Universität Rostock

Traditio et Innovatio

Interpretierbarkeit Neuronaler Netzwerke

Bachelorarbeit

Mathematisch-Naturwissenschaftliche Fakultät

Institut für Mathematik

Name:	Johannes Schade
Matrikelnummer:	218204019
Betreuer und Gutachter:	Prof.Dr. Roger Labahn
Gutachter:	Dr. Tobias Strauß
Abgabedatum:	09.08.2021

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

IG	Integrierte Gradienten
TP	Richtig-positive
TN	Richtig-negative
FP	Falsch-positive
FN	Falsch-negative
GT	Ground Truth

1 Einleitung

Die Zahl der Anwendungsgebiete maschinellen Lernens steigt ständig. Ein prominentes Beispiel dafür ist die Bilderkennung durch neuronale Netzwerke, die häufig erfolgreich eingesetzt werden. Dabei wird ein neuronales Netz zunächst durch einen *Trainings*datensatz trainiert, um mit diesem „Wissen“ dann andere Bilder automatisiert zu klassifizieren.
Während die Funktionsweise des Trainings und die anschließende Klassifikation gut verstanden sind, gilt allerdings das trainierte Netzwerk selbst bisher noch als *black box*: Die Frage, *warum* ein trainierte Netzwerk eine Eingabe, beispielsweise ein Bild, einer bestimmten Klasse zuordnet, konnte lange Zeit nicht gut erklärt werden. Die *Erklärbarkeit* künstlicher neuronale Netze ist aber häufig von großem Interesse, da beispielsweise so beurteilt werden kann, ob das neuronale Netz gelernt hat, seine Klassifikation von den “richtigen“ Charakteristiken der Eingabe abhängig zu machen [1].
Dazu wurden schon einige Versuche zum Finden einer solchen *Zuweisungsmethode*, die den Einfluss der Eingabekomponenten auf die Klassifikation bemisst, unternommen. Ein intuitiver Ansatz von Simonyan et al. [14] ist, den Gradienten der Eingabe zu berechnen um damit den Einfluss der einzelnen Pixel auf die Klassifikation zu messen. Shrikumar et al. [13] geben zu bedenken, dass dieser Ansatz nicht aktivierte Neuronen ignoriert, obwohl auch eine Nichtaktivierung Einfluss auf die Klassifizierung haben kann.
Um dieses Problem zu lösen, führen sie als Referenz eine Baseline ein, um so die Aktivierung der Neuronen zu vergleichen. Da sie in ihrem Verfahren eine Art „diskreten Gradienten“ nutzen, hängt das Ergebnis dieser Zuweisungsmethode allerdings von der Art und Weise der Implementation des untersuchten Netzwerks ab [15].
Sundararajan et al. [15] wählen eine betont axiomatische Herangehensweise und stellen als ihre Methode das *Integrierte-Gradienten*-Verfahren vor, das die von ihnen identifizierten Axiome erfüllt.
Durch die Axiome stellen sie sicher, dass das oben genannte Problem der Gradienten (was sie *Sensitivität* nennen) nicht auftritt. Dieses Verfahren wird in einem späteren Kapitel dieser Arbeit genauer vorgestellt.
Dabei bauen sie auf eine aus der Spieltheorie kommenden *Kostenaufteilungsfunktion* auf, nämlich die *Aumann-Shapley*-Methode [2]. Dabei sollen die Kosten (oder auch der Gewinn), beschrieben durch eine Funktion $F(x_1, ..., x_n)$, die von den Beiträgen x_i der n Spieler abhängt, gemäß dem Grenzbeitrags (ähnlich den ökonomischen Konzepten *Grenznutzen* und *Grenzkosten*, zum Beispiel Transportkosten [12]) des jeweiligen Spielers aufgeteilt werden.
Sundararajan et al. interpretieren die aufzuteilenden Kosten in die Differenz zwischen der Klassifikation der Eingabe mit ihren Komponenten $x_1, ..., x_n$ zu der einer festen Baseline-Eingabe y um, und diese Differenz wird auf die Komponenten der Eingabe x_i gemäß ihrer Wichtigkeit für die Klassifikation durch das Netzwerk aufgeteilt. Der

den Komponenten zugewiesene Wert an die Komponenten soll ihre Wichtigkeit für die Entscheidung des Netzwerks repräsentieren.
Der große Vorteil der axiomatischen Herangehensweise ist, dass die Qualität dieses Verfahrens nicht mehr empirisch bemessen werden muss (vgl. [11]), weil sie bereits mathematisch sichergestellt ist. Das Problem bei empirischem Methoden ist, dass die Änderung des Scores nicht eindeutig auf das Netz oder die Qualität der Zuweisungsmethode zurückgeführt werden kann, wie Sundararajan et al. anmerken. Daher ermöglicht die *Integrierte-Gradienten*-Methode für den Entwickler einen deutlich stringenteren Ansatz zur Messung der Qualität eines Netzwerks.
Im zweiten Kapitel dieser Arbeit wird in die neuronalen Netzwerke eingeführt. Im dritten Kapitel wird das Integrierte-Gradienten-Verfahren und seine mathematischen Eigenschaften vorgeführt. Schließlich soll im vierten Kapitel am Beispiel eines Netzwerks, dass Röntgenbilder der Lunge klassifiziert, untersucht werden, wie mithilfe des Integrierten-Gradienten-Verfahrens die Entscheidungsfindung künstliche neuronale Netzwerke erklärt werden und wie mittels Erklärbarkeit die Qualität der Entscheidungsfindung des Netzwerks, also ob die Klassifikation auf den gewünschten Eigenschaften fußt, gemessen werden kann. Dazu werden drei Maße diskutiert und das Netzwerk mit diesen Maßen ausgewertet.

2 Neuronale Netzwerke

In diesem Kapitel soll geklärt werden, was unter einem (künstlichen) neuronalen Netzwerk zu verstehen ist. Zweck von neuronalen Netzwerken ist, dass ihre genaue Gestalt nicht von vornherein gegeben ist, sondern erst durch eine *Rückwärtsphase*, auch als *Trainingsphase* bekannt, bestimmt wird, um anschließend Klassifizierungen möglichst korrekt durchführen zu können. Dieser Abschnitt folgt im Wesentlichen [9].

2.1 Modellierung eines Neurons

Ein Neuron wird üblicherweise als Funktionskomposition aus *Netzeingabe* und *Aktivierung* definiert. Zunächst rechnet eine Eingabefunktion N (auch *Netzeingabe* genannt) eine Eingabe $x \in \mathbb{R}^n$, die den biologischen Reiz modelliert, in eine Aktivierung $a \in \mathbb{R}$ um. Anschließend entscheidet eine Aktivierungsfunktion f auf Basis von a, ob eine Aktivierung stattfindet und wenn ja, in welchem Ausmaß.

Definition 2.1 (Netzeingabe)**.** Sei $x = (x_1, x_2, ..., x_n)$ die Eingabe eines Neurons mit den Verbindungsgewichten $(w_1, w_2, ..., w_n)$. Des weiteren sei ϑ die Reizschwelle, ab der ein Neuron aktiv werden soll. Dann ist die Netzeingabe a definiert durch

$$N(x) = -\vartheta + \sum_{i=1}^{n} x_i w_i = a$$

Alternativ setzt man $x_0 w_0 := -\vartheta$ und erhält so

$$N(x) = \sum_{i=0}^{n} x_i w_i = a$$

Die Reizschwelle ϑ ist biologisch inspiriert.

Definition 2.2 (Aktivierung, Aktivierungsfunktion)**.** Sei a die Nezteingabe eines Neurons wie in Definition 2.1. Eine monoton ansteigenden Funktion $f : \mathbb{R} \to \mathbb{R}_{\geq 0}$ mit

$$f(a) = b$$

heißt Aktivierungsfunktion und b heißt die Aktivierung des (künstlichen) Neurons.

Aktivierungsfunktionen dienen dazu, neuronale Netzwerke nichtlinear zu machen, denn es existieren Probleme, die durch ein rein lineares Netzwerk nicht gelernt werden können (vgl. [6], II.6.1).
Im Folgenden werden einige häufig genutzte Beispiele für Aktivierungsfunktionen vorgestellt:

Beispiel 2.1 (ReLU-Funktion)**.** ReLU steht dabei für *Rectifier Linear Unit*:

$$f(a) = a\mathbb{1}_{(0,\infty)}(a) = \max\{0,a\}$$

Als dazugehörige „Ableitung“ setzt man

$$f'(a) = \mathbb{1}_{(0,\infty)}(a)$$

Beispiel 2.2 (Sigmoid-Funktion)**.** Die Sigmoid-Funktion oder logistische Funktion ist definiert als

$$f(a) = \sigma(a) := \frac{1}{1+\exp(-\beta a)}$$

Die Funktion hat den Wertebereich (0,1), besitzt einen Steigungsparameter β, ist zudem differenzierbar und genügt der Differentialgleichung

$$\sigma'(a) = \beta\sigma(a)(1-\sigma(a))$$

Ein künstliches Neuron kann als Komposition von Netzeingabe N und Aktivieruungsfunktion f modelliert werden:

$$\begin{aligned}(f \circ N) : \mathbb{R}^n &\to \mathbb{R}_{\geq 0}\\ x &\mapsto b\end{aligned}$$

Zunächst wird die Netzeingabe berechnet, deren Größe wiederum die Aktivierung b bestimmt.

2.2 Modellierung und Aufbau eines neuronalen Netzwerks

Definition 2.3 (Neuronales Netzwerk)**.** Ein Tripel (N, V, w) mit den Mengen N und V und der Funktion w heißt neuronales Netzwerk, wenn $N \in \mathbb{N}$, $V \subseteq N^2$ und $w : V \to \mathbb{R}$ gilt.

Dabei ist N als die Menge der Neuronen, V als die Menge der Verbindungen zwischen den Neuronen und $w(n_1, n_2)$ als das Gewicht zwischen zwei Neuronen $n_1, n_2 \in N$ zu verstehen.
Die Gewichte sind die Parameter des Netzwerks, die in der Trainingsphase angepasst werden. Sie bestimmen letztendlich die Ausgabe des Netzwerks.
Neuronale Netze haben eine *Eingabeschicht*, in der die Eingabe, beispielsweise ein Bild, bestehend aus seinen Pixeln, oder Musik als Zeitreihe von Vektoren, unbearbeitet in das Netz eingegeben werden. Dementsprechend hat eine Eingabeschicht genau so viele Neuronen wie die Eingabe Komponenten hat. Ebenso gibt es eine *Ausgabeschicht*, die letztendlich die Klassifikation repräsentiert. Diese hat dann so viele Neuronen wie es Klassen gibt. Dazwischen liegen meist *verborgene Schichten*, deren Neuronen aus einer

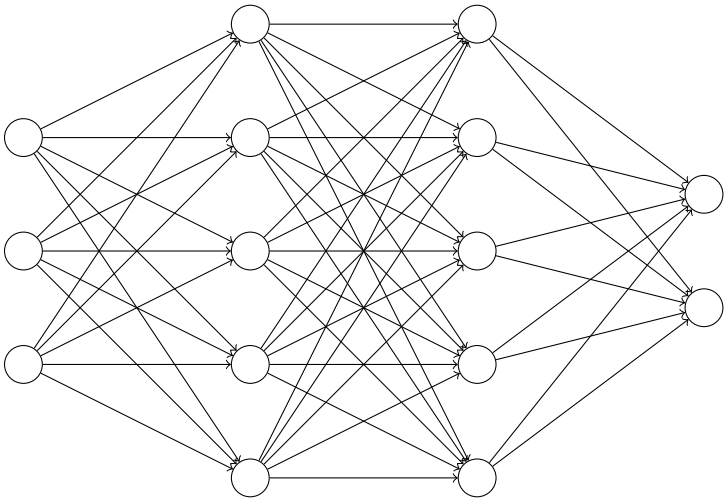

Abb. 2.1. Ein einfaches neuronales Netzwerk. Die Kreise stellen künstliche Neuronen dar, wobei es drei Eingaben gibt (die als Aktivierungen von künstlichen Neuronen aufgefasst werden können). Am Ende wirft das neuronale Netz zwei Ausgaben aus, die ebenfalls als Neuronen aufgefasst werden können. Die Pfeile zwischen den Neuronen repräsentieren die Verbindungen bzw. die Gewichte zwischen den Neuronen.

Linearkombination der Aktivierungen der Neuronen der vorherigen Schicht berechnet wird.
Hängt ein Neuron n_1 von einem anderen Neuron n_2 ab, so nennt man diese Neuronen *verbunden* bzw. man sagt, *es existiert eine Verbindung von n_2 nach n_1* (vgl. Abbildung 2.1). Diese Verbindungen werden mit den oben vorgestellten Gewichten w berechnet.

Beispiel 2.3 (Ziffererkennung)**.** Die Eingabeschicht ist nichts anderes als das eingegebene Bild, auf dem die Ziffer abgebildet ist. Die einzelnen Neuronen sind die Pixel des Bildes und ihre Aktivierung die Graustufe. Haben die Bilder das Format 100×100, dann gibt es $100 \cdot 100$ Eingabeneuronen. Die Ausgabeschicht ist $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ wobei jedes Neuron für eine der zehn Ziffern steht.

Es ist sinnvoll, $V = N^2$ zu setzen und den nicht vorhandenen Verbindungen das Gewicht 0 zuzuweisen. Dann lässt sich w als Matrix auffassen und leicht als Array implementieren.

Wichtige Arten neuronaler Netze

Da in der vorliegenden Arbeit ein bilderkennendes Netzwerk evaluiert werden soll, soll in diesem Abschnitt das dafür besonders geeignete *Faltungsnetzwerk* vorgestellt werden. Dieses gehören den folgenden Klassen an:

Definition 2.4 (Feed-Forward-Netzwerk)**.** Unter einem Feed-Forward-Netzwerk (FFN) versteht man ein neuronales Netzwerk, dessen Neuronen sich in klar abtrennbare Schichten eingeteilt sind. Darunter ist eine Eingabeschicht und eine Ausgabeschicht. Dabei

existieren nur Verbindungen von einer Schicht zu der ihr nachfolgenden Schicht. Mathematisch gesprochen: Sei $N = \dot{\cup}_{i \in I} S_i$, das heißt S_i sind Schichten. Dann folgt für $n_1 \in S_i, n_2 \notin S_{i+1} \implies w(n_1, n_2) = 0$ in einem FFN.

Darüber hinaus können neuronale Netze noch weitere *verborgene Schichten* (engl. *hidden layers*) besitzen, die sich je nach Netzwerktyp unterscheiden:

Definition 2.5 (Mehrschichten-Perzeptron)**.** Ein Feed-Forward-Netzwerkt mit mindestens einer verborgenen Schicht heißt *Mehrschichten-Perzeptron*.

Die Verknüpfung zwischen den Neuronen wird bewirkt, indem die Netzeingabe eines Neurons als Argument die Aktivierung der Neuronen der vorherigen Schicht erhält. Die Aktivierungen der Eingabeschicht bestimmen also die Aktivierungen der zweiten Schicht, die wiederum die Aktivierung der dritten Schicht, bis letztlich die Aktivierungen der vorletzten Schicht die Aktivierungen der Ausgabeschicht und damit die eigentliche Klassifizierung bestimmt. Abbildung 2.1 stellt ein Mehrschichtenperzeptron dar.
Ein Spezialfall der Mehrschichtenperzeptronen sind Faltungsnetzwerke. Sie zeichnen sich durch die Verwendung von *lokalen rezeptiven Feldern*, *gemeinsam genutzter Gewichte* und *Pooling-Schichten* aus
Die Kernidee von *lokalen rezeptiven Feldern* ist, dass jedes Neuron der Schicht $i + 1$ nur mit bestimmten ausgesuchten Neuronen der Schicht i verbunden ist, und zwar die Neuronen, die eine bestimmte quadratische Untermatrix der Netzeingabe bilden. Dieses Feld wird spalten- und zeilenweise über die gesamte Eingabe geschoben. Die Neuronen der Folgeschicht sind dabei ausschließlich mit den Neuronen eines Feldes der vorangehenden Schicht verbunden. Das soll in Abbildung 2.2 verdeutlicht werden.

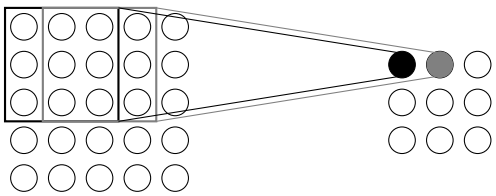

Abb. 2.2. Lokales rezeptives Feld nach [9]

Mathematisch lässt sich die Aktivierung eines Neurons an der Stelle (i,j) durch ein lokales rezeptives Feld der Dimension $h \times h$ wie folgt formulieren:

$$x^*_{i,j} = f\left(b + \sum_{k=0}^{h-1} \sum_{\ell=0}^{h-1} w_{k,\ell} x_{i+k,j+\ell}\right) \tag{2.1}$$

Dabei ist f die Aktivierungsfunktion, b der Bias, $x_{i,j}$ die Komponente der Eingabe an der Position (i,j) und w die Gewichtsmatrix der Verbindungen des $h \times h$ großen lokalen rezpetiven Felds zum Neuron x^* der Folgeschicht. Die Doppelsumme in 2.1 weist die Form einer *diskreten Matrixfaltung* auf und ist Namensgeber für diese Art von neuronalen Netzen.

Die Gewichte $w_{k,\ell}$ in Gleichung 2.1 hängen nicht von den Positionsvariablen i und j der Neuronen ab, werden also positionsunabhängig für alle Neuronen der Folgeschicht gemeinsam verwendet, weshalb auch von *gemeinsam genutzten Gewichten* gesprochen wird. Die Abbildung mittels rezeptiver Felder und einem Satz gemeinsam genutzter Gewichte wird *feature map* oder *Merkmalsabbildung* genannt. Intuitiv gesprochen soll eine feature map ein Merkmal positionsunabhängig im Bild identifizieren können.
Nun kann man die charakteristischste Schicht eines Faltungsnetzwerkes defineren:

Definition 2.6 (Faltungsschicht)**.** Eine *Faltungsschicht* in einem Faltungsnetzwerk ist eine Schicht, die aus mehreren Merkmalsabbildungen besteht. Dabei nutzt jede Merkmalsabbildung ihren eigenen Satz von gemeinsam genutzten Gewichten.

Da zur korrekten Klassifizierung üblicherweise mehrere Merkmale notwendig sind, sollten Faltungsschichten mehrere Merkmalsabbilungen besitzen.
Um Überempfindlichkeit des Netzwerkes zu vermeiden, enthalten Faltungsnetzwerke zusätzlich zu den Faltungsschichten noch sogenannte *Pooling-Schichten*.
Dabei wird die Schicht (die hier ja als Matrix von Neuronen aufgefasst wird) in sich nicht überlappende Untermatrizen aufgeteilt. Jede dieser Untermatrizen wird auf ein Matrixelement abgebildet. Als Abbild kann beispielsweise das Maximum oder der Durchschnitt der Elemente der Untermatrix gewählt werden.

Definition 2.7 (Pooling-Schicht)**.** Eine *Pooling-Schicht* bildet die Ausgabe einer Schicht auf eine kleinere Schicht ab. Dabei werden die Neuronen auf disjunkte gleichgroße Teilmengen aufgeteilt. Jede Teilmenge wird auf ein Neuron abgebildet.

Faltungsnetzwerke bestehen meistens aus sich abwechselnden Faltungs- und Poolingsschichten zwischen Eingabe- und Ausgabeschicht.

Kostenfunktionen

Kostenfunktionen (in Anlehnung an den Sprachgebrauch der mathematischen Optimierung auch Zielfunktionen genannt) sollen messen, wie „falsch“ ein Netzwerk mit seiner Ausgabe liegt. Dabei soll die Kostenfunktion bezüglich der Gewichte w minimiert werden. Diese Minimierung ist ein wesentlicher Bestandteil des (überwachten) Lernprozesses eines Netzes. Ein prominentes und einfach verständliches Beispiel für eine Kostenfunktion ist die Gauß'sche Verlustfunktion:

Beispiel 2.4 (Gauß'sche Verlustfunktion)**.** Sei S die Menge der Trainingspaare (x, t), wobei jedes x eine k-stellige Eingabe in das Netzwerk ist, y die zugehörige Ausgabe und t der n-dimensionale Klassenvektor, nach dem das Netz klassifizieren soll. Dann ist die Gauß'sche Verlustfunktion

$$SSE(S) := \sum_{(x,t)\in S} \sum_{k=1}^{n} (y_k - t_k)^2$$

Diese Funktion summiert also den quadratischen Abstand zwischen jeder Klasse und der Ausgabe auf. In einem perfekt trainierten Netzwerk entspricht die Ausgabe genau dem Klassenvektor t_k und $SSE(S)$ ist 0.

Eine weitere bekannte Verlustfunktion erlaubt für Klassifizierungsaufgaben ist die Kreuzentropie:

Beispiel 2.5 (Kreuzentropie)**.** Mit der Notation aus Beispiel 2.4 ist die Kreuzentropie definiert als:

$$CE(S) := -\sum_{(x,t)\in S}\sum t_k \ln y_k$$

Diese Kostenfunktion lässt sich aus dem Satz von Bayes herleiten und ist daher praktisch, wenn die Ausgabe des Netzes als Wahrscheinlichkeiten für Klassenzugehörigkeiten interpretiert werden sollen. Für eine Herleitung der Kreuzentropie siehe auch [9] 2.1.5 und 2.2.2.
Die Klassifizierungen t_k lassen sich als Ausgabeneuronen als Linearkombination der Aktivierungen der Eingabeneuronen und Gewichte w darstellen. Nun soll die gewählte Kostenfunktion nach den Gewichten w minimiert werden. Dies kann man zum Beispiel mit dem *Gradientenabstiegsverfahren* approximativ bewerkstelligen.

Gradientenabstiegsverfahren

Die Lernphase eines Netzwerks besteht im Wesentlichen darin, dass es bezüglich seiner Gewichte und einer Trainingsmenge optimiert wird. Dazu erhält das Netzwerk Eingaben als Training. Die Ausgabe des Netzes wird dann bezüglich der Gewichte in dem Sinne maximiert, dass die korrekte Klasse einen möglichst hohen Wert und die falschen Klassen möglichst niedrige Werte erhalten. Dies lässt sich durch eine analytische globale Minimierung der gewählten Kostenfunktion erreichen, was jedoch bei großen Netzen (und dementsprechend großen Funktionen mit vielen Variablen) zeit- und ressourcenaufwändig werden kann.
Daher wird in der Praxis häufig das Gradientenabstiegsverfahren verwendet, um ein lokales Minimum zu approximieren:

Definition 2.8 (Gradient)**.** Der Gradient ∇f einer Funktion $f(x_1, x_2, \ldots, x_n)$ ist definiert als der n-dimensionale Vektor

$$(\nabla f)_i = \frac{\partial}{\partial x_i} f$$

Weiter gilt:

- Ist $(\nabla f)_i(x) < 0$, so fällt f mit steigendem x_i. Um sich einem Minimum anzunähern, sollte x_i größer werden.
- Analog sollte bei $(\nabla f)_i(x) > 0$ das x_i kleiner werden um f zu minimieren.
- Bei $(\nabla f)_i(x) = 0$ ist das Minimum von f bezüglich x_i bereits erreicht und das Gradientenabstiegsverfahren kann beendet werden.

Dies lässt sich zusammenfassen zu: Um f zu minimieren, ändere x_i entgegen der Orientierung der partiellen Ableitung.
Möchte man ein beliebiges lokales Minimum von f aus Definition 2.8 approximieren, so kann man wie folgt vorgehen: Man wählt ein beliebiges $x^{(0)} \in \mathbb{R}^n$. Als $x^{(1)}$ wählt man

$$x^{(1)} = x^{(0)} - \varepsilon \nabla f(x^{(0)})$$

für ein kleines $\varepsilon > 0$. Nach mehreren Wiederholungen gelangt man schrittweise näher an das Minimum und hat so ein lokales Minimum approximiert.
In der Praxis werden die Gewichte zu Beginn beliebig, z.B. durch einen Zufallsgenerator, gesetzt. Anschließend wird der Gradient der gewählten Kostenfunktion bezüglich der Gewichte w berechnet (was auch als *Rückwärtsphase* bezeichnet wird).
Schließlich werden Eingaben aus einer Trainingsmenge in den Gradienten eingesetzt und jeweils nach w minimiert. Diesen Prozess wird auch das *Trainieren* des neuronalen Netzwerks genannt.

3 Das Integrierte-Gradienten-Verfahren

Es ist oft von großem Interesse die Komponenten einer Eingabe zu identifizieren, die für die Klassifizierung durch das (trainierte) Netzwerk eine besonders große Rolle spielen. So kann beispielsweise die Qualität des neuronalen Netzes besser beurteilt werden, indem man sich davon überzeugt, dass die Klassifikation des neuronalen Netzes auf den „richtigen Eigenschaften“ der Eingabe fußt, beispielsweise bei Bildern eine bestimmte Region.
Dazu erläutert und präzisiert der folgende Abschnitt [15] mit Anmerkungen aus [8].
Ein klassifizierendes neuronales Netzwerk lässt sich auch als eine Funktion $N : \mathbb{R}^n \to \mathbb{R}^m$ auffassen, wobei eine Eingabe $x \in \mathbb{R}^n$ auf einen Klassenvektor $N(x) \in \mathbb{R}^m$ abgebildet wird, der die Ausgabeschicht repräsentiert. Der Vektoreintrag mit dem größten Eintrag repräsentiert die Klasse, die das Netz der Eingabe zuweist. Sei nun $N(x)_j$ dieser Eintrag, das heißt das Netz hat x die Klasse j zugewiesen.
Um herauszufinden, wie groß der Einfluss der i-ten Eingabekomponenten auf die Klassifizierung in die Klasse j einer Eingabe ist, liegt es nahe zu beobachten, wie stark sich eine Änderung der i-ten Komponenten auf die Klassifizierung auswirkt. Einfach gesagt: Wie stark löst eine Variation der i-ten Komponenten der Eingabe eine Vergrößerung bzw. Verkleinerung von $N(x)_j$ aus?
Zur Interpretierbarkeit des Netzwerks bezüglich der Klasse j genügt es also, die j-te Komponente von $N(x)$ zu betrachten. Wir können also die Funktion N zur Funktion $F : \mathbb{R}^n \to \mathbb{R}; x \mapsto N(x)_j$ „stutzen“. Im Folgenden meint F eine ebensolche Funktion.

Definition 3.1 (Integrierte Gradienten)**.** Sei $x \in \mathbb{R}^n$ der Input und $y \in \mathbb{R}^n$ eine Baseline. Dann ist die i-te Komponente des Integrierten Gradienten von x bezüglich der Baseline y

$$IG_i(x) := (x_i - y_i) \cdot \int_0^1 (\nabla F)_i(y + t \cdot (x - y))\mathrm{d}t$$

wobei $(\nabla F)_i$ die i-te Komponente des Gradienten von $F(x)$ bezeichnet. Die Integrierten Gradienten (IG) sind definiert als die Summe:

$$IG := \sum_{i=1}^{n} IG_i$$

Nun ist zu klären, warum Definition 3.1 eine gute Wahl für die Interpretation neuronaler Netze ist.
Zu diesem Zweck haben Sundararajan et al. Axiome zur sinnvollen Definition von sogennanten *Beiträge* (engl. *attributions*) aufgestellt.

Definition 3.2 (Beitrag)**.** Ein *Beitrag* bezüglich eines Netzwerks F, einer Eingabe $x \in \mathbb{R}^n$ und einer Baseline $y \in \mathbb{R}^n$ ist ein Vektor

$$A_F(x,y) = (a_1, a_2, \ldots, a_n) \in \mathbb{R}^n$$

Weiter heißt a_i der Beitrag von x_i zur Vorhersage $F(x)$.

Solche genannten Beiträge erhält man, indem man sie durch eine *Beitragszuweisungsfunktion* berechnet:

Definition 3.3 (Beitragszuweisungsfunktion)**.** Eine *Beitragszuweisungsfunktion* oder *Zuweisungsfunktion* ist eine Funktion f_F, die eine Eingabe x auf einen Beitrag bezüglich F und eine Baseline y abbildet.

Es ist sinnvoll zu fordern, dass eine Eingabekomponente, deren Änderung auch eine Änderung in der Vorhersage hervorruft, einen Beitrag ungleich 0 aufweist. Das lässt sich im folgenden Axiom formalisieren:

Axiom 1 (Sensitivität)**.** Sei x eine Eingabe und $A_F(x, y)$ sei der Beitrag von x bezüglich F und y. Sollte eine Eingabe x' existieren mit $x_j \neq x'_j$ für ein j und $x_i = x'_i$ für $i \neq j$, dann sollte aus

$$F(x) \neq F(x') \implies a_j \neq 0$$

folgen.

Bildlich gesprochen verhindert die Erfüllung dieses Axioms, dass relevante Eingabekomponenten „übersehen“ werden.
Andersherum sollten Beiträge auch nur dann ungleich 0 sein, wenn die korrespondierenden Eingabekomponenten auch tatsächlich einen Einfluss auf die Vorhersage des Modells haben. Dies führt zu folgendem Axiom:

Axiom 2 (Spezifität)**.** Sei x eine Eingabe und $A_F(x, y)$ sei die Zuweisung von x bezüglich F und y. Gilt für alle Eingaben x' mit $x_j \neq x'_j$ für ein j und $x_i = x'_i$ für $i \neq j$, dann sollte aus

$$F(x) = F(x') \implies a_j = 0$$

folgen.

Bildlich gesprochen wird durch die Erfüllung des Axioms verhindert, das irrelevante Eingabekomponenten irrtümlich als relevant eingestuft werden.
Weiter lässt sich fordern, dass Beiträge für äquivalente Netzwerke identisch sein sollen:

Axiom 3 (Implementierungsinvarianz)**.** Seien F, F' zwei Netzwerke und $\alpha, \beta \in \mathbb{R}$. Eine Zuweisungsmethode ist implementationsinvariant, wenn

$$F(x) = F'(x) \implies A_F(x,y) = A_{F'}(x,y)$$

für beliebige Eingaben x und beliebige Baselines y gilt.

Die Zuweisungsfunktion muss also für Netzwerke, die für gleiche Eingaben gleiche Klassifizierungen ergeben, gleiche Beiträge zuweisen. Beiträge dürfen sich nicht allein durch die Implementierung ändern. Die Methode von [13] ist beispielsweise nicht Implementierungsinvariant.

Axiom 4 (Vollständigkeit)**.** Eine Beitragszuweisungsfunktion $A_F(x,y) = (a_1, a_2, \ldots, a_n)$ ist vollständig, wenn gilt:

$$\sum_{i=1}^{n} a_i = F(x) - F(y)$$

Bildlich gesprochen füllt eine vollständige Zuweisung A_F die „Lücke“ zwischen dem Bild x und der Baseline y. Damit kann die Differenz der Klassifikation zwischen x und y auf die Eingabekomponenten (oder im folgenden Beispiel: Pixel eines Bildes) bilanziert werden. Eine große Zuweisung der Komponente x_i wird so interpretiert, dass x_i einen wesentlichen Teil zum Abstand zwischen x und y beiträgt. Diese Interpretation kommt aus der Spieltheorie zur Aufteilung von Kosten, bzw. Gewinn, die sich auf das ökonomische Konzept des Grenzbeitrags bzw. der Grenzkosten stützt. Siehe dazu auch [2] oder als Beispiel auch [12]. Daher ist in der Praxis auch nicht der Gesamtwert IG, sondern die Zuweisungen IG_i von Bedeutung.

Axiom 5 (Linearität)**.** Seien F, G zwei Netzwerke. Eine Beitragszuweisungsfunktion A_F ist bezüglich des Netzwerkes linear, wenn

$$A_{\alpha F+\beta G}(x,y) = \alpha A_F(x,y) + \beta A_G(x,y)$$

gilt.

Somit bleiben im Netzwerk F vorhandene lineare Zusammenhänge auch in der Zuweisung A_F erhalten.

Eigenschaften der Integrierten Gradienten

Im Folgenden wird gezeigt, dass die Integrierten Gradienten die im vorherigen Kapitel genannten Axiome erfüllen. Darüber hinaus werden andere möglicherweise brauchbare Eigenschaften der Integrierten Gradienten gezeigt. Es folgt die Definition:

Definition 3.4 (Wegintegral zweiter Art)**.** Seien $f : \mathbb{R}^n \to \mathbb{R}^n$ und $\gamma : [a, b] \to \mathbb{R}^n$ stetig differenzierbar. Dann ist das Wegintegral zweiter Art $\int_\gamma f(t)\mathrm{d}t$ definiert als:

$$\int_\gamma f(t)\mathrm{d}t = \int_a^b \langle f(\gamma(t)), \gamma'(t)\rangle \mathrm{d}t = \sum_{i=1}^{n} \int_a^b (f(\gamma(t))_i (\gamma'(t))_i \mathrm{d}t$$

wobei $\langle \cdot, \cdot \rangle$ das euklidische Skalarprodukt bezeichnet.

Nun lässt sich leicht feststellen, dass IG_i ein ebensolches Wegintegral zweiter Art ist, indem man $(f(x))_i = (\nabla F)_i$ und $(\gamma(t))_i = y_i + t(x_i - y_i)$, also den direkten Weg zwischen der Eingabe x und der Baseline y, setzt.
Weiter sei der folgende Satz angegeben:

Satz 1 (Fundamentalsatz der Analysis für Wegintegrale)**.** Sei $f = \nabla F$ und F überall stetig differenzierbar. Für Wegintegrale zweiter Art gilt:

$$\int_\gamma f(t)\mathrm{d}t = F(\gamma(b)) - F(\gamma(a))$$

Beweis. Es gilt

$$\begin{aligned}
\int_\gamma f(t)\mathrm{d}t &= \int_a^b \left[f(\gamma(t))\right]^t \gamma'(t)\mathrm{d}t \\
&= \int_a^b \sum_{i=1}^n (f(\gamma(t))_i(\gamma'(t))_i\mathrm{d}t \\
&= \int_a^b \sum_{i=1}^n \frac{\mathrm{d}}{\mathrm{d}t}(f(\gamma(t))_i\mathrm{d}t \\
&\overset{(*)}{=} \int_a^b \frac{\mathrm{d}}{\mathrm{d}t}(f(\gamma(t))\mathrm{d}t \\
&= F(\gamma(b)) - F(\gamma(a))
\end{aligned}$$

Der entscheidende Schritt ist $(*)$. Dabei ist zu beachten, dass die ursprüngliche Funktion F zwar mehrere Argumente hat; da aber in jedes Argument $\gamma(t)$ eingesetzt wird, hängt die Komposition nur mehr von t ab. Siehe auch [4], S. 225. □

Dieser Satz lässt sich jedoch nicht ohne weiteres auf die Integrierten Gradienten übertragen, da ein Netzwerk F nicht zwingend überall stetig differenzierbar ist, etwa wenn sie ReLU-Funktionen oder max-Pooling-Schichten enthalten [8]. Schwächt man die Aussage so ab, dass F Lipschitz-stetig, der Weg γ dafür aber stetig differenzierbar, und F auf γ fast überall differenzierbar ist (d.h. es gibt maximal abzählbar unendlich viele Punkte, an dem die Differenzierbarkeit nicht gegeben ist), lässt sich die Aussage aus Satz 1 doch treffen. Dazu wird die *Lipschitz-Stetigkeit* benötigt:

Definition 3.5 (Lipschitz-Stetigkeit)**.** Eine Funktion $F : U \subseteq \mathbb{R}^n \to \mathbb{R}^m$ ist Lipschitz-stetig, wenn eine Konstante $L \geq 0$ existiert, sodass gilt:

$$||F(x) - F(y)||_2 \leq L||x - y||_2$$

für alle x,$y \in U$. Dabei ist $||\cdot||$ die euklidische Norm.

Desweiteren wird dazu der Satz von Rademacher benötigt:

Satz 2 (von Rademacher)**.** Sei U eine offene Teilmenge vom $\mathbb{R}^n$ und $F : U \to \mathbb{R}^m$ Lipschitz-stetig. Dann ist F fast überall stetig differenzierbar.

Beweis. Siehe [3]. □

Lerma und Lucas [8] konnten beweisen, dass Satz 1 auch für Lipschitz-stetige F und stetig differenzierbare γ gilt:

Satz 3. Sie $U \subseteq \mathbb{R}^n$ offen und $F : U \to \mathbb{R}$ Lipschitz-stetig und $\gamma(t) : [0,1] \to U$ differenzierbar. Weiter sei F auf γ für fast jedes t differenzierbar. Dann gilt:

$$\int_\gamma \nabla F(t)\mathrm{d}t = F(\gamma(1)) - F(\gamma(0))$$

Beweis. Da γ auf dem kompakten Intervall [0,1] differenzierbar ist, ist γ Lipschitz-stetig, denn da γ stetig differenzierbar ist, existiert eine endliche obere Schranke für γ'. Da die Komposition zweier Lipschitz-stetiger Funktionen wieder Lipschitz-stetig ist, ist $F(\gamma(t))$ wieder Lipschitz-stetig. Da aus Lipschitz-Stetigkeit absolute Stetigkeit folgt, gilt der Fundamentalsatz der Analysis für absolutstetige Funktionen [7]:

$$\int_0^1 F(\gamma(t)'\mathrm{d}t = F(\gamma(1)) - F(\gamma(0))$$

mit der Kettenregel folgt

$$F(\gamma(t))' = F(\gamma(t))\gamma'(t)$$

Damit folgt die Behauptung. □

Daraus folgt nun die Vollständigkeit der Integrierten-Gradienten-Methode:

Satz 4. Wenn F ein Lipschitz-stetiges und auf γ fast überall differenzierbares Netz ist, dann erfüllen die Integrierten Gradienten die Vollständigkeit (siehe Axiom 4).

Beweis. Der Weg $\gamma(t) = y + t(x - y)$ ist offensichtlich stetig differenzierbar. Setze dies in Satz 3 ein. □

Hieraus lässt sich die Sensitivität folgern:

Satz 5. Die Integrierten Gradienten eines stetig differenzierbaren Netzes F erfüllen die Sensitivität.

Beweis. Angenommen, x, x' seien zwei Eingaben mit $x_i = x'_i \ \forall i \neq j$ für ein j und $F(x) \neq F(x')$. Dann ist wegen der Vollständigkeit von $IG(x)$:

$$IG(x) = F(x) - F(y) \neq F(x') - F(y) = IG(x')$$

□

Um die weiteren Eigenschaften zeigen zu können, werden *Wege* und die Klasse der sogenannten *Wegmethoden* anlehnend an [5] und [15] eingeführt:

Definition 3.6 (Wegmethode)**.** Sei $\gamma : [0,1] \to \mathbb{R}^n$ stetig differenzierbar mit $\gamma(0) = x$ und $\gamma(1) = y$. Dann ist die Wegmethode der i. Komponenten bezüglich des Weges γ und der Funktion f gegeben durch:

$$PM_i(x,y,\gamma) := \int_0^1 \frac{\partial f(\gamma(t))}{\partial \gamma_i(t)} \frac{\partial \gamma_i(t)}{\partial t}\mathrm{d}t$$

Das entspricht gerade dem i-ten Summanden des Wegintegrals zweiter Art für ∇f. Setzt man $\gamma(t) = y + t(x - y)$, und legt eine Baseline y fest, so ist $PM_i(\cdot,y,\gamma) = IG_i(\cdot)$. Die Wegmethode hat besonders erstrebenswerte Eigenschaften:

Satz 6. Die einzigen Zuweisungsmethoden, die die Axiome Spezifität, Linearität, Vollständigkeit und Implementierungsinvarianz erfüllen, sind die Wegmethoden $PM = \sum_{i=1}^{n} \alpha_i PM_i$.

Beweis. Siehe [5], Theorem 1. Dabei entspricht *dummy* der Spezifität und *efficiency* der Vollständigkeit. Da Additivität notwendig für Linearität ist, genügt es sie zu zeigen, denn für ein Netwerke f und αf mit $\alpha \in \mathbb{R}$ gilt:

$$PM_{i_{\alpha f}}(x,y,\gamma) = \int_0^1 \frac{\partial \alpha f(\gamma(t)))}{\partial \gamma_i(t)} \frac{\partial \gamma_i(t)}{\partial t} \mathrm{d}t = \alpha \int_0^1 \frac{\partial f(\gamma(t)))}{\partial \gamma_i(t)} \frac{\partial \gamma_i(t)}{\partial t} \mathrm{d}t = \alpha PM_{i_f}(x,y,\gamma)$$

Damit folgt die Homogenität. □

Es stellt sich die Frage, warum die Integrierte-Gradienten-Methode als Spezialfall eine gute Wahl ist. Es wäre wünschenswert, dass zwei Pixel, die die exakt gleiche Rolle bei der Klassifizierung durch das Netzwerk spielen, auch der gleiche Beitrag zugewiesen wird.

Definition 3.7 (Symmetrierhaltende Funktion)**.** Eine Funktion (bzw. Netzwerk) $f : \mathbb{R}^n \to \mathbb{R}^m$ heißt symmetrieerhaltend bezüglich x_i und x_j, wenn für alle $x \in \mathbb{R}^n$

$$f(x) = f(\hat{x})$$

gilt mit $\hat{x}_i = x_j$, $\hat{x}_j = x_i$ und $\hat{x}_k = x_k$, für alle $k \neq i,j$.

Über das Integrierte-Gradienten-Verfahren lässt sich dann sagen:

Satz 7. Ist das Netzwerk $F(x)$ bezüglich zweier Komponenten von x_i und x_j symmetrieerhaltend, so gilt $IG_i(x,y) = IG_j(x,y)$ wenn $x_i = x_j$ und $y_i = y_j$.

Beweis. Ist F bezüglich x_i und x_j symmetrieerhaltend, muss $(\nabla F)_i = (\nabla F)_j$ gelten.

$$\begin{aligned} IG_i &= (x_i - y_i) \int_0^1 (\nabla F)_i (y + t(x - y)) \mathrm{d}t \\ &= (x_j - y_j) \int_0^1 (\nabla F)_i (y + t(x - y)) \mathrm{d}t \\ &= (x_j - y_j) \int_0^1 (\nabla F)_j (y + t(x - y)) \mathrm{d}t \\ &= IG_j \end{aligned}$$

□

Einfach gesprochen wurde bewiesen, dass aus der Vertauschbarkeit zweier Pixel folgt, dass dann diese Pixel durch Integrierte-Gradienten-Methode den gleichen Beitrag zugewiesen bekommen.
Diese Eigenschaft wird von allgemeinen Wegmethoden nicht erfüllt:

Satz 8. Wegmethoden sind im Allgemeinen nicht symmetrieerhaltend im obigen Sinne.

Beweis. Betrachte die Funktion $F(x_1,x_2) = x_1x_2$ und den Weg $\gamma(t) = (t, t^2)^T$. Dann ist F offensichtlich symmetrieerhaltend bezüglich seiner beiden Komponenten und es ist $\nabla F = (x_2, x_1)^T$. Sei die Baseline $y = (0,0)^T$ und die Eingabe $x = (1,1)^T$. Dann ist:

$$PM_F(x,y)_1 = \int_0^1 (\nabla F)_1(\gamma(t))\gamma_1'(t)\mathrm{d}t = \int_0^1 t^2 \cdot t\mathrm{d}t = \int_0^1 t^3\mathrm{d}t = \frac{1}{3}$$

Andererseits ist auch:

$$PM_F(x,y)_2 = \int_0^1 (\nabla F)_2(\gamma(t))\gamma_2'(t)\mathrm{d}t = \int_0^1 t \cdot 2t\mathrm{d}t = 2\int_0^1 t^2\mathrm{d}t = \frac{2}{3}$$

□

Bemerkung 3.1. Es sei angemerkt, dass $\gamma(t)_i \neq y_i + t(x_i - y_i)$ für $I = 1,2,...,n$ sein kann und für eine Wegmethode verwendet dennoch die Integrierten Gradienten hervorbringt. Entscheidend ist, dass γ in jeder Komponente die gleiche Form hat:
Ist ein Weg der Form

$$\gamma(t) = y + \varphi(t)(x - y)$$

mit $\varphi(t) : [0,1] \to \mathbb{R}$ mit $\varphi(0) = 0$, $\varphi(1) = 1$ und φ stetig differenzierbar. Dann gilt für jeden Summanden der Wegintegrale

$$\int_0^1 (\nabla F)_i(\gamma(t))\gamma_i'(t)dt = IG_i(x,y)$$

Mit Substitution erhält man:

$$\begin{aligned}
\int_\gamma (\nabla F)_i(t)\mathrm{d}t &= \int_0^1 (\nabla F)_i(y + \varphi(t)(x - y))(\varphi'(t)(x_i - y_i))\mathrm{d}t && |u := \varphi(t) \\
&= \int_{\varphi(0)}^{\varphi(1)} (\nabla F)_i(y + u(x - y))\varphi'(t)(x_i - y_i)\frac{\mathrm{d}u}{\varphi'(t)} \\
&= (x_i - y_i)\int_0^1 (\nabla F)_i(y + u(x - y))\mathrm{d}u && |u := t \\
&= (x_i - y_i)\int_0^1 (\nabla F)_i(y - t(x - y))\mathrm{d}t \\
&= IG_i(x,y)
\end{aligned}$$

Satz 7 hebt die Bedeutung der IG_i hervor. Während alle Pfadmethoden in ihrer Summe das gleiche ergeben, lassen bei den Integrierten Gradienten unterschiedliche Zuschreibungen der Pixel den Schluss zu, dass diese Pixel eine unterschiedlich große Rolle in Klassifikation haben.
Sundararajan et al. führen in ihrer Arbeit noch einen Beweis dafür an, dass das Integrierte-Gradienten-Verfahren die einzige Wegmethode sei, die symmetrieerhaltend im obigen Sinne ist. Ein Problem dabei ist unter anderem ihre unpräzise Definition der Symmetrieerhaltung. Lerma und Lucas [8] konnten zeigen, dass mit einer Definition analog zu der Definition 3.7 und der Zusatzbedingung des monotonen Steigens bzw. Fallens von

γ (dass γ also in allen ihren Komponenten monoton steigend beziehungsweise fallend ist) diese Eindeutig gegeben ist, aber nur dann, wenn die i-te und j-te Komponenten von Eingabe x bzw. Baseline y übereinstimmen. Für den Fall, dass die Komponenten nicht übereinstimmen, konnten sie ein Gegenbeispiel angeben.
Dies tut der Sinnhaftigkeit des Integrierten-Gradienten-Verfahrens keinen Abbruch, da es eine sehr einfache Variante der Wegmethoden ist, die Symmetrieerhaltung gewährleistet.

4 Anwendung

In diesem Kapitel soll mit dem Integrierte-Gradienten-Verfahren die Qualität der Entscheidungsfindungsqualität eines Netzwerks untersucht werden, das Röntgenbilder der Lunge in die Kategorien „krank“ und „gesund“ klassifiziert. Zudem wird an Beispielen gezeigt, wie die Integrierten Gradienten visualisiert werden können.

4.1 Aufgabenstellung

Das im vorhergehenden Kapitel vorgestellte Integrierte-Gradienten-Verfahren soll dazu dienen, das Verhalten neuronaler Netzwerke transparenter und so besser verständlich machen. Damit eröffnet sich auch die Möglichkeit zu bewerten, ob das Netzwerk seine Klassifizierung auf die gewünschten Eigenschaften der Eingabe stützt. Zur Berechnung der Integrierten Gradienten wurde eine eigens für diese Aufgabenstellung modifizierte Version des Codes von Nain [10] verwendet.
Zu diesem Zweck wurde ein von der Firma Planet artifical intelligence GmbH (Planet AI) aus Rostock bereitgestelltes Netzwerk untersucht, das Röntgenbilder der Lunge in die beiden Klassen „krank“ und „gesund“ klassifiziert. Es handelt sich um eine Abwandlung des `InceptionV3`-Netzwerks von Keras [16]. Es ost ein Faltungsnetzwerk, das mit 8000 Röntgenbildern trainiert wurde und hat von 1000 Validierungsbildern 83,53% richtig klassifiziert.
Ebenfalls von Planet AI bereitgestellt wurde jeweils ein Satz von insgesamt 200 Trainings- und Validierungsbildern von denen 100 auffällige Regionen beinhalteten. Für jedes Bild, das auffällige Regionen aufwies, lagen Markierungen dieser Regionen in Gestalt von Rechtecken vor. Diese Rechtecke wurden von Ärzten gezogen.
In diesem Kapitel soll gezeigt werden, wie mithilfe des Integrierte-Gradienten-Verfahrens die Entscheidungen des Netzwerks für „krank“ erklärt werden und damit die Entscheidungsfindungsqualität beurteilt werden kann. Dazu wurden ausschließlich die Bilder aus dem Trainings- und Validierungsdatensatz verwendet, die auffällige Regionen beinhalten. Es werden im Folgenden die drei Maße F_1-Score, ein Hybrid aus Accuracy und F_1-Score sowie die Accuracy des größten Perzentils angewendet und kritisch gewürdigt.

4.2 Visualisierungen

In diesem Abschnitt wird demonstriert, wie die Integrierten Gradienten von den oben erläuterten Röntgenbildern dargestellt werden können. In Abbildung 4.1 wurde die Matrix von Beiträgen eines der Bilder für verschiedene Schwellenwerte eines Maximal- und Minimalperzentils dargestellt. Im linken Bild, das nicht weiter prozessiert wurde, lassen

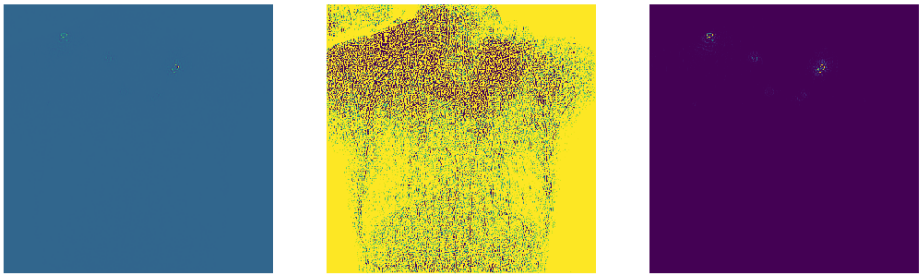

Abb. 4.1. Verschiedene Visualisierungen eines Integrierten Gradienten. Im linken Bild wurde er nicht bearbeitet, im mittleren wurden die Beiträge unter dem 10. und über dem 30. Perzentil abgeschnitten, im rechten wurden die Beiträge zwischen dem 85. und 99,99. Perzentil abgeschnitten.

sich zwei Stellen mit Beiträgen besonders hoher Intensität feststellen. Im mittleren Bild lassen sich die Umrisse des Oberkörpers und auch die Lungenflügel gut erkennen. Im rechten Bild, dessen Schwellenperzentile dem in Kapitel 4.3 beschriebenen Rendering schon nahe kommt, lassen sich zwei Schwerpunkte gut identifizieren.
In Abbildung 4.2 wurden mithilfe der Visualiserungsklasse von Nain [10] die Integrierten Gradienten in ihre Ursprungsbilder integriert. Dabei wurden alle Werte über dem 99,5. Perzentil auf diesen Wert abgeschnitten. Um die großen Beiträgen besser hervorzuheben, wurden zudem alle Beiträgen unter dem 97. Perzentil abgeschnitten. Die grünen Punkte markieren große Beiträgen und damit Pixel von besonders großem Belang für die Klassifizierung durch das Netzwerk (bezüglich der Klassifikation des schwarzen Bildes durch das Netzwerk). Diese Art der Darstellung verdeutlicht den potentiellen Nutzen für den Anwender, beispielsweise als Diagnostiktool.

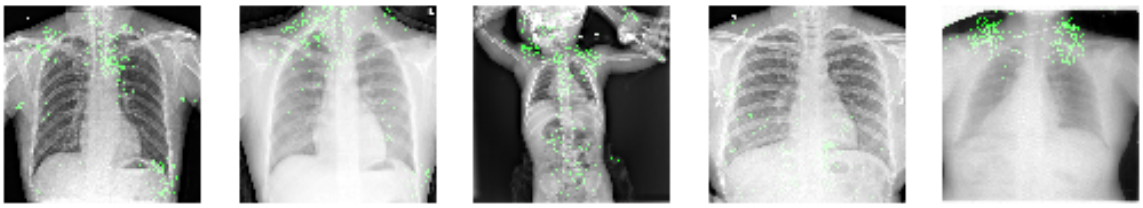

Abb. 4.2. Verschiedene Integrierte Gradienten über ihre Ursprungsbilder gelegt.

4.3 Auswertung

Die durch das Integrierte-Gradienten-Verfahren errechneten IG_{ij} geben Aufschluss über dir Rolle der ij-ten Komponente der Eingabe bei der Klassifizierung durch das Netzwerk. In diesem Kapitel soll erörtert werden, wie die Integrierten Gradienten quantitativ erfasst werden könne.

4.3.1 Problematik

Im Integrierte-Gradienten-Verfahren erhält jede Komponente der Eingabe, in diesem konkreten Fall die Pixel des eingegebenen Bildes, einen Beitrag $IG_{ij} \in \mathbb{R}$, die beziffern soll, wie viel dieses Pixel zur Klassifikation beiträgt. Anders gesagt beziffert IG_{ij} den Beitrag zur Klassifikationsdifferenz zwischen Eingabe und Baseline. Setzt man voraus, dass die Baseline eine niedrigere Klassifikation durch das Netzwerk erhält als die Eingabe, bedeuten größere IG_{ij} einen bedeutsameren positiven Beitrag (das Pixel hat die Differenz vergrößert), negative IG_{ij} einen negativen Beitrag (das Pixel hat die Differenz verringert).
Dementsprechend lassen sich Häufungen betragsmäßig hoher IG_{ij} als eine Region betrachten, die für die Klassifikation durch das Netzwerk wichtig sind und erklären damit die Klassifikation des Netzwerks.
Ein naheliegender Gedanke zur Untersuchung der Entscheidungsqualität ist, wie groß die betragsmäßige Übereinstimmung der großen IG_{ij} mit den Ground Truths (GT), also den von Ärzten markierten Regionen, ist. Der Einfachheit halber werden in dieser Arbeit nur große IG_{ij} betrachtet, die diejenigen Pixel repräsentieren, die besonders *für* die getroffene Klassifikation als "krank" sprechen. Dazu muss zum einen ein Schwellenwert festgelegt werden, ab welchem Wert ein IG_{ij} als „groß" und damit in hohem Maße relevant gilt. Dabei ist zu beachten, dass dieser Schwellenwert nicht zu hoch angesetzt werden sollte, da sonst Regionen übersehen werden könnten. Er sollte aber auch nicht zu niedrig gewählt werden, denn dann würden große Teile des Bilds als bedeutsam für die Klassifikation eingestuft und so keine valide Aussage gemacht werden können.
Zum anderen muss eine Definition dafür gefunden werden, ab wann eine GT mit großen IG_{ij} übereinstimmt, oder allgemeiner: ab wann eine Menge von IG_{ij} nicht nur ein für das Netzwerk kritisches Pixel, sondern eine kritische Region anzeigt, also eine auffällige Region *vorhersagt*. Wünschenswert wäre eine Menge von großen IG_{ij} die räumlich nahe beieinander liegen, denn vereinzelte IG_{ij} können Ausreißer sein. Für solche Cluster sollte zudem eine Mindestgröße festgelegt werden, denn je kleinere Cluster man berücksichtigt, desto eher beeinflussen Ausreißer die Auswertung und desto weniger wahrscheinlich haben die IG_{ij} ihren großen Wert wegen der meist eher großflächigen GTs angenommen.
Wichtig ist zudem zu definieren, ab wann eine GT als vorhergesagt und ab wann eine Vorhersage als korrekt gewertet wird. Dabei ist zu berücksichtigen, dass die GTs durch von Ärzten gezogene Boxen repräsentiert werden und so keiner exakten Logik folgen. Die Boxen variieren unabhängig von der eigentlichen Auffälligkeiten in ihrer Größe. Darüber hinaus sind manchmal mehrere kleine Auffälligkeiten in einer großen Box zusammengefasst. Das macht es schwierig, ohne Verzerrung die Definition beispielsweise an einen Mindestflächeninhalt des Schnitts zwischen GT und Vorhersage oder die Fläche

der GT, die von Vorhersagen geschnitten wurde, zu knüpfen.

4.3.2 Maße

In diesem Abschnitt sollen die in dieser Arbeit verwendeten Maße vorgestellt werden. Außerdem soll erläutert werden, wie der oben beschriebenen Problematik begegnet wird.

F_1-Score

Um eine *erklärende Region*, also eine regionale Häufung von Pixeln mit hohen Beiträgen, zu identifizieren, muss einerseits ein Schwellenwert gefunden werden, ab wann ein Beitrag IG_{ij} als groß und damit bedeutend genug gewertet wird, andererseits muss eine Methode gefunden werden, die räumliche Häufungen von solchen großen Beiträgen identifiziert. Die Extraktion solcher Häufungsregionen ist erstrebenswert, da sie weniger anfällig für stochastische Ausreißer ist und die zusätzliche Informationen, wie die räumliche Größe einer Vorhersage, enthält.
In der vorliegenden Arbeit wird als Schwellenwert für die Mindestgröße eines IG_{ij} das p-te Perzentil mit $p \in [0{,}100]$ des Integrierten Gradienten eines Bildes gewählt. So werden mit beispielsweise dem 95. Perzentil als Schwelle nur die 5% der größten IG_{ij} betrachtet und so Pixel von besonders großer Bedeutung extrahiert.
Um ganze Regionen zu extrahieren, um so die räumliche Nähe zu berücksichtigen, werden die Integrierten Gradienten jedes Bildes mittels eines `connectedComponents`-Befehls gefiltert und anschließend mit einer Bounding Box umrandet werden. Als zweiter Schwellenwert sollte die Mindestgröße einer solchen Bounding Box berücksichtigt werden, weil anderenfalls viele sehr kleine Vorhersagen in Gestalt von Bounding Boxes gemacht werden.
Es muss noch festgelegt werden, ab wann eine GT als vorhergesagt gilt. Um eine Art obere Schranke zu ermitteln, wurde in der vorliegenden Arbeit die niedrigschwellige Definition verwendet, dass ein Schnitt mit einem positiven Flächeninhalt zwischen GT und Vorhersage als Übereinstimmung von GT und Vorhersage gesetzt wird.
Ein Maß zur Beurteilung von Netzwerken ist der F_1-Score. Dieses Maß hat den Vorteil, dass er der in Abschnitt 4.3.1 erläuterten Problematik der zu wählenden Schwellenwerte begegnet. In dieser Arbeit sind dies das oben erläuterte p-te Perzentil und die Mindestfläche A_{min} der Bounding Boxes.
Der F_1-Score ist als Komposition der beiden Scores Precision und Recall definiert:

$$\mathrm{F}_1 = \frac{2 \cdot \mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$$

Precision und Recall werden wiederum wie folgt definiert:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$
$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

Dabei stehen TP für die Anzahl der Richtig-positiven (*true-positives*), FP für die Menge der Falsch-positiven und FN Menge für die der Falsch-negativen. In diesem Zusammenhang ist :

$$\text{TP} = \{\text{GT} : \text{GT vorhergesagt}\}$$

Komplementär dazu ist:

$$\text{FN} = \{\text{GT} : \text{GT nicht vorhergesagt}\}$$

Entsprechend gilt dann:

$$\text{FP} = \{\text{Vorhersage} : \text{Vorhersage entspricht keiner GT}\}$$

Die Precision misst also das Verhältnis der richtigen Vorhersagen an allen Vorhersagen und gibt so, ihrem Namen gerecht werdend, über die Genauigkeit der Vorhersagen Auskunft. Sie ist monoton wachsend in TP und nimmt ihren maximalen Wert 1 an, wenn alle Vorhersagen GTs entsprechen. Sie nimmt ihr Minimum 0 an, wenn keine einzige Vorhersage einer GT entspricht.
Der Recall misst das Verhältnis der richtigen Vorhersagen an der Menge der GTs. Sie nimmt ebenfalls Werte zwischen 1 und 0 an, wobei er monoton in TP wächst und 1 wird, wenn alle GTs vorhergesagt werden, und 0 wenn keine GT vorhergesagt wird.
Dementsprechend erreicht F_1 sein Maximum genau dann, wenn Precision und Recall 1 sind, und ihr Minimum wenn Precision und Recall 0 sind. F_1 wird dann groß, wenn Precision und Recall große Werte nahe 1 annehmen.
Um der in Abschnitt 4.3.1 geschilderten Problematik zu begegnen, soll der F_1-Score maximiert und so ein Art „Kompromiss“ zwischen Precision und Recall gefunden wird. Denn je größter das Perzentil p gewählt wird, desto größer wird der Recall und desto geringer die Precision. Die Umkehrung würde für ein zu klein gewähltes p gelten. Ähnlich sinkt mit Vergößerung der Mindestfläche A_{min} die Anzahl der Vorhersagen der Recall, während die Precision steigt. Daher lassen sich durch das Maximieren von F_1 geeignete p und A_{min} finden. Die Parameter werden p und A_{min} ermittelt, indem F_1 wie in Abschnitt **??** für alle Kombinationen aus $p \in \{70, 7580, 85, 90, 95, 99, 99.3, 99.5, 99.7, 99.9\}$ und $A_{\text{min}} \in \{1, 5, 10, 15, 20, 30, 40{,}50, 60,\ 70, 80, 100\}$ für den Trainingsdatensatz berechnet werden.
Wie bereits in diesem Abschnitt impliziert, ist entspricht TP den korrekt vorhergesagten GTs:

$$\text{TP} = |\{\text{Vorhersage} : \text{stimmt mit einer GT überein}\}|$$

FP entspricht dann den Vorhersagen, die keine GT vorhersagen:

$$\text{FP} = |\{\text{Vorhersage} : \text{stimmt mit keiner GT überein}\}|$$

FN entspricht dann denjenigen GTs, die nicht vorhergesagt wurden:

$$\text{FN} = |\{\text{GT} : \text{stimmt mit keiner Vorhersage überein}\}|$$

Da F_1 über den Recall jede nicht entdeckte GT bestraft, gibt F_1 Auskunft darüber,

wie gut das eigentlich binär klassifizierende Netzwerk Objekte erkennen kann. Auch das stellt in gewisser Hinsicht schon ein Maß für das eigentlich binär klassifizierende Netzwerk dar: ist das Netz ideal trainiert, macht es seine Entscheidung für die Klassifikation „krank“ von gerade den Pixeln abhängig, die eine erkrankte Stelle anzeigen und berücksichtigt die anderen Pixel nicht (siehe Axiome 1 und 2).
Man kann einwenden, dass dies eigentlich ein zu strenges Kriterium für das binär klassifizierende Netzwerk ist. Für ein solches Netzwerk ist ein Lernprozess bereits erfolgreich, wenn *eine* GT korrekt vorhergesagt wird, die zur Klassifikation als „krank“ führt. Dennoch ist ein guter F_1-Wert ein nicht notwendiges, jedoch hinreichendes Kriterium für die Qualität eines binär klassifizierendes Netzwerks.

F_1-Accuracy-Hybrid

Für ein binär klassifizierendes Netzwerk genügt es, eine GT pro Bild zu erkennen. Dazu soll ein Hybrid aus dem F_1-Score und der *Accuracy* dienen. Die Accuracy wird wie folgt definiert:

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} = \frac{\text{TP+TN}}{|\text{Grundgesamtheit}|}$$

TP, FP, FN und TN (*true-negatives*) beziehen sich hier nicht mehr auf GTs, sondern auf Bilder. Wie in der obigen Formel impliziert addieren sich die vier Zahlen zur Mächtigkeit der Grundgesamtheit auf.
Einfach gesprochen ist die Accuracy der Anteil der richtig eingestuften Elemente an der Gesamtmenge der Elemente, also das, was man gemeinhin als Genauigkeit kennt.
Da in dieser Arbeit untersucht werden soll, inwiefern die GTs, die als Cluster von großen IG_{ij} definiert werden, übereinstimmen, sind TP, TN, FP und FN passend zu definieren. Da nur für Bilder mit tatsächlich auffälligen Regionen GTs untersucht werden, muss die Menge dieser Bilder die Grundgesamtheit bilden:

$$\text{Grundgesamtheit} = \{\text{Bilder mit GT}\}$$

Bei dem vorliegenden Netzwerk handelt es sich um einen binären Klassifikator, der Eingaben auf Wahrscheinlichkeiten abbildet, „krank“ zu sein. Von diesem Netzwerk lässt sich daher nur erwarten, mindestens eine GT pro Bild korrekt vorherzusagen, nicht jedoch alle. Damit lässt sich TP definieren als:

$$\text{TP} = |\{\text{Bild} \in \text{Grundgesamtheit} : \text{mindestens eine GT vorhergesagt}\}|$$

TN sind in diesem Zusammenhang die Bilder, bei denen korrekterweise keine Vorhersage getroffen wurde, da keine GT vorliegt. Per Definition der Grundgesamtheit werden jedoch nur Bilder mit GT betrachtet. Daher gilt:

$$\text{TN} = 0$$

Um festzulegen, wann eine Vorhersage als getroffen und wann eine Vorhersage als korrekt gilt, lässt sich der F_1-Score verwenden. Somit erhält man als Maß den Anteil der

Bilder, bei denen mindestens eine GT korrekt vorhergesagt wurde und erhält damit eine Art Performanz. Dabei pflanzt sich jedoch das Problem des F_1-Scores fort, dass auch dann nicht vorhergesagte GTs bestraft werden, wenn bereits eine GT im Bild korrekt vorhergesagt wird. Dies kann zu einer Fehlgewichtung zugunsten des Recalls führen, was sich wiederum in einer schlechten Precision niederschlägt. Das führt wiederum dazu, dass viele Vorhersagen gemacht werden, die keiner GT entsprechen. Im Vergleich zur menschlichen Wahrnehmung wird die Qualität des Netzwerks damit also eher überschätzt.

Accuracy des Top-Pixels

Um das Problem des Recalls zu umgehen, kann man im Bild anstatt ganze Regionen von IG_{ij} zu extrahieren, nur das *Top-Pixel*, also das Pixel mit dem größten Beitrag IG_{ij}, berücksichtigen. Das entspricht dem Sonderfall $p = 100$ und $A_{\text{min}} = 0$ des vorangehenden Abschnitts. Es wird also nur noch ein Pixel (oder in seltenen Fällen einigen wenigen Pixeln) pro Bild betrachtet und überprüft, ob es in einer GT liegt und damit die Accuracy berechnet:

$$\text{Accuracy} = \frac{\{\text{Bilder} \in \text{Grundgesamtheit} : \text{Top-Pixel liegt in GT}\}}{|\text{Grundgesamtheit}|}$$

Die Maximierung des F_1-Scores entfällt, da die Problematik des Konflikts zwischen Precision und Recall nicht mehr exisitert. Da nur die Pixel des 100. Perzentil betrachtet werden, das in Regel nur ein Pixel ist, entfällt die Gefahr, zu weite Teile des Bildes als Vorhersage zu markieren. Damit ist die Verzerrungsgefahr durch die Falschgewichtung über den Recall gebannt. Andererseits ist zu bedenken, dass die einzeln betrachteten Pixel mit deutlich höherer Wahrscheinlichkeit Ausreißer sein könnten als ganze Cluster. Dadurch, dass weitere Pixel mit großen IG_{ij} ignoriert werden, wird die Anzahl der Bilder mit mindestens einer richtigen Vorhersage im Vergleich zur menschlichen Wahrnehmung eher unterschätzt.

4.3.3 Vorverarbeitung und Umsetzung

Um Ausreißer aus den IG zu entfernen, wurde jede IG mit einem Gauß-Filter ($\sigma = 5$) gefiltert und so geglättet.
Es wurden die Integrierten Gradienten bezüglich der Klasse „krank“ berechnet.
Pro Bild wurde das p-te Perzentil der Integrierten Gradienten berechnet. Es wurden alle IG_{ij}, die darunter lagen, auf 0 und alle darüber auf 1 gesetzt. Anschließend wurde sie mit `connectedComponents` weiterverarbeitet, um so Cluster zu erhalten (siehe Abb. 4.4). Anschließend wurde für jedes Cluster eine Bounding Box berechnet, welche eine Vorhesage repräsentiert.
Eine Bounding Box wurde nur bei der weiteren Auswertung berücksichtigt, wenn sie einen Mindestflächeninhalt von A_{min} aufwies. Eine Vorhersage durch den F_1-Score und dem F_1-Accuracy-Hybrids galt dann als übereinstimmend, wenn der Schnitt zwischen GT (ebenfalls durch ein Rechteck repräsentiert) und Vorhersage größer 0 war. In Abbildung 4.3 wurde die Repräsentation von GTs und Vorhersagen visualisiert.

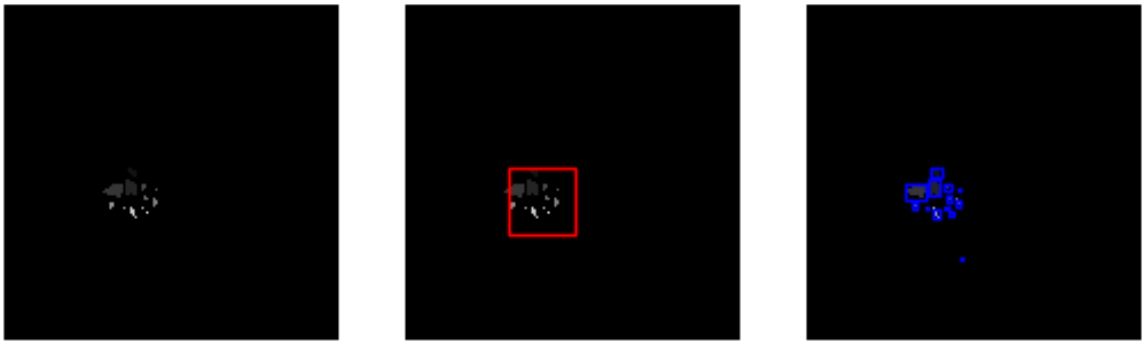

Abb. 4.3. Cluster, GTs und Bounding Boxes. Von links nach rechts: gerenderte Integrierte Gradienten (gIG), gIG mit von Ärzten markierten Regionen, gIG mit Bounding Box

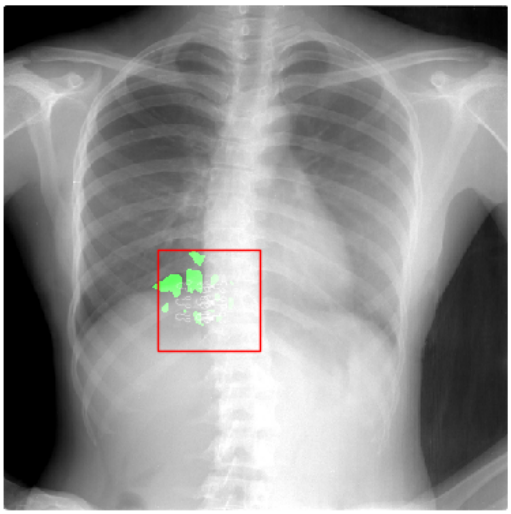

Abb. 4.4. Verabeitete IGs über Ursprungsbild gelegt, mit GT als rotes Rechteck.

Als Baseline wurde wie in [15] nahegelegt, die 0-Matrix, also das schwarze Bild, gewählt. Die Baseline hatte eine Klassifikation von ungefähr 0,2.

4.3.4 Ergebnisse und Diskussion

Mit jedem der drei Maße wurde jeweils der Satz alle Bilder, die eine GT beinhalten, gemessen.
Die Ergebnisse der Maximierung des F_1-Scores sind in Tabelle 4.1 zusammengefasst. Es ist zu erkennen, dass vor allem der Recall einen hohen Wert annimmt, ein erheblicher Anteil der erkrankten Regionen von einer Bounding Box geschnitten wurde. Die Precision von etwa 0.5 gibt an, dass aber etwas nur die Hälfte dieser Bounding Boxes tatsächlich eine erkrankte Region geschnitten hat. Die dabei verwendeten Parameter sind $p = 99.5$, also das 99.5. Perzentil der Integrierten Gradienten, sowie eine Mindestfläche A_{min} von 70 Pixel.

Der Recall ist deutlich größer als die Precision. Es wurden also unter diesen Parametern über 80% der auffälligen Regionen über die Bounding Boxes gefunden, aber nur 50% Bounding Boxes stimmen mit einer GT überein. Es ist also zu einem gewissen Grad der Effekt eingetreten, dass auf Kosten der Präzision „zu viele“ Vorhersagen gemacht wurden, was dem Recall zugute kommt.

Tabelle 4.1. maximaler F_1-Score und Accuracy

F1	Precision	Recall	$\text{Accuracy}_{\text{Ges}}$	p	$A_{\min}$
0.6360	0.5187	0.8219	0.8900	99.5	70

Die Accuracy ist bei 0.89, es wurden also bei knapp 90% der Bilder mindestens eine der auffällige Regionen von einer Bounding Box geschnitten.
Die Werte für die Accuracy des Top-Pixels sind in Tabelle 4.2 aufgelistet. Dieser Fall ist interessant, weil hier der Recall, der nicht erkannte GTs bestraft, vernachlässigt werden kann. Für die Beurteilung des Netzwerks genügt es, mindestens eine GT pro Bild zu erkennen.

Tabelle 4.2. Accuracy beim 100. Perzentil

Precision	$\text{Accuracy}_{\text{Ges}}$	$\text{Accuracy}_{\text{train}}$	$\text{Accuracy}_{\text{val}}$
0.6250	0.6300	0.6250	0.6363

Der F_1-Score von 63.3% gibt Auskunft über die Objekterkennungsqualitäten des Netzwerks. Das weist mit 0,8219 einen hohen Recall auf, aber auch eine niedrige Precision von 0,5187. Während über 80% der GTs korrekt vorhergesagt wurden, waren nur gut 50% der Vorhersagen korrekt. Das zeigt, dass sich das Netz nur bedingt zur Objekterkennung eignet. Andererseits ist der F_1-Score ein hartes Maß für ein binär Klassifizierendes Netzwerk, da er fordert, alle GTs zu identifizieren. Es könnte jedoch sein, dass das Netzwerk gelernt hat, nur mindestens eine GT richtig vorherzusagen um so zur korrekten Klassifizierung zu kommen.
Die Accuracy, die die Parameter des F_1-Score übernimmt, ist mit knapp 90% hoch; in 90% der Bilder wurde also mindestens eine GT korrekt vorhergesagt. Die geringe Precision bedeutet jedoch, dass viele Vorhersagen Falsch-positive sind, was sich in einem hohen Recall und dadurch in einer höhere Accuracy niederschlägt. Dennoch hat das Maß hat die Aussagekraft, wie viele der Bilder *unter anderem* aufgrund der GTs korrekt klassifiziert wurden.
Betrachtet man das 100. Perzentil, ergibt sich eine Accuracy von gut 60%. Es liegen also 60% der größten Perzentile des Integrierten Gradienten eines Bildes in einer GT. Der Verzerrungseffekt durch den Recall tritt hier nicht auf. Zudem bringt de Betrachtung von (in der Regel) nur einem Pixel mit sich, dass nur eine Vorhersage pro Bild getroffen wird und damit andere potentiell richtige Vorhersagen nicht berücksichtigt werden. Da bei den meisten Bildern davon ausgegangen werden kann, dass deutlich weniger als 60%

der Fläche GTs sind, ist hier nicht von einem Zufallseffekt auszugehen. Insofern kann hier von einer unteren Schranke der Accuracy gesprochen werden.
Für dieses Netzwerk kann man also von einer Accuracy zwischen 60% und 90% ausgehen. Der oben als Unzulänglichkeit interpretierte Effekt durch den Recall kann aber zum Vorteil genutzt werden: Die Kombination aus hoher Accuracy und hohem Recall spricht für den potentiellen Nutzen in der medizinischen Diagnostik, da viele GTs in vielen Bildern erkannt werden. Zwar wäre auch hier eine hohe Precision wünschenswert, aber nicht unbedingt notwendig.
Es sollte jedoch auch die Rolle der Baseline berücksichtigt werden, die auch als Hyperparameter verstanden werden kann. In dieser Arbeit wurde, wie von Sundararajan et al. [15] vorgeschlagen, das schwarze Bild als Baseline verwendet. Grundsätzlich sind jedoch auch andere Baselines vorstellbar; Nain [10] implementiert beispielsweise auch eine Funktion, die die Integrierten Gradienten zufällig erzeugter Baselines mittelt. Diese Varianten stellen jedoch äußerst unnatürliche Referenzen dar. Das kann Verzerrungen zur Folge haben: Man führe sich den Fall vor Augen, dass ein Bild schwarze Pixel an einer Stelle aufweist, die bei Lungen üblicherweise hell ist und auf eine Krankheit hinweist. Dies lässt im Extremfall IG_{ij} allein wegen dem Vorfaktor $(x_{ij} - y_{ij})$ sehr klein oder sogar 0 werden, obwohl möglicherweise das Integral groß ist. Ähnliche verzerrende Skalierungseffekte durch den Vorfaktor könnten auch bei zufälligen Baselines auftreten, wenn viele Pixel der Baseline für ihre Position untypische Werte annehmen. Daher könnte es für zukünftige Untersuchungen vorteilhaft sein, ein Röntgenbild einer gesunden Lunge mit möglichst niedrigem Score oder eine Kombination aus solchen Bildern als Baseline oder Baselines zu wählen.

5 Fazit und Ausblick

5.1 Fazit

In der vorliegenden Arbeit sollten die Integrierte-Gradienten-Methode als Ansatz zur Interpretation und Qualitätsüberprüfung von neuronalen Netzen vorgestellt und erprobt werden. Dieses Verfahren eignet sich wegen der axiomatischen und nicht-statistischen Herangehensweise besonders gut. Diese Analyse wurde an einem neuronalen Netzwerk, das Röntgenbilder der Lunge binär in „gesund“ und „krank“ klassifiziert, durchgeführt. In Kapitel 3 und 4 wurde gezeigt, wie die Integrierten Gradienten interpretiert und mithilfe eines Schwellenwerts dazu genutzt werden können, die für die Klassifikation besonders wichtigen Pixeln und Regionen zu extrahieren und so die Entscheidung des Netzwerks erklärbar zu machen.
In Kapitel 4 wurden drei Maße diskutiert, mit denen überprüft werden kann, ob das Netzwerk seine Entscheidungen auf die gewünschten Charakteristika der Bilder stützt. Der F_1-Score gibt Auskunft über die Objekterkennungsqualitäten des Netzwerks. Ein hoher Score ist zwar ein nicht notwendiges, aber hinreichendes Kriterium für eine gute Entscheidungsqualität, denn ein ideal trainiertes Netzwerk würde seine Entscheidungen genau vond von den relevanten Pixeln abhängig machen. Er ist jedoch für ein binär klassifizierendes Netzwerk ein sehr hartes Maß, da durch die Komponente Recall jede nicht vorhergesagte GT bestraft wird.
Die Kombination aus F_1-Score und Accuracy ist ein insofern besseres Maß, dass die Anzahl der Bilder gezählt wird, in denen mindestens eine GT korrekt vorhergesagt wird. Da der Recall dieses Kriterium nicht ideal wiedergibt, schmälert sich die Aussagekraft in der Hinsicht, dass dier Recall auf Kosten der Präzision überbewertet wird. Dennoch gibt die Accuracy darüber Auskunft, bei welchem Anteil der Bilder die GTs zur Klassifikation beigetragen haben.
Eine andere Version der Accuracy lässt sich berechnen, indem man pro Bild nur das Top-Pixel betrachtet und prüft, ob es in einer GT liegt. Zwar wird dadurch der in Häufungen von bedeutenden Pixeln innewohnenden Information verworfen, aber die durch den Recall entstehende Problematik entfällt. Man erhält so eine Art untere Schranke, wie stark das Netzwerk sich in seiner Klassifikation auf die GTs stützt.
Für das in dieser Arbeit verwendete Netzwerk ergibt sich eine Accuracy zwischen 60% und 90% mit einem moderaten F_1-Score. Die Kombination aus hoher Accuracy von 90% und hohem Recall macht trotz der niedrigen Precision den potentiellen Nutzen dieser Methode in der medizinischen Diagnostik deutlich.
Zusammengefasst lässt sich sagen, dass es möglich ist, mittels des Integrierten-Gradienten-Verfahrens für die Klassifikation durch das Netzwerk wichtige Charakteristika zu extrahieren und damit die Entscheidungsqualität von Netzwerken zu messen.

5.2 Ausblick

In dieser Arbeit konnten nur einige Ansätze zur Beurteilung eines Netzwerks durchgeführt werden. Es bleiben noch einige Fragen offen und Probleme zu lösen.
Ein Problem ist die durch den Recall verursachte Verzerrung. Ideal wäre, die Parameter durch die Maximierung einer Funktion zu identifzieren, die nur dann bestraft, wenn in einem Bild keine Ground Truth vorhergesagt wird. Ein sehr einfacher Ansatz wäre, den Recall durch die Funktion der Form $1 - {}^{k}/_{n}$, wobei n die Gesamtzahl der Bilder und k die Zahl der Bilder, in denen keine GT vorhergesagt wurde, bezeichnen. Auch diese Funktion hätte die Eigenschaft, im besten Fall 1, im schlechtesten Fall 0 und in der Anzahl der Richtig-positiven monoton steigend zu sein. Zudem könnte man die Gewichtung von Precision und Recall (oder seinem Ersatz) verändern. Beispielsweise könnte in der medizinischen Diagnostik der Recall wichtiger sein als die Precision, damit keine Auffälligkeiten zu übersehen werden.
In der vorliegenden Arbeit wurde eine niedrigschwellige Definition mit der Existenz eines Schnitts von zwei Boxen, die GTs beziehungsweise GTs umschließen, verwendet. Für eine genauere Analyse könnten andere Methoden untersucht werden, beispielsweise mit einer Mindestfläche des Schnitts oder Mindestanteil an der Gesamtfläche einer Box.
Mit Baselines wurde in dieser Arbeit nicht experimentiert. In dieser Arbeit wurde eine einfache, aber unnatürliche Baseline gewählt, die die Ergebnisse verzerren könnte. Es ist zu vermuten, dass eine natürliche Baseline, also eine Baseline, die typisch für eine Eingabe in das Netzwerk ist, bessere Ergebnisse hervorbringt als künstliche wie das schwarze Bild. Im konkreten Fall der Röntgenbilder könnten beispielsweise Mittellungen besonders gesunder und niedrig klassifizierter Röntgenbilder als Baseline verwendet werden.

Literatur

[1] Amina Adadi und Mohammed Berrada. „Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)“. In: *IEEE Access* 6 (2018), S. 52138–52160. DOI: 10.1109/ACCESS.2018.2870052.

[2] Robert J. Aumann und Lloyd S. Shapley. *Values of non-atomic games*. A Rand Corporation research study. Princeton, NJ: Princeton Univ. Press, 1974. ISBN: 0-691-08103-4. URL: http://www.loc.gov/catdir/enhancements/fy1504/72004038-d.html.

[3] Herbert Federer. *Geometric measure theory*. Reprint of the 1969 ed. Classics in mathematics. Berlin u. a.: Springer, 1996. ISBN: 9783540606567.

[4] Otto Forster. *Analysis 3: Maß- und Integrationstheorie, Integralsätze im IRn und Anwendungen*. 7., überarb. Aufl. 2012. Aufbaukurs Mathematik. Wiesbaden: Vieweg+Teubner Verlag, 2012. ISBN: 9783834823748. DOI: 10.1007/978-3-8348-2374-8. URL: http://gbv.eblib.com/patron/FullRecord.aspx?p=969971.

[5] Eric J. Friedman. „Paths and consistency in additive cost sharing“. In: *International Journal of Games Theory* 32.4 (2004). ISSN: 0020-7276. DOI: 10.1007/s001820400173.

[6] Ian Goodfellow, Yoshua Bengio und Aaron Courville. *Deep Learning. Das umfassende Handbuch: Grundlagen, aktuelle Verfahren und Algorithmen, neue Forschungsansätze*. 1st ed. mitp Professional. Frechen: MITP, 2018. ISBN: 9783958457010. URL: https://ebookcentral.proquest.com/lib/gbv/detail.action?docID=5598176.

[7] A. N. Kolmogorov und Fomin. S.V. *Introductory Real Analysis*. Dover Books on Mathematics, 1970.

[8] Miguel Lerma und Mirtha Lucas. *Symmetry-Preserving Paths in Integrated Gradients*. URL: http://arxiv.org/pdf/2103.13533v1.

[9] Johannes Michael. „Visuelle Aufmerksamkeitsmodelle basierend auf neuronalen Netzwerken“. Diss. Rostock: Univeristät Rostock, 2015. URL: https://doi.org/10.18453/rosdok_id00002317.

[10] Akash Kumar Nain. *Model interpretability with Integrated Gradients*. 2020. URL: https://keras.io/examples/vision/integrated_gradients/.

[11] Wojciech Samek u. a. „Evaluating the Visualization of What a Deep Neural Network Has Learned“. In: *IEEE transactions on neural networks and learning systems* 28.11 (2017), S. 2660–2673. DOI: 10.1109/TNNLS.2016.2599820.

[12] Dov Samet, Yair Taumann und Israel Zang. „An Application of the Aumann-Shapley Prices for Cost Allocation in Transportation Problems“. In: *Mathematics of Operations Research* 9.1 (1984), S. 25–42. URL: 1984.

[13] Avanti Shrikumar, Peyton Greenside und Anshul Kundaje. „Learning Important Features Through Propagating Activation Differences“. In: *PMLR 70:3145-3153* (2017). URL: http://arxiv.org/pdf/1704.02685v2.

[14] Karen Simonyan, Andrea Vedaldi und Andrew Zisserman. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2013. URL: http://arxiv.org/pdf/1312.6034v2.

[15] Mukund Sundararajan, Ankur Taly und Qiqi Yan. *Axiomatic Attribution for Deep Networks*. 2017. URL: http://arxiv.org/pdf/1703.01365v2.

[16] Christian Szegedy u. a. *Rethinking the Inception Architecture for Computer Vision*. 2015. URL: http://arxiv.org/pdf/1512.00567v3.

Thesen zur Bachelorarbeit

Interpretierbarkeit

Neuronaler Netzwerke

Johannes Schade

1.
2.
3.