Fachhochschule Nordwestschweiz
Hochschule für Technik

Bachelor Thesis

24FS_I4DS27: Adversarial Attacks
Wie kann KI überlistet werden?

Windisch, 20. Mai 2024

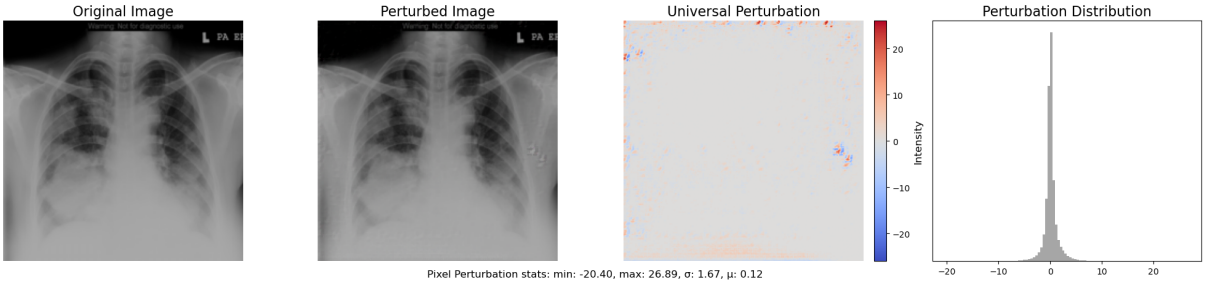

Abbildung 1: Dieses Bild soll noch ersetzt werden

Studenten	Si Ben Tran Gabriel Torres Gamez
Fachbetreuer	Daniel Perruchoud Stephan Heule
Auftraggeber	i4Ds
Projektnummer	24FS_I4DS

Fachhochschule Nordwestschweiz, Hochschule für Technik

Abstract

abstract

eigenleistung
uap alg

Vorwort und Dank

danke

Inhaltsverzeichnis

1 Einleitung

Deep Learning ist ein wesentliches Teilgebiet des maschinellen Lernens und der Künstlichen Intelligenz. In der heutigen Zeit nehmen Machine- und Deep Learning-Modelle eine immer zentralere Stellung in der Gesellschaft ein und werden als Schlüsseltechnologien der Industrie 4.0 angesehen. Die Fähigkeit, aus Daten zu lernen, ermöglicht diesen Modellen, in verschiedenen Anwendungsbereichen wie im Gesundheitswesen, der visuellen Erkennung, der Textanalyse und dem autonomen Fahren einen erheblichen Mehrwert zu generieren [1]. Trotz der faszinierenden Möglichkeit und Erfolge, die uns die Künstliche Intelligenz bringt, wurde eine beunruhigende Eigenschaft dieser Modelle festgestellt. Forscher haben herausgefunden, dass durch einfaches Hinzufügen von bestimmten kleine Störungen, beispielsweise in Form von Veränderungen eines Pixels in einem Bild, die Deep Learning Modelle extrem schlechte Ergebnisse liefern [2]. In der Bildklassifikation können selbst geringfügige Veränderungen vorgenommen werden, die für das menschliche Auge kaum wahrnehmbar sind. Diese können dazu führen, dass das Modell eine falsche Klassifikation voraussagt.[3].

“In einer Welt, wo ML & AI mehr und mehr Einfluss auf unser Leben insbesondere in sensitiven Bereichen wie Gesundheitswesen und Finanzen nimmt, sind die Implikationen von Adversarial Attacks von enormer Tragweite [3].“

2 Grundlagen, Stand der Forschung

2.1 Adversarial Attacks

Adversarial Attacks sind ein faszinierendes Phänomen im Bereich des Deep Learning, das in den letzten Jahren zunehmend an Bedeutung gewonnen hat. Diese Angriffe beziehen sich auf gezielte Manipulationen von Eingabedaten, die darauf abzielen, neuronale Netzwerke zu täuschen und falsche Vorhersagen zu erzwingen. Sie werfen wichtige Fragen zur Robustheit und Sicherheit von Deep Learning Modellen auf und haben weitreichende Implikationen für deren praktische Anwendungen.

Erste Paper mit Perturbationen z.B. Paper von Goodfellow

2.1.1 Grundlagen

2.1.2 Allgemeine Beispiele für Adversarial Attacks

2.1.3 Adversarial Attacks auf Bildklassifikation

2.2 Universal Adversarial Attacks auf Bildklassifikation

2.2.1 Grundlagen

2.2.2 Unser Hauptpaper

2.2.3 Bisherige Herausforderungen bei der Verteidigung

3 Daten

In unserer Arbeit konzentrieren wir uns primär auf zwei Datensätze, die jeweils medizinische Bilder von Patienten enthalten, die entweder an COVID-19 erkrankt sind (im COVIDx CXR-4 Datensatz) oder an Gehirntumoren leiden (im Brain Tumor Datensatz). Diese Datensätze weisen die Gemeinsamkeit auf, dass sie medizinische Bildinformationen enthalten.

3.1 COVID-19

COVID-19, verursacht durch das Coronavirus SARS-CoV-2, ist eine hochansteckende Atemwegserkrankung, die erstmals Ende 2019 identifiziert wurde. Die Symptome reichen von milden Anzeichen wie Husten und Fieber bis hin zu schweren Krankheitsverläufen, wie Lungenentzündung und akutem Atemnotsyndrom. Aufgrund der hohen Übertragbarkeit und der potenziell schweren Verläufe hatte die Pandemie weitreichende Auswirkungen auf das globale Gesundheitssystem, die Wirtschaft und das tägliche Leben der Menschen.

3.2 COVIDx CXR-4

COVIDx CXR-4 [4] ist ein öffentlicher Datensatz für COVID-19-Diagnostik mit Röntgenbildern, der 84,818 Bilder von 45,342 Patienten enthält. COVIDx CXR-4 ist, nach Kenntnisstand der Autoren, der grösste und vielfältigste öffentlich verfügbare COVID-19-Datensatz für Röntgenbilder und soll die Forschung unterstützen, um Klinikern im Kampf gegen COVID-19 zu helfen.

Patienten, mit einem negativen Befund erhalten das Klassenlabel Null, bei einem positiven Befund das Klassenlabel Eins. Diese Unterscheidung ist wichtig und relevant für unsere binäre Klassifikation. Die Röntgenbilder beschränken sich auf den Brustkorb des jeweiligen Menschen.

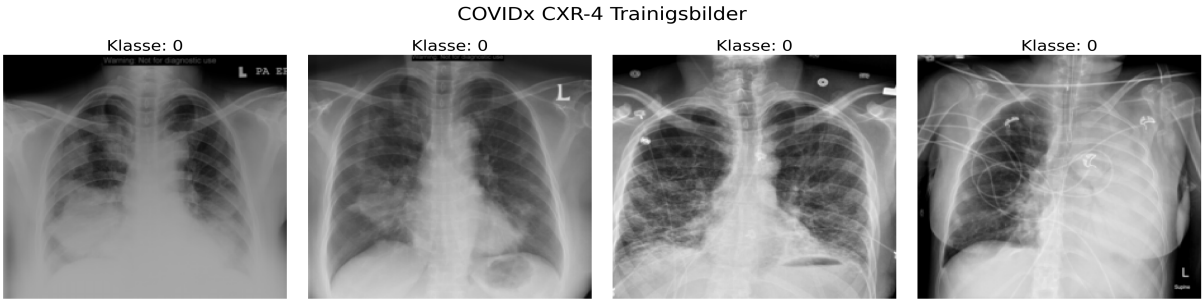

Abbildung 2: Beispiele von negativen Covid Patienten vom COVIDx CXR-4 Datensatz

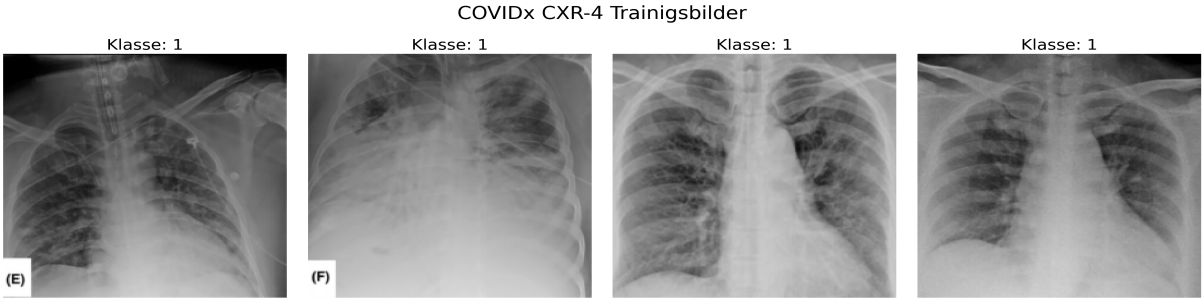

Abbildung 3: Beispiele von positiven Covid Patienten vom COVIDx CXR-4 Datensatz

3.2.1 Datenpartitionierung

Die Datenpartitionierung ist bereits durch die Struktur vorgegeben. Wir stellen fest, dass die Klassenverteilung von positiven und negativen Labels für die Validierung und den Testdatensets fast gleich verteilt ist, mit einem Verhältnis von 50% positive sowie 50% negative Fälle.

Partition	Anzahl Bilder		Klassenverteilung		Positiv-Verhältnis
	Absolut	Relativ	Positiv	Negativ	
Train	67863	0.8001	57199	10664	0.8429
Validation	8473	0.0999	4241	4232	0.5005
Test	8482	0.1000	4241	4241	0.5000

Tabelle 1: Klassenverteilung von COVIDX-CXR4

3.2.2 Datenexploration

In den dargestellten Histogrammen sehen wir die Pixelverteilung der Röntgenbilder von positiven 3 und negativen 2 Patienten. Jedes Histogramm repräsentiert die Intensitätsverteilung der Pixelwerte in den jeweiligen Röntgenaufnahmen. Die x-Achse jedes Histogramms zeigt die Grauwertintensität von 0 bis 255, während die y-Achse die Anzahl der Pixel für jede Intensität darstellt.

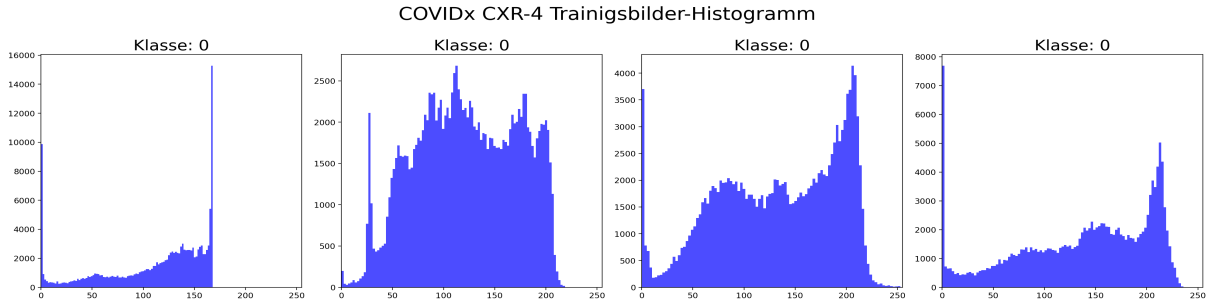

Abbildung 4: Histogramm der Pixelverteilung von Abbildung 2

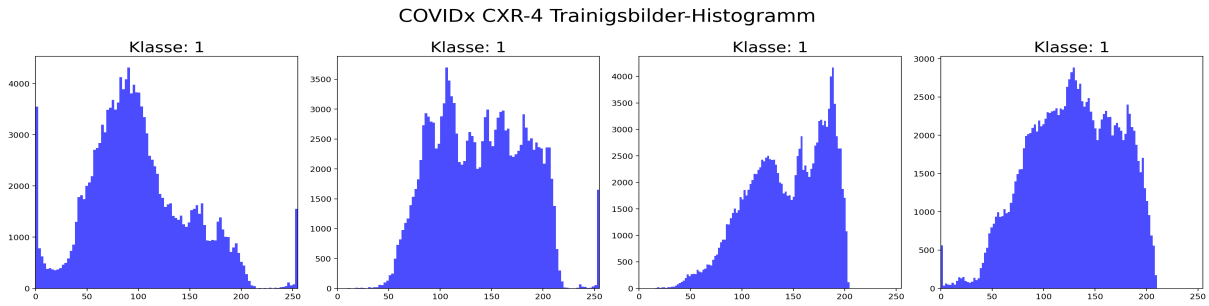

Abbildung 5: Histogramm der Pixelverteilung von Abbildung 3

Die Histogramme der Röntgenbilder variieren stark von Bild zu Bild. Tiefe Pixelwerte repräsentieren vermutlich Weichgewebe, während höhere Pixelwerte harte Knochengewebe darstellen

3.3 Gehirntumor

Gehirntumoren sind abnormale Wucherungen von Zellen im Gehirn, die entweder gutartig oder bösartig sein können. Diese Tumoren können sowohl im Gehirn selbst entstehen als auch als Metastasen von anderen Krebsarten in den Kopf wandern. Die Symptome variieren je nach Tumorart, -grösse und -lokalisation und können Kopfschmerzen, Übelkeit, Sehstörungen, Krampfanfälle und kognitive Beeinträchtigungen umfassen. Aufgrund ihrer Komplexität und der sensiblen Lage im zentralen Nervensystem stellen Gehirntumoren eine erhebliche Herausforderung für die medizinische Diagnostik und Behandlung dar und haben oft weitreichende Auswirkungen auf die Lebensqualität der Betroffenen.

3.4 Brain Tumor Datenset

Der **Brain Tumor Classification (MRI)-Datensatz** [5] umfasst 3.260 bereinigte und augmentierte, T1-gewichtete, kontrastverstärkte MRI-Bilder zur Identifikation und Klassifikation von Gehirntumoren.

Die MRI-Bilder in der Abbildung zeigen Beispiele von Gehirnscans, die zur Klassifikation von Gehirntumoren verwendet werden. Positive Fälle werden mit der Klasse 1 und negative Fälle mit der Klasse 0 gekennzeichnet. Eine genaue Beschreibung der Klassenzusammenfassung findet sich in Kapitel 3.5.4. In den Beispielabbildungen 6 und 7 erkennen wir, dass im Datensatz Saggital-, Frontal- und Transversalebenen des Kopfes enthalten sind.

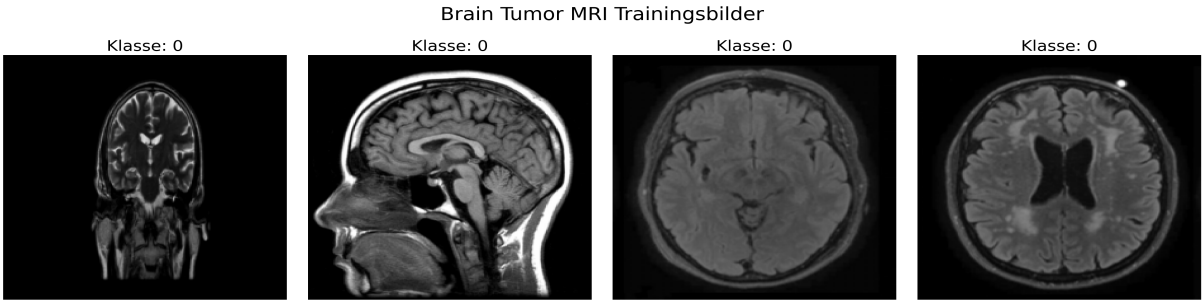

Abbildung 6: Beispiele von negativen Gehirntumor Patienten vom Brain Tumor Datensatz

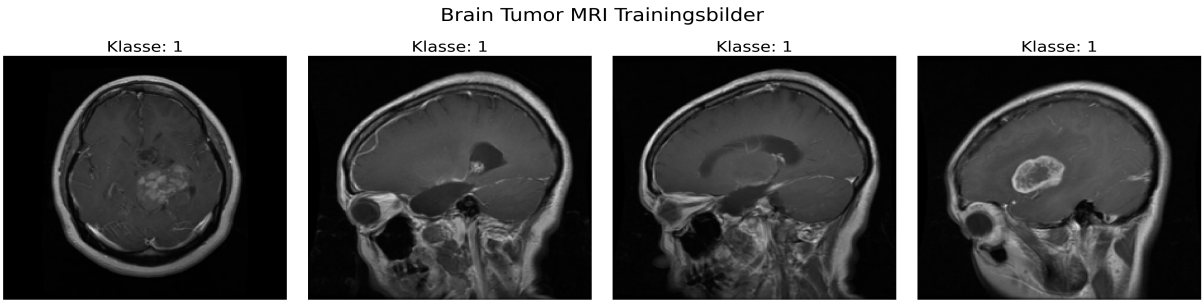

Abbildung 7: Beispiele von positiven Gehirntumor Patienten vom Brain Tumor Datensatz

3.5 Gehirntumor Arten im Datenset

Im Datensatz für Gehirntumoren gibt es vier Klassen. Drei davon sind positive Fälle, in denen Gehirntumoren vorhanden sind. Eine Klasse ist negativ und beinhaltet keine Gehirntumoren. Die drei positiven Klassen sind Gliome, Maligne Tumoren und Pituary.

3.5.1 Gliome

Gliome sind eine Gruppe von Hirntumoren, die im Glia-Gewebe (Gewebe im Nervensystem) entstehen. Sie sind die häufigsten primären Hirntumoren bei Erwachsenen. Gliome werden nach ihren Ursprungszellen in Astrozytome, Oligodendrogliome und Ependymome unterteilt. Die Symptome variieren je nach Lage und Grösse des Tumors und können Kopfschmerzen, Anfälle und neurologische Defizite umfassen. Die Behandlung erfolgt in der Regel durch eine Kombination von Operation, Strahlentherapie und Chemotherapie.

3.5.2 Malignent

Maligne Tumoren sind bösartige Wucherungen, die unkontrolliert wachsen und in umliegendes Gewebe eindringen können. Sie haben die Fähigkeit, Metastasen zu bilden, was bedeutet, dass sie sich auf andere Teile des Körpers ausbreiten können. Die Behandlung von malignen Tumoren umfasst in der Regel eine Kombination aus Operation, Strahlentherapie und Chemotherapie. Die Prognose hängt von der Art, dem Stadium und der Lage des Tumors ab.

3.5.3 Pituary

Hypophysenadenome sind gutartige Tumoren der Hypophyse, einer kleinen Drüse im Gehirn, die wichtige Hormone produziert. Obwohl sie meist nicht bösartig sind, können sie aufgrund ihrer Lage und Hormonproduktion erhebliche gesundheitliche Probleme verursachen. Symptome können Kopfschmerzen, Sehstörungen und hormonelle Ungleichgewichte umfassen. Die Behandlung umfasst in der Regel eine Operation und, in einigen Fällen, medikamentöse Therapie oder Strahlentherapie.

3.5.4 Datenpartitionierung

Der Datensatz enthält vier Klassen, drei davon sind Tumorklassen und eine Klasse ist kein Tumor. Die Verteilung der Klassen ist in der Tabelle 2 zu sehen. Ersichtlich ist, dass in dem Datensatz positive Tumorklassen deutlich mehr vertreten sind als keine Tumore. Da wir uns für unsere These auf die binäre Klassifikation stützen, fassen wir die drei Gehirntumoren Klassen Pituitary, Glioma, Meningioma zu einem positiven Gehirn Tumor Klasse und kein Tumor als negative Klasse und erhalten somit die Tabelle 3.

Partition	Klassenverteilung			
	Pituitary	Glioma	Meningioma	kein Tumor
Train	662	661	658	317
Validation	165	165	164	78
Test	74	100	115	105

Tabelle 2: Ursprüngliche Klassenaufteilung von Gehirntumoren

Die Verteilungsverhältnisse waren ursprünglich 70.4% Trainings- und 29.6% Testbilder. Da ein analoger Datensatz, wie beim COVIDx, einfacher zu handhaben ist, haben wir den ursprünglichen Testdatensatz weiter unterteilt, und zwar in 17.5% Validierungs- und 12.1% Testdaten. Bevor wir die positiven Klassen zusammengefasst haben, herrschte eine Klassenimbalance, die wir bei der Partitionierung in Train-, Validierung, und Testset mitberücksichtigt haben. Die Tabelle 3 zeigt die Anzahl an Bilder in absolut, relativ und die Klassenverteilung von positive, negative Tumorbilder für jede Datenpartition auf.

Partition	Anzahl Bilder		Klassenverteilung		Positiv-Verhältnis
	Absolut	Relativ	Positiv	Negativ	
Train	2298	0.704	1981	317	0.862
Validation	572	0.175	494	78	0.864
Test	394	0.121	289	105	0.736

Tabelle 3: Binäre Klassenverteilung von Gehirntumoren

3.5.5 Datenexploration

Die Histogramme der Hirntumor-Klassifikation sehen im Vergleich zum COVIDx CXR-4-Datensatz deutlich unterschiedlich aus. So ist klar in den Beispielen ersichtlich, dass die Intensität von niedrigeren Pixelwerten sehr hoch.

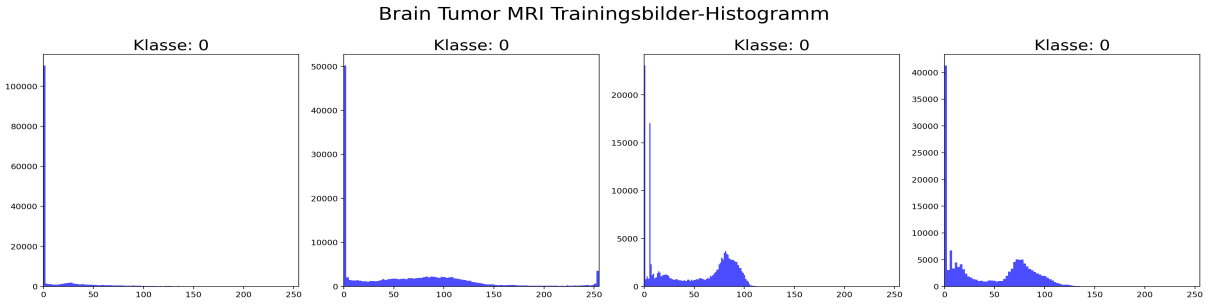

Abbildung 8: Histogramm der Pixelverteilung von Abbildung 6

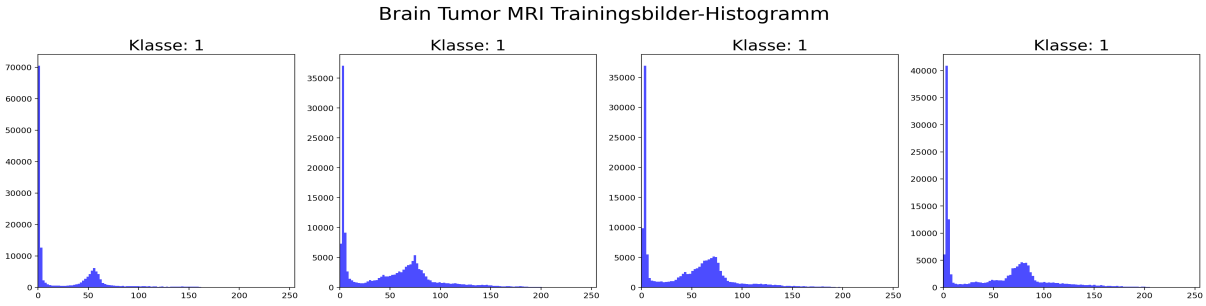

Abbildung 9: Histogramm der Pixelverteilung von Abbildung 7

Die niedrigen Pixelwerte in einem Bild deuten darauf hin, dass viele dunkle Bereiche vorhanden sind. Es liegt nahe, dass dies der Hintergrund der MRI-Bilder ist und somit einen Grossteil der Informationen ausmacht, die keine relevanten Details enthalten.

4 Methodik

In diesem Abschnitt widmen wir uns den angewandten Methoden unserer Arbeit und erläutern die verwendeten Modelle sowie ihre Metriken und Algorithmen.

Die Visualisierung 10 stellt unsere Arbeit einfach dar und gibt einen Überblick, um ein grundlegendes Verständnis dafür zu vermitteln, worum es in der Arbeit geht.

Abbildung 10: Simplifizierte Version unseres Vorgehens

Eine weiterführende, detailreichere Visualisierung ist 11, die die oben gezeigte Visualisierung weiter aufschlüsselt und genauer beschreibt.

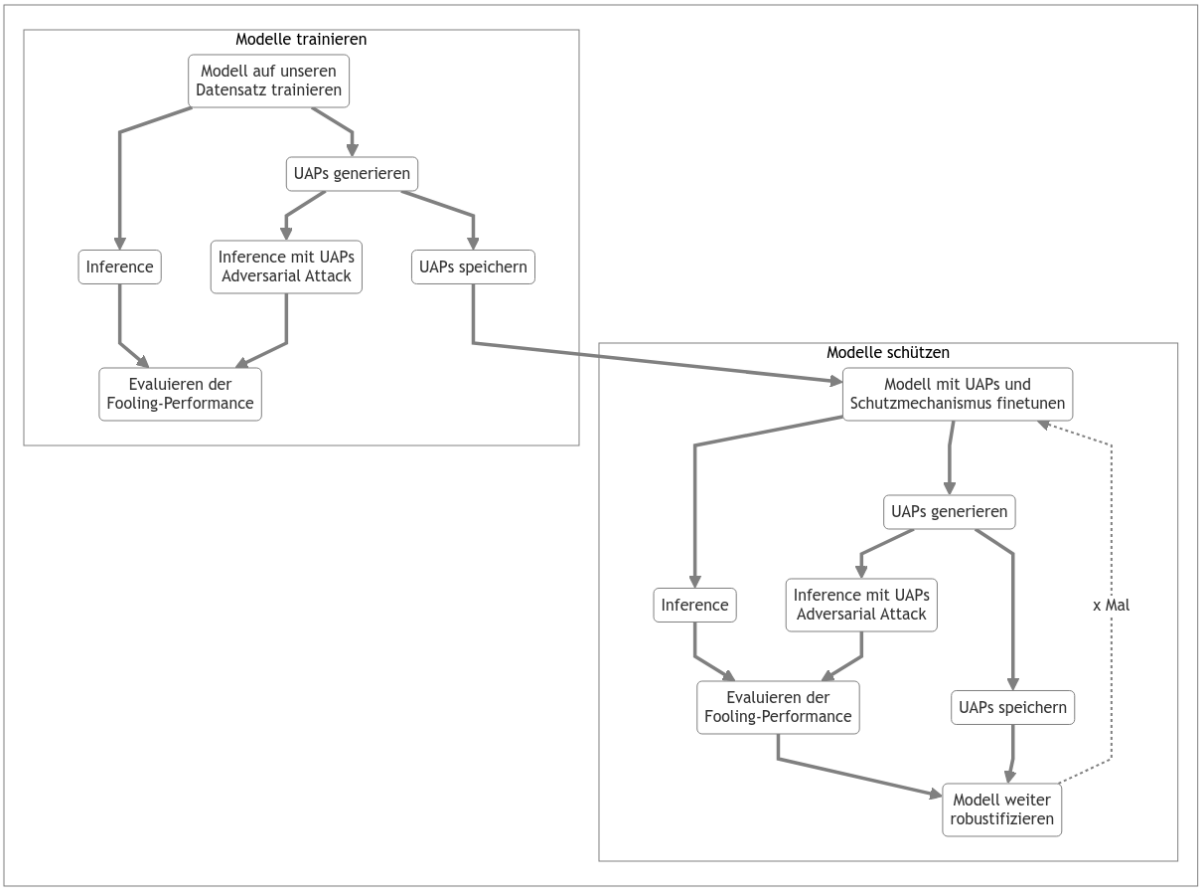

Abbildung 11: Methodik Übersicht

4.1 Technische Umsetzung

Für die technische Umsetzung unserer Bachelorarbeit haben wir auf bewährte und bekannten Data Science Frameworks zurückgegriffen. Darunter gehören: Numpy, Pandas, Matplotlib, & Seaborn. Für die Modelle sowie die Verarbeitung von Bildern nutzen wir PyTorch sowie die Erweiterung PyTorch-Lightning und tracken den Trainingsprozess sowie unsere Modelle in Weights & Bias.

4.2 Preprocessing

Der Preprocessing Schritt erlaubt es uns, die Rohdaten so zu transformieren, dass sie optimal für das Training oder die Verarbeitung durch unser Modell vorbereitet sind. In diesem Kapitel wird insbesondere auf zwei Arten von Preprocessing eingegangen: Preprocessing vor dem Training 4.2.1 und Preprocessing mit Perturbation 4.2.2.

4.2.1 Preprocessing vor dem Training

Alle Bilder werden vor dem Training durch eine Preprocessing Pipeline vorbereitet. Dabei werden die Bilder, zu einer Grösse von 224 Pixelhöhe und 224 Pixelbreite, mit Antialiasing verarbeitet.

Antialiasing ist eine Technik, die in der Computergrafik verwendet wird, um den Aliasing-Effekt zu entfernen. Der Aliasing-Effekt ist das Auftreten von gezackten Kanten oder „Jaggies“ in einem gerasterten Bild.

4.2.2 Preprocessing mit Perturbation

Die erzeugten Perturbationen werden durch eine elementweise Addition dem jeweiligen Eingangsbild hinzugefügt. Dadurch werden die Bilder manipuliert und für den Angriff auf die Modelle vorbereitet. Die Addition der Perturbationen kann durch das Erstellen eigener PyTorch Custom Preprocessing Klassen codiert werden. Dies erlaubt uns, diese einfach in die Pipeline hinzuzufügen.

4.3 Klassifikationsmodelle

In der Untersuchung der Universal Adversarial Perturbations wurden verschiedene Klassifikationsmodelle verwendet, um die Generalisierbarkeit über verschiedene Architekturen hinweg zu bewerten. Die ausgewählten Modelle umfassen eine breite Palette von Architekturen, darunter ResNet [6], DenseNet [7], EfficientNetV2 [8] und ein Transformer-basiertes Modell, der Vision Transformer (ViT) [9], welches durch die PyTorch Library zur Verfügung gestellt wird.

In jeder dieser Klassifikationsfamilien haben wir zwei bis drei verschiedene Modelle trainiert und evaluiert. Durch die Verwendung einer Vielzahl von Modellen stellen wir sicher, dass die Ergebnisse nicht spezifisch für eine bestimmte Architektur sind. Auf diese Weise können wir verallgemeinerbare Schlussfolgerungen über universelle adversarial Perturbationen ziehen.

In unserer Erarbeitung wurden Modelle, die zu keinem erfolgreichen Training geführt haben, aussortiert und nicht mehr in Betracht gezogen. Zwei der betroffenen Modellfamilien waren AlexNet [10] und VGG [11]. Die Gründe für ein misslungenes Training wurden in dieser Arbeit nicht weiterverfolgt.

4.3.1 ResNet

Das ResNet, auch bekannt als Residual Neural Network wurde Kaiming et al. [6] entwickelt und erreichte einen Fehler von 3.57% auf dem ImageNet Testdatensatz. Dieses Ergebnis führte zum Sieg der Klassifikationsaufgabe des ILSVRC 2015. Beim ResNet werden Residual Connections verwendet werden, um sogenannte „Skip Connections“ zwischen den Schichten des neuronalen Netzwerks zu etablieren. Diese Skip Connections ermöglichen es, den direkten Fluss von Informationen zwischen den Schichten zu erleichtern, was dazu beiträgt, das Problem des Verschwindens von Gradienten während des Trainings zu mildern.

Durch die Verwendung von Residual Connections können tiefe neuronale Netzwerke effektiver trainiert werden, da sie es ermöglichen, dass der Gradient leichter rückwärts durch das Netzwerk fliessen kann. Dies führt oft zu einer besseren Konvergenz und verhindert das Auftreten von Degradationsproblemen, die bei sehr tiefen neuronalen Netzwerken auftreten können.

Die Grundidee hinter Residual Connections ist es, die ursprüngliche Eingabe eines Blocks mit der Ausgabe desselben Blocks zu addieren. Dadurch wird eine Art „Shortcut“ geschaffen, der es dem Netzwerk ermöglicht, die Identität zu lernen, falls dies für die beste Leistung erforderlich ist. Dieses Konzept hat sich als äusserst effektiv erwiesen und hat dazu beigetragen, die Leistung von tiefen neuronalen Netzwerken erheblich zu verbessern.

4.3.2 DenseNet

Das DenseNet, kurz für Dense Convolutional Network, wurde von Huang et al. [7] eingeführt und stellt eine Weiterentwicklung von Residual Neural Networks (ResNets) dar. Im Gegensatz zu ResNets, die Residual Connections verwenden, um „Skip Connections“ zwischen Schichten herzustellen, verbindet DenseNet jede Schicht direkt mit allen nachfolgenden Schichten in einem Feedforward-Muster. Dieser dichte Verbindungsaufbau ermöglicht eine effizientere Informationsübertragung zwischen den Schichten und fördert die Wiederverwendung von Merkmalen durch das gesamte Netzwerk.

Durch die direkten Verbindungen zwischen allen Schichten kann DenseNet das Problem des Informationsverlusts während des Trainings reduzieren und die Gradientenflussstabilität verbessern. Dies führt oft zu einer besseren Nutzung der verfügbaren Daten und kann die Genauigkeit und Effizienz des Modells erhöhen.

Die grundlegende Idee hinter DenseNet besteht darin, dass jede Schicht nicht nur Ausgaben von vorherigen Schichten empfängt, sondern auch ihre eigenen Ausgaben an alle nachfolgenden Schichten weitergibt. Dieser Ansatz ermöglicht es dem Netzwerk, eine tiefe Hierarchie von Merkmalen zu lernen und dabei die Anzahl der zu optimierenden Parameter zu reduzieren, was zu kompakteren Modellen und besserer Generalisierung führt.

4.3.3 EfficientNetV2

EfficientNetV2 ist eine verbesserte Version des EfficientNet-Modells, das von Tan et al. [8] eingeführt wurde. Es basiert auf der Idee, ein optimales Gleichgewicht zwischen Modellgrösse und Leistung zu finden, indem eine skalierbare Compound Scaling-Strategie verwendet wird.

Im Gegensatz zu früheren Ansätzen zur Skalierung von CNNs, die sich hauptsächlich auf die Tiefe, Breite und Auflösung der Netzwerke konzentrierten, berücksichtigt EfficientNetV2 auch die Netzwerkstruktur und verwendet verbesserte Bausteine wie EffcientConv, eine effizientere Variante der Standard-Convolution. Diese Verbesserungen tragen dazu bei, die Leistung und Effizienz des Modells weiter zu steigern.

EfficientNetV2 hat gezeigt, dass es mit weniger Parametern als andere Modelle vergleichbare oder sogar bessere Leistungen auf verschiedenen Bilderkennungsaufgaben erreichen kann. Diese Effizienz macht es zu einer attraktiven Wahl für Ressourcenbeschränkte Umgebungen oder Anwendungen, bei denen schnelle Inferenzzeiten erforderlich sind.

4.3.4 ViT

Das Vision Transformer (ViT) ist ein kürzlich eingeführtes Modell für die Bildklassifizierung, das von Dosovitskiy et al. [9] vorgestellt wurde. Im Gegensatz zu traditionellen Convolutional Neural Networks (CNNs) verwendet ViT eine transformerbasierte Architektur, die ursprünglich für die Verarbeitung von Sequenzen in natürlicher Sprache entwickelt wurde.

ViT zerlegt das Eingabebild in Patches und behandelt sie als Token einer Sequenz, die dann von einem Transformer-Encoder verarbeitet wird. Diese Patch-Embedding-Technik ermöglicht es ViT, die räumlichen Beziehungen zwischen den Bildpixeln effektiv zu erfassen und sie in einen hochdimensionalen Raum zu projizieren, in dem sie von den Transformer-Blöcken verarbeitet werden.

Durch die Verwendung von Transformer-Architekturen kann ViT eine hohe Skalierbarkeit und Flexibilität aufweisen und ist in der Lage, komplexe Muster in Bildern zu erfassen, die für herkömmliche CNNs möglicherweise schwierig zu modellieren sind. Dies hat zu beeindruckenden Leistungen auf verschiedenen Bildklassifizierungsaufgaben geführt und zeigt das Potenzial von Transformer-Modellen für die visuelle Verarbeitung.

4.4 Metriken

Metriken spielen eine zentrale Rolle bei der Bewertung und Beurteilung der Leistung von Klassifikationsmodellen. In diesem Kapitel werden verschiedene Metriken vorgestellt, die zur Evaluierung von Klassifikationsmodellen verwendet werden. Des Weiteren stellen wir auch Metriken vor, die für uns von grosser Relevanz sind in Bezug auf Adversarial Attacks, wie bspw. die Fooling Rate im Kapitel 4.4.7

4.4.1 Konfusion Matrix

Die Konfusion Matrix ist ein Werkzeug, das verwendet wird, um die Vorhersagen eines Klassifikationsmodells mit der tatsächlichen Wahrheit zu vergleichen. Dabei werden die Vorhersagen des Modells in vier Kategorien unterteilt:

1. True Positives (TP): Elemente, die korrekterweise als zur Zielklasse gehörend erkannt wurden.
2. True Negatives (TN): Elemente, die korrekterweise als nicht zur Zielklasse gehörend erkannt wurden.
3. False Positives (FP): Elemente, die fälschlicherweise als zur Zielklasse gehörend vorhergesagt wurden, obwohl sie es in Wirklichkeit nicht sind.
4. False Negatives (FN): Elemente, die fälschlicherweise nicht als zur Zielklasse gehörend erkannt wurden, obwohl sie es eigentlich sind.

Nachfolgende die Darstellung einer Konfusion Matrix, mit der Anordnung von PyTorch Torchmetrics.

		Predicted label	
		Negativ	Positiv
Actual label	Negativ	TN	FP
	Positiv	FN	TP

Tabelle 4: Binäre Konfusionsmatrix

4.4.2 Precision

Precision (Präzision) misst das Verhältnis der korrekt identifizierten positiven Instanzen zu allen Instanzen, die vom Modell als positiv klassifiziert wurden. Mit anderen Worten, Precision gibt an, wie viele der als positiv identifizierten Fälle tatsächlich positiv sind. Die Formel für Precision ist:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

4.4.3 Recall

Recall hingegen misst das Verhältnis der korrekt identifizierten positiven Instanzen zu allen tatsächlich positiven Instanzen. Mit anderen Worten, Recall gibt an, wie viele der tatsächlich positiven Fälle vom Modell korrekt identifiziert wurden. Die Formel für Recall ist:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

4.4.4 Specificity

Specificity ist eine weitere wichtige Metrik zur Bewertung der Leistung von Klassifizierungsmodellen. Im Gegensatz zu Precision und Recall, die sich auf die Leistung des Modells bei der Identifizierung von positiven Instanzen konzentrieren, misst die Spezifität die Fähigkeit des Modells, negative Instanzen korrekt zu identifizieren.

Die Spezifität gibt das Verhältnis der korrekt identifizierten negativen Instanzen zur Gesamtzahl der tatsächlich negativen Instanzen an. Anders ausgedrückt, Spezifität gibt an, wie viele der tatsächlich negativen Fälle vom Modell korrekt identifiziert wurden. Die Formel für Spezifität ist:

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{3}$$

4.4.5 AUROC

Die AUROC (Area under the receiver operating characteristic curve) ist eine Metrik zur quantitativen Bewertung der Leistungsfähigkeit von Klassifizierungsmodelle. Die zugrundeliegende ROC Kurve (Receiver Operating Characteristic Curve) visualisiert das Verhältnis zwischen der True Positive Rate (TPR) und der False Positive Rate (FPR) eines Klassifizierungsmodells über verschiedene Schwellenwerte für die Klassifikation. Hierbei variiert die TPR als Prozentsatz der korrekt identifizierten positiven Fälle im Verhältnis zur Gesamtzahl der tatsächlich positiven Fälle, während die FPR den Anteil der fälschlicherweise als positiv klassifizierten Fälle im Verhältnis zur Gesamtzahl der tatsächlich negativen Fälle beschreibt.

$$\text{True Positive Rate (TPR)} = \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4}$$

$$\text{False Positive Rate (FPR)} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{5}$$

Die AUROC quantifiziert die Gesamtleistung des Klassifizierungsmodells, indem sie die Fläche unter der ROC Kurve berechnet. Ein Wert von 0.5 deutet auf ein schlechtes Modell hin, das äquivalent zu einer zufälligen Klassifikation ist, während ein Wert von 1 eine perfekte Klassifikation ohne Fehler bedeutet. Die AUROC ermöglicht somit eine quantitative Vergleichbarkeit verschiedener Klassifizierungsmodelle.

4.4.6 F1-Score

Der F1-Score ist eine Metrik, die Precision und Recall kombiniert, um die Gesamtleistung eines Klassifizierungsmodells zu bewerten. Er ist besonders nützlich, wenn ein ausgewogenes Verhältnis zwischen Precision und Recall angestrebt wird.

Der F1-Score wird als harmonisches Mittelwert von Precision und Recall berechnet und berücksichtigt sowohl falsch positive als auch falsch negative Vorhersagen. Die Formel für den F1-Score lautet:

$$\text{F1} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{6}$$

Durch die Berücksichtigung von Precision und Recall ermöglicht der F1-Score eine umfassendere Bewertung der Leistung eines Klassifizierungsmodells. Ein hoher F1-Score zeigt an, dass das

Modell sowohl eine hohe Precision als auch einen hohen Recall aufweist, was darauf hindeutet, dass es sowohl präzise als auch umfassend bei der Identifizierung von positiven Instanzen ist.

Der F1-Score ist besonders nützlich in Situationen, in denen ein Ungleichgewicht zwischen den Klassen vorliegt oder wenn sowohl Precision als auch Recall von Bedeutung sind, wie zum Beispiel bei der Erkennung von Betrug oder bei medizinischen Diagnosen.

4.4.7 Fooling Rate

Die Fooling Rate quantifiziert die Erfolgsrate von adversarial Angriffen auf ein Modell. Sie wird als Prozentsatz der Bilder definiert, die nach einer Modifikation anders klassifiziert werden als vorher. Eine Fooling Rate von 80% bedeutet beispielsweise, dass 80 von 100 modifizierten Bildern anders als die unveränderten Bilder klassifiziert werden. Eine hohe Fooling Rate zeigt eine hohe Vulnerabilität des Modells gegenüber den generierten adversarial Attacken an. Mathematisch wird die Fooling Rate folgendermassen definiert:

$$\text{Fooling Rate}(t) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{\{(\hat{y}_i \geq t) \neq (\hat{y}_{\text{adv},i} \geq t)\}} \tag{7}$$

wobei:

t, ist die für die Klassifikation gewählte Threshold.

$\hat{y}_i$, ist die Vorhersage des Modells für das Originalbild.

$\hat{y}_{\text{adv},i}$, ist die Vorhersage des Modells für das perturbierte Bild.

N, ist die Gesamtanzahl an Datenpunkte.

4.4.8 Matrizennorm

Vektoren repräsentieren nicht nur Richtungen, sondern auch Längen. Diese Längenmessung wird durch die sogenannte Norm ermöglicht. Für einen Vektor v wird die Norm wie folgt berechnet:

$$\|v\| = \sqrt{\sum_{i=1}^{n} v_i^2} \tag{8}$$

Matrizen sind eine Erweiterung dieses Konzepts. Ähnlich wie bei Vektoren können wir die „Grösse“ einer Matrix mit einer entsprechenden Norm messen. Die Norm einer Matrix A wird durch die Formel definiert:

$$\|A\|_P = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^p} \tag{9}$$

Hier steht p für einen Parameter, der die Art der Norm festlegt. Beispielsweise entspricht $p = 2$ der Frobeniusnorm. Wenn p gegen unendlich konvergiert, erhalten wir die Maximumnorm, die dem höchsten Wert in der Matrix entspricht:

$$\|A\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^{n} |a_{ij}| \tag{10}$$

4.5 Modell Training

In diesem Kapitel erklären wir, wie wir unsere Baseline Modelle trainiert haben, sowie alle Komponenten, welche wir verwendet haben.

4.5.1 Loss-Funktion

Für das Modelltraining verwenden wir den Binary Cross Entropy Loss. Die BCE Loss wird minimiert, wenn das Modell eine perfekte Vorhersage trifft (d.h. wenn $\hat{y}_i = y_i$ für alle i), und wird grösser, je weiter von entfernt ist).
Die BCE Loss ist definiert als:

$$L_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}(y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)) \tag{11}$$

folgende Parameter stehen für:

y_i, ist das tatsächliche Label des i-ten Datenpunkts
$\hat{y}_i$, ist die Vorhersage des Modells des i-ten Datenpunkts
N, ist die Gesamtanzahl an Datenpunkte

4.5.2 Optimizer

Für die Optimierung der Modellparameter verwenden wir den Adam Optimizer von Kingma et al. [12]. Adam ist ein Optimierungsalgorithmus, der für die Anpassung stochastischer Zielfunktionen mittels Gradientenverfahren entwickelt wurde. Er kombiniert Techniken wie Momentum und RMSprop, um die Lernrate anzupassen und die Konvergenz des Modells zu verbessern. Zudem ist es einfach zu implementieren, effizient und benötigt wenig Speicherplatz. Adam zeigt gute praktische Ergebnisse und ist mit anderen Methoden der stochastischen Optimierung vergleichbar.

Muss ich genau Zittieren oder Paraphrasieren, vorallem bei der nachfolgenden Zusammenfassung?

4.5.3 Hyperparamteroptimierung und Modellselektion

Für das Training iterieren wir durch alle Modelle und Datensätze, die in den Kapiteln 4.3: Klassifikationsmodelle und 3: Daten beschrieben sind. Aufgrund der Grösse des COVIDx CXR-4-Datensatzes im Vergleich zum Brain Tumor Datenset-Datensatz wählen wir pro Trainingsepoche nur zufällig 15% seines Trainingssubsets aus. Für jede Kombination aus Modell und Datensatz starten wir drei Trainingsdurchläufe, wobei wir die Learningrate von 10^{-5}, 10^{-4} bis 10^{-3} variieren.

Wegen des hohen Rechenaufwands und begrenzter Ressourcen verzichten wir auf Hyperparameteroptimierung bei L2-Regularisierung und Dropout vor dem letzten Klassifikationslayer und deaktivieren diese. Stattdessen speichern wir den besten Checkpoint jedes Modells und Datensatzes, basierend auf dem niedrigsten Validierungsverlust, und setzen diese als unsere Baseline-Klassifikationsmodelle ein.

4.6 UAP Algorithmus

Unser Algorithmus für Universal Adversarial Perturbation (UAP) basiert auf dem Ansatz von Moosavi-Dezfooli et al. [13]. Ziel ist es, eine Perturbation zu entwickeln, die auf neue Bilder übertragbar ist. Dies erreichen wir, indem wir iterativ durch eine Auswahl von Trainingsbildern gehen und den jeweils kürzesten Weg zur Entscheidungsgrenze jedes Bildes finden. Die dabei erzeugten Perturbationen werden summiert, um eine universelle Perturbation zu formen.

4.6.1 Loss-Funktion

Das Optimierungsproblem wird in unserer Implementierung durch eine selbst definierte Loss-Funktion gelöst, welche die Norm der Perturbationsmatrix und die inverse Binary Cross Entropy Funktion minimiert. Die Grundidee hierbei ist, dass mit der Norm die aktuelle Perturbation so klein wie möglich gehalten wird und mit der Inversen der Binary Cross Entropy Funktion die aktuelle Perturbation an die Entscheidungsgrenze gebracht wird.

Unsere Verlustfunktion optimiert den Tensor Δv und sieht wie folgt aus:

$$L_{UAP} = \lambda_{norm} \cdot ||v + \Delta v||_p + \frac{1}{L_{BCE}(\hat{y}, \hat{y}_{\text{adv}}) + \epsilon} \tag{12}$$

Wobei der hier verwendete Binary Cross Entropy wie folgt minimiert werden kann:

$$L_{BCE} = -(\hat{y} \cdot \log(\hat{y}_{\text{adv}}) + (1 - \hat{y}) \cdot \log(1 - \hat{y}_{\text{adv}})) \tag{13}$$

wobei:

L_{UAP}, ist der gesamte Loss.

L_{BCE}, bezeichnet den Binary Cross Entropy Loss.

λ_{norm}, ist der Regularisierungsparameter.

v, ist die universelle gegnerische Störung.

Δv, ist die Änderung der Störung v.

$||v + \Delta v||_p$, repräsentiert die L_p-Norm der Perturbation $v + \Delta v$.

$\hat{y}$, ist die Vorhersage des Modells für das Originalbild (img).

$\hat{y}_{adv}$, ist die Vorhersage des Modells für das perturbierte Bild (img $+ v + \Delta v$).

ϵ, ist eine kleine positive Konstante für numerische Stabilität.

Da unser Algorithmus eine Batch Size von 1 erfordert und nach jeder Berechnung die Backpropagation durchführt, muss bei der Loss-Funktion kein Mittelwert berechnet werden. $\hat{y}$ sowie $\hat{y}_{adv}$ beziehen sich auf das aktuelle Bild im Algorithmus.

Für ϵ wird ein kleiner Wert wie 10^{-6} gewählt, der eine Division durch null verhindert, wenn der Output unseres Modells sowohl mit als auch ohne Störung gleich ist. Dies ist vor allem bei der Berechnung der allerersten Perturbation ein Problem.

4.6.2 Technische UAP Umsetzung

Die technische Umsetzung des Prozesses zur Generierung der UAP Bilder ist in der folgenden Grafik 12 dargestellt:

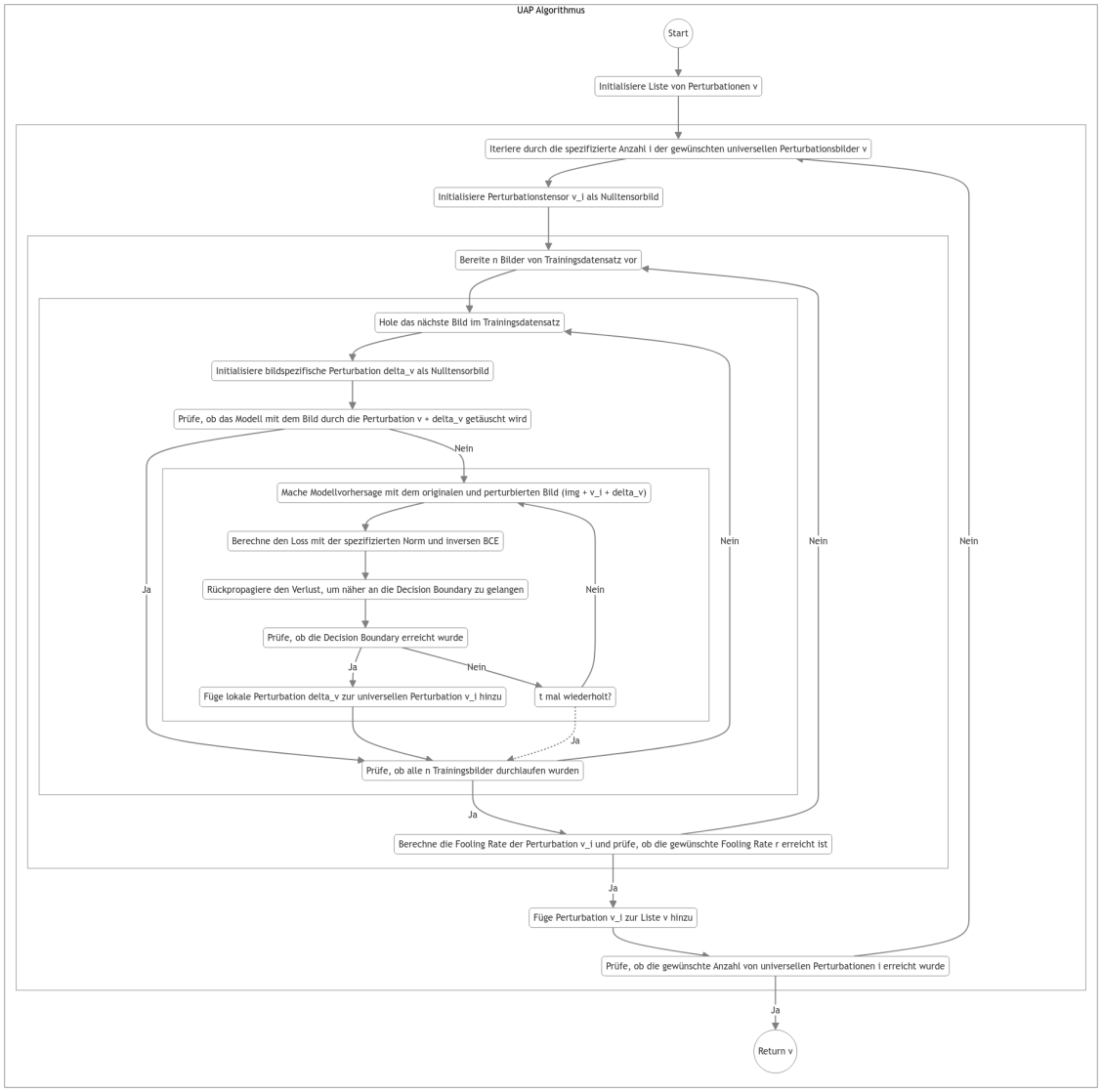

Abbildung 12: UAP Algorithmus

Wobei man folgende Parameter selbst bestimmen muss:

- i, ist die Anzahl Perturbationsbilder, welche generiert werden sollen.
- n, ist die Anzahl Trainingsbilder, welche für die Generierung verwendet werden.
- t, ist die Anzahl Versuche, eine bildlokale Perturbation zu finden
- r, ist der prozentuelle Anteil von Trainingsbilder, welche durch die Perturbationen getäuscht werden soll, damit die Perturbation v_i gespeichert wird.
- p, ist der Normparameter der L_p Norm.
- λ_{norm}, siehe Loss-Funktion.
- ϵ, siehe Loss-Funktion.

4.7 Schutzmechanismen

Im Kapitel Schutzmechanismen befassen wir uns mit dem Thema, wie man unsere anfällige trainierten Modelle aus Kapitel 4.5 fine tunen bzw. Schutzmechanismen anwenden, um das Modell robuster gegenüber unsere adversarial Training machen.

4.7.1 Data Augmentation

4.7.2 Input Ensembles

4.7.3 Adversarial Training

Adversarial; Training beschreiben die wir verwenden, um unsere Modelle stabiler zu machen

4.7.4 Übersicht Modell Finetuning

In der Abbildung 13 ist der Iterationsschritt der Pipeline zu sehen. Die Pipeline wird dabei x-mal durchlaufen, was bedeutet, dass die Robustifizierung ebenfalls x-mal für das Modell und Datensatz durchgeführt wird.

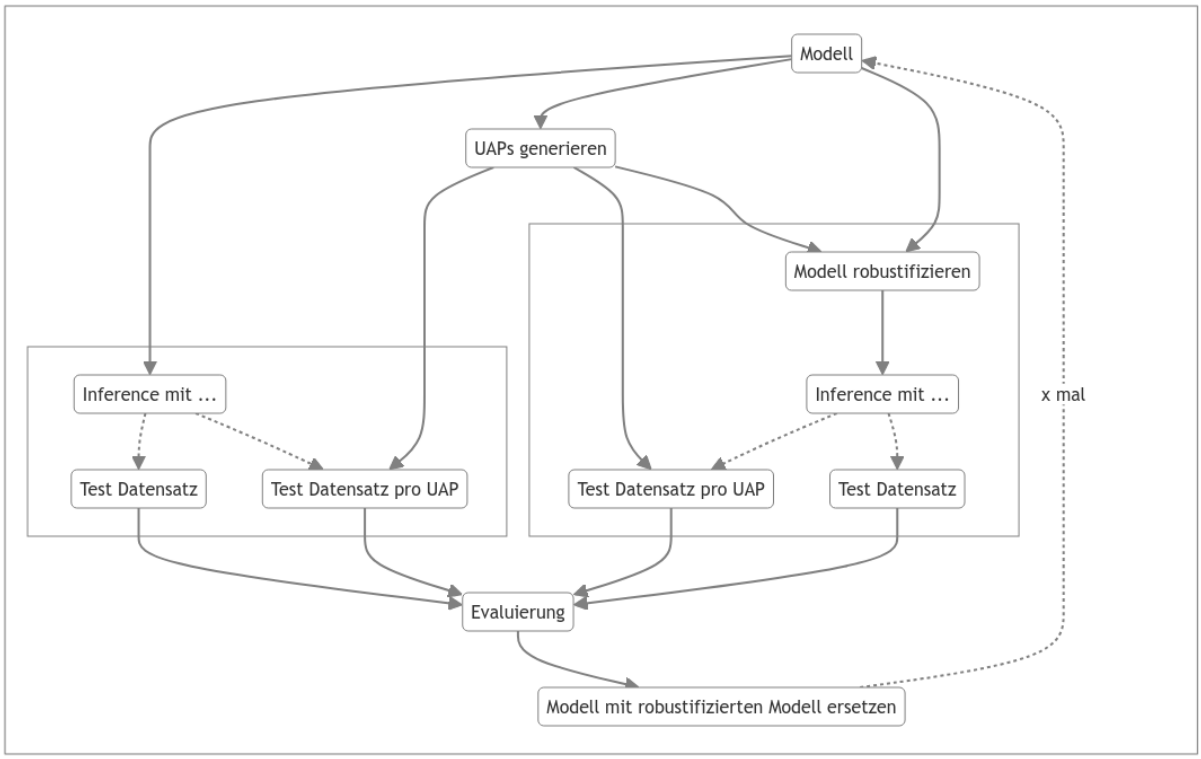

Abbildung 13: Übersicht der Evaluierungspipeline

- **Modell**: Startpunkt der Pipeline, unser trainiertes Modell auf unserem Datensatz.
- **UAPs generieren**: UAP werden generiert, um das Modell robuster zu machen.
- **Modell robustifizieren**: Adversarial Training wird angewendet, um das Modell zu robustifizieren. Siehe Kapitel **??**.
- **Inference mit...**: Erfolgt über zwei parallele Wege:
  - Einer führt zur Inferenz mit dem ursprünglichen Modell, jeweils für den Testdatensatz mit und ohne UAP.

  - Der andere Weg führt zur Inferenz mit dem robustifizierten Modell, ebenfalls für Testdatensatz mit und ohne UAP.

- **Evaluierung**: Im Evaluierungsschritt werden vier Pakete durch die Inferenz erzeugt und miteinander verglichen.

5 Resultate

Welche Grafiken sind sinnvoll

Unsicherheit und Straeungen

5.1 Ergebnisse des Modelltrainings

5.2 Anfälligkeit von ungeschützten Modellen

5.3 Anfälligkeit von geschützten Modellen

5.3.1 Data Augmentation

5.3.2 Adversarial Training

5.3.3 Input Ensembles

5.3.4 Weitere Verteidigungsmechanismen

Detektion von Adversarial Bilder durch Bildanalysen, wie Fouriertransformation.

6 Diskussion und Ausblick

6.1 Multiklassifkation

Notiz: Diskussion zur Loss Funktion und Algorithmus für Multiklassifkation statt Binary Cross Entropy können wir für Multiklassifkationen die Cross Entropy berücksichtigen. Für Prediction, mit 0.5 Tresholding, kann man bei Multiklassifkationen Softmax nehmen und anschliessend Argmax.

7 Glossar

Damit die Wörter auftauchen im Glossar oder Akronyme, müssen dabei folgende befehle ausgeführt werden:

\Gls{algorithmus} Algorithmus
\gls{algorithmus} algorithmus
\Glspl{algorithmus} Algorithmuss

\acrlong{uap} Universal Adversarial Perturbation
\acrshort{uap} UAP
\acrfull{bce} Binary Cross Entropy (BCE)

Glossar

algorithmus Eine präzise Anweisung oder eine Folge von Anweisungen zur Lösung eines Problems oder zur Ausführung einer Aufgabe in endlicher Zeit. 27

Akronyme

BCE Binary Cross Entropy. 20, 21, 26, 27

CE Cross Entropy. 26

UAP Universal Adversarial Perturbation. 21–24, 27

8 Literatur

[1] I. H. Sarker, "Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions", *SN Computer Science*, Jg. 2, Nr. 6, S. 420, 18. Aug. 2021, ISSN: 2661-8907. DOI: 10.1007/s42979-021-00815-1. Adresse: https://doi.org/10.1007/s42979-021-00815-1 (besucht am 26. Feb. 2024).

[2] C. Szegedy, W. Zaremba, I. Sutskever u. a., *Intriguing properties of neural networks*, 19. Feb. 2014. arXiv: 1312.6199[cs]. Adresse: http://arxiv.org/abs/1312.6199 (besucht am 21. Feb. 2024).

[3] D. Perruchoud und S. Heule, *24FS_I4DS27: Adversarial Attacks - Wie kann KI überlistet wer- den?*, 12. Nov. 2023.

[4] Y. Wu, H. Gunraj, C.-e. A. Tai und A. Wong, *COVIDx CXR-4: An Expanded Multi-Institutional Open-Source Benchmark Dataset for Chest X-ray Image-Based Computer-Aided COVID-19 Diagnostics*, 29. Nov. 2023. arXiv: 2311.17677[cs,eess]. Adresse: http://arxiv.org/abs/2311.17677 (besucht am 14. März 2024).

[5] S. Bhuvaji, A. Kadam, P. Bhumkar, S. Dedge und S. Kanchan, *Brain Tumor Classification (MRI)*, 2020. DOI: 10.34740/KAGGLE/DSV/1183165. Adresse: https://www.kaggle.com/dsv/1183165.

[6] K. He, X. Zhang, S. Ren und J. Sun, *Deep Residual Learning for Image Recognition*, 10. Dez. 2015. arXiv: 1512.03385[cs]. Adresse: http://arxiv.org/abs/1512.03385 (besucht am 14. März 2024).

[7] G. Huang, Z. Liu, L. van der Maaten und K. Q. Weinberger, *Densely Connected Convolutional Networks*, 28. Jan. 2018. arXiv: 1608.06993[cs]. Adresse: http://arxiv.org/abs/1608.06993 (besucht am 14. März 2024).

[8] M. Tan und Q. V. Le, *EfficientNetV2: Smaller Models and Faster Training*, 23. Juni 2021. DOI: 10.48550/arXiv.2104.00298. arXiv: 2104.00298[cs]. Adresse: http://arxiv.org/abs/2104.00298 (besucht am 19. März 2024).

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov u. a., *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*, 3. Juni 2021. DOI: 10.48550/arXiv.2010.11929. arXiv: 2010.11929[cs]. Adresse: http://arxiv.org/abs/2010.11929 (besucht am 19. März 2024).

[10] A. Krizhevsky, I. Sutskever und G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", in *Advances in Neural Information Processing Systems*, Bd. 25, Curran Associates, Inc., 2012. Adresse: https://papers.nips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html (besucht am 14. März 2024).

[11] K. Simonyan und A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, 10. Apr. 2015. arXiv: 1409.1556[cs]. Adresse: http://arxiv.org/abs/1409.1556 (besucht am 14. März 2024).

[12] D. P. Kingma und J. Ba, *Adam: A Method for Stochastic Optimization*, 29. Jan. 2017. DOI: 10.48550/arXiv.1412.6980. arXiv: 1412.6980[cs]. Adresse: http://arxiv.org/abs/1412.6980 (besucht am 6. Mai 2024).

[13] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi und P. Frossard, *Universal adversarial perturbations*, 9. März 2017. arXiv: 1610.08401[cs,stat]. Adresse: http://arxiv.org/abs/1610.08401 (besucht am 7. März 2024).

9 Anhang

- **GitHub Repositories**
  https://github.com/AdversarialAttacks
- **Weights & Bias**
  https://wandb.ai/24FS_I4DS27
- **UAP Algorithmus**
  UAP Algorithmus

Ehrlichkeitserklärung

wir erklären hiermit, dass wir den vorliegenden Leistungsnachweis selber und selbständig verfasst haben,

- dass wir sämtliche nicht von uns selber stammenden Textstellen und anderen Quellen wie Bilder etc. gemäss gängigen wissenschaftlichen Zitierregeln[1] korrekt zitiert und die verwendeten Quellen klar sichtbar ausgewiesen haben;
- dass wir in einer Fussnote oder einem Hilfsmittelverzeichnis alle verwendeten Hilfsmittel (KI-Assistenzsysteme wie Chatbots[2], Übersetzungs-[3] Paraphrasier-[4] oder Programmierapplikationen[5]) deklariert und ihre Verwendung bei den entsprechenden Textstellen angegeben haben;
- dass wir sämtliche immateriellen Rechte an von uns allfällig verwendeten Materialien wie Bilder oder Grafiken erworben haben oder dass diese Materialien von uns selbst erstellt wurden;
- dass das Thema, die Arbeit oder Teile davon nicht bei einem Leistungsnachweis eines anderen Moduls verwendet wurden, sofern dies nicht ausdrücklich mit der Dozentin oder dem Dozenten im Voraus vereinbart wurde und in der Arbeit ausgewiesen wird;
- dass wir uns bewusst sind, dass unsere Arbeit auf Plagiate und auf Drittautorschaft menschlichen oder technischen Ursprungs (Künstliche Intelligenz) überprüft werden kann;
- dass wir uns bewusst sind, dass die Hochschule für Technik FHNW einen Verstoss gegen diese Eigenständigkeitserklärung bzw. die ihr zugrundeliegenden Studierendenpflichten der Studien- und Prüfungsordnung der Hochschule für Technik verfolgt und dass daraus disziplinarische Folgen (Verweis oder Ausschluss aus dem Studiengang) resultieren können.

Si Ben Tran	Datum
Gabriel Torres Gamez	Datum

[1] z.B. APA oder IEEE
[2] z.B. ChatGPT
[3] z.B. Deepl
[4] z.B. Quillbot
[5] z.B. Github Copilot