Fakultät Ingenieurwissenschaften
Labor für Kooperative, automatisierte Verkehrssysteme (KAV)

Verteidigungsmaßnahmen gegen Modell-Inversionsangriffe

Bachelorarbeit

von

Hannes Weber

Aschaffenburg, 7. Dezember 2023

Autor:

Hannes Weber
Am Lindenbrunnen 17
D-97846 Partenstein

Matrikel-Nr.: 2220472
Studiengang Software Design (Bachelor)

Prüfer:

Prof. Dr.-Ing. Konrad Doll
Kooperative, automatisierte Verkehrssysteme (KAV)

Zweitprüfer:

Prof. Dr.-Ing. Ulrich Brunsmann

Technische Hochschule Aschaffenburg
Fakultät Ingenieurwissenschaften
Würzburger Straße 45
D-63743 Aschaffenburg

Ehrenwörtliche Erklärung

Hannes Weber

Am Lindenbrunnen 17
D-97846 Partenstein

Hiermit erkläre ich, dass ich die von mir vorgelegte Arbeit mit dem Thema „*Verteidigungsmaßnahmen gegen Modell-Inversionsangriffe*“ selbstständig verfasst habe, dass ich die verwendeten Quellen, Internet-Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen und Bildern –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Aschaffenburg, den 7. Dezember 2023

Hannes Weber

Danksagung

Hiermit möchte ich mich ausdrücklich bei Herrn Prof. Dr. Konrad Doll und Herrn Prof. Dr. Ulrich Brunsmann für die Unterstützung während meiner Arbeit bedanken. Des Weiteren ...

Inhaltsverzeichnis

Abkürzungen

ML	*Maschinelles Lernen*
CNN	*Convolutional Neural Network*
DCGAN	Deep Convolutional Generative Adversarial Network
KNN	*Künstlich-Neuronales Netzwerk*
NN	*Neuronales Netzwerk, eng.: Neural Network*

Kapitel 1

1 Einleitung

1.1 Motivation

Inmitten der fortscheitenden Digitalisierung greift man immer häufiger auf Lösungen zurück, die diverse Aufgaben in Arbeits- und Lebensbereich unterstützen, vereinfachen und sogar ersetzen. Von Medizin, in der beispielsweise Diagnoseverfahren zur frühzeitigen Erkennung von Krankheiten eingesetzt werden, bis zur Automobilindustrie, die schon jetzt auf selbstfahrende Kraftfahrzeuge setzt. Durch all die neuen Entwicklungen und Einsatzgebiete von künstlichen Intelligenzen gewinnt diese unaufhaltsam an Bedeutung.

Trotz des enormen Potenzials, das diese Technologien in ihren jeweiligen Anwendungsgebieten mit sich bringen, treten vermehrt Herausforderungen hinsichtlich Sicherheit und Robustheit der Systeme auf. Ein zentraler Aspekt ist die Notwendigkeit der Risikominimierung durch das Sicherstellen der Vertrauenswürdigkeit von KI-Systemen. Ein weiterer wichtiger Punkt in der Entwicklung dieser Systeme ist die Robustheit gegenüber Angriffen. KI-Systeme können anfällig für Manipulationen und gezielte Angriffe von außen oder durch Fehlkonfigurationen sein. Dabei spielt die Implementierung und Integration von bestimmten Sicherheitsmechanismen eine entscheidende Rolle, um die Integrität der Systeme zu gewährleisten. Insgesamt ist die Sicherheit im Bereich des maschinellen Lernens von zentraler Bedeutung, wordurch das Vertrauen der Nutzer gestärkt und die breite Integration dieser Technologien in verschiedener Bereichen vorangetrieben wird. Diese Arbeit legt den Fokus auf die Herausforderung der Verdeutlichung von Sicherheits- und Robustheitsimplementierungen in Bild-Klassifikationsmodellen. In einem globalen System, in dem immer mehr Aspekte des täglichen Lebens in die „Hände“ von KI-Systemen gegeben werden, ist es umso wichtiger, diese mit Anbetracht auf Sicherheit zu implementieren, wie auch zu überwachen. Eine Unsicherheit eines Systems kann hierbei in schlimmen Fällen sogar zu Verletzungen der Privatsphäre einzelner Personen führen. Daher wird diese Arbeit gesondert Angriffsvektoren und Bedrohungen von Klassifikationsmodellen behandeln. Darüber hinaus werden innovative Ansätze zur Bekämpfung möglicher Angriffsvektoren und Schwachstellen aufgezeigt, die sowohl Sicherheit, als auch Robustheit von neuronalen Netzen erhöhen. Insgesamt soll diese Arbeit dazu beitragen, dem Leser ein grundlegendes Verständis über Angriffe, deren Auswirkungen, wie auch Verteidigungsmaßnahmen zu liefern.

1.2 Aufgabenstellung

Hintergrund

Da KI-Systeme immer mehr Anwendung finden, werden diese häufiger Ziele von Cyber-Angriffen. Es gibt viele verschiedene Angriffsvektoren auf diese Systeme, zu denen auch das Erlangen von zugrundeliegenden Informationen über Daten des Trainingsprozesses von Neuronalen Netwerken gehört. Da viele Modelle mit sensiblen Daten trainiert werden, ist es wichtig, dass ein potentieller Angreifer durch das Modell keinen Zugruff auf Daten erlangen kann. Der „EU-AI Act“ enthält unter anderem Anforderungen an KI-Systeme in Bezug auf Robustheit und Cybersicherheit. Daher bringt dieser nicht nur die Verantwortung sichere Modelle zu trainieren mit sich, sondern auch die Sicherstellung, dass genutzte Trainingsdaten privat gehalten werden. Um Modelle während der Bereitstellung abzusichern, müssen die verschiedenen Angriffsvektoren und die entsprechenden Abwehrmaßnahmen bekannt sein.

Ziel der Arbeit

Während der Thesis sollen folgende Fragen beantwortet werden:

- Welche Angriffsmöglichkeiten gibt es, um Daten von deployten Modellen zu extrahieren?
- Wie kann man sich gegen diese Attacken schützen?
- Showcase zu Verteidigungstechniken und deren Effektivität gegenüber Inversions-Angriffen.

Methodischer Ansatz

- Onboarding
- Recherche über verschiedene Attacken und Verteidigungsmöglichkeiten
- Vergleich von verschiedenen Angriffen auf unterschiedliche Modell-Architekturen
- Implementierung eines Showcases für mindestens einen Angriff auf mindestens eine Modell-Architektur:
  - Zeigen, wie ein solcher Angriff funktioniert.
  - Kann man sich gegen einen solchen Angriff verteidigen?
  - Wie effektiv sind die Verteidigungsstrategien?

1.3 Aufbau der Arbeit

Die Arbeit unterteilt sich in drei Hauptbestandteile. Dazu gehört zum einen der SStand der Technik", worin aktuelle Technologien und Modelle dargestellt werden. Hauptsächlich wurde der Wissenstand aus anderen Papern entnommen und spiegelt somit den "ResearchingTeil der Arbeit wieder. ....

Eine weitere Hauptkomponente der Arbeit ist die Implementierung eines Modells und dessen Angriffsvektoren. Mit der Implementierung der möglichen Verteidigungsmaßnahmen der Schwachstellen wird die Kehrseite aufgezeigt, mit der eine mögliche Absicherung stattfinden kann. ...

Durch die Darstellung und Auswertung der herausgefundenen Ergebnisse, geht die Arbeit zu Ende. Dabei werden die wichtigsten Erkenntnisse aus den Implementierungen und Researching-Tasks zusammengefasst, und anschaulich dargestellt. Dies soll den Vorteil mit sich bringen, dem Leser einen Überblick des neu erweorbenen und eine finale Aussicht zu bieten. ...

Der Aufbau der Arbeit lehnt sich an die Struktur von **Bar-Shalom** an.

Kapitel 2

2 Grundlagen

2.1 Maschinelles Lernen

Ein Teilgebiet der künstlichen Intelligenzen ist der Bereich des Maschinellen Lernens, welcher die Verarbeitung von Daten addressiert, um beispielsweise Vorhersagen in Situationen auf Basis von bestimmten Informationen zu treffen. Im Jahr 1959 wurde dieser von Arthur Samuel geprägt, der die Herausforderung annahm, ein Schachspiel mit Hilfe einer Maschine zu lösen. (Joshi 2020, S. 4) Das besondere hierbei ist das Erlernen des richtigen Lösungswegs, welcher nicht durch bestimmte Bedingungen gesteuert, sondern auf Basis verschiedener Informationen erlernt wird, die über diverse Quellen beigefügt werden. Maschinelles Lernen findet in vielen Bereichen Anwendung, von Spracherkennung, Bilderkennung bis hin zu selbstfahrenden Kraftfahrzeugen. Diese Technologie wird zunehmend wichtiger in unserer stetig vernetzter werdenden Welt und trägt dazu bei, komplexer werdende Probleme zu lösen und immer innovativere Lösungen in verschiedenen Branchen zu entwickeln.

Im Wesentlichen geht es um die Entwicklung von Algorithmen, die dem Computer das Erkennen von Mustern und Zusammenhängen in Daten ermöglichen, wordurch ohne menschliche Intervention Aufgaben erledigt werden können. Daher trainiert man Algorithmen auf Basis großer Datenbestände, wodurch das jeweilige Modell so generalisiert wie möglich angepasst wird, sodass die erlernten Fähigkeiten auf neu erhobene Daten angewendet werden können. Das Erlernen des nötigen Wissens basiert hierbei auf drei wesentlichen Faktoren, die die Qualität des Trainings beeinflussen. Neben den Informationen, die aus Daten gewonnen werden, nutzt man eine Kennzahl, die einen Vergleich zwischen dem aktuellen und idealen Verhalten herstellt, um mit dem dritten Faktor - einem Rückkopplungsmechanismus - das programm anzuleiten eine verbesserte Leistung in Folgeergebnissen zu erzielen. (Joshi 2020, S. 4)

Um ein Modell trainieren zu können, wird ein geeignetes maschinelles Lernverfahren ausgewählt, das darauf abzielt, die Ausgaben eines KI-Systems in Bezug auf bestimmte Systemeingaben im Laufe des Lernprozesses zu optimieren. Unterschieden wird hautpsächlich in der Art und Weise, wie die „Kritik“ präsentiert wird, die zur Verbesserung des Verhaltens der jeweiligen künstlichen Systeme führen soll. Ein Training basiert auf sogenannten Hyperparametern, die Rahmenbedingungen des Prozesses festlegen. Dies kann zum Beispiel die Anzahl der Epochen, die Lern-Rate oder auch die Batch-Größe sein. Im Folgenden werden drei prägende Lernverfahren näher beschrieben.

2.1.1 Überwachtes Lernen

Überwachtes Lernen (eng.: supervised learning) stellt einen wesentlichen Bereich des maschinellen Lernens dar, um aus Informationen, bestehend aus Datenpunkten $X = x_1, x_2, \ldots, x_n$ mit einem zugehörigen Label aus $Y = y_1, y_2, \ldots, x_n$, das nötige Wissen für bestimmte Anwendungsfälle zu erlangen. Anwendung findet dieses Verfahren in Systemen, die einen bestimmten Input I verarbeiten, um darauf basierend eine Vorhersage über das zugehörige Label abzugeben. Mögliche Arten von neuronalen Netzwerken, die über diese Methode trainiert wurden, sind beispielsweise Bildklassifikatoren. Dabei ist es die Aufgabe des Modells M, basierend auf einem bestimmten Input I mit Hilfe erlernter Muster und Zusammenhänge der Trainingsdaten eine Vorhersage über die Kategorie oder das Label von I zu treffen.

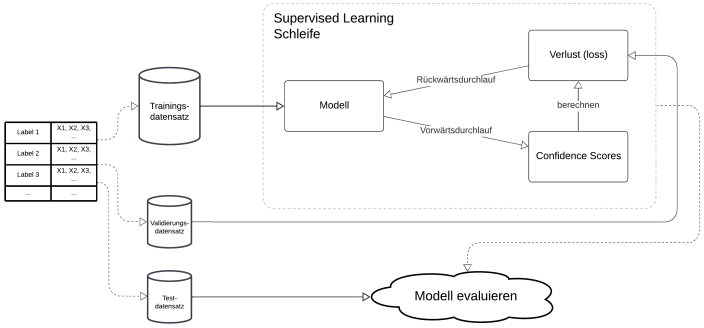


Bild 2.1: *Prozessvisualisierung: überwachtes Lernen*

Der *Ablauf* des Trainings lässt sich wie folgt beschreiben: nach Beenden von Datensammlung, Datenvorverarbeitung und Datensatztrennung in einen Trainings-, Validierungs- und Testdatensatz beginnt der eigentliche Teil des Trainings, wobei die Trainings- und Validierungsdaten für die Aktualisierung der interenen Modelleinstellungen, wie Gewichtungen und Schwellenwerte, verwendet werden. Die Optimierung der Parameter wird auf Basis der Verlustfunktion mit Hilfe eines Rückwärtsdurchlaufes (eng.: Backward Propagation) durchgeführt, die mit der Differenz zwischen Vorhersagen des Modells aus dem Vorwärtsdurchlauf (eng.: Forward Propagation) und den realen Labeln berechnet wird.

Die *Qualität* des Trainings wird meist anhand der Genauigkeit von Vorhersagen bestimmt, welche ausdrückt, wie gut die Klassifikation von im Training noch nicht gesehenen Daten funktioniert.

$$\text{Accuracy} = \frac{\text{Anzahl korrekter Vorhersagen}}{\text{Gesamtanzahl der Vorhersagen}}$$

Weitere Metriken können die Präzision (eng.: Precision), der Recall oder auch der F1-Score sein. Neben Messungen der Modell-Genauigkeit kann man die Qualität des trainierten Modells auch anhand der Verlustfunktion (eng.: loss-function) oder der 'Area Under the Curve' (AUC), welche die Fläche unter der 'Receiver Operating Characteristic' (ROC) – dem Verhältnis zwischen der True- und False-Positive-Rate – messen.

2.1.2 Unüberwachtes Lernen

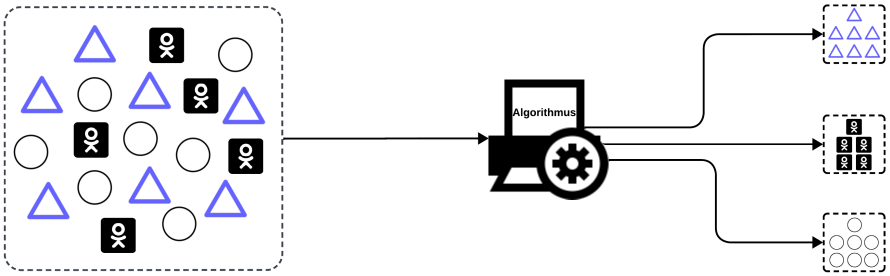


Bild 2.2: *Resultat eines auf unüberwachtem Lernen basierenden Clusterings*

Unüberwachtes Lernen (eng.: unsupervised learning) stellt einen anderen bedeutenden Bereich im maschinellen Lernen dar, der sich von überwachtem Lernen (2.1.1) dahingegen unterscheidet, dass keine expliziten Labels für die Trainingsdaten bereitgestellt werden. Diese Methode wird angewendet, wenn das Ziel darin besteht, Muster, Strukturen oder Zusammenhänge in den Daten zu finden, ohne dabei Kategorien oder Label vorzugeben, weshalb basierend auf Informationen eines Datensatzes, bestehend aus Datenpunkten $X = x_1, x_2, \ldots, x_n$ ohne dazugehöriges Label, trainiert wird. Das Modell soll dabei auf natürliche Art und Weise Strukturen, Muster und Zusammenhänge innerhalb der Eingabedaten ohne vorherige Kenntnisse der Zielvariable erlernen. „Aufgabe ist es hier, passende Repräsentationen zu finden, die z. B. die Erkennung von Charakteristika in Datenmengen, Wiedererkennung von Ausnahmen oder die Erstellung von Prognosen ermöglichen."(Lorenz 2020, S. 5) Im Gegensatz zu Überwachtem Lernen (2.1.1) geht es nicht um das Zurdnen von Mustern in vorhandenen Kategorien, sondern um das Auffinden von Clustern in einer bestimmten Datenmenge X. Dabei werden Modellparameter nicht über eine Verlustfunktion, die durch die Differenz zwischen tatsächlichem und vorhergesagtem Label berechnet wird, dargestellt, sondern die Parameter werden für die repräsentation inhärenter Muster in den Daten angepasst. Das Aufteilen des Datensatzes in 3 unabhängige Teile ist hierbei nicht notwendig, da keine Kategorien/Label für die jeweiligen Datenpunkte vorhanden sind. Es ist nur für die Auswertung des Algorithmus sinnvoll einen zweiten Datensatz zu erstellen, um damit die Qualität zu bewerten.

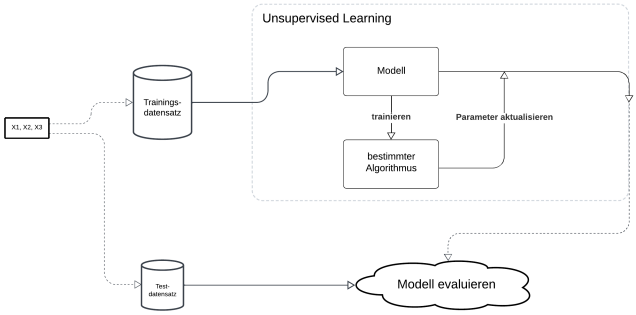


Bild 2.3: *Prozessvisualisierung: unüberwachtes Lernen*

Der *Ablauf* des Trainigs gestaltet sich wie folgt: Nach Abschluss von Datensammlung und Vorverarbeitung beginnt der eigentliche Prozess des unüberwachten Lernens. Im Gegensatz zum überwachten Lernen (2.1.1) gibt es hierbei keine vordefinierten Zielvariablen. Den Datenpunkten $X = x_1, x_2, \ldots, x_n$ ist also initial keine Kategorie beziehungsweise kein Label zugeordnet, da die Menge Y bist dato nicht existent ist. Im Gegensatz zu überwachtem Lernen nutzt man hier Algorithmen wie beispielsweise „k-Nearest Neighbors (KNN)“ (Joshi 2020, S. 38), um eine bestimmte Operation auf den zu behandelnden Datensatz auszuführen. Mit Hilfe des Algorithmus wird ein bestimmtes Modell dahingehend trainiert, dass es einen Datensatz zum Beispiel in bestimmte Kategorien unterteilen kann. Dafür analysiert der Algorithmus die jeweiligen Datenpunkte und versucht diese mit Hilfe von Zusammenhängen und verschiedenen extrahierten Merkmalen zu interpretieren. Anhand des dabei erlangten Wissens werden Modellparameter aktualisiert, wonach das unüberwachte Lernen beendet ist. Ein Ergebnis kann hierbei beispielsweise ein kategorisierter Datensatz sein.

Die *Qualität* eines Modells ist schwieriger zu messen als bei überwachtem Lernen 2.1.1, da keine klar definierte Zielvariable vorliegt, mit Hilfe welcher eine Genauigkeitsanalyse durchgeführt werden kann. Dennoch lässt sich auch dieses mit verschiedenen Metriken evaluieren. Zum einen kann man mit dem sogenannten Silhouette Score (Shahapure und Nicholas 2020) Clustering-Algorithmen evaluieren, da dieser das Zusammenpassen der verschiedenen Datenpunkte innerhalb eines Clusters bewertet. Je höher dieser Wert, desto besser sind die jeweiligen Cluster definiert. Neben Metriken, die durch bestimmte Berechnungen repräsentiert werden, lässt sich die Qualität eines Modells auch durch visuelle Methoden wie der Auswahl geeigneter Plots messen, die die Zugehörigkeit verschiedener Datenpunkte darstellen.

2.1.3 Bestärkendes Lernen

Das bestärkende Lernen (eng.: reinforcement learning) ist ein maschinelles Lernverfahren, das sich von überwachtem und unüberwachtem unterscheidet. Es basiert auf der Interaktion eines Agenten mit seiner Umgebung und ist auf Rückmeldungen aus dieser angewiesen. Im Gegensatz zu überwachten Lernverfahren gibt es keine vorgegebenen Label für die Datenpunkte, weshalb Reinforcement Learning nicht als vollständig überwacht betrachtet werden kann. Gleichzeitig fehlt es den Datenpunkten an der Strukturierung, die für unüberwachte Lernverfahren typisch ist. Die Interaktion mit der Umgebung erfolgt durch den Agenten, der verschiedene Erfahrungen sammelt, während dieser Aktionen ausführt. Das Hauptziel des Algorithmus besteht darin, den Agenten so zu trainieren, dass er in seiner spezifischen Umgbung optimale Aktionen durchführt, um eine maximale kumulative Belohung über die Dauer des Lernprozesses zu erzielen. Diese Belohnungen und Strafen werden von der Umgebung basierend auf den vom Agenten durchgeführten Aktionen übergeben. Damit der Agent das Ziel der maximalen kumulativen Belohnung erreicht, erfordert dies eine geschickte Balance zwischen der Erkundung neuer Aktionen und der Ausführung bekannter, belohnungsreicher Aktionen. Dabei muss der Agent seine Strategie kontinuierlich anpassen und verbessern. Bestärkendes Lernen wird in diversen Anwendungen eingesetzt, wie beispielsweise in Spielstrategien, in autonom-fahrenden Kraftfahrzeugen oder im Bereich der Robotik. Das Lösen komplexer Aufgaben hat durch den Einsatz tiefer neuronaler Netzwerke im 'Deep Reinforcement Learning'-Bereich zu bedeutendem Fortschritt geführt. Im Allgemeinen ist das bestärkende Lernen eine mächtige Methode, um intelligente Agenten für das Lösen komplexer Probleme in dynamischen Umgebungen zu trainieren. Diese sind dann in der Lage, optimale Entscheidungen zu treffen.

Zu den wichtigsten Bestandteilen des bestärkenden Lernverfahrens zählen neben dem Agenten und der Umgebung auch Belohnungen, Zustände, Zustandsübergänge und Aktionen. Die Teile

des Verfahrens lassen sich in zwei Gruppen unterteilen: zum einen der Agent mit den jeweiligen Aktionen und die Umgebung, mit den Zuständen und Belohnungen. Diese Umgebung bestimmt die neuen Zustände anhand der Veränderung basierend auf die Aktion, welche vom Agenten ausgeführt wird. Ein Zustand beschreibt hier die genaue Konfiguration der Umgebung zu einem bestimmten Zeitpunkt t. Ein Beispiel für einen Zustand kann das aktuelle Schachbrett mit den Positionen der jeweiligen Figuren sein, die durch eine Aktion und dem damit zusammenhängenden Zustandsübergang verändert wurden.

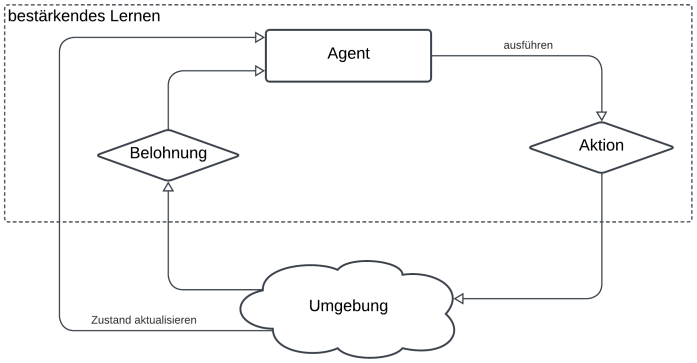


Bild 2.4: *Prozessvisualisierung: bestärkendes Lernen*

Der *Ablauf* des Trainings kann wie folgt beschrieben werden: Zu Beginn des Algorithmus setzt man die Parameter des Lernalgorithmus und des Agenten, wie Q-Werte, Policy oder andere relevante Variablen, auf einen bestimmten Startwert. Der Agent beobachtet hier den aktuellen Zustand der Umgebung mit einer bestimmten Methode (zum Beispiel visuell), um relevante Informationen für die Entscheidungsfindung aufzunehmen. Basierend auf seine interne Policy oder Schätzungen der Q-Werte des beobachteten Zustands der Umgebung wählt der Agent eine Aktion, um einen neuen Zustand herbeizuführen. Die Umgebung verändert ihren Zustand und gibt eine Belohnung, wie auch den neuen Zustand, an den Agenten zurück. Der Agent aktualisiert seine internen Modelle, um die Qualität seiner Entscheidungen zu verbessern, indem er den neuen Zustand analysiert und Belohnungen interpretiert. Beim Wählen der nächsten Aktion steht der Agent vor der Entscheidung der Exploration einer neuen Aktion und der Exploitation einer bereits durchgeführten. Der Prozess, beginnend mit der Auswahl einer bestimmten Aktion, wiederholt sich für eine gegebene Anzahl an Epochen beziehungsweise Schritten bis zur Konvergenz des Modells oder dem Erreichen einer akzeptablen Leistung.

Die Bewertung der *Qualität* eines mit bestäkendem Lernen trainierten Modells stellt sich aufgrund der fehlenden Übertragbarkeit von traditionellen Metriken, wie Genauigkeit und Verlust, als komplexe Aufgabe dar. Eine Art der Leistungsbewertung kann mit Hilfe der kumulativen Belohnung sein, wobei ein höherer Wert darauf hindeutet, dass der Agent optimale Aktionen gelernt hat. Daneben kann man auch verschiedene andere Ansätze wie zum Beispiel Visualisierungen nutzen, um die Qualität eines Modells zu bewerten.

2.2 Bilderkennung /-klassifikation

Ein bedeutender Bereich von maschinellem Lernen – und damit Teilbereich von künstlicher Intelligenz – ist die Bildklassifizierung (eng.: image classification). Diese Technik ermöglicht

Computern die automatisierte Analyse und Kategorisierung beziehungsweise Zurodnung bestimmter Klassen von Bildern. Das Hauptziel eines Bildklassifizierungsalgorithmus liegt darin, einem Computer das Erkennen von Mustern und Merkmaln in Bildern beizubringen, um darauf basierend korrekt zu entscheiden, welcher vordefinierten Klasse das jeweilige Bild angehört.

Die Vielfalt von Bildklassifikations-Anwendungen ist enorm groß. Von medizinischen Möglichkeiten, wie Krebserkennung, bis hin zu Systemen für das Erkennen verschiedener Verkehrsteilnehmer, die sich in der Umgebung eines autonom fahrenden Kraftfahrzeuges bewegen. Durch das Einbringen von derartigen Lösungen können komplexe Aufgaben in nahezu Echtzeit gelöst und menschliche Fehler minimiert werden.

Der Bildklassifizierungsprozess besteht aus mehreren Schritten. Zunächst wird eine große Menge an Bildern gesammelt, welche in einen Trainings- und Testdatensatz aufgeteilt wird. Während Trainingsdaten für den Trainingsprozess verwendet werden, wird der Testdatensatz für die Validierung und Leistungsbewertung des trainierten Modells verwendet, um sicherzustellen, dass das Modell auch im Training ungesehene Daten richtig interpretiert. Während des Trainingsprozesses werden die Parameter des neuronalen Netzwerks meist mit einem überwachten Lernalgorithmus optimiert. Das Modell lernt die Identifizierung relevanter Merkmale um Bilder korrekt zu klassifizieren. Nach dem Training kann das Modell auf neue, nicht gesehene Bilder angewendet werden, um Vorhersagen über die Klassen-Zugehörigkeit des jeweiligen Inputs durchzuführen.

2.2.1 Convolutional Neural Networks (CNN)

Eine spezielle Art neuronaler Netzwerke sowie ein DeepLearning Algorithmus inspiriert von einem visuellen System lebender Organismen. (Balas, Kumar und Srivastava 2020, S. 519) Ein solches CNN ist variabel – mit geringem Änderungsaufwand – in jeglichen Bereichen der Bild- und Sprachverarbeitung einsetzbar. Das Training eines CNNs basiert meist auf überwachten Lernalgorithmen (2.1.1) und beansprucht durch die Diversität und Vielfalt der Daten enorm hohe Rechenleistung, geschuldet von meist großen Eingangsdaten und hoher Parameteranzahl des Modells.

Standardmäßig ist der Aufbau eines Convolutional Neural Networks für die Bildklassifizierung in zwei verschiedene Teile zu gliedern. Zum einen die Merkmalsextrahierung, welche wie folgt aufgebaut ist: einzelne oder mehrere Blöcke von Faltungs- und Vereinigungsschichten werden aneinander gereiht. Eine Faltungsschicht (eng.: convolutional layer) hat die Aufgabe, Merkmalsrepräsentationen zu erlernen, indem über einen Faltungskern mehrere Merkmalsbilder extrahiert werden. Um die Dimensionen zu reduzieren verwendet man die sogenannten Vereinigungsschichten (eng.: pooling layer), welche einen kleinen Bereich der Ausgabe einer Faltungsschicht entgegennehmen und diesen durch eine bestimmte Abtastung zu einer einzigen Ausgabe transformieren, wodurch auch nötige Parameter reduziert werden. Der zweite Teil des Netzwerks umfasst Schichten zur Klassifizierung des in Teil I generierten Outputs. Dazu gehören vollvermaschten Schichten, die zuvor extrahierte Rückgaben entgegennehmen, um darauf basierend ein Ergebnis – und damit die endgültige Ausgabe des CNNs – ableiten zu können.

Die Leistung eines solchen Netzwerks hängt von verschiedenen Faktoren ab, die für Optimierungszwecke verändert und verfeinert werden können. Dazu zählen beispielsweise Aktivierungsfunktionen, Normalisierungen, Verlustfunktionen, ...

2.2.2 Merkmalextrahierung

Die Mermalextrahierung (eng.: feature extraction) in einem neuronalen Netzwerk wird mit Hilfe von Faltungsschichten ermöglicht, welche den jeweiligen Input I mit einem sogenannten Faltungskernel, der durch ein Raster diskreter Werte oder Zahlen beschrieben wird, filtern. Jeder Wert der Matrix legt eine bestimmte Gewichtung des Kernels fest, die während des Trainings angepasst werden kann. Initial gibt es die Möglichkeit das Training mit zufälligen Kernelgewichtungen zu starten oder selbst Werte zu definieren, abhängig von genutzter Methode.

Der vereinfachte *Ablauf* einer Mermalextrahierung ist wie folgt: Der vordefinierte Filterkern der Größe $M \times N$ wird mit einer bestimmten Schrittgröße von Links nach Recht bewegt. Dabei wird während jedem Schritt ein gewichteter Durchschnitt berechnet, welcher den aktuellen Teil des Bildes repräsentiert. Dies wird bis Beendigung der gesamten Inputverarbeitung wiederholt. Die neuen Berechnungen speichert man in einer Matrix, die am Ende des Vorgangs das Resultat des Gesamtfilters darstellt. In der Regel wendet man eine nicht-lineare Aktivierungsfunktion auf die Ergebnisse der Faltung an, um Linerität in den Daten zu vermeiden.

-1	0	1
-2	0	2
-1	0	1

Bild 2.5: *Sobel-Filter (horizontal) mit vordefinierten Gewichtungen*

Ein Beispiel für einen gängigen Filterkern ist der horizontal angeordnete Sobel-Filter (Bild 2.5). Dieser wird verwendet, um horizontale Gradienten und Kanten in einem bestimmten Input zu detektieren. Der Filter berechnet bei jeder Position einen gewichteten Durchschnitt, womit im Laufe des Prozesses Kanten erkannt werden können. Dies ist der Fall, da die Gewichtungen des Filter so gewählt sind, dass diese besonders Empfindlich auf Veränderungen in horizontaler Richtung reagieren. Dadurch werden horizontale Kanten hervorgehoben, aber auch vertikale unterdrückt. Durch das Nutzen eines vertikalen Sobel-Filters ist das Verhalten umgekehrt. Nach Beenden des Durchgangs werden im Resultat, einer Feature-Map, die detektierten Kanten dargestellt.

2.3 Angriffsmöglichkeiten auf Neuronale Netzwerke

Neuronale Netzwerke, obwohl leistungsfähig und vielseitig einsetzbar, eröffnen verschiedene Möglichkeiten für Angriffsvektoren, die von potenziellen Angreifern ausgenutzt werden können, um erheblichen Schaden anzurichten. Die Angriffe reichen von sogfältig konzipierten Manipulationen des Trainisngsprozesses bis hin zu raffinierten Methoden oder Modellinferenzen. Das ganze bringt Konsequenzen für Entwickler und Unternehmen von Beeinträchtigung der KI-System-Verfügbarkeit bis hin zu gravierenden Datenschutzverletzungen von Involvierten mit sich.

2.3.1 Modell-Inversionsangriffe

Hier soll was über Modell-Inversionsangriffe stehen (Bsp. usw.) ...

2.3.1.1 Angriffsziel

Hier soll was über Ziel von diesem Angriff geschreiben werden (Bsp. usw.) ...

2.3.1.2 Angriffsvektoren

Hier soll geschrieben werden, wie der Angriff ausgeführt werden kann (Bsp. usw.) ...

2.3.1.3 Verteidigungsstrategien

Hier sollen mögliche Verteidigungsstrategien beleuchtet werden ...

Kapitel 3

3 Stand der Technik

3.1 Forschungsergebnisse

Kapitel 4

4 Implementierung

4.1 Experimentierumgebung

Modell-Art	Hyp 1	Hyp 2	Hyp 3
VGG-16	Inhalt 1	Inhalt 2	Inhalt 3
DCGAN	Inhalt 4	Inhalt 5	Inhalt 6
StyleGAN	Inhalt 7	Inhalt 8	Inhalt 9

Tabelle 4.1: *Hyperparametervergleich verschiedener Modelle*

4.2 Funktionalität des Codes

Hier soll die Funktionalität des Codes beschrieben werden ...

```
import numpy as np

def incmatrix(genl1,genl2):
  m = len(genl1)
  n = len(genl2)
  M = None #to become the incidence matrix
  VT = np.zeros((n*m,1), int)  #dummy variable

#compute the bitwise xor matrix
  M1 = bitxormatrix(genl1)
  M2 = np.triu(bitxormatrix(genl2),1)

  for i in range(m-1):
    for j in range(i+1, m):
      [r,c] = np.where(M2 == M1[i,j])
        for k in range(len(r)):
          VT[(i)*n + r[k]] = 1;
          VT[(i)*n + c[k]] = 1;
          VT[(j)*n + r[k]] = 1;
          VT[(j)*n + c[k]] = 1;

          if M is None:
            M = np.copy(VT)
          else:
            M = np.concatenate((M, VT), 1)

          VT = np.zeros((n*m,1), int)
```

```
28
29  return M
```

Listing 4.1: *Python example*

4.3 Code-Beschreibung

Hier soll beschrieben werden, wie der Code strukturiert ist, nach welchen Pattern gearbeitet wurde und was der Code oberflächlich macht...

5 Kapitel 5

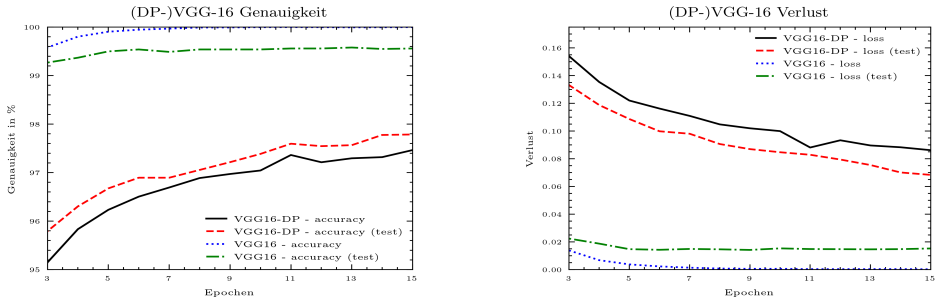
Ergebnisse

5.1 Beobachtungen

In diesem Kapitel werden umfassende Beobachtungen präsentiert, die eine detaillierte Bewertung der Leistung und Charakteristika des entwickelten Modells ermöglichen. Die Beobachtungen erstrecken sich über verschiedene Aspekte, einschließlich der Modellleistung nach dem Training, der Qualität der generierten Bilder sowie der Schnelligkeit und Qualität von Angriffen auf das Modell.

5.1.1 Modellleistung

Im Folgenden werden Metriken dargestellt und verglichen, mit Hilfe derer die Leistung der verschiedenen Modelle nach Abschluss der Trainingsroutinen bewertet werden kann.



(a) Modell-Genauigkeit *(b) Verlust*

Bild 5.1: *Verlust- und Genauigkeitsgegenüberstellung eines auf den MNIST-Datensatz normal und privat trainierten Modells*

Die *Genauigkeit* (Bild 5.1a) bezüglich der Test- und Trainingsdaten nimmt – wie erwartet – im Laufe des Trainingsprozesses zu. Während das „normale Training“ mit einer Test-Genauigkeit von $\approx 99,0\%$ startet und bei $\approx 99,5\%$ nach etwa 7 Epochen stagniert, erreicht das „neuronale Netzwerk mit differentieller Privatsphäre“ zu Beginn einen deutlich niedrigeren Wert von $\approx 91,5\%$ und stagniert während der trainierten 15 Epochen noch nicht. Dabei ist allerdings zu Vermuten, dass die Genauigkeit während weiteren Epochen ansteigt, was aufgrund begrenzter Ressourcen nicht getestet werden konnte. Die hohe Genauigkeit nach nur einer Trainingsepoche

ist auf das Transfer-Learning zurückzuführen, wobei Gewichtungen des Modells schon anhand eines anderen, großen Datensatzes vortrainiert sind, und nicht mit Trainingsstart neu initialisiert werden müssen.

Der *Verlust* (Bild 5.1b) während des Trainings zeigt einen ähnlichen Verlauf wie die oben beschriebene Test-Genauigkeit der beiden Modellarten. Nach etwa 7 Epochen beginnt dieser bei der Durchführung des „normalen Trainings“ zu konvergieren und deutet darauf hin, dass das Training erfolgreich durchgeführt wurde. Im Gegensatz dazu ist wiederum bei dem „Modell mit differentieller Privatsphäre“ zu beobachten, dass eine Konvergenz der Verlustfunktion nach 15 Epochen nicht erkannt werden kann. Auch hier – wie bei Test-Genauigkeit – lässt sich vermuten, dass eine Erhöhung der Trainingsdurchläufe eine Minimierung der Verlust-Funktion herbeiführt. Zudem lässt sich bei beiden Modellen aus dem Verlauf der Verlustfunktion ableiten, dass die Modelle keine Überanpassung bezügliche der Trainingsdaten vorweisen.

Das „normale Training“ kann aufgrund der konvergierenden Verlust- und Genauigkeitsverte nach 7 Epochen beendet werden, da die Parameter nahezu vollständig an den Datensatz (Bild 5.1 - MNIST) angepasst sind. Dahingegen sollte man aufgrund der unvollständigen Parameteranpassung im Modell mit differentieller Privatsphäre die Anzahl der Epochen erhöhen.

5.1.2 Bildqualität

Aufgrund der verschiedenen Angriffs-Verfahren variiert die Genauigkeit der Bilder in den unterschiedlichen Durchgängen, wobei die Qualität - abhängig von der Komplexität des Generative-Adversarial Networks - gleich ist. Die Genauigkeit des neu generierten Bildes, das aus privaten Daten abgeleitet ist, wird mit Hilfe des Confidence-Scores bezüglich Ziel-Klasse bestimmt. Im Folgenden werden einige Vergleiche über die Bildqualität nach der Ausführung des Angriffs basierend auf einem Klassifizierungsmodell für Zahlen und Gesichtern aufgezeigt.

Die *Qualität* der Bilder ist abhängig von genutztem GAN (Generative adversarial network), das für die Generierung verwendet wird. Generierungen basierend auf dem DCGAN sind aufgrund geringerer Parameter, kürzerer Trainingsdauer und anderer Faktoren qualitativ nicht so hochwertig wie die des genutzten StyleGANs.

(a) *Generierung der Zahl 0 auf Basis eines DCGAN*

(b) *Generierung einer Frau mit Hilfe eines StyleGANs*

Bild 5.2: *Bildqualität der GAN-Modelle*

Die Bilder des DCGAN basierten Generators sind etwas schwach mit vergleichsweise wenigen Pixeln aufgelöst (siehe Bild 5.2a). Dahingegen lassen sich Bilder des StyleGAN Netzwerks in einer deutlich besseren Auflösung von bis zu 200 × 200 Pixeln generiern. Wie im Bild 5.2b zu erkennen, wird die Qualität der Bilder durch die Reduktion der Pixel deutlich verschlechtert, was aber zu einer effektiveren Angriffsdauer führt. Im Gegensatz zu den generierten Ziffern handelt es sich bei den Gesichtern um 3-Kanal Bilder (RGB), weshalb diese farblich zu betrachtet sind.

5.1.3 Angriffsperformance

Dieses Kapitel befasst sich mit der Auswertung beider Angriffe nach der Durchführung, wobei im Folgenden ein Vergleich mit dem Ziel der Stärken- und Schwächenbeleuchtung aufgestellt wird. Um die Angriffe besser bewerten zu können, werden zum einen visuelle, aber auch metrische Daten begutachtet, wozu beispielsweise die Genauigkeit der neu generierten Bilder bezüglich der Ziel-Kategorie des „angegriffenen“ Modells zählt.

5.1.3.1 Bildqualität der Angriffe

5.1.3.2 Angriffsstatistiken

5.1.4 Auswertung der Verteidigungsstrategie

5.2 Rückschlüsse

Hier sollen die Rückschlüsse stehen ...

Kapitel 6

6 Zusammenfassung und Ausblick

Kapitel 7

7 Anhang

Hier sind noch LaTeX-Beispiele für die Verwendung:

...

Tabelle erzeugen:

Tabelle 7.1: *Beispieltabelle*

Spalte 1	Spalte 2	Spalte 3	Spalte 4
Inhalt 1	Inhalt 2	Inhalt 3	Inhalt 4
Inhalt 5	Inhalt 6	Inhalt 7	Inhalt 8

...

Quellen verlinken: Test **Bar-Shalom**. Test **Goldhammer**. Test **WHO-2004**.

...

Kapitel verlinken: Wie in Abschnitt 5 beschrieben wird ...

...

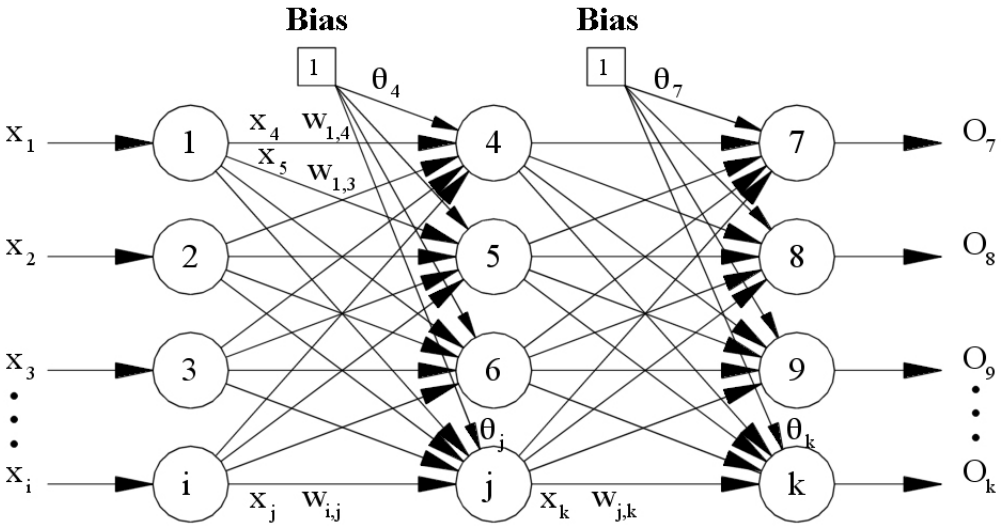


Bild 7.1: *Beispielbild*

Bilderverzeichnis

Tabellen

(Lorenz 2020)

Literatur

Bücher

Joshi, Ameet V (2020).
Machine Learning and Artificial Intelligence.
Cham: Springer International Publishing.
ISBN: 978-3-030-26621-9 978-3-030-26622-6.
DOI: 10.1007/978-3-030-26622-6.
URL: http://link.springer.com/10.1007/978-3-030-26622-6 (besucht am 24. 11. 2023).
Lorenz, Uwe (2020).
Reinforcement Learning: Aktuelle Ansätze verstehen - mit Beispielen in Java und Greenfoot.
Berlin, Heidelberg: Springer Berlin Heidelberg.
ISBN: 978-3-662-61650-5 978-3-662-61651-2.
DOI: 10.1007/978-3-662-61651-2.
URL: http://link.springer.com/10.1007/978-3-662-61651-2 (besucht am 04. 10. 2023).

Sammlungen

Balas, Valentina E., Raghvendra Kumar und Rajshree Srivastava, Hrsg. (2020).
Recent Trends and Advances in Artificial Intelligence and Internet of Things.
Bd. 172.
Intelligent Systems Reference Library.
Cham: Springer International Publishing.
ISBN: 978-3-030-32643-2 978-3-030-32644-9.
DOI: 10.1007/978-3-030-32644-9.
URL: http://link.springer.com/10.1007/978-3-030-32644-9 (besucht am 29. 11. 2023).

Konferenzbeiträge

Shahapure, Ketan Rajshekhar und Charles Nicholas (Okt. 2020).
„Cluster Quality Analysis Using Silhouette Score“.
In: *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*.
2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA).
sydney, Australia: IEEE,
S. 747–748.
ISBN: 978-1-72818-206-3.
DOI: 10.1109/DSAA49011.2020.00096.
URL: https://ieeexplore.ieee.org/document/9260048/ (besucht am 28. 11. 2023).