HAW HAMBURG

BACHELORTHESIS
Eike Hannes Meyer

Evaluation von Visualisierungsmethoden als Testverfahren für CNNs am Beispiel Bildklassifikation

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Eike Hannes Meyer

Evaluation von Visualisierungsmethoden als Testverfahren für CNNs am Beispiel Bildklassifikation

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Angewandte Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Bettina Buth
Zweitgutachter: Prof. Dr. Kai von Luck

Eingereicht am: 28.12.2020

Eike Hannes Meyer

Thema der Arbeit

Evaluation von Visualisierungsmethoden als Testverfahren für CNNs am Beispiel Bildklassifikation

Stichworte

Software Testing, Black Box Tests, XAI, CNN, LRP, Image classification, Class Model Visualization

Kurzzusammenfassung

Das Verhalten heute vielfach verwendeter neuronaler Netze wird im Gegensatz zu traditioneller Software nicht konkret definiert, sondern hängt insbesondere von den verwendeten Trainingsdaten ab. Dadurch entziehen sich diese Komponenten traditionellen Testmethoden und es ist schwierig Vertrauen in solche Systeme aufzubauen. In dieser Arbeit werden Visualisierungsmethoden für neuronale Netze und ihre Entscheidungen als Analogie zu Black-Box Testmethoden angewandt und deren Nutzen evaluiert. Dafür wurde ein CNN zur Bildklassifikation unter Verwendung natürlicher und veränderter Daten mit der Methode Layer-Wise Relevance Propagation untersucht und die Ergebnisse interpretiert. Es hat sich bestätigt, dass nicht nur die eigentliche Instanz einer Klasse als Entscheidungsmerkmal für eine Klassifikation verwendet wird, sondern auch weitere Artefakte. Die Anwendung von LRP kann somit Hinweise auf Probleme in den Trainingsdaten geben.

Eike Hannes Meyer

Title of Thesis

Evaluation of visualization methods as testing techniques for CNNs using the example of image classification

Keywords

Software Testing, Black Box Tests, XAI, CNN, LRP, Image classification, Class Model Visualization

Abstract

The behavior of widely used neuronal networks is not explicitly defined but is dependent on the training data . Thereby traditional test methods can not be used for those components and the build up of trust is difficult. In this work methods for the visualization of neural networks and their decisions are used and evaluated as analogies for black-box tests. For this a CNN for image classification was inspected under the use of natural and manipulated data and via the method of Layer-Wise Relevance Propagation. This led to the confirmation of the presumption that NNs do not explicitly use the instance of a class as a feature for the classification process, but rather also other artifacts. The use of LRP can consequently be used to find hints on problems with the training data.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Heutzutage werden künstliche neuronale Netze (NNs) in vielen Bereichen eingesetzt. Gerade in kritischen Anwendungsdomänen wie Medizin, autonom fahrenden Fahrzeugen und der Verarbeitung von persönlichen Daten ist es aber wichtig, dass das Verhalten der Netze verifiziert und validiert werden kann. Insbesondere der Aufbau von Vertrauen in NNs ist wichtig und wird unter dem Aspekt der Explainable Artificial Intelligence (XAI) betrachtet. Dieser Zweig der künstlichen Intelligenz beschäftigt sich mit der Interpretierbarkeit und dem Verständnis vom Aufbau und den getroffenen Entscheidungen intelligenter Systeme. In dieser Arbeit wird untersucht, ob sich Methoden eines spezifischen Teils der XAI, sogenannte „post-hoc“ Interpretationen [16], eignen, um Probleme mit der Auswahl der Trainingsdaten und somit auch dem Verhalten eines Netzes aufzudecken. Diese Art der Interpretation betrachtet das trainierte Netz als Black-Box und versucht zu erklären, anhand welcher Kriterien eine Entscheidung getroffen wurde.

1.1 Problemstellung und Motivation

Neuronale Netze sind nicht wie traditionelle Software sequentielle Abfolgen von Anweisungen, sondern stellen einen gerichteten Graphen dar, der auf Basis von errechneten Kantengewichten Entscheidungen fällt. Im Gegensatz zu traditioneller Software, deren Verhalten explizit definiert wird, entsteht das Verhalten eines NNs auf Basis seiner Topologie und insbesondere der verwendeten Trainingsdaten. Aufgrund dieses Aufbaus entziehen sich neuronale Netze traditionellen Testansätzen, wie zu Beispiel funktionalen Komponententests [18].
Die Qualität der trainierten neuronalen Netze, Modelle genannt, hängt von der Architektur des Netzes und zu einem großen Teil von den Daten ab, mit denen das Netz trainiert wurde. Eine vollständige Analyse dieser Daten ist sehr zeitintensiv, da ein Domänenexperte diese, in der Regel sehr große Datenmenge, vollständig durchsehen müsste. Diese

Datenbasis kann verschiedene Probleme aufweisen. In den Daten könnten Sondersituationen fehlen, die aber durchaus in der Realität auftreten. Weiterhin können in den Daten Korrelationen enthalten sein, die es dem Netz zwar erleichtern seine Aufgabe zu erfüllen, aber in der Realität nicht immer genau so auftreten. Deutlich wird dies anhand einiger problematischer Beispiele: Die Software *COMPAS* [22] wird von amerikanischen Gerichten verwendet, um zu bewerten, ob ein Angeklagter zu einem Wiederholungstäter wird. Die Software schrieb Schwarzen oft höhere Rückfallrisiken als anderen Ethnien zu, obwohl Beobachtungen in der Realität zeigten, dass dies nicht der Fall ist [17]. Ein Beispiel im Bereich der Bildklassifikation ist der Datensatz „PASCAL VOC2007 “ [8], bei dem ein Großteil der verwendeten Daten der Klasse „Pferd“ ein Wasserzeichen in einer Ecke enthielten, die trainierte Netze sofort als wichtiges Merkmal erkannten und somit eine Korrelation zwischen diesem Wasserzeichen und Pferden herstellten [14].
Diese Arbeit beschäftigt sich mit der Verwendung und Eignung von Visualisierungsmethoden als Testmethodik für das Entscheidungsverhalten und somit auch für eine implizite Evaluierung der Trainingsdaten von neuronalen Netzen zur Bildklassifizierung. Durch eine Versuchsreihe wird untersucht, ob diese Methoden geeignet sind, um Probleme in neuronalen Netzen zur Bildklassifikation und deren Trainingsdaten aufzudecken. Anhand von natürlichen und veränderten Bilddaten wird versucht sichtbar zu machen, dass NNs andere Merkmale als nur die eigentliche Instanz einer Klasse zur Erkennung und Klassifikation nutzen und dass dies zu Problemen mit der Klassifikation führen kann.
Das Ziel dieser Arbeit ist es festzustellen, ob neuronale Netze mithilfe von Visualisierungsmethoden auf diese Probleme hin untersucht werden können. Insbesondere stellt sich die Frage, welche Arten von Problemen aufgedeckt werden können und was mögliche Konsequenzen dieser Probleme sind. Konkreter soll festgestellt werden, inwieweit Artefakte in den Trainingsdaten von neuronalen Netzen einen Einfluss auf die Klassifikation haben und ob dieser Verhalt mit dem Ansatz der „Layer-wise Relevance Propagation“ Methodik (LRP) dargestellt werden kann. Folgen, wie etwa false-positives, also die Klassifikation von Bildern als eine Klasse ohne tatsächliche Instanz und false-negatives, die Zuordnung zu einer anderen Klasse als tatsächlich im Bild vorhanden, werden in dieser Arbeit explizit untersucht. LRP und andere Visualisierungsmethoden können dabei helfen im Zusammenhang mit bereits trainierten Netzen Vertrauen im Bezug auf ihr Entscheidungsverhalten aufzubauen. Auf die Lösung der besagten Probleme wird im Hauptteil der Arbeit nicht explizit eingegangen; eine Übersicht zu möglichen Ansätzen erfolgt im Ausblick.

1.2 Einordnung

Im Kontext dieser Arbeit wird das Verhalten eines neuronales Netzes zur Bildklassifikation betrachtet. Da das Verhalten des Netzes auf den verwendeten Traingsdaten basiert, werden diese implizit mit untersucht. Bei dem hier untersuchten neuronalen Netz handelt es sich um ein „Convolutional Neural Network“, also eine spezielle Netztopologie. Diese Art der Topologie hat in den letzten Jahren die besten Ergebnisse für den Bereich der Bildklassifikation erzielt [13] und stellt heutzutage, mit einigen Änderungen, de facto den Standard dar [25][7]. Das untersuche Model in dieser Arbeit wird, trotz eigenem Entwurf, als Black-Box behandelt, mit dem Ziel, die im Abschnitt 3 vorgestellten, Visualisierungsmethoden auf ihre Eignung als Black-Box Testmethode hin zu untersuchen. Black-Box Testmethode bedeutet hier insbesondere, das kein Wissen über die innere Struktur des Testobjekts vorausgesetzt wird. Neuronale Netze gelten im Allgemeinen ohnehin als undurchsichtig, da sie keine detaillierten Erklärungen für ihre Entscheidungen geben und auch die Kenntnis über ihre gesamte interne Struktur keine menschenmögliche Interpretation zulässt [19] [16].
Das in dieser Arbeit verwendete Netz wurde vom Autor auf die Klassifikation von zehn verschiedenen Klassen trainiert und stellt somit einen spezifischen Anwendungsfall dar, der aber keinen konkreten Bezug zu einem übergeordneten Anwendungsszenario hat. Vielmehr stellen diese Klassen eine Teilmenge eines typischen Benchmark Datensatzes für Bildklassfikation dar. Eine genaue Struktur ist in 4.2 zu finden. Mithilfe der von Bach et al. entwickelten Visualisierungsmethode „Layer-Wise Relevance Propagation“ [3] wird anhand einzelner Bilder ein, von Menschen interpretierbarer, Erklärungsversuch für das Klassifikationsverhalten des Netzes erzeugt. Es existieren viele dieser Visualisierungsmethoden, die jedoch nicht alle auf jede Problemstellung und jede Netztopologie anwendbar sind [16] [33]. Diese Methoden entstammen dem Forschungsbereich der Explainable Artificial Intelligence (XAI), der sich unter anderem mit der Nachvollziehbarkeit und Erklärung von getroffenen Entscheidungen neuronaler Netze beschäftigt [16] [21]. Diese Erklärungen sollen leicht von Menschen interpretierbar sein, was zur Folge hat, dass die meisten Methoden dieses Bereichs Visualisierungen der Entscheidungen eines Netzes sind [16] [18]. Da das Verhalten und der Entscheidungsfluss eines NNs auf dessen erlernten Gewichten basiert, die auf Basis der Trainingsdaten berechnet werden, werden somit implizit die Traingsdaten untersucht.

1.3 Abgrenzung und Related Work

Diese Arbeit baut auf den Methoden verschiedener Autoren auf, die ihre Methoden eher in einem theoretischen Rahmen vorstellten [3] [27] [16]. Einige Autoren, die Methoden für die Visualisierung von neuronalen Netzen und deren Verhalten entwickeln oder untersuchen, tun dies um diese Netze zu optimieren oder einen Interpretrierungsansatz geben zu können [16]. Die Betrachtung der Methoden finden also in einem theoretischen Rahmen statt, um beispielweise Anforderungen an Visualisierungsmethoden aufzustellen [16] [20] [33]. Insbesondere findet die Betrachtung dieser Methoden als Analogie zu Black-Box Tests in den genannten Arbeiten nicht statt.
In der Arbeit [14] wurde der PASCL-VOC 2007 Datensatz [8], der für die Anwesenheit von Wasserzeichen auf Bildern der Klasse „Pferd“ bekannt ist, mit Hilfe verschiedener Visualisierungsmethoden untersucht. Die Autoren stellten zwar Probleme mit dem Netz zur Klassifikation in einzelnen Beispielen dar, untermauerten diese Ergebnisse jedoch nicht durch weitere Experimente. Diese Arbeit soll die Erkenntnisse weiterführen und bestärken. Dies soll zum einen durch weitere Methoden und weitere Versuche geschehen, zum anderen anhand des viel genutzten „ImageNet“ [25] Datensatzes und seiner Teilmenge „ImageNette“ [12]. Obwohl auch dieser Datensatz kritisch in Bezug auf die faire Verteilung von Daten untersucht wurde und bezogen auf die demographische Repräsentation der Kategorien, die Menschen enthalten können, hinterfragt wird, gilt der „ImageNet“ [25] Datensatz als Benchmark für das Training von Netzen zur Bildklassifikation [26].
Nicht betrachtet werden in dieser Arbeit Analogien zu White-Box Testmethoden. Einige Autoren schlagen auch Testmethodiken für neuronale Netze vor, die die Struktur des Netzes selbst betrachten und Abdeckung von Neuronen und Kanten auf dieser messen und Testfälle auf Basis dieser Abdeckung entwerfen [29].
Diese Arbeit beschränkt sich auf die Untersuchung von CNNs zur Bildklassifikation. Es werden keine anderen Arten des maschinellen Lernens oder andere Problemstellungen betrachtet.

1.4 Aufbau der Arbeit

Zu Beginn der Arbeit 1.1 wurde die Problembeschreibung und die Motivation für diese Arbeit beschrieben. In Abschnitt 1.2 erfolgte eine genauere Einordnung des Themas und eine Abgrenzung zu anderen Arbeiten . Im Grundlagenteil 2 werden die benötigten

Grundlagen beschrieben, die für den Versuchsaufbau benötigt werden, sowie verwendete Werkzeuge. Im dritten Teil 3 wird die vorgeschlagene Methodik vorgestellt, analysiert und Alternativen zu dieser diskutiert. Im Abschnitt 4 folgt die genaue Beschreibung des eigentlichen Versuchsaufbaus, der Umgebung und der Versuchsobjekte. Im Anschluss erfolgen die Präsentation der Ergebnisse in Bildform, deren Beschreibung und die Diskussion der Ergebnisse. Abschließend wird im letzten Kapitel ein Fazit aus der Arbeit gezogen und ein Ausblick gegeben.

2 Grundlagen

In diesem Abschnitt werden die benötigen Grundlagen zu den Themen neuronale Netze, deren Aufbau und verwendete Werkzeuge zur Implementierung vorgestellt. Zuerst wird eine generelle Problembeschreibung für Bildklassifikation gegeben. Danach wird der Aufbau eines neuronalen Netzes und der Vorgang des Lernens beschrieben. Dazu gehört die Beschreibung der Struktur und Funktionsweise künstlicher Neuronen und deren Anordnung in Schichten, welche wiederum unterschiedliche Aufgaben erfüllen. Weiterhin werden die verwendeten Werkzeuge und Bibliotheken vorgestellt, die für die Erstellung des untersuchten Modells, die Manipulation von Bilddaten und die Untersuchung des Modells verwendet wurden.

2.1 Bildklassifikation

Bildklassifikation bezeichnet die Aufgabe, einem Bild einer Klasse aus einer vorher bestimmten Menge zuzuordnen [15]. Damit ist eine 1:1-Zuordnung von einer Klasse zu einem Bild gemeint, also kann einer beliebige Eingabe nur genau eines der vorgegeben Klassenlabel zugeordnet werden. Es geht also darum, für die Gesamtheit aller Pixel eines Eingabebildes herauszufinden, was diese darstellt. Mit Bildklassifikation ist nicht die Lokalisierung dieser Klasseninstanz in einem Bild oder die Zählung mehrerer Instanzen, die sogenannte Multi-label Classification, gemeint. Die Lösung solcher Aufgaben basiert allerdings auf dem Problem der in dieser Arbeit untersuchen Bildklassifikation.

Abbildung 2.1: Bildklassifikation. Dem Bild wird das Label „Fallschirm“ zugeordnet. Quelle: n03888257_4233.JPEG aus [25].

Es gibt eine Vielzahl an Herausforderungen bei der Klassifikation von Bildern. Objekte in einem zu klassifizierenden Bild können aus verschiedenen Winkeln aufgenommen sein, sodass ihre Form je nach Winkel variiert. Weiterhin kann die Größe dieser Objekte unterschiedlich sein. Diese Größenunterschiede beziehen sich sowohl auf den Platz, den ein Objekt im Bild einnimmt, als auch auf die Größe eines Objektes in der echten Welt. Ebenso können die Objekte teilweise verdeckt sein oder einzelne Objekte einer Klasse unterscheiden sich stark in ihrer Form, sodass sie nicht einem bestimmten Stereotyp entsprechen. Für eine Verarbeitung durch Computer muss ein solches Bild in eine andere Form gebracht werden. Die Form ist im Normalfall eine pixelweise Darstellung mit Farb- und Helligkeitswerten für jeden Pixel im Bild. Selbst bei kleinen Veränderungen im Bild ändern sich diese Werte. Schon eine Verschiebung eines Objekts in einem Bild führt zu einer anderen Pixelrepräsentation.
Es gibt keine festgelegte Vorgehensweise um die Aufgabe der Bildklassifikation maschinell zu lösen. Aufgrund der beschriebenen Herausforderung wird eine gute Verallgemeinerungsfähigkeit der Lösungsmethoden benötigt. Eine Vorgehensweise sind „Nearest-Neighbour“ Methoden. Hierbei wird ein Eingabedatum mit bereits bekannten Daten verglichen und dann der naheliegendsten Klasse zugeordnet. So kann zum Beispiel aus den Farbwerten eines neuen Bildes ein Wert berechnet werden, mithilfe dessen dann die Distanz zu bereits bekannten Bildern berechnet wird. Anhand dieser Distanz wird dann die Klassifikation vorgenommen [7].
In den letzten Jahren hat sich herausgestellt, dass künstliche neuronale Netze Bilder schneller richtig klassifizieren können, als andere Methoden [13]. Einige dieser Netze sind bei der Erkennung von Zahlen und Objekten vorbereiteten Menschen sogar ebenbürtig [28].

2.2 Neuronale Netze

Neuronale Netze sind eine besondere Art vorwärts gerichteter Graphen, deren Aufbau und Parameter das Verhalten des fertigen Netzes bestimmen. Dieses Netz besteht aus künstlichen Neuronen, die Schichten bilden und über bestimmte Arten von Kanten miteinander verbunden sind. Die Kantengewichte dieses Netzes werden während des sogenannten „Trainings“ so verändert, dass sich das Netz dem gewünschten Verhalten annähert. Die in diesem Abschnitt zusammengefassten Informationen stammen aus den folgenden Werken:

- Deep Learning: A Practitioner's Approach [23]
- Grundkurs Künstliche Intelligenz : Eine praxisorientierte Einführung [7]
- Deep Learning [10]

Die einfachste Form eines Netzes besteht aus einer Eingabe-, einer versteckten- und einer Ausgabeschicht sowie den Kanten zwischen diesen Schichten.

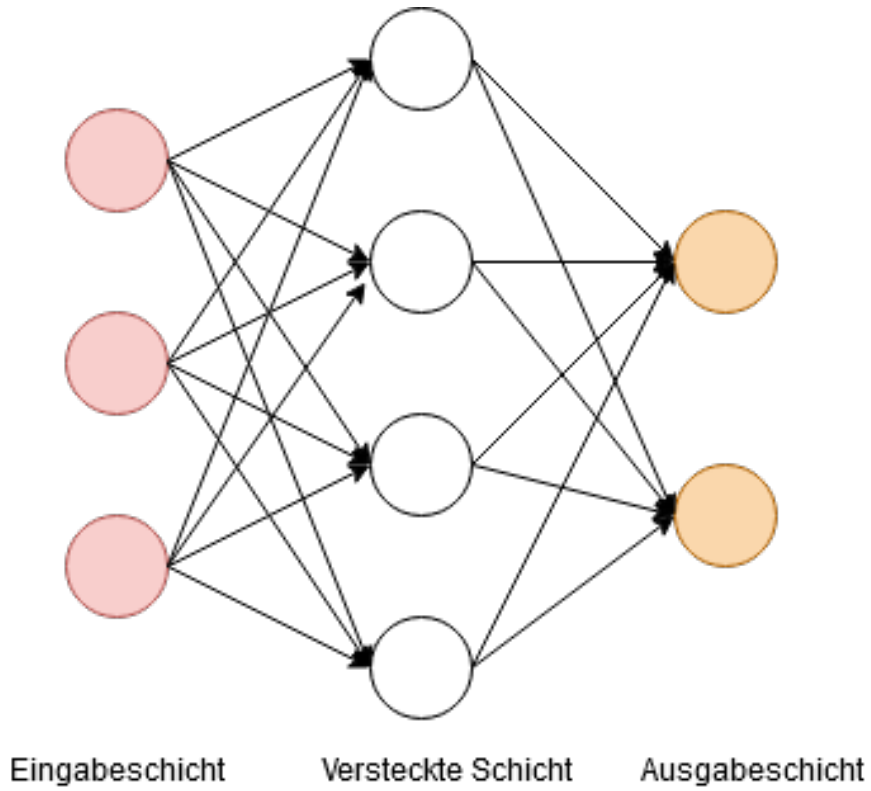

Abbildung 2.2: Ein einfaches künstliches neuronales Netz. Vom Autor.

Dieses Kapitel stellt in keiner Weise einen vollständigen und tiefen Einblick dar, sondern gibt nur die nötigsten Informationen zum Verstehen dieser Arbeit wieder. Für weitere Informationen zu diesem Thema wird auf die oben genannten Werke verwiesen.

2.2.1 Künstliches Neuron

Das künstliche Neuron, oft auch nur Neuron, stellt die kleinste Einheit in einem neuronalen Netz dar. Es stellt die Knoten in dem vorwärts gerichteten Netzwerk dar. In seiner Grundfunktion ist es dem natürlichen Neuron in einem Säugetierhirn nachempfunden. Es erhält Signale von anderen Neuronen und sendet selber ein Signal auf Basis der eingegangenen weiter. Dabei gibt es nicht wie in der Biologie nur die Möglichkeit zu feuern oder nicht zu feuern, also ein binäres Signal, sondern auch andere mathematische Funktionen, die gewählt werden können, um dieses Verhalten zu bestimmen.
Die Grundidee besteht darin, dass Neuronen Schichten und somit ein Netz bilden, Signale

von anderen Neuronen erhalten, verarbeiten und diese weitergeben. Die Signale werden dem Neuron über eingehende Kanten übermittelt. Diese Kanten sind mit Gewichten versehen, welche mit dem ursprünglichen Signal multipliziert werden. Da Neuronen über viele eingehende Kanten verfügen, die alle mit individuellen Gewichten multipliziert werden, lässt sich die Eingabe gut als Vektor aller eingehenden Kantengewichte W_i betrachten. Diese Gewichte werden mit den jeweiligen Aktivierungen der Neuronen multipliziert, hier als Vektor A_i bezeichnet. Die summierten Eingabewerte in ein Neuron werden mit $W_i * A_i$ bezeichnet.
Um die Aktivierung eines Neurons zu berechnen, wird auf die jeweilige Eingabesumme nun eine Aktivierungsfunktion f angewandt. Es gibt viele Aktivierungsfunktionen, die je nach Schicht verwendet werden. In dieser Arbeit werden die Softmax und die Rectifier-Funktion verwendet.

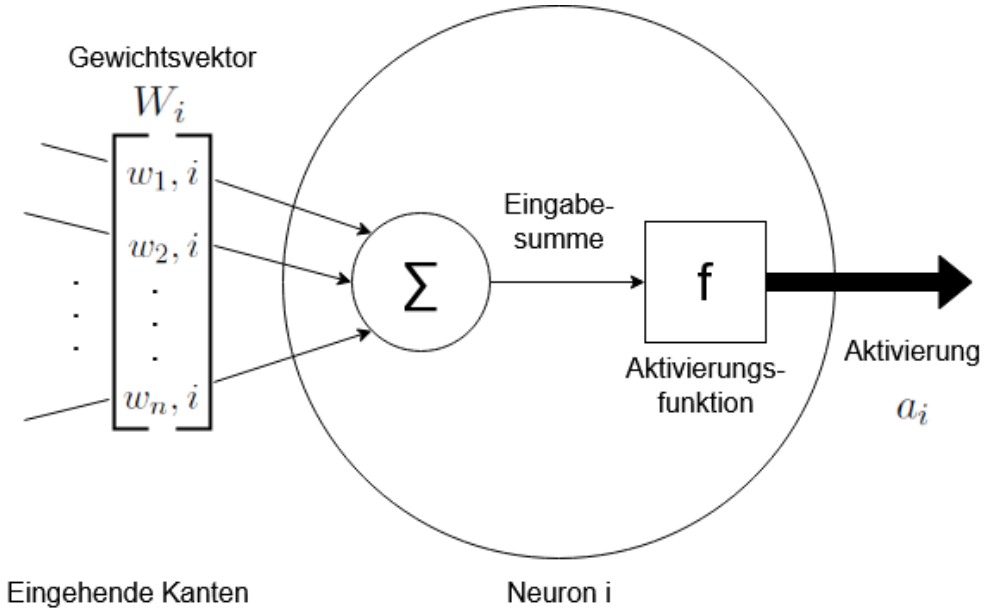

Abbildung 2.3: Ansicht eines künstlichen Neurons. Vom Autor nach [23].

Die Softmax-Funktion wird in der letzten Schicht bei Multiklassen-Klassifikationsproblemen mit nicht-zusammenhängenden Klassen verwendet, da die Funktion ihre Argumente in Wahrscheinlichkeiten umwandelt, die summiert 1 ergeben. So entsteht eine Wahrscheinlichkeitsverteilung über alle Ausgabeklassen. Die Softmax-Funktion ist definiert als:

$$softmax(z_j) = \frac{e_j^z}{\sum_{k=1}^{K} e_k^z}$$

Die Rectifier-Funktion(ReLU) wird in den versteckten Schichten verwendet und gibt nur den positiven Anteil seiner Argumente aus. Heutzutage gilt die Verwendung von ReLU als State-of-the-art, da sich gezeigt hat, dass die Funktion in vielen verschiedenen Situationen gute Ergebnisse erzielt [23]. Die Funktion ist definiert als:

$$relu(x) = max(0, x)$$

Die Aktivierung a_i eines einzelnen Neurons ergibt sich somit als:

$$a_i = f(W_i * A_i)$$

2.2.2 Schichten eines neuronalen Netzes

Neuronale Netze sind eine besondere Art vorwärts gerichteter Graphen, dessen Knoten in sogenannten Schichten angeordnet sind. Schichten stellen eine Gruppierung von Neuronen dar, die untereinander nicht verbunden sind, sondern nur über ausgehende Kanten zur nächsten Schicht und eingehende Kanten von der vorigen Schicht verfügen. Typischerweise verwenden alle Neuronen einer Schicht auch dieselbe Aktivierungsfunktion. Die Kanten zwischen den Schichten verfügen über Gewichte, welche während des Trainings angepasst werden. Diese Gewichte sind reelle Zahlen, die mit den Aktivierungen der verbundenen Neuronen multipliziert werden.

Eingabeschicht

Die Eingabeschicht ist verantwortlich für die Eingabe der Daten in das Netz. Typischerweise werden die Daten in Vektorform übergeben. Die Anzahl der Neuronen in dieser Schicht entspricht den Dimensionen der Eingabe. So entspricht beispielsweise für die Eingabe von Bilddaten die Anzahl der Neuronen der Auflösung des Bildes. Im Falle von Farbbildern ist die Anzahl der Neuronen dreimal größer, für jeden Farbkanal kommt nochmal der Betrag der Auflösung hinzu. Ein Netz verfügt über genau eine Eingabeschicht, auf die weitere Schichten einer anderen Art folgen können.

Ausgabeschicht

Die Ausgabeschicht ist für die finale Ausgabe des Netzes zuständig. Im Falle einer Klassifikationsaufgabe wird als Aktivierungsfunktion „softmax“ verwendet, um eine Wahrscheinlichkeitsverteilung über alle Klassen zu erhalten. Die Anzahl der Neuronen dieser Schicht entspricht in diesem Fall der Anzahl der zu lernenden Klassen. Ein Netz verfügt über genau eine Ausgabeschicht.

Convolutional Layer

Convolutional Layer, zu deutsch etwa Faltungsschichten, sind seit einiger Zeit de facto Standard im Bereich der Bildverarbeitung durch neuronale Netze [13]. Sie werden auch in anderen Bereichen verwendet, in denen Strukturen mit räumlichen oder zeitlichen Zusammenhang bestehen. Die Idee hinter den Convolutional Layers ist, dass ein vorher bestimmter Bereich eines Eingabedatums auf einmal betrachtet wird. Diese Aufgabe übernehmen sogenannte Filter mit fester Größe, die Matrizen repräsentieren und deren Elemente während des Trainings verändert werden.
Mit diesen Filtern erfolgt eine sogenannte „Convolution“ oder Faltung. In der Mathematik bezeichnet eine Faltung die Zusammenfassung von zwei Mengen an Informationen. Während dieser Faltung laufen die vorher definierten Filter schrittweise über das gesamte Eingabedatum und erzeugen eine neue Ausgabe, basierend auf ihren gelernten Elementen.
Es hat sich gezeigt, dass diese Filter Muster in den Eingabedaten erkennen, wobei verschiedene Filter verschiedene Muster erkennen. Filter von hohen convolutional layers, die dichter an der Eingabeschicht sind, erkennen einfache Muster, wie etwa horizontale oder vertikale Kanten. Filter tieferer Schichten erkennen komplexere Muster, wie etwa Kanten, dann Kreise und sogar Gesichter und Ähnliches.

Pooling Layer

Pooling Layer werden genutzt, um die Größe der Datenrepräsentation zu verringern. Im Fall von Bilddaten werden die Breite und Höhe der Daten beim Durchreichen im Netz verkleinert. Pooling Layer werden normalerweise zwischen aufeinanderfolgenden Convolutional Layers eingefügt. Ein Pooling Layer verwendet eine vorher definierte Funktion und eine Filtergröße zur Zusammenfassung von Informationen. Die Filtergröße bestimmt

den Faktor der Verkleinerung. Die Funktion bestimmt, auf welche Art die Informationen zusammengefasst werden. In dieser Arbeit werden nur 2x2 MaxPooling Layer verwendet, was bedeutet, dass aus einem Bereich von zwei mal zwei Pixeln, nur der höchste Wert weitergegeben wird und die Dimension der Eingabe halbiert.

Dense Layer

Ein Dense Layer bezeichnet eine Schicht, in der jedes Neuron der Schicht mit jedem Neuron der Folgeschicht verbunden ist. Vor 2012 war die Verwendung dieser Art Schicht auch im Bereich der Bildklassifizierung Standard, jedoch gilt seit diesem Jahr die Verwendung von Convolutional Layers als State-of-the-art, da sich damit bedeutend bessere Ergebnisse erzielen lassen[13]. Dennoch werden in dieser Arbeit, wie heutzutage oft, vor der Ausgabeschicht mehrere Dense Layer verwendet, um die Ergebnisse der vorigen Schichten zusammenzuführen.

2.3 Überwachtes Lernen und Training

Überwachtes Lernen ist der Prozess durch Algorithmen aus Datenbeispielen mit dazugehörigen Labels generalisierende Informationen zu ziehen und somit ein Modell zu bilden. Die Datenbeispiele werden hier Trainingsdaten genannt. Sie sind die Menge der Daten, die einem neuronalen Netz zum Erstellen struktureller Informationen zur Verfügung gestellt werden. In dem Beispiel dieser Arbeit stellen diese strukturellen Informationen Gewichte an den Kanten des neuronalen Netzes dar. Der Prozess zur Anpassung der, zu Beginn zufällig initialisierten, Gewichte basierend auf den Trainingsdaten, wird als Training bezeichnet. Weiterhin gibt es Validierungsdaten. Diese werden nicht für den Trainingsprozess verwendet, sondern für die Berechnung einer Bewertung der Verallgemeinerungsfähigkeit des Modells. Diese Daten werden in vielen Arbeiten auch Testdaten genannt, dieser Begriff sollte aber nicht mit dem Testen von Software verwechselt oder gar gleichgestellt werden [18].
Ein neuronales Netz kann als eine Funktion betrachtet werden, die aus einer Eingabe ein Ergebnis berechnet. Diese Funktion besitzt eine hohe Anzahl von Parametern, die erlernt und angepasst werden müssen, um ein gutes Modell der Abbildung von einem Eingabe- zu einem Ausgabewert zu geben. Die Parameter sind die Gewichte aller Kanten sowie die Filter der Convolutional Layers. Im Beispiel eines vorwärtsgerichteten Netzes

zur Bildklassifikation ist das Argument der Funktion einer Vektorrepräsentation aus den Pixelfarbwerten eines Bildes und die Ausgabe eine Wahrscheinlichkeitsverteilung über die vorher definierten Klassen.
Das eigentliche Lernen des Netzes basiert auf der Minimierung einer vorher definierten Fehlerfunktion, über alle Parameter des Netzes. Für diese werden die Trainingsdaten benötigt, die ein Eingabedatum mit einem korrekten Ausgabewert verbinden. Nach einem Durchlauf dieser Daten durch das Netz wird geprüft, wie weit der eigentliche Zielwert vom derzeitigen abweicht. Hieraus wird der Fehler des Netzes berechnet, auf Basis dessen sämtliche Gewichte des Netzes angepasst werden. Die Anzahl Epochen gibt eine der möglichen Abbruchbedingungen für den Lernprozess des Netzes an.
Die Anpassung der Gewichte geschieht durch einen stochastischen Gradientenabstieg unter Verwendung des Backpropagation-Algorithmus. Der Gradientenabstieg nähert sich schrittweise an ein Minimum der Fehlerfunktion des neuronalen Netzes an. Der Gradient der Kostenfunktion an einer Stelle gibt die Richtung des steilsten Anstiegs an. Beim Gradientenabstieg wird dem negativen Gradienten gefolgt, um die Fehlerfunktion zu minimieren. Der Backpropagation-Algorithmus wird genutzt um die Kosten auf die einzelnen Gewichte der Kanten zurückzuführen. Dies geschieht unter Verwendung der Kettenregel. Das Ziel der beiden Verfahren ist es, die Kostenfunktion, die von allen Parametern des Netzes abhängig ist, zu minimieren um so ein lokales Minimum der Funktion zu finden.
Das Training eines Netzes, also die Anpassung der Parameter der Funktion, findet in Epochen statt. Eine Epoche bezeichnet den vollständigen Durchlauf aller ausgewählten Trainingsdaten durch das Netz und die daraus resultierenden Veränderungen. Die Trainingsdaten müssen nicht alle auf einmal pro Epoche durch das Netz gereicht werden, sondern können in Batches unterteilt werden. Für jedes Batch erfolgt ein vollständiger Durchlauf, was zu einer häufigeren Anpassung der Gewichte führt.

2.4 Verwendete Werkzeuge

In diesem Abschnitt werden Bibliotheken, Frameworks und Programme vorgestellt, die in dieser Arbeit verwendet wurden. Die Auswahl erfolgte aufgrund von leichter Verwendbarkeit, Veränderbarkeit und kostenloser Verfügbarkeit.

2.4.1 Keras

Keras ist eine in Python geschriebene Open-Source Bibliothek für maschinelles Lernen [4]. Diese Bibliothek stellt Bausteine zur Verfügung, mit denen auch tiefe neuronale Netze schnell entwickelt werden können.
Keras bietet die Möglichkeit direkt ganze Schichten eines Netzes zu implementieren, ohne auf einzelne Neuronen achten zu müssen. Lediglich die Anzahl der Eingabe- und Ausgabeneuronen muss hierbei beachtet werden. Schichten können etwa Convolutional, Dense- oder Pooling Layer darstellen. Die Parameter für die einzelnen Schichten sind leicht einstell- und veränderbar.
Keras bietet Implementationen von Loss-Funktionen und der Berechnung von Gradienten, die für den Lernprozess eines Netzes elementar sind. Ebenso sind eine Vielzahl von Aktivierungsfunkionen bereits implementiert. Der eingebaute „ImageDataGenerator“ vereinfacht das Laden von Trainings- und Validierungsdaten und bietet die Möglichkeit, Duplikate der geladenen Bilder zu verändern, sogenannte „Data Augmentation“, sodass mehr Trainingsdaten entstehen und das zu trainierende Netz hoffentlich besser abstrahiert. Änderungen umfassen hierbei Operationen wie horizontales oder vertikales Spiegeln, Rotationen, Veränderungen der Helligkeit und weitere.

2.4.2 GIMP

Das „GNU Image Manipulation Program“ [30] (GIMP) ist ein plattformunabhängiges Bildbearbeitungsprogramm, das unter der GNU Public License [9] steht und somit frei verfügbar ist. Es bietet Möglichkeiten zur pixelweisen Bearbeitung von Bildern und der Einführung von Ebenen in diese. Ebenen ermöglichen es einzelne Bildelemente abzulegen und erlauben es, dass bei Bearbeitungen nur derzeit ausgewählte Ebenen verändert werden. Weiterhin kann GIMP zur Verzerrung und Rotation von gesamten Bildern oder einzelnen Teilen verwendet werden. GIMP verfügt über Auswahlwerkzeuge in verschiedenen Formen, die es ermöglichen Bildteile auszuwählen um sie weiter zu bearbeiten. Besonders diese Funktion zur Erzeugung von pseudo-randomisierten Mustern sind für diese Arbeit relevant.
GIMP erlaubt es zufälliges plastisches Rauschen in Bilder einzufügen. Die GIMP Render Filter erzeugen Muster, die vollkommen unabhängig von allen Bildteilen sind. Das erzeugte Rauschen ist also zufällig und stellt ein vollkommen neues Muster dar. Diese Art von Rauschen ist nützlich, da die Bildteile, die durch zufällige Muster ersetzt werden

sollen, über keinen logischen Zusammenhang mit den anderen Teilen des Bildes verfügen. In der Arbeit [24] hat sich gezeigt, dass die Verwendung von derartigem Rauschen sogar positive Effekte auf die Genauigkeit eines Netzes hat. Weiterhin wurde nachgewiesen, dass die Verwendung von Rauschen in einzelnen Bilderteilen das Netz dazu bewegt, mehr Aufmerksamkeit auf die gewünschten Merkmale eines Bilder zu lenken.
Das Rauschen wurde in dieser Arbeit verwendet, um die untersuchten Bilder gezielt zu Manipulieren. Für einige Versuche wurden gezielt entweder die eigentliche Klasseninstanz aus dem Bild oder alles ausser dieser Instanz entfernt, um diese Bildteile isoliert betrachten zu können. Eine genauere Erklärung zur Verwendung des plastischen Rauschens findet sich in Abschnitt 4.

2.4.3 iNNvestigate

Obwohl neuronale Netze erfolgreich sind, was ihre Anwendungsdomänen betrifft, werden sie auch heute noch immer größtenteils als Black Boxes betrachtet. Dennoch gibt es bereits einige Methoden, die Teile der inneren Funktion erklären und sogar Entscheidungsgrundlagen in Eingabedaten von Modellen darstellen können. „iNNvestigate“ [2] bietet Implementierungen dieser Methoden an, die leicht auf eigene Modelle anwendbar sind, sofern diese mit Keras entwickelt wurden. Die Bibliothek ist in der Programmiersprache Python verfasst, und bietet nach Installation und Import ein Interface zu den darin liegenden Algorithmen zur Analyse fertiger Modelle.
Die Bibliothek bietet unter anderem Implementierungen der folgenden Methoden:

- SmoothGrad
- Deconvnet
- PatternNet
- Layer-wise Relevance Propagation

All diese Methoden generieren eine Art „Heatmap“, also ein Abbild eines Eingabebildes, in welchem bestimmte Pixel markiert wurden, indem sie eine Rückrechnung von der Ausgabeschicht und ihrer Werte zu den Neuronen der Eingabeschicht durchführen.
Von diesen genannten Methoden wird in dieser Arbeit die Implementierung von Layer-wise Relevance Propagation für die Versuche verwendet. Eine genauere Erklärung der Funktionsweise findet sich im nächsten Abschnitt 3. Hier wird auch eine erste Evaluation der Methode vorgenommen.

3 Analyse von Visualisierungsmethoden

In diesem Kapitel wird eine Auswahl von Visualisierungsmethoden vorgestellt und analysiert, die später auf ihre Eignung als Testverfahren geprüft werden. Wie im Kapitel 2.2 beschrieben, wird das Verhalten von neuronalen Netzen nicht explizit definiert, sondern anhand von Testdaten gelernt. Das bedeutet, dass der Programmcode keinen Hinweis über die Entscheidungslogik enthält. Aufgrund dieser Eigenschaft werden andere Testmethoden als für traditionelle Software benötigt.
Die hier vorgestellten und analysierten Methoden verwenden die Gewichte und die Aktivierung in der Ausgabeschicht eines fertig trainierten Netzes um Visualisierungen von den Entscheidungen eines Netzes darzustellen. Die Auswahl erfolgte aufgrund eigener Erfahrung sowie des zu leistenden Aufwandes für ein Ergebnis und der Interpretierbarkeit der Ergebnisse.
Wie in anderen Arbeiten bereits beobachtet und beschrieben, hängt die Wahl der Methode auch von der Aufgabenstellung und Topologie des betrachteten Netzes ab [33] [19] [21]. In dieser Arbeit werden nur Methoden betrachtet, die auf CNNs zur Bildklassifikation angewendet werden können. Folgende Methoden wurden in Betracht gezogen:

- Layer-Wise-Relevance Propagation
- Class Model Visualization

3.1 Layer-Wise Relevance Propagation

Layer-Wise Relevance Propagation (LRP) ist eine von Bach et al. [3] entworfene Methode zur Erklärung von Entscheidungen neuronaler Netzwerke. Für ein Eingabedatum, in dieser Arbeit ein zu klassifizierendes Bild, kann die Entscheidung eines bereits trainierten neuronalen Netzes visualisiert werden. Die Grundidee ist, dass LRP die internen Gewichte eines fertig trainierten Netzes nutzt, um eine Rückrechnung von der Ausgabeschicht zur Eingabeschicht durchzuführen und so die Relevanz der einzelnen Eingabeneuronen

auf eine Entscheidung aufzeigt.
Der Großteil der Informationen stammt aus einem Sammelwerk [20] über das Interpretieren und die Visualisierung von maschinellem Lernen, sowie der Originalarbeit [3].

3.1.1 Heatmap

Layer-Wise Relevance Propagation basiert, wie andere Visualisierungsmethoden auch, auf pixelweiser Zerlegung der Entscheidung eines Modells. Damit ist die Berechnung des Beitrages eines einzelnen Pixels x_d des Eingabebildes x auf eine Entscheidung $f(x)$ eines Modells gemeint. Mithilfe dieser Berechnung wird jeder Pixel entsprechend seines Beitrages im Originalbild gefärbt. Dies funktioniert, da jeder Pixel der Eingabe einem Neuron der Eingabeschicht entspricht. Dieses neue Bild wird Heatmap genannt. Die Beiträge oder auch Relevanz R, einzelner Pixel wird in drei Gruppen unterschieden:

- $R > 0$ positiver Beitrag (in dieser Arbeit rot)
- $R < 0$ negativer Beitrag (in dieser Arbeit blau)
- $R = 0$ kein Beitrag (in dieser Arbeit weiß)

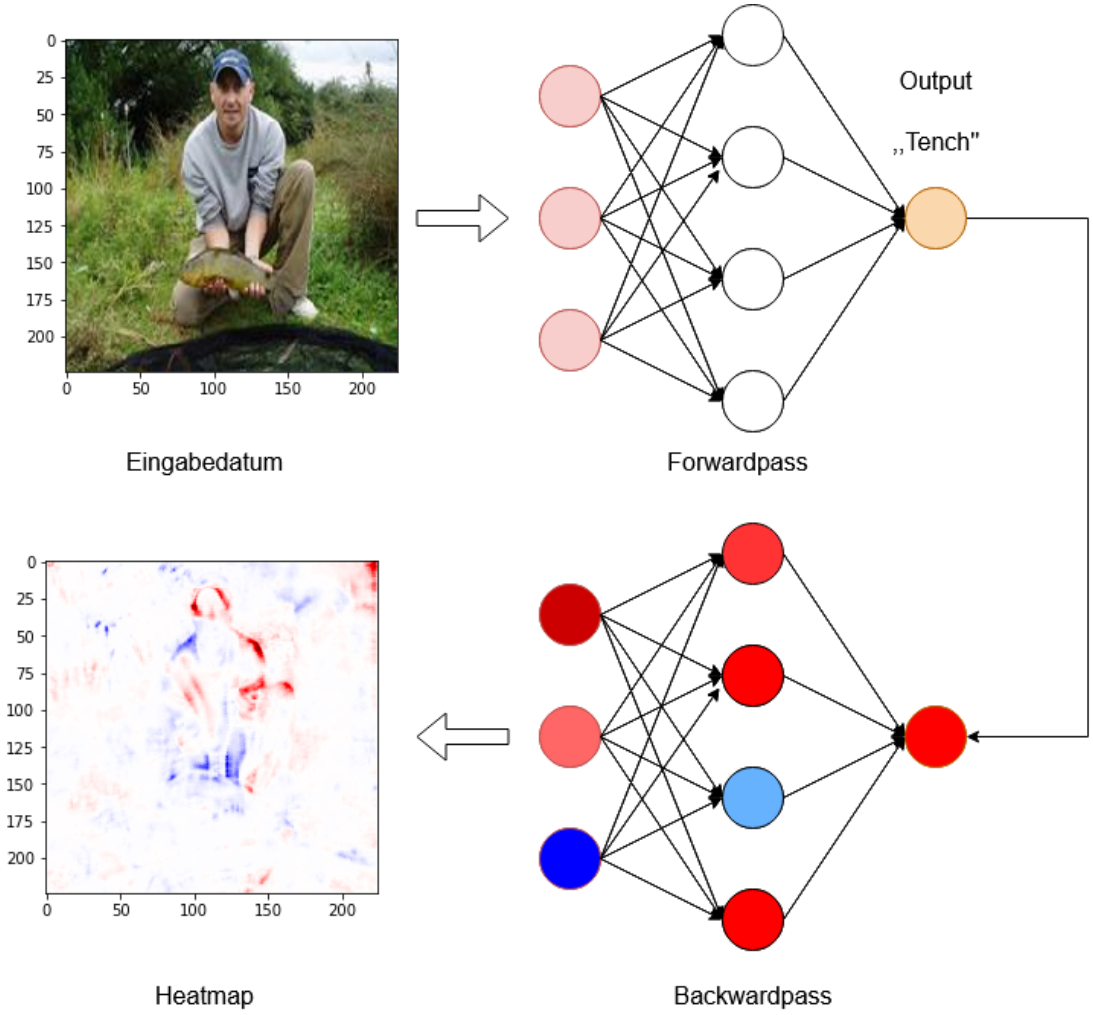

Abbildung 3.1: Erzeugung einer Heatmap durch einen Backwardpass. Vom Autor nach [20].

3.1.2 Funktion

LRP arbeitet mit einem Backwardpass, der sich ausgehend von der Ausgabeschicht über alle Schichten bis zur Eingabeschicht verbreitet. Dieser Backwardpass unterliegt einer conversation property, also der Eigenschaft, dass Werte die ein Neuron empfangen hat, auch in gleichem Maße an die darüberlegende Schicht weitergereicht werden müssen. Von den Autoren wird die Verwendung der ReLU (2.2.1) Aktivierungsfunktion in den inneren Schichten des Netzes empfohlen.
Das Weiterreichen der Relevanz R beginnt an einem Ausgabeneuron, dessen Aktivierung

als Relevanz verwendet wird. Von diesem Punkt aus wird rückwärts durch das Netz gegangen, um für jedes Neuron jeder Schicht die Relevanz, anhand der Gewichte und der Aktivierung zu berechnen. Diese Berechnung folgt der einfachsten LRP Regel, auch LRP-0 genannt:

$$R_j = \sum_k \frac{a_j w_{j_k}}{\sum_{0,j} a_j w j_k} R_k$$

j und k bezeichnen Neuronen von aufeinanderfolgenden Schichten, wobei das Neuron k sich dichter an der Ausgabeschicht befindet, also während der Klassifizierung seine Werte von Neuron j erhalten hat. R_j bezeichnet somit die zu berechnende Relevanz eines Neuron einer höheren Schicht, R_k die Relevanz des derzeit betrachteten Neurons, der tieferen Schicht. a bezeichnet die Aktivierung eines Neurons, w das Gewicht an der Kante zwischen zwei Neuronen.
In dieser Formel bezeichnet der Zähler den Einfluss von Neuron j auf Neuron k. Der Nenner ist die Summe aller Einflüsse der Neuronen der tieferen Schicht. Durch diese Division wird die zuvor beschriebene conversation property sichergestellt. Die Summe $\sum_k$ gibt an, dass die Relevanz R_j von der dem Einfluss aller Neuronen der tieferen Schicht abhängt.

3.1.3 Vorläufige Evaluation von LRP

Diese Methode wurde ausgewählt, da bereits in der Arbeit [14] durch einige Beispiele der Nutzen und die Effektivität dieser Methode zur Visualisierung dargestellt wurde. Weiterhin verspricht die zuvor beschriebene conversation property, dass die Heatmaps aus dieser Methode tatsächlich die Entscheidungen des Netzes darstellen können. Wie in 2.2 beschrieben, wird einem Modell ein Bild in Pixeldarstellung übergeben und diese Pixel werden zum Finden einer Korrelation zwischen diesem Bild und einer der vorher definierten Klasse verwendet. LRP erzeugt eine Heatmap, in der alle Pixel des Eingangsbildes vorhanden sind, sodass die Heatmap leicht von Menschen zu interpretieren ist.
Erste eigene Experimente bei der Anwendung dieser Methode haben gut interpretierbare Ergebnisse geliefert. Nach einigen Versuchen konnten auch direkt Hinweise für die Bestätigung der These gefunden werden, dass neuronale Netze auch Artefakte, statt der eigentlichen Klasse, zur Klassifikation nutzen. Nach anfänglicher Betrachtung weiterer, zu LRP ähnlicher, Methoden wurden diese ausgeschlossen, da die Ergebnisse schwieriger

zu interpretieren wirkten. Insbesondere eine weitere Familie oft angewandter Methoden zur Visualisierung von Heatmaps, wie etwa Guided GradCAM oder SmoothGrad, wurde in einer Arbeit durchleuchtet und diese hab Hinweise, dass diese Methoden Ergebnisse erzeugen, die nichts mit den Gewichten oder den Trainingsdaten des betrachteten Modells zu tun haben [1]. Aufgrund dieser Hinweise wurde im Rahmen dieser Arbeit auf diese Methoden verzichtet.

3.2 Class Model Visualization

Genau wie LRP verwendet Class Model Visualization (CMV) die Gewichte des zu untersuchenden fertig trainierten neuronalen Netzes. Nach dem Trainingsprozess sind diese Gewichte statisch und werden vom Netz für eine Klassifikation verwendet. Die Idee hinter CMV ist, dass die Gewichte des Modells genutzt werden, um für eine Klasse eine Art Prototyp zu visualisieren. Dieser Prototyp, im Weiteren Class Model genannt, wird mithilfe der maximalen Aktivierung des, der Klasse zugeordneten Ausgabeneurons, generiert. Anders als bei LRP wird nicht ein einzelnes Bild verwendet, um auf diesem die Relevanz einzelner Pixel zu visualisieren, sondern es entsteht exakt ein Bild pro Ausgabeklasse.
Die Information zur beschriebenen Methode in diesem Abschnitt stammen aus den Werken:

- Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps [27]
- Visualizing Higher-Layer Features of a Deep Network [6]

3.2.1 Class Model

Abbildung 3.2: Class Models eines CNNs, trainiert auf dem ImageNet Datensatz. Aus [27].

Die Abbildung 3.2 zeigt einige Beispiele von Class Models. Dies sind Bilder einer Klasse, die das entsprechende Ausgabeneuron des betrachteten Modells maximal aktivieren. Das Class Model I wird wie folgt berechnet:

$$\arg\max_I S_c(I) - \lambda \|I\|_2^2 \tag{3.1}$$

Der arg max Operator dient dazu, ein Class Model I zu finden, welches den zweiten Teil der Formel maximiert. Bei der Eingabe von I soll das trainierte Netz anhand seiner Gewichte einen hohen Score S_c erreichen. Das Netz soll sich maximal sicher sein, dass das Bild I zur Klasse c gehört. Der letzte Teil der Formel $\lambda \|I\|_2^2$ ist der L2-Regularisierungsterm. Dieser sorgt dafür, dass das produzierte Bild keine extremen Werte enthält. Die Ausgabe wird damit glatter und leichter vom Menschen interpretierbar. Ein Class Model wird durch eine Art Backpropagation eines leeren Bildes durch ein Netz produziert, nur, dass sich im Gegensatz zum Trainingsprozess nicht die Gewichte, sondern die Pixel des Bildes ändern.

3.2.2 Vorläufige Evaluation von CMV

Diese Methode wurde gewählt, da sie ein geringen Anwendungsaufwand suggerierte. Anstatt einzelne Bilder pro Klasse stichprobenartig untersuchen zu müssen, wie bei LRP,

muss nur für jede vorher definierte Klasse genau eine Anwendung der Methode stattfinden. Dies verringert den Zeitaufwand und verspricht auch aussagekräftigere Ergebnisse, da das gesamte Modell im Bezug auf eine Klasse untersucht wurde, statt nur die Relevanz einzelner Pixel in einem Bild. Wenn ein Netz die Anwesenheit von Artefakten für eine Klassifizerung nutzt und somit eine Korrelation zwischen den Artefakten und einer Klasse gibt, müssten diese Artefakte auch im Class Model dieser Klasse aufzufinden sein. Nach anfänglichen Experimenten zeigte sich jedoch schnell, dass die aus dem hier verwendeten Modell zur Bildklassifikation entstandenen Bilder nicht zu Interpretieren waren. Interpretiertbarkeit vom Menschen ist aber eine essentielle Anforderung an Visualisierungsmethoden [16], besonders im Bereich des Testens, da eine Evaluation sowohl aus der korrekten Klassifikation als auch aus der Nachvollziehbarkeit einer Entscheidung bestehen sollte. Die Nachvollziehbarkeit der Entscheidung ist hierbei kaum gegeben, da die durch CMV entstandenen Bilder sehr abstrakt sind. Diese Bilder sind auch in guten Fällen nicht leicht zu interpretieren, wie Experimente im Ursprungspaper zeigten [27].
Weitere Recherche und Experimente legten nahe, dass eine Veränderung der Netztopologie vorgenommen werden müsste, um bessere Ergebnisse zu erzielen. Diese Veränderung würde jedoch aus Vergleichbarkeitsgründen mit LRP abgelehnt, da sich dort bereits mit der derzeitigen Architektur 4.2 gute Ergebnisse zeigten. Aufgrund dieser Unterschiede in der Qualität erster Ergebnisse und aus Zeitgründen wurde auf eine weitere Untersuchung dieser Methode verzichtet. Es wurden mehrere Implementierungen dieser Methode erprobt, jedoch zeigten alle ungenügende Ergebnisse. Es kann nicht ausgeschlossen werden, dass die Werkzeuge fehlerfrei funktionieren oder falsch angewandt wurden.

4 Versuchsaufbau

In diesem Abschnitt werden die einzelnen Versuche beschrieben, mit denen die trainierten neuronalen Netze untersucht wurden. Die Anwendung der in Abschnitt 3 beschriebenen Methoden wird dabei als eine Art Blackbox Test betrachtet, da lediglich Ein- und Ausgabedaten des Models betrachtet und interpretiert werden. Mit den Versuchen soll untersucht werden, ob sich die in 3 beschriebene Methode LRP eignet, um Probleme mit bereits trainierten CNNs aufzuzeigen. Die Verwendung der auch im Abschnitt 3 vorgestellten Methode der Class Model Visualization wurde nach einer längeren ersten Evaluation verworfen, da eine Änderung an der Netztopologie sowie weiterer Zeitaufwand geleistet werden müsste, damit diese Methode im Rahmen dieser Arbeit sinnvoll angewendet werden könnte. Es wird angenommen, dass Modelle nicht nur die eigentliche Instanz einer Klasse zur Klassifikation verwenden, sondern auch andere Artefakte und deren Charakteristika [14]. Die hier verwendeten Methoden untersuchen somit die Trainingsdaten und versuchen eine Aussage über deren Qualität zu treffen. Mit der LRP Methode sollen also Probleme mit diesen Trainingsdaten aufgezeigt werden, wie etwa der Fall, dass viele Bilder einer Klasse ähnliche Artefakte enthalten, die nicht zur eigentlichen Klasseninstanz gehören. Es wird angenommen, dass die Anwesenheit gleichförmiger Artefakte dazu führt, dass das Netz nach dem Training auch diese Artefakte zur Klassifizierung verwendet [14].

4.1 Datensatz

Als Trainingsdatensatz wurde „ImageNette“ [12] gewählt. Dieser Datensatz ist eine Teilmenge der „ImageNet“ [25] Bilddatenbank, welche etwa 21.000 Klassen enthält. Zu jeder dieser Klassen sind Fotos zugeordnet, welche ein Objekt oder eine Aktivität darstellen. Durch die hohe Auflösung sind diese Bilder sehr gut von Menschen erkennbar.
Von 2010 bis 2017 wurde jährlich ein Wettbewerb für Bildklassifikation und Objekterkennung mit diesem Datensatz abgehalten, sodass es viele Modelle gibt, mit denen Vergleiche

angestellt werden können [25]. Im Jahr 2012 wurde der Wettbewerb von einer Gruppe gewonnen, die ein CNN als Netz-Architektur für Bildklassifikation genutzt haben [13]. In den folgenden Jahren waren die Architekturen mit der höchsten Genauigkeit immer CNNs [25].
Die Teilmenge „ImageNette“ [12] wurde gewählt, da durch die geringere Anzahl an Klassen und die kleinere Auflösung ein schnelleres Training eines CNNs möglich ist. Die Auflösung ist dennoch groß genug, um vom Menschen gut erkennbar und interpretierbar zu sein. Gerade bei der Verwendung von LRP ist dies wichtig, da einzelne Pixel in der Heatmap markiert werden. Der Datensatz enthält 10 Klassen mit jeweils etwa 900 Bildern als Trainingsdaten. Die Klassen sind „tench“, „springer“, „casette player“, „chainsaw“, „church“, „French horn“, „garbage truck“, „gas pump“, „golf ball“ und „parachute“. Jedes einzelne Bild ist maximal 244 Pixel breit und hoch, mit einer minimalen Breite und Höhe von 160 Pixeln.
Die Klasse „tench“ wurde ausgewählt, da auf mehr als 55% der Trainingsdaten Menschen sichtbar sind. Da die Menschen nicht zur Klasse gehören, stellen sie Artefakte dar. Der Prozentsatz stammt aus einer manuellen Zählung im Rahmen dieser Arbeit. Durch die Größe, Form und Anzahl der Artefakte im Trainingsdatensatz wird angenommen, dass diese einen Einfluss auf die Klassifizierung der Klasse haben, obwohl sie kein Teil der Instanz sind.
Die Klasse „golf ball“ ist gut geeignet, da keine gemeinsamen Artefakte auf den Bildern der Klasse sichtbar sind. Durch die die Abwesenheit gemeinsamer Artefakte und die homogene Form wird angenommen, dass nur die eigentliche Klasseninstanz einen Einfluss auf die Klassifizierung hat.

Abbildung 4.1: ILSVRC2012_val_00013633.JPEG „golf ball“ aus [25]

Abbildung 4.2: ILSVRC2012_val_00034826.JPEG „french horn“ aus [25]

Abbildung 4.3: ILSVRC2012_val_00018207.JPEG „springer“ aus [25]

Abbildung 4.4: ILSVRC2012_val_00009346.JPEG „tench“ aus [25]

Abbildung 4.5: Bilder von vier der zehn „ImageNette“ Klassen [12].

4.2 Modell zur Bildklassifikation

Für die Lösung des Problems der Bildklassifikation wurde in dieser Arbeit ein neuronales Netz gewählt. Wie im Kapitel 2.1 beschrieben, haben sich neuronale Netze in den letzten Jahren als genaueste Methode herausgestellt, um dieser Aufgabe nachzukommen. Die Architektur des Netzes wurde selbst entworfen, orientiert sich aber grob am VGG-Net16 [28]. Dieses Netz erzielte bei ImageNet [25] Wettbewerben hohe Genaugkeiten. Da ImageNette [12] eine Teilmenge der ImageNet [25] Daten ist, bietet die Architektur von VggNet16 eine gute Basis für ein eigenes Netz.
Alle Versuche wurden auf dem Modell mit derselben Architektur durchgeführt. Lediglich für einen Versuch wurden andere Trainingsdaten gewählt und ein neues Training fand statt. Dies ist einer der Gründe für das Training und die eigene Archtitektur eines Netz. Sowohl die Trainings- als auch die Validierungsdaten mussten komplett selbst wählbar sein. Dies verhinderte den Einsatz eines bereits verfügbaren und trainierten Modells. Weiterhin ist für Versuche eine möglichst große Kontrolle über die Versuchsobjekte und -umgebung wünschenswert.
Die Architektur beider Modelle folgt grob dem Aufbau von VggNet16 [28], jedoch wurden eigene Veränderungen anhand von Optimierungstechniken aus Patterson und Gibson 17 [23], sowie eigenen Anpassungen der Hyperparameter vorgenommen. Das Netz wurde mit einer Batchgröße von 16 Bildern trainiert. Die Anzahl der Trainingsepochen beträgt 37, wobei jedoch ein Optimum der Kostenfunktion und der Genauigkeit auf den Validierungsdaten bereits nach 17 Epochen auftrat. Das Netz wurde so lange trainiert, bis die Genauigkeit 20 Epochen lang kein neues Maximum erreicht hat.

Layer (type)	Output Shape	Param #
conv2d_12 (Conv2D)	(None, 224, 224, 64)	1792
conv2d_13 (Conv2D)	(None, 224, 224, 64)	36928
max_pooling2d_6 (MaxPooling2	(None, 112, 112, 64)	0
batch_normalization_6 (Batch	(None, 112, 112, 64)	256
conv2d_14 (Conv2D)	(None, 112, 112, 128)	73856
conv2d_15 (Conv2D)	(None, 112, 112, 128)	147584
max_pooling2d_7 (MaxPooling2	(None, 56, 56, 128)	0
batch_normalization_7 (Batch	(None, 56, 56, 128)	512
conv2d_16 (Conv2D)	(None, 28, 28, 256)	295168
conv2d_17 (Conv2D)	(None, 14, 14, 256)	590080
max_pooling2d_8 (MaxPooling2	(None, 7, 7, 256)	0
batch_normalization_8 (Batch	(None, 7, 7, 256)	1024
conv2d_18 (Conv2D)	(None, 4, 4, 512)	1180160
conv2d_19 (Conv2D)	(None, 2, 2, 512)	2359808
max_pooling2d_9 (MaxPooling2	(None, 1, 1, 512)	0
batch_normalization_9 (Batch	(None, 1, 1, 512)	2048
conv2d_20 (Conv2D)	(None, 1, 1, 512)	2359808
conv2d_21 (Conv2D)	(None, 1, 1, 512)	2359808
max_pooling2d_10 (MaxPooling	(None, 1, 1, 512)	0
batch_normalization_10 (Batc	(None, 1, 1, 512)	2048
flatten_2 (Flatten)	(None, 512)	0
dense_4 (Dense)	(None, 8192)	4202496
dropout_2 (Dropout)	(None, 8192)	0
dense_5 (Dense)	(None, 4096)	33558528
dropout_3 (Dropout)	(None, 4096)	0
dense_6 (Dense)	(None, 10)	40970

Total params: 47,212,874
Trainable params: 47,209,930
Non-trainable params: 2,944

Abbildung 4.6: Mit keras erzeugte Zusammenfassung der Architektur des Modells.

Beide Modelle erreichten eine Genauigkeit von 78% auf den Validierungsdaten. Die Validierungsdaten wurden vorher ausgewählt und nicht während des Trainings verwendet. Das bedeutet also, dass die Modelle 78% der nicht gesehenen Bilder korrekt klassifiziert haben. Gründe hierfür sind etwa die recht geringe Anzahl an Trainingsdaten. Für das Erlernen von zehn Klassen sind lediglich 9469 Bilder vorhanden. Andere Metriken, als die erzielte Genauigkeit auf den Validierungsdaten, wurden für diese Arbeit nicht betrachtet, da das Modell nur als Versuchsobjekt dient und die Genauigkeit des Modells nichts mit den eigentlichen Versuchen zu tun hat. Dennoch bedingen diese Tatsachen die Aussagekraft der nachfolgenden Versuche.

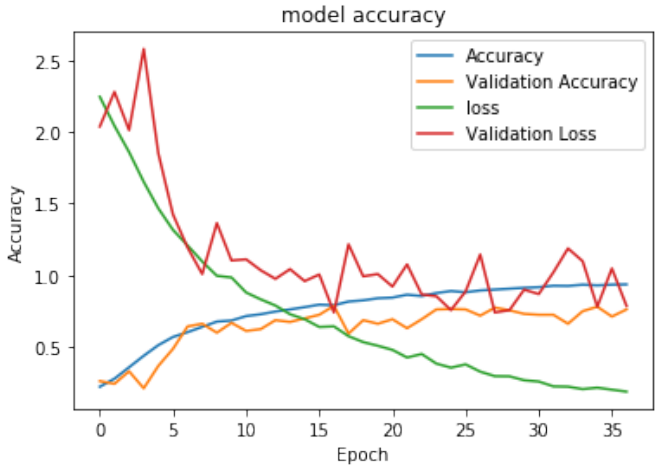

Abbildung 4.7: Mit Pyplot erzeugter Verlauf der Trainingsgenaugkeiten

4.3 Veränderung der Bilddaten

Für einige Versuche wurden Teile der Trainings- und Validierungsdaten manuell verändert. Für LRP-E2 und LRP-E3 wurden die Änderungen mit GIMP [30] vorgenommen. Die Änderungen der Trainingsdaten für LRP-E4 wurden mit einem Script vorgenommen, welches die Python Imaging Library (PIL) [5] verwendet, um Farbwerte in den ursprünglichen Daten zu ändern.

4.3.1 Veränderung mit GIMP

Um genauere Untersuchungen vorzunehmen, welchen Teil eines Bildes das oben beschriebene Modell verwendet, um eine Klassifikation vorzunehmen, wurden Teile des Bildes entfernt und durch von GIMP erzeugtes Rauschen ersetzt [31]. In Tremblay2018 [32] wurde gezeigt, dass das Austauschen von Teilen des Bildes, die nicht zur eigentlichen Instanz gehören, durch Rauschen einen positiven Einfluss auf die Genauigkeit eines Modells hat.
Für LRP-E2 wurde die eigentliche Instanz einer Klasse aus Bildern der Validierungsdaten entfernt und durch Rauschen ersetzt, um dem Netz nur den nicht zur Klasse gehörigen Bereich übergeben zu können.
Für LRP-E3 wurde alles, außer der eigentlich Instanz, aus Bildern der Validierungsdaten entfernt und durch Rauschen ersetzt, um dem Netz nur die Instanz einer Klasse übergeben zu können, ohne etwaige Artefakte, die in den Trainingsdaten aufgetreten sein können.

4.3.2 Veränderungen mit PIL

Für LRP-E4 wurde ein Skript entwickelt, welches mithilfe der Python Imaging Library [5] Bilddaten einliest und die Farbwerte in einem quadratischen Bereich in der linken oberen Ecke auf den RGB Farbwert (255, 0 , 0) setzt. Diese Veränderung soll einen Zeitstempel oder ein Wasserzeichen simulieren, wie es manchmal auf Fotos zu finden ist.
Diese synthetischen Artefakte wurden in 25% aller Trainingsdaten der Klasse „golf ball“ eingefügt und sollen dem Netz eine Gemeinsamkeit in den Trainingsdaten einer Klasse zeigen, die kein Teil der eigentlich Klasse sind. Diese Klasse wurde ausgewählt, da der Großteil der Bilder keine offensichtlichen Artefakte enthält, die in jedem Bild vorkommen. Weiterhin besitzen die Instanzen dieser Klasse eine charakteristische Form und nahezu keine Varianz in dieser. Aufgrund der Kugelform ist der Winkel, aus dem das Foto aufgenommen wurde, irrelevant.

4.4 Evaluation von Layer-wise Relevance Propagation

In den folgenden Abschnitten werden Versuche durchgeführt, in denen ein bereits trainiertes Netz anhand der Eingabe manuell ausgewählten Trainingsdaten untersucht wird.

Hierbei werden sowohl unveränderte Bilder aus den Validierungsdaten verwendet, als auch manuell veränderte Bilder, um erfolgte Beobachtungen zu spezifizieren.

4.4.1 Untersuchung mit unveränderten Trainings- und Validierungsdaten (LRP-E1)

In diesem Versuch wird ein Model verwendet, welches auf unveränderten Bildern des Datensatzes trainiert wurde. Als Input zur Klassifizierung werden unveränderte Bilder aus den Validierungsdaten genutzt.

Abbildung 4.8: Eine modifizierte Version von ILSVRC2012_val_00006697.JPEG aus [25]

Abbildung 4.9: Eine modifizierte Version von ILSVRC2012_val_00037861.JPEG aus [25]

Abbildung 4.10: Zwei manuell veränderte Bilder der Klasse „Tench“ . In rot die eigentliche Klasse, in gelb große Artefakte die in etwa 55% der Trainingsbilder vorkommen.

Es werden Bilder der Klasse „Tench“ verwendet, bei der auf vielen Trainingsbildern das gleiche Artefakt sichtbar ist. In diesem Fall ist auf über 55% der Trainingsbilder mindestens der Oberkörper oder das Gesicht eines Menschen zu sehen. Das häufige Vorkommen von Artefakten, die nicht Teil der Klasseninstanz sind, kann darauf hinweisen, dass das neuronale Netz auch andere Teile des Bildes zur Klassifizierung nutzt. Dies kann aufgrund des häufigen kombinierten Auftretens der Artefakte und der Klasseninstanz geschehen, sodass das eigentlich zu lernende Merkmal nicht eindeutig ist.
Das Model wird genutzt, um eine Klassifizierung der Bilder vorzunehmen. Dabei wird

die prozentuale Sicherheit der Entscheidung notiert. Danach wird eine Heatmap der Entscheidung mithilfe von Layer-wise Relevance Propagation erzeugt.

4.4.2 Untersuchung von Validierungsdaten ohne Klasseninstanz (LRP-E2)

In diesem Versuch wird dasselbe Modell wie in LRP1 4.4.1 verwendet. Als Input werden händisch ausgewählte und veränderte Bilder aus den Validierungsdaten genutzt. Dieser Versuch dient zur Bekräftigung der These, dass Netze Artefakte nutzen um die eigentliche Klasse zu erkennen.
Es werden dieselben Bilder wie im ersten Versuch verwendet, allerdings werden diese manuell so verändert, dass die eigentliche Instanz einer Klasse aus dem jeweiligen Bild entfernt wird. Der so entstandene leere Bereich wird mit zufälligem Rauschen gefüllt.

Abbildung 4.11: ILSVRC2012 _val_00037861.JPEG aus [25]. Die Instanz der Klasse wurde durch Rauschen ersetzt.

Nach dieser Änderung wird das Model genutzt, um eine Klassifizierung der Bilder vorzunehmen, dabei wird die prozentuale Sicherheit der Entscheidung notiert. Danach wird eine Heatmap der Entscheidung mithilfe von Layer-wise Relevance Propagation erzeugt.

4.4.3 Untersuchung von Validierungsdaten mit zufälligem Hintergrund (LRP-E3)

In diesem Versuch wird dasselbe Model wie in LRP14.4.1 verwendet. Als Input werden händisch ausgewählte und veränderte Bilder aus den Validierungsdaten genutzt. Dieser

Versuch dient zur Bekräftigung der Ergebnisse aus Versuch LRP1.
Es werden dieselben Bilder wie im ersten Versuch verwendet, allerdings werden diese manuell so verändert, dass nur die eigentliche Instanz einer Klasse im Bild vorhanden ist. Der restliche Bereich wird mit zufälligem Rauschen gefüllt.

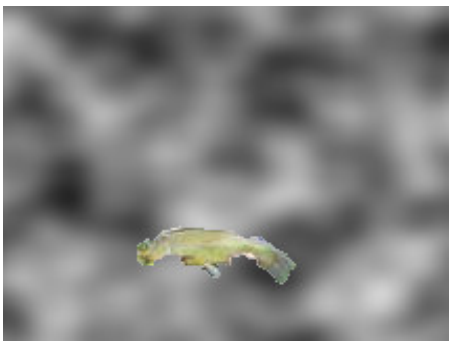

Abbildung 4.12: ILSVRC2012_val_00037861.JPEG aus [25]. Alles außer der Klasseninstanz wurde durch zufälliges Rauschen ersetzt.

Nach dieser Änderung wird das Model genutzt, um eine Klassifizierung der Bilder vorzunehmen, dabei wird die prozentuale Sicherheit der Entscheidung notiert. Danach wird eine Heatmap der Entscheidung mithilfe von Layer-wise Relevance Propagation erzeugt.

4.4.4 Untersuchung mit synthetischen Artefakten in Trainings- und Validierungsdaten (LRP-E4)

Für diesen Versuch wird ein neues Model, derselben Architektur, mit manuell veränderten Daten trainiert. Diese veränderten Bilddaten enthalten in der linken oberen Ecke ein rotes Viereck. Dieses Viereck soll synthetische Artefakte, wie etwa ein Wasserzeichen oder einen Zeitstempel im Foto darstellen.

Abbildung 4.13: ILSVRC2012 _val_00036171.JPEG aus [25]. In das Bild wurde über einen Automatismus ein rotes Viereck eingefügt

Nach dieser Änderung wird das Model genutzt, um eine Klassifizierung eines Bildes dieser Klasse ohne synthetisches Artefakt vorzunehmen. Danach wird eine Heatmap der Entscheidung mithilfe von Layer-wise Relevance Propagation erzeugt.
Als weiterer Schritt wird eine Klassifizierung eines Bildes mit Artefakt vorgenommen und eine Heatmap von der Klassifizierungsentscheidung erzeugt.

5 Beobachtungen und Diskussion

In diesem Kapitel werden die Ergebnisse aus dem vorigen Kapitel 4 genauer betrachtet. Im ersten Teil werden Teile der Ergebnisse jedes einzelnen Versuches dargestellt und beschrieben. Dies umfasst eine Gegenüberstellung und Beschreibung der Originaldaten und deren durch LRP erzeugte Heatmaps, sowie die manuell veränderten Bilddaten und deren Heatmaps.
Die Klassifikation der Bilder und die Erzeugung der Heatmaps erfolgten am im Abschnitt 4.2 beschriebenen neuronalen Netz zur Bildklassifikation von ImageNette [12] Bildern. Im zweiten Teil werden die Beobachtungen interpretiert und Ergebnisse einzelner Versuche miteinander verglichen.

5.1 Beobachtungen der Versuche mit LRP

In diesem Abschnitt werden Teile der Ergebnisse aus denen in 4 beschrieben Versuche vorgestellt. Alle Bilder stammen aus dem ImageNette [12] Datensatz und werden im Original oder in manipulierter Form dargestellt. Die Sicherheit, mit der das Modell das Bild klassifiziert hat, oft auch „score“ genannt, ist an einigen Bildern vermerkt. Es werden nur ausgewählte Bilder dargestellt, die restlichen Ergebnisse befinden sich im Anhang.
Rote Farbwerte in einer Heatmap markieren Pixel im Eingabebild, die besonders relevant für die ausgegebene Klassifikation eines Bildes durch das Modell waren. Blaue Farbwerte in einer Heatmap markieren Pixel im Eingabebild, die gegen die ausgegebene Klassifikation eines Bildes durch das Modell sprechen.

5.1.1 Beobachtungen von LRP-E1

Hier wurden die unveränderten Bilder vom Modell klassifiziert und eine Heatmap durch LRP erzeugt.

Abbildung 5.1: ILSVRC2012_val_00006697.JPEG aus [25]

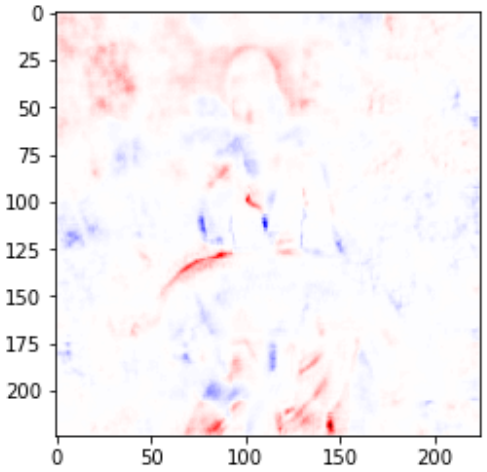

Abbildung 5.2: Die mit LRP erzeugte Heatmap desselben Bildes

Abbildung 5.3: Ein unverändertes Bild der Klasse „tench“ mit einem Menschen als Artefakt und die durch LRP erzeugte Heatmap

Das Bild 5.1 wurde, mit einer Sicherheit von 99,98%, korrekt klassifiziert. Auf der Heatmap ist zu sehen, wie der Kopf, die rechte Schulter und das Knie des Menschen im Bild relevant für Klassifikation des Bildes als „tench“ sind. Außer dem Oberteil des Fischkopfes spricht die eigentliche Instanz der Klasse gegen diese Klassifikation.

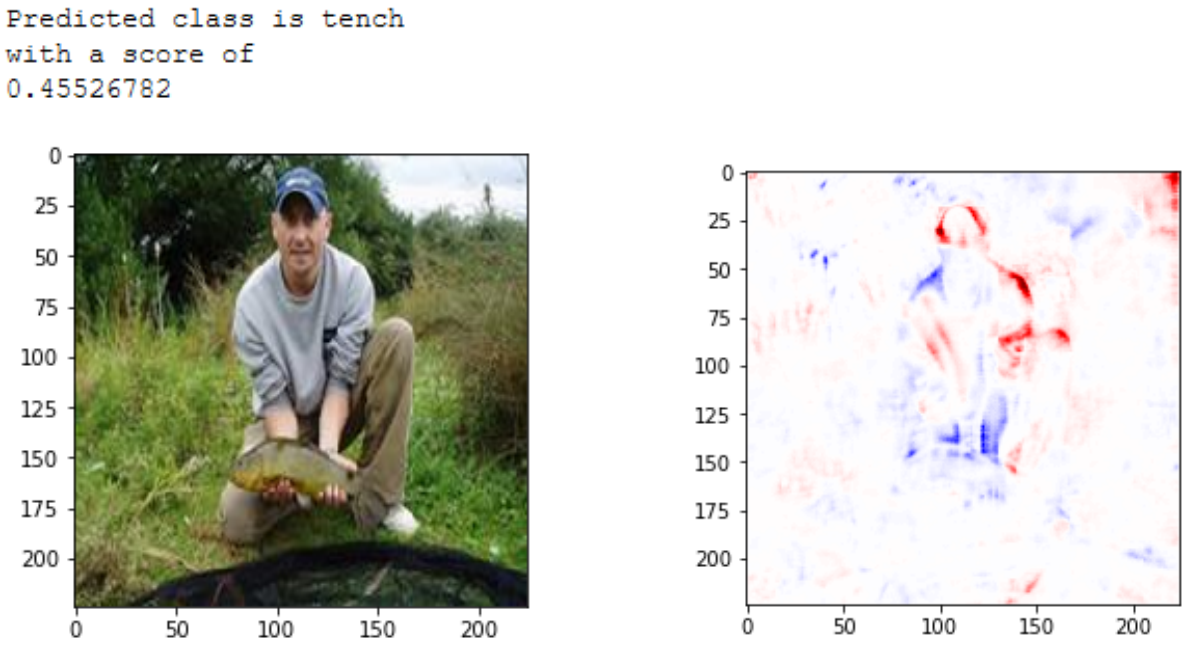

Abbildung 5.4: ILSVRC2012_00009379.JPEG.aus [25]

Abbildung 5.5: Die mit LRP erzeugte Heatmap desselben Bildes

Abbildung 5.6: Ein weiters unverändertes Bild der Klasse „tench“ mit einem Menschen als Artefakt und die durch LRP erzeugte Heatmap

Das Bild 5.4 wurde, mit einer Sicherheit von 45,53%, korrekt klassifiziert. Auf dieser ist zu sehen, wie der Kopf und die Knie des Menschen relevant für die Klassifikation waren. In diesem Beispiel spielte außerdem die obere Hälfte des Kopfes des eigentlich zu klassifizierenden Fisches eine große Rolle. Weiterhin scheint ein Teil der Birken im Hintergrund relevant für die korrekte Klassifikation zu sein.

5.1.2 Beobachtungen von LRP-E2

Hier wurden die Bilder, aus denen die eigentliche Klasseninstanz entfernt wurde, vom Modell klassifiziert und eine Heatmap durch LRP erzeugt.

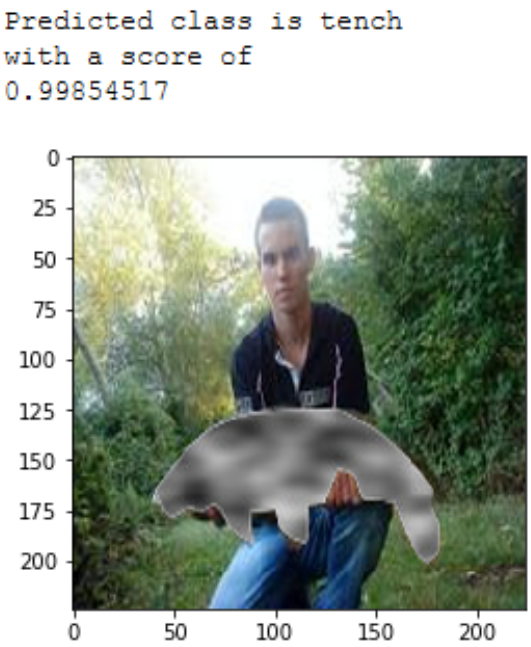

Abbildung 5.7: Eine modifizierte Version von ILSVRC2012_val_00037861.JPEG aus [25]

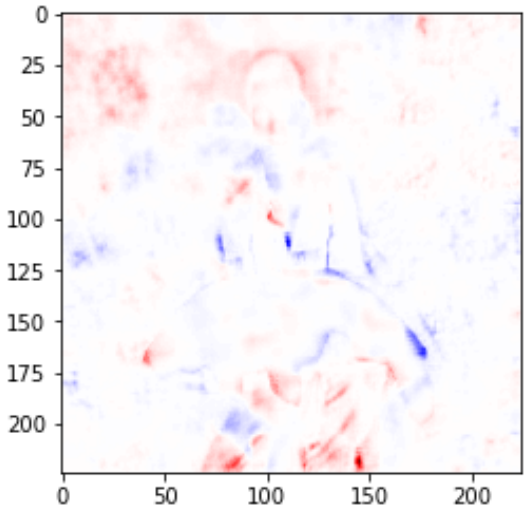

Abbildung 5.8: Die durch LRP erzeugte Heatmap desselben Bildes

Abbildung 5.9: Ein Bild, aus dem die eigentliche Instanz der Klasse „tench“ entfernt wurde und die durch LRP erzeugte Heatmap

Das Bild 5.7 wurde mit einer Sicherheit von 99,85% richtig als „tench“ klassifiziert. Dabei zeigt die Heatmap, wie schon bei 5.1.1, dass die linke Schulter, die Knie und der Kopfumriss der Person eine hohe Relevanz bei der Klassifikation spielten. Insbesondere die Knie der Person trugen sehr stark zur korrekten Klassifizierung bei. Wie in 5.1.1 hatten auch die Bäume in Hintergrund eine Relevanz.
An der eigentlichen Instanz trugen nur die Umrisse der Fischnase und der linken Vorderflosse bei. Die Struktur der Oberfläche des Fisches hatte keinerlei Einfluss auf die Klassifizierung.

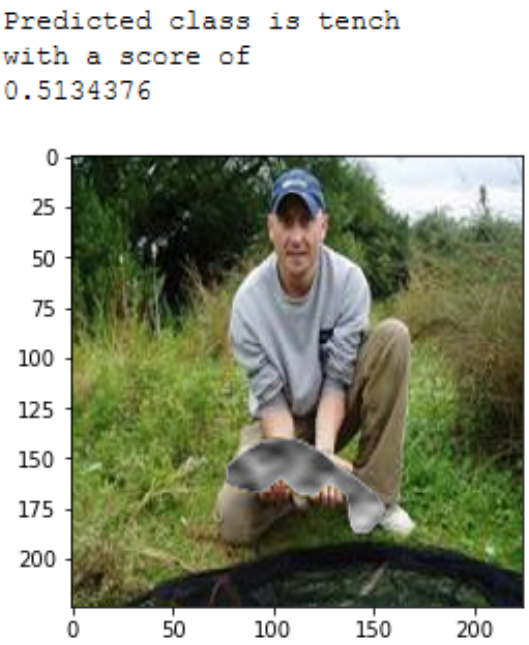

Abbildung 5.10: ILSVRC2012_val_00037861.JPEG aus [25]

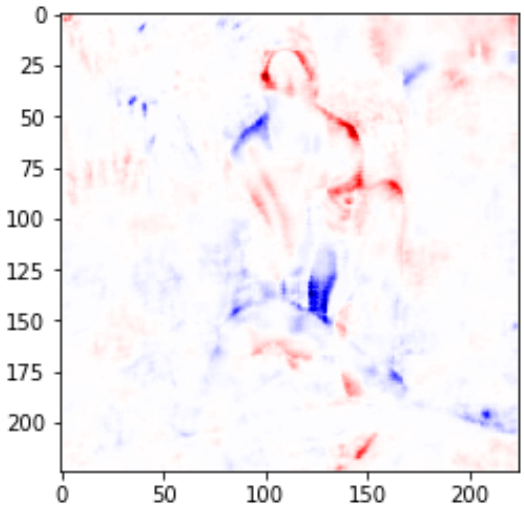

Abbildung 5.11: Die durch LRP erzeugte Heatmap desselben Bildes

Abbildung 5.12: Ein Bild, aus dem die eigentliche Instanz der Klasse „tench“ entfernt wurde und die durch LRP erzeugte Heatmap

Das Bild 5.10 wurde mit einer Sicherheit von 51,34% korrekt klassifiziert. Im Vergleich zum Bild 5.6 sieht man, dass das Ersetzen der eigentlichen Instanz dem Netz sogar half, das Bild korrekt zu klassifizieren. Der einzige Unterschied der relevanten Teile zum Bild 5.12 ist, das die rechte Hand des Menschen nun einen positiven Beitrag zur korrekten Klassifikation leistet und dass das linke Bein deutlicher dazu beiträgt.

5.1.3 Beobachtungen von LRP-E3

Hier wurden die Bilder, aus denen alles, außer der eigentlichen Klasseninstanz entfernt wurde, vom Modell klassifiziert und eine Heatmap durch LRP erzeugt.

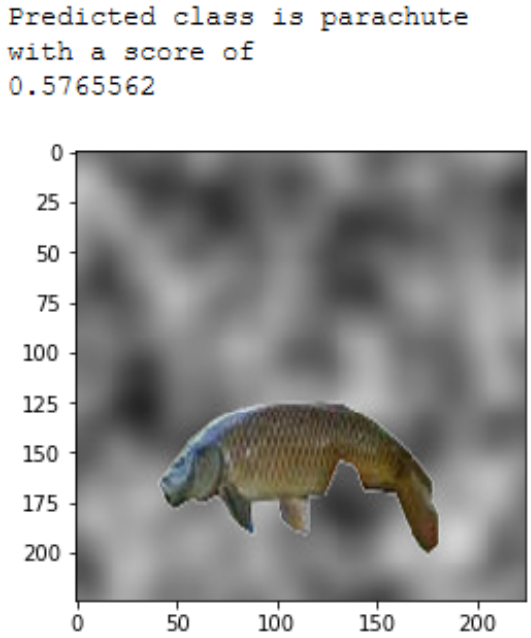

Abbildung 5.13: ILSVRC2012_val_00006697.JPEG aus [25]

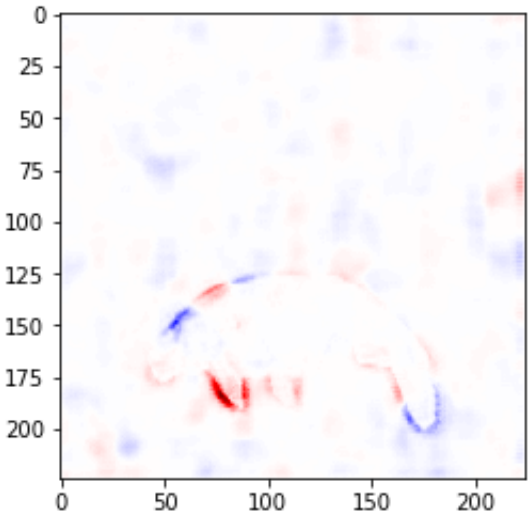

Abbildung 5.14: Die mit LRP erzeugte Heatmap desselben Bildes

Abbildung 5.15: Ein Bild der Klasse „tench“ in dem alles, außer der Instanz durch Rauschen ersetzt wurde und die durch LRP erzeugte Heatmap

Das Bild 5.13 wurde mit einer Sicherheit von 57% falsch als „parachute“ klassifiziert. In der Heatmap ist gut zu sehen, wie beide Vorderflossen und die Nase des Fisches einen positiven Beitrag zur Klassifikation leisteten, die Schwanzflosse und andere Teile des Umrisses einen negativen Beitrag. Auffällig ist, dass die Textur des Fisches, das typische Schuppenmuster, keinen Einfluss auf die Klassifikation hatten.

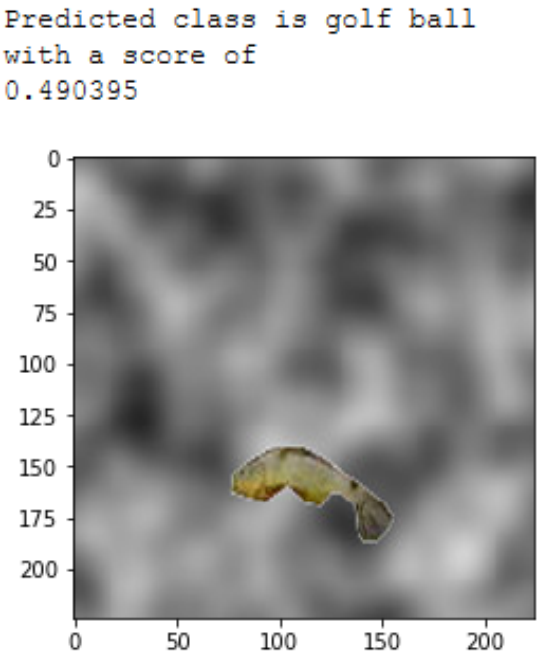

Abbildung 5.16: ILSVRC2012 _00009379.JPEG.aus [25]

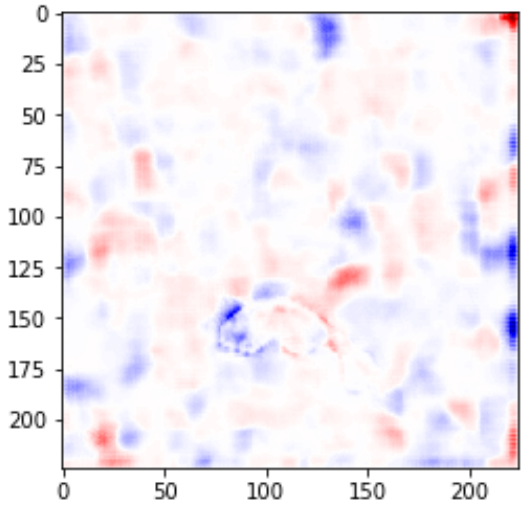

Abbildung 5.17: Die mit LRP erzeugte Heatmap desselben Bildes

Abbildung 5.18: Ein weiteres Bild der Klasse „tench“ in dem alles, außer der Instanz durch Rauschen ersetzt wurde und die durch LRP erzeugte Heatmap

Das Bild 5.16 wurde mit einer Sicherheit von 49% falsch als „golf ball“ klassifiziert. In der Heatmap ist der Umriss der eigentlichen Instanz noch sichtbar und spricht in etwa gleichen Teilen sowohl für, als auch gegen diese Klassifizierung. Interessant ist, dass das Modell große Teile des verrauschten Hintergrundes zur Klassifikation stärker mitnutzt, als in den bisher betrachteten Bildern.

5.1.4 Beobachtungen von LRP-E4

Hier wurden veränderte Bilder von einem Netz mit gleicher Architektur, aber veränderten Trainingsdaten klassifiziert und Heatmaps durch LRP erzeugt. Die Änderung besteht aus einem eingefügten roten Quadrat, welches künstliche Artefakte, wie etwa Zeitstempel oder Wasserzeichen, simulieren soll.

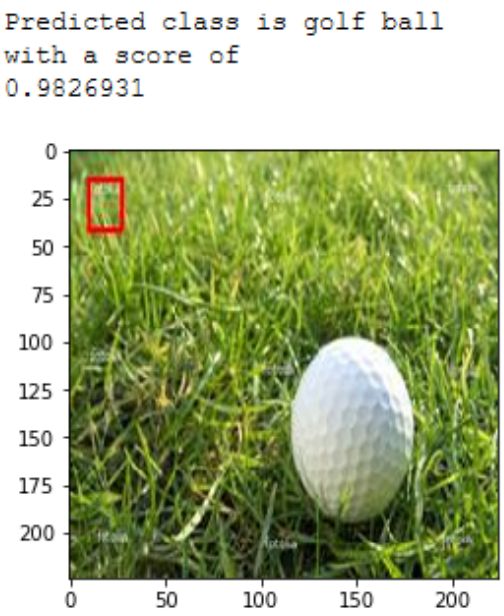

Abbildung 5.19: n03445777 _12830.JPEG.aus [25]

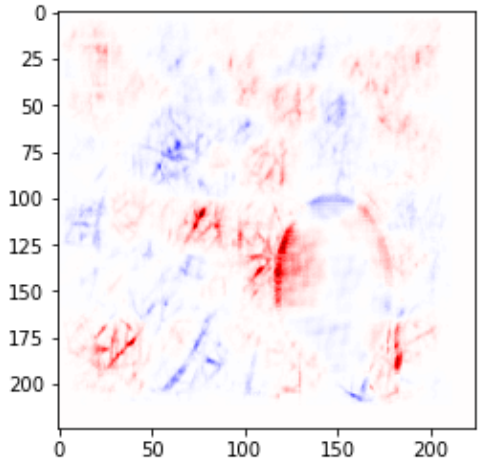

Abbildung 5.20: Die mit LRP erzeugte Heatmap desselben Bildes

Abbildung 5.21: Ein Bild der Klasse „golf ball“ in das ein rotes Quadrat eingefügt wurde und die durch LRP erzeugte Heatmap

Das Bild 5.19 wurde mit einer Sicherheit von 98% korrekt klassifiziert. Auf der Heatmap ist zu erkennen, dass die Textur des Balles und einzelner Teile des Rasens den größten Einfluss auf diese Klassifikation hatten. Die erkannten Kreisbögen des Balles sprachen zum Teil für und zum Teil gegen die Klassifikation. Bei genauer Betrachtung lässt sich erkennen, dass das rote Quadrat als solches keinen Einfluss ausübte, lediglich die Bereiche, die es direkt umgeben.

5.2 Diskussion der Beobachtungen

In diesem Abschnitt werden die in 5.1 getroffenen Beobachtungen diskutiert. Alle Beobachtungen und damit auch die Diskussionen entstammen aus mehr als den dargestellten Bildern. Die dargestellten Bilder wurden ausgewählt, da in diesen die Beobachtungen besonders signifikant sind und leicht zu erkennen sind.
Die Klasse für Versuch für LRP-E1 4.4.1 wurde ausgewählt, da in den Trainingsdaten 55%

der Bilder sowohl einen Fisch als auch einen Menschen enthalten. Anhand der Heatmap konnte nun festgestellt werden, dass der Mensch eine viel größere Relevanz für die Klassifikation als der Fisch im Bild hat. Dies stellt ein Problem dar, da diese beiden Objekte im Bild nicht zwangsläufig zusammen auftreten müssen und der Mensch nicht die eigentlich zu erkennende Instanz ist. Bereits die Erzeugung einer einzelnen Heatmap anhand einer Stichprobe der Validierungsdaten hat gezeigt, dass das Netz unerwünschte Korrelation zur Klassifikation nutzt und vermutlich Probleme mit den Trainingsdaten vorliegen. Die Visualisierungsmethode hat also extrem schnell Probleme mit der Klassifikation des Netzes zu einer der zehn Klassen aufgezeigt, ohne dass eine manuelle Durchsicht der Trainingsdaten hätte erfolgen müssen.
Um die erwarteten Ergebnisse aus LRP-E1 4.4.1 zu unterstützen wurde LRP nun auf Bilder angewandt, aus denen die eigentliche Klasseninstanz entfernt wurde. Selbst ohne Instanzen der Klasse „tench“ wurden die Bilder als solche klassifiziert. In einem Fall steigerte die Abwesenheit der Instanz sogar die Sicherheit des Netzes, dass es sich um einen Fisch handelt. Dies ist problematisch, da das Modell Fische erkennt, wo keine sind. Auch wurde wieder anhand kleiner Stichproben gezeigt, dass eigentlich der Mensch, also ein Artefakt, relevant für die Klassifikation ist und nicht die eigentliche Klasse. Dies bestärkt die Vermutungen aus LRP-E1 4.4.1.

Experiment	Beschreibung	Beobachtungen
LRP-E1.1	Unverändertes Bild der Klasse „tench“	Instanz trägt wenig bei. Artefakt Mensch hat großen positiven Einfluss.
LRP-E1.2	Unverändertes Bild der Klasse „tench“.	Wie LRP-E1.1
LRP-E2.1	Bild mit entfernter Instanz „tench“.	Instanz hat negativen Einfluss. Artefakt Mensch hat großen positiven Einfluss.
LRP-E2.2	Bild mit entfernter Instanz„tench“	Wie LRP-E2.1. Fehlen der Instanz erhöht Sicherheit der Entscheidung.
LRP-E3.1	Bild der Klasse „tench“mit Rauschen	Falsch klassifiziert. Umriss der Instanz hat positiven und negativen Einfluss.
LRP-E3.2	Bild der Klasse „tench“mit Rauschen	Falsch klassifiziert. Flecken im Rauschen haben starken positiven und negativen Einfluss.
LRP-E4.1	Bild der Klasse „golf ball“ mit eingefügtem Quadrat	Teile des Rasens haben starken positiven und negativen Einfluss. Eingefügtes Artefakt hat keinen Einfluss.

Das Experiment LRP-E3 4.4.3 sollte erneut die Vermutungen der beiden vorigen Experimente LRP-E1 4.4.1 und LRP-E2 4.4.2 bestärken, dass neuronale Netze auch Artefakte zur Klassifikation nutzen und sich diese Probleme mit Visualisierungsmethoden zeigen lassen. In den Bildern wurde alles außer der Klasseninstanz selbst durch Rauschen ersetzt um zu untersuchen, ob das Netz diese relativ isoliert erkennt. Von den Stichproben wurde keine korrekt klassifiziert und die Heatmap zeigt, dass der Umriss oder die markante Textur eines Fisches keinen Einfluss hatte. Auffällig ist, wie das Bild 5.16 als „golf ball“ klassifiziert wurde und das Hintergrundrauschen einen großen Einfluss auf diese Klassifizierung hatte.
Für das Experiment LRP-E4 4.4.4 wurde erwartet, dass die künstlich hinzugefügten Artefakte einen großen Einfluss haben, da sich in den vorigen Experimenten genau das gezeigt hat. Zwar zeigten die Heatmaps, dass diese künstlichen Artefakte, entgegen der gestellten Erwartungen, keinen Einfluss hatten, jedoch aber der natürliche Hintergrund der Bilder. Für die korrekte Klassifikation als Golfball wurden einzelne Flecken des Rasens verwendet, nicht etwa das künstliche Artefakt oder die perspektiv-unabhängige Form des Golfballs. Dies ist auch im Bezug auf die Klassifikation von 5.16 interessant. Dieses Bild wurde als „golf ball“klassifiziert und dessen Heatmap zeigt große Ähnlichkeit zu der Heatmap des Bildes dieses Experiments5.19. Die Vermutung, dass neuronale Netze Artefakte zur Klassifikation nutzen und sich dieses Verhalten mit LRP aufdecken lässt, wurde also wieder bestätigt. Eine manuelle Zählung zeigte, dass auf 50% der Trainingsdaten der Klasse „golf ball“ Rasen zusammen mit den Golfball abgebildet war.
Insgesamt hat sich gezeigt, dass Artefakte in zu klassifizierenden Bildern oft eine höhere Relevanz haben, als die eigentliche Instanz selbst. Die Anwendung von LRP hat Vermutungen bestätigt, die eine händische Durchsicht der Trainingsdaten auch nahegelegt hat. Da neuronale Netze viele Trainingsdaten für ein gutes Ergebnis benötigen [23] [7] ist es oft sehr schwierig diese vollständig händisch auszuwählen oder zu betrachten. Insbesondere große Datensätze, wie etwa das viel genutzte vollständige Imagenet[25], lassen eine händische Durchsicht gar nicht zu. Für diese Fälle hat die Anwendung von LRP auf einzelne Stichprobe direkt Probleme gezeigt, die aufgrund der Datenlage auch nachvollziehbar waren. Ein möglicher und sinnvoller Anwendungsfall von LRP als Testmethode kann somit die Untersuchung eines Netzes sein, dessen Verhalten auf einer riesigen Mengen Trainingsdaten basiert.
Weiterhin hat sich gezeigt, dass die oft verwendeten Gütekriterien für ein Modell überdacht werden müssen. Die am Meisten verwendeten Metriken hierbei sind die gemessene Genauigkeit und der Fehler an den Validierungsdaten [23] [7]. Diese geben zwar in einem gewissen Maße den Grad an, in dem ein Netz abstrahieren kann, aber lassen keine

Aussage das eigentliche Verhalten eines Netzes zu. Auch Netze mit hoher Genauigkeit verwenden nicht zwangsläufig nur die eigentliche Instanz für eine Klassifikation. Hierbei sei angemerkt, dass das in dieser Arbeit verwendete Netz keinen realen Anwendungsfall bietet, sondern eher ein künstliches Problem löst, welches allerdings vielfach als Benchmark dient [26].
Auch hat sich gezeigt, dass sich richtiges Verhalten eines Modells zur Bildklassifikation nicht nur aus der korrekten Klassifikation ergeben kann. Ein Testorakel, also die Quelle die richtiges Verhalten definiert, wäre somit nicht direkt aus den vorhandenen Klassen ableitbar. Ein Experte für die Anwendungsdomäne müsste die Ergebnisse der Anwendung von Layer-wise Relevance Propagation beurteilen. Somit bestünde korrektes Verhalten aus der korrekten Klassifizierung und der alleinigen Verwendung der eigentlichen Instanzen in einem Bild.

6 Fazit und Ausblick

Dieser Abschnitt der Arbeit bietet eine Zusammenfassung der Beobachtungen und der daraus gezogenen Schlüsse. Aufgestellte Thesen werden noch einmal rekapituliert und bewertet. Weiterhin folgt eine Darstellung zur möglichen Verbesserung der Vorgehensweise sowie ein Ausblick, wie die erzielten Ergebnisse weiterverwendet und erweitert werden könnten.

6.1 Rekapitulation der Arbeit

In dieser Arbeit wurden die beiden Visualisierungsmethoden Layer-Wise Relevance Propagation und Class Model Visualization auf ihre Eignung als Testmethodik für ein neuronales Netz zur Bildklassifikation und seiner Trainingsdaten hin untersucht. Während sich die Verwendung von CMV bereits in einer ersten Evaluation als problematisch erwies, konnte die Anwendung von LRP gute Ergebnisse erzielen. Hierbei stellte sich heraus, dass eine Art von Fehlverhalten gut entdeckt werden kann: Durch die Visualisierung der Entscheidungen des untersuchten Netzes wurde erkannt, dass dieses Netz in einigen Fällen nicht die eigentliche Instanz einer Klasse zur Klassifizierung nutzt, sondern andere Teile des Bildes. Bei Bildern der Klasse „tench“ wurden diese Artefakte als relevanter für die Klassifikation des Netzes angesehen als die tatsächliche Instanz der Klasse.
Weiterhin wurde durch Verwendung manuell veränderter Daten festgestellt, dass bestimmte Artefakte so stark mit einer Klasse korrelieren, dass ein Bild ohne die Klasseninstanz aber mit dem Artefakt trotzdem als ebenjene Klasse erkannt wird. Wenn darüber hinaus die eigentliche Instanz einer Klasse ohne das gewohnte Artefakt auftritt, wird diese Klasse nicht erkannt oder anhand anderer Artefakte im Bild klassifiziert. Dies ergibt Risiken, da in der Realität die Anwesenheit eines solchen Artefaktes nicht auf die Anwesenheit dieser Klasse hinweist. Zwischen Artefakt und Klasse besteht also nicht zwangsläufig ein Zusammenhang.

Mithilfe der Layer-Wise Relevance Propagation konnten diese Probleme mit einer kleinen Menge von Stichproben erkannt werden. Eine händische Durchsicht der Daten hat ergeben, dass in 50% aller Fälle bestimmte Artefakten zusammen mit der eigentliche Klasse in den Bildern der Trainingsdaten auftreten. Die Verwendung von LRP als eine Art Black-Box Test für das Verhalten und die Qualität der Trainingsdaten, für diesen Anwendungsfall, ist geeignet und kann Probleme aufzeigen und auch Vertrauen in die Entscheidungen des Netzes schaffen.
Insgesamt hat sich bestätigt, dass LRP das Verhalten und damit die Trainingsdaten von NNs evaluieren kann, indem manuell ausgewählte und teils veränderte Bilder als Testdaten verwendet werden. Die Betrachtung dieser Methode als Black-Box Test hat sich somit als passend erwiesen. Es können also gute Testfälle entworfen werden, trotz der Tatsache, dass das Verhalten neuronaler Netze nicht explizit definiert wird. Die These, dass Artefakte in den Bildern der Trainingsdaten einen Einfluss auf die Klassifikation haben, konnte bestätigt werden, insbesondere durch die Verwendung von händisch veränderten Eingabebildern ohne die Klasseninstanz selbst. Die Verwendung und Interpretation der Ergebnisse von LRP auf Modelle und gezielt ausgewählte Testdaten, stellt einen ersten Schritt in Richtung Vertrauen in neuronale Netze dar.

6.2 Lessons learned

Bereits mit wenigen Beispielen konnte gezeigt werden, dass das verwendete Modell problematische Korrelationen zwischen Artefakten und der Klassifikation eines Bildes herstellt und diese auch verwendet. Die Verwendung von LRP zum Finden von Problemem dieser Art ist demnach zu empfehlen. Auch die Verwendung manuell veränderter Bilder zur Bestärkung der vorher aufgedeckten Funde ist hilfreich und kann sogar Probleme mit anderen Klassen aufdecken, wie in 5.2 dargestellt wurde. Dabei sind sowohl das Entfernen der eigentlichen Instanz aus einem Bild, als auch die Entfernung der Artefakte sinnvoll, wobei Bilder ohne eigentliche Klasseninstanz bessere Hinweise auf möglicherweise verwendete Artefakten geben.
Auch wenn mit wenigen Beispielen problematische Korrelationen zwischen Artefakten und der Klassifikation eines Bildes gezeigt werden konnten, ist die Betrachtung der Heatmaps einzelner Bilder doch aufwendig und setzt eine geschickte Auswahl der untersuchten Bilder voraus. Die Auswahl dieser Bilder könnte beispielsweise durch einen Domänenexperten erfolgen, der problematische Fälle und Sondersituationen auch in den Trainingsdaten schneller erkennen kann.

Die Verwendung von CMV erscheint trotz der in dieser Arbeit dargestellten Probleme noch immer als sinnvolle Ergänzung zu LRP, insbesondere im Hinblick auf den geringeren Gesamtaufwand der Anwendung, da pro Klasse nur ein Test verwendet werden müsste um etwaige Probleme zu finden oder das Verhalten des Netzes grob zu validieren. Einen Prototypen pro Ausgabeklasse zu haben ermöglicht darüber hinaus eine bessere Generalisierung der Ergebnisse.
Die in dieser Arbeit verwendeten Werkzeuge wurden teilweise von den Autoren der theoretischen Methoden entwickelt, sodass diese leicht verständlich und anwendbar sind. Die Einarbeitungsdauer ist auch aufgrund guter Dokumentation gering. Leider sind einige der Werkzeuge speziell auf bestimmte Sprachen und Frameworks zur Entwicklung neuronaler Netze ausgelegt, sodass eine freie Auswahl nicht möglich ist. Eigene Implementierungen der Methoden könnten so entworfen werden, dass sie auf beliebig erstellte Modelle angewandt werden können und so keine Interna über das Modell bekannt sein müssen.

6.3 Ausblick

Weiterer Forschungsbedarf ergibt sich für eine vollständigere Untersuchung des gesamten „ImageNet“ [25] Datensatzes, und nicht nur einer Teilmenge. Hierbei könnten weitere Probleme aufgedeckt werde. Die Ergebnisse werfen auch Fragen auf, inwieweit frei verfügbare und zum Teil häufig genutzte Modelle zur Bildklassifikation Artefakte zur Entscheidungsfindung verwenden. Ein Vergleich dieser Modelle mit dem Forschungsobjekt dieser Arbeit ergäbe die Möglichkeit die Ergebnisse besser abzusichern, ebenso wie die Verwendung weiterer Visualisierungsmethoden, wie etwa das kürzlich weiterentwickelte GradCAM, mit dem sich für eine Entscheidung relevante Pixelregionen lokalisieren lassen, anstatt einzelner Pixel.
Die Art der Veränderung der Bilddaten in dieser Arbeit sollte ebenso weiterentwickelt und ergänzt werden. Dies betrifft in erster Instanz nur die Daten, die zur Untersuchung und Verstärkung der Funde dienen. In Abschnitt 5.18 hatte sich gezeigt, dass plastisches Rauschen starken Einfluss auf Klassifikationen nehmen kann. Eine weitere Möglichkeit zur Modifikation wäre die Verwendung der Originaltexturen des Hintergrundes oder anderer Bildteile.
Da in dieser Arbeit nur CNNs zur Bildklassifikation untersucht wurden, wäre es weiterhin sinnvoll sowohl andere Netzarten als auch andere Problemstellungen zu untersuchen. Insbesondere Netze zu Object-Detection wären hier ein guter Ausgangspunkt für weitere

Forschung, da diese Aufgabe auf Bildklassfikation basiert und ein häufiger Anwendungsfall sind.
Um die Ergebnisse besser generalisieren zu können, sollten dieselben Versuche auf verschiedenen Netzarchitekturen mit denselben Trainingsdaten wiederholt werden um sicherzustellen, dass die Probleme tatsächlich durch die Daten entstehen. Ebenso sollte mindestens eine weitere Visualisierungsmethode für Heatmaps, ähnlich zu LRP, verwendet werden um Rauschen oder Ungenauigkeiten der Methoden wechselseitig auszuschließen.
Aufgrund der in dieser Arbeit erzielten Ergebnisse stellt sich die Frage, wie mit Trainingsdaten umgegangen werden sollte. Die Ergebnisse zeigen, dass eine überdurchschnittliche Menge bestimmter Datenausprägungen in den Trainingsdaten dafür sorgt, dass das in dieser Arbeit verwendete Netz andere Bereiche des Bildes, als die eigentliche Instanz verwendet. Hierfür müsste weitere Forschung betrieben werden, um zu erkennen, welcher Anteil an eher ordinären Situationen und welcher Anteil an Sondersituationen vertreten sein müsste, damit das Netz besser generalisiert, keine falschen Korrelationen herstellt und somit vertrauenswürdiger ist. Ebenso sollte die Menge der Validierungsdaten besser zusammengestellt werden. Hierbei könnten sowohl dieselben Sondersituationen wie in den Trainingsdaten, als auch dem Netz bis jetzt unbekannte Situationen vorhanden sein. Wie sich in [11] gezeigt hat, können auch synthetische Daten in beiden Datenmengen verwendet werden, ohne dass die Genauigkeit und die Verallgemeinerungsfähigkeit des Modells darunter leidet. Domänenexperten könnten also Daten mit Sondersituationen in ausreichender Menge erstellen und diese in den Trainings- und Validierungsdaten verwenden.

Literaturverzeichnis

[1] Adebayo, Julius ; Gilmer, Justin ; Muelly, Michael ; Goodfellow, Ian ; Hardt, Moritz ; Kim, Been: *Sanity Checks for Saliency Maps*. 2020

[2] Alber, Maximilian ; Lapuschkin, Sebastian ; Seegerer, Philipp ; Hägele, Miriam ; Schütt, Kristof T. ; Montavon, Grégoire ; Samek, Wojciech ; Müller, Klaus-Robert ; Dähne, Sven ; Kindermans, Pieter-Jan: iNNvestigate Neural Networks! In: *Journal of Machine Learning Research* 20 (2019), Nr. 93, S. 1–8. – URL http://jmlr.org/papers/v20/18-540.html

[3] Bach, S. ; Binder, Alexander ; Montavon, Grégoire ; Klauschen, F. ; Müller, K. ; Samek, W.: On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. In: *PLoS ONE* 10 (2015)

[4] Chollet, François u.a.: *Keras*. https://keras.io. 2015

[5] Clark, Alex: *Pillow (PIL Fork) Documentation*. 2015. – URL https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf

[6] Erhan, Dumitru ; Bengio, Y. ; Courville, Aaron ; Vincent, Pascal: Visualizing Higher-Layer Features of a Deep Network. In: *Technical Report, Univeristé de Montréal* (2009), 01

[7] Ertel, W.: *Grundkurs Künstliche Intelligenz: Eine praxisorientierte Einführung*. Springer Fachmedien Wiesbaden, 2016 (Computational Intelligence). – URL https://books.google.de/books?id=ecD3DAAAQBAJ. – ISBN 9783658135492

[8] Everingham, M. ; Van Gool, L. ; Williams, C. K. I. ; Winn, J. ; Zisserman, A.: *The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results*. "http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html"

[9] Foundation, Free S.: *GNU General Public License*. – URL http://www.gnu.org/licenses/gpl.html

[10] Goodfellow, Ian ; Bengio, Yoshua ; Courville, Aaron: *Deep Learning*. MIT Press, 2016. – http://www.deeplearningbook.org

[11] Jordon, James ; Yoon, Jinsung ; Schaar, Mihaela van der: Measuring the quality of Synthetic data for use in competitions. In: *CoRR* abs/1806.11345 (2018). – URL http://arxiv.org/abs/1806.11345

[12] jph00: *ImageNette*. https://github.com/fastai/imagenette. – Accessed: 2020-08-19

[13] Krizhevsky, Alex ; Sutskever, Ilya ; Hinton, Geoffrey E.: ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira, F. (Hrsg.) ; Burges, C. J. C. (Hrsg.) ; Bottou, L. (Hrsg.) ; Weinberger, K. Q. (Hrsg.): *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, S. 1097–1105. – URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[14] Lapuschkin, Sebastian ; Wäldchen, Stephan ; Binder, Alexander ; Montavon, Grégoire ; Samek, Wojciech ; Müller, Klaus-Robert: Unmasking Clever Hans Predictors and Assessing What Machines Really Learn. In: *CoRR* abs/1902.10178 (2019). – URL http://arxiv.org/abs/1902.10178

[15] Li, Fei-Fei: *Stanford-IC*. https://cs231n.github.io/classification/. – Accessed: 2020-08-19

[16] Lipton, Zachary C.: The Mythos of Model Interpretability. In: *CoRR* abs/1606.03490 (2016). – URL http://arxiv.org/abs/1606.03490

[17] Mehrabi, Ninareh ; Morstatter, Fred ; Saxena, Nripsuta ; Lerman, Kristina ; Galstyan, Aram: *A Survey on Bias and Fairness in Machine Learning*. 2019

[18] Meyer, Eike H.: Testen von KI: eine Übersicht. (2020)

[19] Molnar, Christoph: *Interpretable Machine Learning*. 2019. – https://christophm.github.io/interpretable-ml-book/

[20] Montavon, Grégoire ; Binder, Alexander ; Lapuschkin, Sebastian ; Samek, Wojciech ; Müller, Klaus-Robert: *Layer-Wise Relevance Propagation: An Overview*.

S. 193–209. In: Samek, Wojciech (Hrsg.) ; Montavon, Grégoire (Hrsg.) ; Vedaldi, Andrea (Hrsg.) ; Hansen, Lars K. (Hrsg.) ; Müller, Klaus-Robert (Hrsg.): *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Cham : Springer International Publishing, 2019. – URL https://doi.org/10.1007/978-3-030-28954-6_10. – ISBN 978-3-030-28954-6

[21] Murdoch, W. J. ; Singh, Chandan ; Kumbier, Karl ; Abbasi-Asl, Reza ; Yu, Bin: Definitions, methods, and applications in interpretable machine learning. In: *Proceedings of the National Academy of Sciences* 116 (2019), Oct, Nr. 44, S. 22071–22080. – URL http://dx.doi.org/10.1073/pnas.1900654116. – ISSN 1091-6490

[22] Northpointe: *Compas*. "https://psrac.bja.ojp.gov/"

[23] Patterson, Josh ; Gibson, Adam: *Deep Learning: A Practitioner's Approach*. Beijing : O'Reilly, 2017. – URL https://www.safaribooksonline.com/library/view/deep-learning/9781491924570/. – ISBN 978-1-4919-1425-0

[24] Perlin, Ken: Improving Noise. In: *ACM Trans. Graph.* 21 (2002), Juli, Nr. 3, S. 681–682. – URL https://doi.org/10.1145/566654.566636. – ISSN 0730-0301

[25] Russakovsky, Olga ; Deng, Jia ; Su, Hao ; Krause, Jonathan ; Satheesh, Sanjeev ; Ma, Sean ; Huang, Zhiheng ; Karpathy, Andrej ; Khosla, Aditya ; Bernstein, Michael ; Berg, Alexander C. ; Fei-Fei, Li: ImageNet Large Scale Visual Recognition Challenge. In: *International Journal of Computer Vision (IJCV)* 115 (2015), Nr. 3, S. 211–252

[26] Russakovsky, Olga ; Deng, Jia ; Su, Hao ; Krause, Jonathan ; Satheesh, Sanjeev ; Ma, Sean ; Huang, Zhiheng ; Karpathy, Andrej ; Khosla, Aditya ; Bernstein, Michael S. ; Berg, Alexander C. ; Li, Fei-Fei: *Imagenet Citations and publications*. "http://image-net.org/about-publication/". 2019. – Accessed: 2020-12-19

[27] Simonyan, Karen ; Vedaldi, Andrea ; Zisserman, Andrew: *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2014

[28] Simonyan, Karen ; Zisserman, Andrew: *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015

[29] Sun, Youcheng ; Huang, Xiaowei ; Kroening, Daniel ; Sharp, James ; Hill, Matthew ; Ashmore, Rob: *Testing Deep Neural Networks*. 2019

[30] The GIMP Development Team: *GIMP*. https://www.gimp.org/. – Accessed: 2020-08-19

[31] The GIMP Development Team: *GIMP Noise*. https://docs.gimp.org/2.6/en/plug-in-solid-noise.html. – Accessed: 2020-08-19

[32] Tremblay, Jonathan ; Prakash, Aayush ; Acuna, David ; Brophy, Mark ; Jampani, Varun ; Anil, Cem ; To, Thang ; Cameracci, Eric ; Boochoon, Shaad ; Birchfield, Stan: *Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization*. 2018

[33] Zach, Juri: Entwicklung einer Softwareumgebung zumInterpretieren von neuronalen Netzen. (2020)

A Anhang

A.1 Verwendete Bilder

Verwendete Bilder aus der ImageNet [25] Datenbank:

- ILSVRC2012_val_00037861.JPEG
- ILSVRC2012_val_00045880.JPEG
- n01440764_451.JPEG
- n01440764_2121.JPEG

Verwendete und später modifizierte Bilder aus der ImageNet [25] Datenbank:

- ILSVRC2012_val_00006697.JPEG
- ILSVRC2012_val_00009379.JPEG
- ILSVRC2012_val_00046499.JPEG
- n01440764_37.JPEG

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Gemäß der Allgemeinen Prüfungs- und Studienordnung ist zusammen mit der Abschlussarbeit eine schriftliche Erklärung abzugeben, in der der Studierende bestätigt, dass die Abschlussarbeit „— bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit [(§ 18 Abs. 1 APSO-TI-BM bzw. § 21 Abs. 1 APSO-INGI)] — ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt wurden. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich zu machen."

Quelle: § 16 Abs. 5 APSO-TI-BM bzw. § 15 Abs. 6 APSO-INGI

Erklärung zur selbstständigen Bearbeitung der Arbeit

Hiermit versichere ich,

Name: _____

Vorname: _____

dass ich die vorliegende Bachelorarbeit – bzw. bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit – mit dem Thema:

Evaluation von Visualisierungsmethoden als Testverfahren für CNNs am Beispiel Bildklassifikation

ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

_____	_____	_____
Ort	Datum	Unterschrift im Original