction*. Dez. 2019. URL: https://chicagoagentmagazine.com/2019/12/24/robots-expected-to-overtake-humans-in-construction-by-2025/ (Zugriff am 08. 06. 2022).

[17] H. Fercher. *1. Platz: Palfinger Structural Inspection*. Section: Digital Award. Jan. 2020. URL: https://www.diepresse.com/5875131/1-platz-palfinger-structural-inspection (Zugriff am 07. 06. 2022).

[18] Y. Fujita, Y. Mitani und Y. Hamamoto. „A Method for Crack Detection on a Concrete Structure“. In: *18th International Conference on Pattern Recognition (ICPR'06)*. Bd. 3. Aug. 2006, S. 901–904. DOI: 10.1109/ICPR.2006.98.

[19] X. Gao und B. Jin. „Research on Crack Detection Based on Improved UNet“. In: *2020 7th International Conference on Information Science and Control Engineering (ICISCE)*. Dez. 2020, S. 2098–2103. DOI: 10.1109/ICISCE50968.2020.00412.

[20] C. Gates und P. Matthews. „Data Is the New Currency“. In: *Proceedings of the 2014 New Security Paradigms Workshop*. NSPW '14. New York, NY, USA: Association for Computing Machinery, Sep. 2014, S. 105–116. DOI: 10.1145/2683467.2683477.

[21] T. I. GmbH. *Wo liegt der Unterschied zwischen Artificial Intelligence, Machine Learning und Deep Learning? - News - TWT Interactive*. URL: https://www.twt.de/news/detail/wo-liegt-der-unterschied-zwischen-artificial-intelligence-machine-learning-und-deep-learning.html (Zugriff am 02. 06. 2022).

[22] I. Goodfellow. *Deep learning: das umfassende Handbuch : Grundlagen, aktuelle Verfahren und Algorithmen, neue Forschungsansätze*. 1. Auflage.. Frechen: mitp Verlags GmbH & Co., 2018. ISBN: 978-3-95845-702-7.

[23] Q. Gu, Z. Li und J. Han. *Generalized Fisher Score for Feature Selection*. arXiv:1202.3725. Feb. 2012. DOI: 10.48550/arXiv.1202.3725.

[24] B. Gupta, P. Mittal und T. Mufti. „A Review on Amazon Web Service (AWS), Microsoft Azure & Google Cloud Platform (GCP) Services“. In: März 2021. ISBN: 978-1-63190-292-5.

[25] E. El-hariri, N. El-Bendary und S. Taie. „Automated Pixel-Level Deep Crack Segmentation on Historical Surfaces Using U-Net Models“. In: *Algorithms* 15 (Aug. 2022), S. 281. DOI: 10.3390/a15080281.

[26] K. He, X. Zhang, S. Ren und J. Sun. *Identity Mappings in Deep Residual Networks*. arXiv:1603.05027 [cs]. Juli 2016. URL: http://arxiv.org/abs/1603.05027 (Zugriff am 17. 08. 2022).

[27] Y.-A. Hsiel und Y.-C. Tsai. *DAU-Net: Dense Attention U-Net for Pavement Crack Segmentation*. Pages: 2256. Sep. 2021. DOI: 10.1109/ITSC48978.2021.9564806.

[28] D. Hubel und T. Wiesel. *Brain and Visual Perception: The Story of a 25-Year Collaboration*. en. Google-Books-ID: 8YrxWojxUA4C. Oxford University Press, Okt. 2004. ISBN: 978-0-19-803916-7.

[29] A. Hug. *Schweizer Autobahnbrücken am Limit*. de. Section: Wissen. Apr. 2014. URL: https://www.srf.ch/wissen/technik/schweizer-autobahnbruecken-am-limit (Zugriff am 02. 01. 2023).

[30] Ian Goodfellow, Yoshua Bengio und Aaron Courville. *DeepLearningBook*. URL: https://www.deeplearningbook.org/contents/intro.html (Zugriff am 18. 04. 2022).

[31] J. McCarthy, M. L. Minsky, N. Rochester und C.E. Shannon. *A Proposal For The Darthmouth Summer Research Project On Artificial Intelligence*. URL: http://www-formal.stanford.edu/jmc/history/dartmouth/dartmouth.html (Zugriff am 16. 05. 2022).

[32] S. Jadon. „A survey of loss functions for semantic segmentation“. In: *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. arXiv:2006.14822. Okt. 2020, S. 1–7. DOI: 10.1109/CIBCB48159.2020.9277638.

[33] F. Jun, L. Jiakuan, S. Yichen, Z. Ying und Z. Chenyang. „ACAU-Net: Atrous Convolution and Attention U-Net Model for Pavement Crack Segmentation“. In: *2022 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI)*. Juli 2022, S. 561–565. DOI: 10.1109/ICCEAI55464.2022.00120.

[34] E. Karaaslan, U. Bagci und F. N. Catbas. „Artificial Intelligence Assisted Infrastructure Assessment using Mixed Reality Systems“. In: *Transportation Research Record* 2673.12 (Dez. 2019). Publisher: SAGE Publications Inc, S. 413–424. DOI: 10.1177/0361198119839988.

[35] A. Krizhevsk, Ilya Sutskeve und Geoffrey E. Hinton. *ImageNet classification with deep convolutional neural networks | EndNote Click*. en. URL: https://click.endnote.com/viewer?doi=10.1145%2F3065386&token=WzM2MTM1NTEsIjEwLjExNDUvMzA2NTM4NiJd.FfC3rQN8SoWrlEEb-GB_cU94EdA (Zugriff am 06. 08. 2022).

[36] Y. Lecun, L. Bottou, Y. Bengio und P. Haffner. „Gradient-based learning applied to document recognition“. In: *Proceedings of the IEEE* 86.11 (Nov. 1998). Conference Name: Proceedings of the IEEE, S. 2278–2324. DOI: 10.1109/5.726791.

[37] M. Leonhardt, N. Pauen, L. Kirnats, J.-N. Joost, J. Frisch und C. van Treeck. „Implementierung von KI-Basierten Referenzprozessen für die Computergestützte Objekterkennung im Gebäude“. de. In: (). Publisher: Verlag der Technischen Universität Graz. DOI: 10.3217/978-3-85125-786-1-72.

[38] Y. Liu, M. Hou, A. Li, Y. Dong, L. Xie und Y. Ji. „Automatic Detection of Timber-Cracks in wooden Architectural Heritage using YOLOv3 algorithm“. In: *ISPRS International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLIII-B2-2020 (Aug. 2020), S. 1471–1476. DOI: 10.5194/isprs-archives-XLIII-B2-2020-1471-2020.

[39] Manuel Eberhardt. *Unfälle im Bauwesen | BAUAkademie Österreich*. URL: https://www.bauakademie.at/CMSArtikel.aspx?LI1=24 (Zugriff am 18. 08. 2022).

[40] P. Mazzoni, R. A. Andersen und M. I. Jordan. „A more biologically plausible learning rule than backpropagation applied to a network model of cortical area 7a“. In: *Cerebral Cortex (New York, N.Y.: 1991)* 1.4 (Aug. 1991), S. 293–307. DOI: 10.1093/cercor/1.4.293.

[41] *Neural Networks and Deep Learning*. de. URL: https://www.springerprofessional.de/neural-networks-and-deep-learning/16073524 (Zugriff am 25. 05. 2022).

[42] A. van Opbroek, M. A. Ikram, M. W. Vernooij und M. de Bruijne. „Transfer Learning Improves Supervised Image Segmentation Across Imaging Protocols“. In: *IEEE Transactions on Medical Imaging* 34.5 (Mai 2015). Conference Name: IEEE Transactions on Medical Imaging, S. 1018–1030. DOI: 10.1109/TMI.2014.2366792.

[43] OpenAI. *CLIP: Connecting Text and Images*. Jan. 2021. URL: https://openai.com/blog/clip/ (Zugriff am 17. 08. 2022).

[44] A. Oppermann. *Batch-Normalisierung in Deep Learning*. Aug. 2021. URL: https://artemoppermann.com/de/batch-normalisierung-in-deep-learning/ (Zugriff am 18. 08. 2022).

[45] O. Ore und M. Sposato. „Opportunities and risks of artificial intelligence in recruitment and selection“. In: *International Journal of Organizational Analysis* ahead-of-print.ahead-of-print (Jan. 2021). DOI: 10.1108/IJOA-07-2020-2291.

[46] Ç. F. Özgenel. „Concrete Crack Images for Classification“. In: 1 (Jan. 2018). Publisher: Mendeley Data. DOI: 10.17632/5y9wdsg2zt.1.

[47] K. Palanisamy, D. Singhania und A. Yao. *Rethinking CNN Models for Audio Classification*. arXiv:2007.11154. Nov. 2020. DOI: 10.48550/arXiv.2007.11154.

[48] Y. Pan und L. Zhang. „Roles of artificial intelligence in construction engineering and management: A critical review and future trends“. In: *Automation in Construction* 122 (Feb. 2021), S. 103517. DOI: 10.1016/j.autcon.2020.103517.

[49] C. Pantofaru und M. Hebert. *A Comparison of Image Segmentation Algorithms*. Journal Abbreviation: Robotics Institute Publication Title: Robotics Institute. Jan. 2005.

[50] Papers with Code. *An Overview of Semantic Segmentation Models*. en. URL: https://paperswithcode.com/methods/category/segmentation-models (Zugriff am 11. 08. 2022).

[51] K. Pavel, B. Alexandra und B. Petr. *Supporting data for Deep Learning and Machine Vision based approaches for automated wood defect detection and quality control*. Type: dataset. Apr. 2021. DOI: 10.5281/ZENODO.4694695.

[52] F. Pilny. *Risse und Fugen in Bauwerken*. de. Springer-Verlag, Nov. 2013. ISBN: 978-3-7091-2295-2.

[53] L. Prechelt. „Early Stopping - But When?“ In: (März 2000). DOI: 10.1007/3-540-49430-8_3.

[54] PricewaterhouseCoopers. *Herausforderungen der deutschen Bauindustrie 2021*. URL: https://www.pwc.de/de/digitale-transformation/herausforderungen-der-deutschen-bauindustrie.html (Zugriff am 18. 08. 2022).

[55] qinnzou. *DeepCrack: Learning Hierarchical Convolutional Features for Crack Detection*. original-date: 2020-02-12T04:17:32Z. Apr. 2022. URL: https://github.com/qinnzou/DeepCrack (Zugriff am 14. 04. 2022).

[56] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger und I. Sutskever. *Learning Transferable Visual Models From Natural Language Supervision*. arXiv:2103.00020. Feb. 2021. DOI: 10.48550/arXiv.2103.00020.

[57] I. Rajagopal. *Batch Normalization — Speed up Neural Network Training*. en. Juni 2018. URL: https://medium.com/@ilango100/batch-normalization-speed-up-neural-network-training-245e39a62f85 (Zugriff am 03. 01. 2023).

[58] S. Reddi, S. Kale und S. Kumar. „On the convergence of Adam and Beyond“. In: *International Conference on Learning Representations*. 2018. DOI: 10.48550/arXiv.1904.09237.

[59] L. Reinprecht. „Wood Durability and Lifetime of Wooden Products“. In: *Wood Deterioration, Protection and Maintenance*. Hoboken, New Jersey: Wiley-Blackwell, Aug. 2016, S. 1–27. DOI: 10.1002/9781119106500.ch1.

[60] Y. Ren, J. Huang, Z. Hong, W. Lu, J. Yin, L. Zou und X. Shen. „Image-based concrete crack detection in tunnels using deep fully convolutional networks“. en. In: *Construction and Building Materials* 234 (Feb. 2020), S. 117367. DOI: 10.1016/j.conbuildmat.2019.117367.

[61] Reuven Rubinstein und Dirk Kroese. *The Cross-Entropy Method*. en. URL: https://link.springer.com/book/10.1007/978-1-4757-4321-0 (Zugriff am 10. 08. 2022).

[62] O. Ronneberger, P. Fischer und T. Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. arXiv:1505.04597. Mai 2015. URL: http://arxiv.org/abs/1505.04597 (Zugriff am 18. 08. 2022).

[63] S. J. Russell, P. Norvig, E. Davis und D. Edwards. *Artificial intelligence: a modern approach*. Third edition, Global edition. Prentice Hall series in artificial intelligence. Boston: Pearson, 2016. ISBN: 978-1-292-15396-4.

[64] *RVS 13.71. Überwachung, Kontrolle und Prüfung von Kunstbauten - Straßenbrücken: Ausgabe August 1995*. de. Google-Books-ID: a9RonQAACAAJ. Österreichische Forschungsgemeinschaft Straße und Verkehr (FSV), 1995.

[65] S. Sahoo. *Residual blocks — Building blocks of ResNet*. en. Sep. 2022. URL: https://towardsdatascience.com/residual-blocks-building-blocks-of-resnet-fd90ca15d6ec (Zugriff am 16. 02. 2023).

[66] S. Santurkar, D. Tsipras, A. Ilyas und A. Madry. *How Does Batch Normalization Help Optimization?* arXiv:1805.11604. Apr. 2019. DOI: 10.48550/arXiv.1805.11604.

[67] J. Schulz. *Architektur der Bauschäden: Schadensursache - Gutachterliche Einstufung - Beseitigung - Vorbeugung - Lösungsdetails*. de. Wiesbaden: Springer Fachmedien Wiesbaden, 2020. ISBN: 978-3-658-27653-9 978-3-658-27654-6. DOI: 10.1007/978-3-658-27654-6. URL: http://link.springer.com/10.1007/978-3-658-27654-6 (Zugriff am 14. 06. 2022).

[68] Sebastian Raschka und Vahid Mirjalili. *Machine Learning mit Python und Keras, TensorFlow 2 und Scikit-learn*. 3. Aufl. Frechen: mitp Verlags GmbH & Co. KG, 2021.

[69] G. Sedlacek und M. Paschen. „Die Bedeutung einer qualifizierten Bauwerksprüfung“. In: *Stahlbau* 78.8 (2009), S. 584–592. DOI: 10.1002/stab.200910072.

[70] G. Sedlacek und M. Paschen. „Die Bedeutung einer qualifizierten Bauwerksprüfung“. In: *Stahlbau* 78.8 (2009), S. 584–592. DOI: 10.1002/stab.200910072.

[71] S. K. Sinha und P. W. Fieguth. „Segmentation of buried concrete pipe images“. en. In: *Automation in Construction* 15.1 (Jan. 2006), S. 47–57. ISSN: 0926-5805. DOI: 10.1016/j.autcon.2005.02.007. URL: https://www.sciencedirect.com/science/article/pii/S0926580505000464 (Zugriff am 22. 08. 2022).

[72] U. Smoltczyk. „Empfehlung Nr. 2 des Arbeitskreises “Geotechnik historischer Bauwerke und Naturdenkmäler” der DGGT: Baugrundbedingte Risse und Verformungen an historischen Bauwerken“. In: *Bautechnik* 81.1 (2004), S. 17–24. DOI: 10.1002/bate.200490001.

[73] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever und R. Salakhutdinov. „Dropout: A Simple Way to Prevent Neural Networks from Overfitting“. In: *Journal of Machine Learning Research* 15.56 (2014), S. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html (Zugriff am 06. 06. 2022).

[74] Statista Research Department. *Österreich – Brücken auf Autobahnen nach Brückenart 2021*. URL: https://de.statista.com/statistik/daten/studie/786454/umfrage/bruecken-auf-autobahnen-in-oesterreich-nach-brueckenart/ (Zugriff am 21. 06. 2022).

[75] Strucinspect. *Infrastructure Lifecycle Hub*. en-US. URL: https://strucinspect.com/ (Zugriff am 02. 10. 2022).

[76] V. Sze, Y.-H. Chen, T.-J. Yang und J. Emer. „Efficient Processing of Deep Neural Networks: A Tutorial and Survey“. In: *Proceedings of the IEEE* 105 (März 2017). DOI: 10.1109/JPROC.2017.2761740.

[77] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke und A. Rabinovich. „Going deeper with convolutions“. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, Juni 2015, S. 1–9. ISBN: 978-1-4673-6964-0. DOI: 10.1109/CVPR.2015.7298594.

[78] K. Team. *Keras documentation: Layer activation functions*. URL: https://keras.io/api/layers/activations/ (Zugriff am 07. 06. 2022).

[79] TensorFlow. *TensorFlow*. URL: https://www.tensorflow.org/ (Zugriff am 12. 08. 2022).

[80] G. Thagunna. „Building cracks – causes and remedies“. In: 04.01 (2015), S. 6. ISSN: 2319-5347.

[81] H. Urban, N. Breitschopf und C. Schranz. „Entwicklung und Validierung eines AR-Abnahmetools für die örtliche Bauaufsicht am Beispiel der Technischen Gebäudeausrüstung“. In: *Bauingenieur* 97.11 (2022), S. 353–361. DOI: 10.37544/0005-6650-2022-11-35.

[82] A. Veit, M. J. Wilber und S. Belongie. „Residual Networks Behave Like Ensembles of Relatively Shallow Networks“. In: *Advances in Neural Information Processing Systems*. Bd. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper/2016/hash/37bc2f75bf1bcfe8450a1a41c200364c-Abstract.html (Zugriff am 17. 08. 2022).

[83] S. v. d. Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart und T. Yu. „scikit-image: image processing in Python“. In: *PeerJ* 2 (Juni 2014). Publisher: PeerJ Inc., e453. DOI: 10.7717/peerj.453.

[84] Z. Wei, K. Jia, X. Jia, A. Khandelwal und V. Kumar. „Global River Monitoring Using Semantic Fusion Networks“. In: *Water* 12 (Aug. 2020), S. 2258. DOI: 10.3390/w12082258.

[85] P. Wennker. „Künstliche Intelligenz – Eine kurze Geschichte“. In: *Künstliche Intelligenz in der Praxis: Anwendung in Unternehmen und Branchen: KI wettbewerbs- und zukunftsorientiert einsetzen*. Hrsg. von P. Wennker. Wiesbaden: Springer Fachmedien, 2020, S. 1–8. DOI: 10.1007/978-3-658-30480-5_1.

[86] M. Werner. *Digitale Bildverarbeitung: Grundkurs mit neuronalen Netzen und MATLAB®-Praktikum*. Wiesbaden: Springer Fachmedien, 2021. DOI: 10.1007/978-3-658-22185-0.

[87] I. H. Witten, E. Frank und M. A. Hall. *Data mining: practical machine learning tools and techniques*. 3rd ed. Morgan Kaufmann series in data management systems. OCLC: ocn262433473. Burlington, MA: Morgan Kaufmann, 2011. ISBN: 978-0-12-374856-0.

[88] www.ait.ac.at. *RIBET – AIT Austrian Institute Of Technology*. de. URL: https://www.ait.ac.at/themen/baudynamik-und-bauwerksshybewertung/projects/ribet (Zugriff am 02. 10. 2022).

[89] Y. Nesterov. *Introductory Lectures on Convex Optimization*. en. URL: https://link.springer.com/book/10.1007/978-1-4419-8853-9 (Zugriff am 10. 08. 2022).

[90] W. Yu, Y. Li, Y. Hongtao und B. Qian. „The Centerline Extraction Algorithm of Weld Line Structured Light Stripe Based on Pyramid Scene Parsing Network". In: *IEEE Access* PP (Juli 2021), S. 1–1. DOI: 10.1109/ACCESS.2021.3098833.

[91] A. S. Zanke. „Building Cracks: Causes and Preventions". In: *International Journal of Engineering Applied Sciences and Technology* 5.8 (Dez. 2020). DOI: 10.33564/IJEAST.2020.v05i08.025.

[92] L. Zhang, F. Yang, Y. Daniel Zhang und Y. J. Zhu. „Road crack detection using deep convolutional neural network". In: *2016 IEEE International Conference on Image Processing (ICIP)*. Sep. 2016, S. 3708–3712. DOI: 10.1109/ICIP.2016.7533052.

[93] Y. Zhang. „Safety Management of Civil Engineering Construction Based on Artificial Intelligence and Machine Vision Technology". In: *Advances in Civil Engineering* 2021 (Dez. 2021). Publisher: Hindawi, e3769634. DOI: 10.1155/2021/3769634.

[94] R. Zhao, B. Qian, X. Zhang, Y. Li, R. Wei, Y. Liu und Y. Pan. „Rethinking Dice Loss for Medical Image Segmentation". In: *2020 IEEE International Conference on Data Mining (ICDM)*. Nov. 2020, S. 851–860. DOI: 10.1109/ICDM50108.2020.00094.

[95] A. Zheng und Amanda Casari. *Merkmalskonstruktion für Machine Learning - Prinzipien und Techniken der Datenaufbereitung*. 1. Aufl. Heidelberg: dpunk.verlag GmbH, 2019. ISBN: 978-3-96009-093-9.

[96] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh und J. Liang. „UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation". In: *IEEE Transactions on Medical Imaging* 39.6 (Juni 2020), S. 1856–1867. DOI: 10.1109/TMI.2019.2959609.

[97] P. Zschech, C. Sager, P. Siebers und M. Pertermann. „Mit Computer Vision zur automatisierten Qualitätssicherung in der industriellen Fertigung: Eine Fallstudie zur Klassifizierung von Fehlern in Solarzellen mittels Elektrolumineszenz-Bildern". In: *HMD Praxis der Wirtschaftsinformatik* 58.2 (Apr. 2021), S. 321–342. ISSN: 2198-2775. DOI: 10.1365/s40702-020-00641-8.