R. Reussner, A. Koziolek, R. Heinrich (Hrsg.): INFORMATIK 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2021 841

Risiken für die Privatheit aufgrund von Maschinellem Lernen*

Verena Battis[1], Lukas Graner[1]

Abstract: Maschinelle Lernverfahren sind aus unserem Alltag fast nicht mehr wegzudenken - selbstlernende Verfahren finden bereits in nahezu allen Bereichen des Lebens Anwendung. In vielen Fällen werden dabei auch private und/oder sensible Informationen verarbeitet. Da selbstlernende Verfahren in der Regel auf sich nicht überschneidenden Datenmengen trainiert und später angewendet werden, ging man lange davon aus, dass es nicht möglich sei, vom finalen Modell Rückschlüsse auf die zum Training verwendeten Daten zu ziehen. Ergebnissen aus der jüngeren Forschung demonstrieren jedoch, dass es sich bei dieser Annahme um einen Trugschluss handelt. Die vorliegende Arbeit erläutert welche Risiken sich für die Privatheit des Einzelnen im Rahmen von maschinellen Lernverfahren ergeben und wie dem unerwünschten Abgreifen von sensiblen Informationen bereits in der Trainingsphase entgegen gesteuert werden kann.

Keywords: Privatheit; Privatsphäre; Risiken; maschinelles Lernen; Angriffe

1 Motivation

Maschinelles Lernen (ML) ist ein Teilgebiet der künstlichen Intelligenz und beschreibt eine Reihe von Lernalgorithmen, die versuchen Strukturen in Daten zu erkennen, um basierend auf diesen Mustern bspw. Klassifizierungs- oder Regressionsaufgaben zu lösen. Der Einsatz von Verfahren des maschinellen Lernens bietet sich immer dann an, wenn die zu lösenden Probleme zu komplex oder zu umfassend sind, um sie analytisch beschreiben zu können [Do18]. Gleichzeitig bedeuten größere Datenmengen auch, dass mehr Informationen zum Trainieren der Lernalgorithmen zur Verfügung stehen, was tendenziell zu besseren Modellen und effizienteren Schätzungen führt [ST17]. Ein maschinelles Lernverfahren welches in den letzten Jahren infolge seiner Flexibilität und guten Generalisierungsfähigkeit besonders an Beliebtheit gewonnen hat, sind sogenannte Neuronale Netze. Diese finden aufgrund ihrer Fähigkeit selbst komplexe, nicht-lineare Funktionen zuverlässig approximieren zu können, in den verschiedensten Bereichen Anwendung – ob im Verarbeiten und Analysieren natürlicher Sprachen, zur Bild- oder Gesichtserkennung oder zum Aufspüren von Anomalien. Diese Fähigkeit, welche häufig mit dem Begriff *Kapazität* beschrieben wird [BV19], ist es aber auch, welche die Privatheit des Individuums bedroht. Modelle mit nicht ausreichender

* Diese Forschungsarbeit wurde vom Bundesministerium für Bildung und Forschung (BMBF) und vom Hessischen Ministerium für Wissenschaft und Kunst (HMWK) im Rahmen ihrer gemeinsamen Förderung für das Nationale Forschungszentrum für angewandte Cybersicherheit ATHENE unterstützt.

[1] Fraunhofer-Institut für Sichere Informationstechnologie SIT, Rheinstraße 75, 64295 Darmstadt
vorname.nachname@sit.fraunhofer.de

doi:10.18420/inf2020_75

Kapazität können den in den Trainingsdaten enthaltenen Zusammenhang nur schwer oder gar nicht abbilden, wohingegen eine zu hohe Kapazität dazu führen kann, dass das Neuronale Netz Eigenschaften der Trainingsdaten auswendig lernt [Go16].
Üblicherweise sind die Datensätze auf denen ML-Algorithmen trainiert und später angewendet werden, disjunkt. In der Konsequenz sollte es nicht möglich sein, vom finalen Modell Rückschlüsse auf die zum Training verwendeten Daten zu ziehen, was einer Anonymisierung der verwendeten Trainingsdaten gleichkommen würde [WBH19]. Dass in großen Datenbeständen - selbst in solchen aus gering strukturierten oder gar unstrukturierten Daten - entscheidende Verknüpfungen gefunden werden können, welche das Herstellen von Personenbezügen ermöglichen, ist bereits hinreichend bekannt [HG16; NS08]. Moderne ML-Angriffe gehen allerdings noch einen Schritt weiter. Hier werden bestimmte Eigenschaften des ML-Algorithmus bewusst ausgenutzt, z.B. die zum Teil extreme Kapazität eines Neuronalen Netzes, um Informationen bezüglich der zum Training verwendeten Daten in Erfahrung zu bringen und somit die Privatheit der Datensubjekte zu kompromittieren.

Im Folgenden werden drei Arten von Rückschlüssen und die korrespondierenden Angriffe vorgestellt: Model Inversion, Membership Inference sowie Model Extraction. Der Fokus der vorliegenden Arbeit liegt auf den Risiken von maschinellen Lernverfahren, weshalb mögliche Gegenmaßnahmen abschließend nur in aller Kürze angesprochen werden.

2 Angriffe auf die Privatheit

2.1 Begriffserklärung

Angenommen ein maschinelles Lernmodell wird auf einem vertraulichen und unveröffentlichten Datensatz trainiert und anschließend zur Nutzung bereitstellt. Allgemein wird zwischen zwei Szenarien unterschieden: ob das Modell vollständig veröffentlicht wird oder ob dem Nutzer lediglich Nutzungszugriff auf das Modell gewährt wird, bspw. über eine API. Wird das Modell an sich veröffentlicht, kann der Nutzer das Modell nach Belieben befragen und besitzt darüber hinaus volles Wissen über den verwendeten Algorithmus, die Architektur und die Parameter des Modells. Man spricht in diesem Kontext von einem *White-Box Zugriff*. Im Gegensatz dazu kann der Nutzer in einem sogenannten *Black-Box Setting* das Modell zwar ebenfalls mit seinen eigenen Datenpunkten befragen, um eine Ausgabe zu erhalten, verfügt darüber hinaus aber über keinerlei Wissen bezüglich des verwendeten Modells, dessen Architektur oder verwendeter Parameter [BR18; KK+12; Sa19]. Je mehr Wissen der Angreifer über das Zielmodell hat, desto größer ist die Chance auf einen erfolgreichen Angriff. Dementsprechend gestaltet sich ein Black-Box Angriff als herausfordernder als ein White-Box Angriff.

Weiter gehen wir davon aus, dass es sich bei dem maschinellen Lernmodell um ein Klassifikationsmodell handelt, dessen Ausgabe aus Wahrscheinlichkeitswerten besteht. Zum einen geben diese Werte an, welcher Klasse bzw. welchem Attribut der eingegebene

Datenpunkt zugeordnet wird. Zum anderen bedeuten Werte nahe 1 auch, dass sich das Modell bezüglich seiner Entscheidung sicherer ist. Resultiert in einem Klassifizierungsproblem mit n Klassen bspw. eine Wahrscheinlichkeit von $1/n$ für eine bestimmte Klasse, so ist sich das Modell wesentlich unsicherer bezüglich seiner Entscheidung, als wenn es einen Datenpunkt mit einer Wahrscheinlichkeit von 0,98 einer der Klassen zuweist.

Die folgenden Beispiele stellen zwar ausschließlich Angriffe auf Neuronale Netze dar, die beschriebenen Ansätze sind jedoch ebenfalls auf eine Reihe anderer maschineller Lernalgorithmen anwendbar.

2.2 Model Inversion

Die Idee der Model Inversion ist es, das Modell selbst zu nutzen, um gezielt Datenpunkte, die zum Training verwendet wurden, zu rekonstruieren. Je nach Intention des Angreifers bedarf es nicht einmal zwangsläufig einer vollständigen Rekonstruktion der Daten, sondern nur bestimmter Eigenschaften. Eine vollständige und perfekte Rekonstruktion des Trainingsdatensatzes würde eine massive Verletzung der Privatheit der Datensubjekte darstellen. Betrachten wir die Software *Faception*, welche von dem gleichnamigen israelischen Konzern vermarktet wird [Fa19]. *Faception* ist ein maschinell lernendes Modell, welches anhand von Gesichtsbildern Rückschlüsse auf die Persönlichkeit der jeweiligen Person schließt. Die Entwickler werben damit, dass ihr Modell anhand eines einfachen Fotos entscheiden kann, ob es sich hierbei um einen Wissenschaftler, einen Bingo-Spieler, einen Pädophilen oder um einen Terroristen handelt [Fa19]. Gerade mit Blick auf die beiden letztgenannten Kategorien kann ein Angreifer ein besonders hohes Interesse daran haben, die zum Training des Modells verwendeten Gesichtsbilder möglichst akkurat zu rekonstruieren.

Da der Angreifer bei dieser Form des Angriffs im Extremfall von einem kleindimensionalen Ergebnisvektor (i.d.R. n Wahrscheinlichkeitswerte gemäß der n zuzuordnenden Klassen) auf einen hochdimensionalen Input zurückschließen muss, steigen die Erfolgschancen des Angriffs je mehr Informationen bezüglich des Modells dem Angreifer vorliegen. Im Optimalfall verfügt der Angreifer über einen *White-Box Zugriff* und kann die Gradienten im Neuronalen Netz direkt berechnen. Diese Gradienten, also alle partiellen Ableitungen einer multivariaten Funktion, symbolisieren den Effekt, den eine marginale Änderung in den Inputwerten (wie etwa einzelner Pixelfarbwerte eines Inputbildes) auf die Ausgabe des Modells hat und stellen die Grundlage vieler Model Inversion Ansätze dar.

Es sei bekannt, dass das Modell Personen anhand von Bildern n unterschiedlichen Klassen zuordnet. Ein Angreifer wäre demnach daran interessiert zu wissen, wie eine Person aussieht, die bspw. zum Training der Klasse 'Terrorist' verwendet wurde. Ein möglicher Ansatz, um einen Model Inversion Angriff durchzuführen ist, ausgehend von einem „leeren“ Startbild (bspw. ein vollständig schwarzes Bild) das Modell wiederholt zu befragen [Sa18]. Verfügt der Angreifer über einen White-Box Zugriff kann dieser nicht nur sämtliche Ausgabewerte beobachten, sondern auch die Gradienten berechnen. Mit Hilfe dieser Gradienteninformation

modifiziert der Angreifer schrittweise die Pixelwerte des Eingabebildes dahingehend, dass die Ausgabekonfidenz für die Zielklasse (hier: 'Terrorist') maximiert wird. Wird das Modell anschließend mit dem finalen, über Pixeloptimierung generierten Bild befragt, wird es dieses der Zielklasse mit einer sehr großen Sicherheit, also einem hohem Wahrscheinlichkeitswert, zuordnen. Hierbei wird naiverweise angenommen, dass das so generierte Bild eine Instanz der Trainingsdaten darstellt oder diese für die Zielklasse zumindest angemessen repräsentiert. Dies ist allerdings nur in einigen wenigen Sonderfällen korrekt, wie im Folgenden demonstriert.

In der akademischen Literatur stößt man häufig auf das Beispiel von Fredrikson et al., die die Trainingsdaten eines Gesichtserkennungsmodells rekonstruiert haben [FJR15]. Allerdings stellt gerade dieser Fall eines Gesichtserkennungsmodells einen Extremfall dar, da jede Ausgabeklasse des Modells eine individuelle Person repräsentiert. D.h. alle Trainingsdatenpunkte einer jeden Klasse sind nur unterschiedliche Aufnahmen der gleichen Person. Wenn der Angreifer also versucht, wie oben beschrieben, einen Input zu generieren, welcher vom Modell mit einer hohen Konfidenz der Zielklasse zugeordnet wird, so bildet er im Grunde einen Mittelwert über alle Trainingsbilder jener Klasse - und nicht wie gewollt, einen tatsächlichen Trainingsdatenpunkt. Im Fall von Fredrikson et al. wurde das anzugreifende Modell auf einem sehr kleinen, standardisierten Datensatz trainiert, welcher ausschließlich aus Frontalaufnahmen besteht. In Kombinationen mit einer naiven Modellarchitektur, die so in der Praxis keine Anwendung findet, resultieren künstlich generierte Bilder, die häufig auch von Menschen wiedererkannt werden können (vgl. Abb. 2, Spalte 2). Sobald die Daten innerhalb der Klassen jedoch eine größere Variation aufweisen oder eine komplexere Modellarchitektur genutzt wird (wie etwa ein Faltendes Netzwerk (engl. Convolutional Network), sind jene Rekonstruktionen, die basierend auf dem Ansatz von [FJR15] gewonnen wurden, nicht mehr als Objekte, geschweige denn spezifische Instanzen des Trainingssets wiederzuerkennen (vgl. Abb. 1, sowie Abb. 2 Spalte 3).

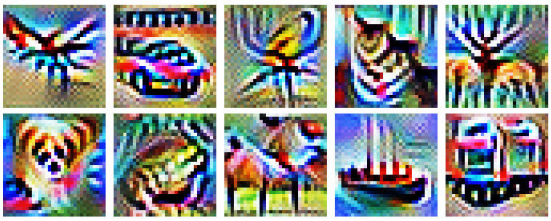

Abb. 1: Model Inversion Angriff auf den CIFAR 10 Datensatz, in Anlehnung an [FJR15]. Die rekonstruierten Klassen sind: Oben - Flugzeug, Auto, Vogel, Katze, Hirsch. Unten - Hund, Frosch, Pferd, Schiff, LKW.

Ergebnisse aus unserer eigenen Forschung (*GenMoIn*) zeigen jedoch, dass erweiterte Angriffsansätze trotz erschwerter Bedingungen - wie z.B. aufgrund hoher Modellkomplexität - dennoch Erfolge erzielen können. Beispielsweise kann der Model-Inversion-Prozess durch das Berücksichtigen von Hintergrundwissen optimiert werden. In den meisten Fällen stammen die für ein zu lösendes Problem verwendeten (Trainings-)Daten nur aus einer

bestimmten Domain. Im Falle einer Gesichtserkennung kann ein Angreifer bspw. davon ausgehen, dass das Trainingsset nur Portraitfotos beinhaltet. Bilder, die etwas anderes als menschliche Gesichter darstellen, wie etwa Objekte oder Rauschen, können von Grund auf ignoriert werden. Unser Ansatz, *GenMoIn*, macht sich genau dieses Vorwissen über die Verteilung der Daten zu Nutze. Dazu bedienen wir uns sogenannten *Generator Netzwerken*, die unter anderem als Teil eines *Generative Adversarial Networks* (GAN) [Go14] trainiert werden können. Ein solcher Generator erhält als Eingabe lediglich einen Vektor und erstellt daraus einen Datenpunkt - etwa ein Bild. Die Werte dieses *Code-Vektors* beschreiben und bestimmen dabei in einer komprimierten Form den resultierenden Datenpunkt. Für den Prozess der Model Inversion wird der Generator vor das anzugreifende Modell geschaltet, sodass die Ausgabe des Generators als Eingabe für das Zielmodell fungiert. Ab hier verläuft *GenMoIn* analog zu dem Angriff nach Fredrikson et al. [FJR15]. Allerdings wird in *GenMoIn* anstatt einzelner Pixelwerte der Code-Vektor selbst schrittweise optimiert. Jeder weitere Schritt verändert somit den Input-Code und damit auch den zu rekonstruierenden Datenpunkt, sodass die Konfidenz des Modells bezüglich der Zielklasse maximiert wird. Der Generator repräsentiert hierbei einen Vorfilter, welcher nur Datenpunkte aus einer bekannten Verteilung generieren kann.

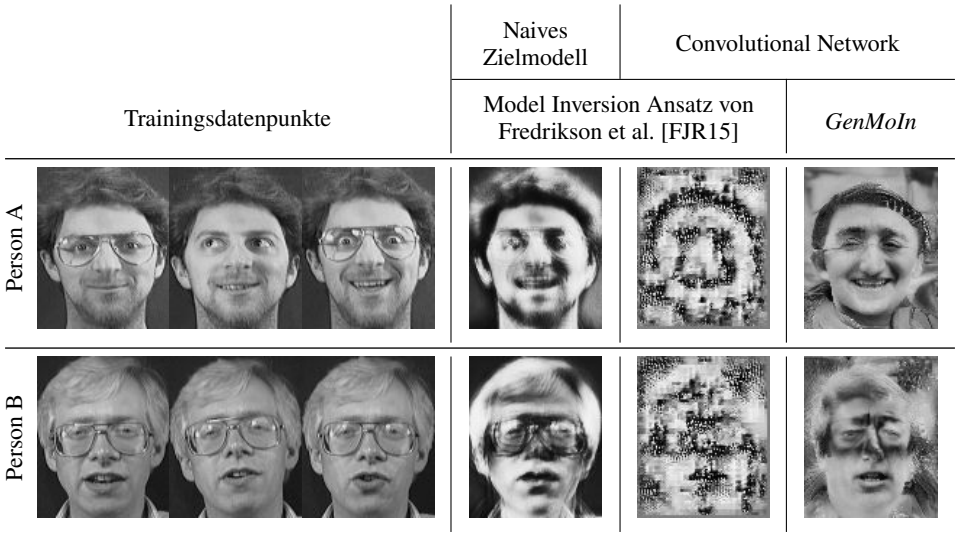

Abb. 2: Model Inversion Angriff auf den ATT Faces Datensatz. Jede Zeile, und somit Person, repräsentiert eine individuelle Klasse. Die linke Spalte stellt jeweils eine Teilmenge der entsprechenden Trainingsbilder dar. Die zweite Spalte zeigt Ergebnisse des Angriffs von Fredrikson et al. [FJR15] bei dem das Zielmodell nur aus einer Eingabeschicht, unmittelbar gefolgt von einer Ausgabeschicht, besteht. Die beiden letzten Spalten beziehen sich auf ein komplexeres Zielmodell mit zwei Convolutional Layern, wobei links Ergebnisse des Ansatzes nach Fredrikson et al. [FJR15] (Spalte 3) und rechts (Spalte 4) die eines unserer eigenen Angriffsansätze (*GenMoIn*) dargestellt sind.

Allerdings ist zu beachten, dass der Generator die Vielfalt der Datenverteilung möglichst umfassend und ohne Lücken gelernt hat - also ein ausreichendes Ausgabespektrum abdeckt. Ist dies nicht der Fall, etwa als Folge eines sogenannten *Mode Collapse*, kann dies zu stark verfälschten Ergebnissen der Model Inversion führen oder eine Rekonstruktion gänzlich unmöglich machen.

Im Fall eines Gesichtserkennungsmodells kann ein potentieller Angreifer davon ausgehen, dass der Trainingsdatensatz ebenfalls aus Aufnahmen von menschlichen Gesichtern bestanden hat. Gemäß dieses Vorwissens verwenden wir für unseren Angriff auf das ATT-Faces-Zielmodell ein Generator Netzwerk, welches Gesichter generiert. Die StyleGAN-Architektur [KLA19] gilt zusammen mit ihrer Weiterentwicklung StyleGAN2 [Ka19] als wegweisend für bildgenerierende Modelle. Deshalb, und weil ebendieses *Generative Adversarial Network* auf einer sehr großen, heterogenen Menge an Gesichtsbildern trainiert wurde [St19], haben wir den Generator des StyleGANs als Vorfilter für unseren Ansatz ausgewählt.

Mittels *GenMoIn* lassen sich auch aus komplexeren Modellen Rekonstruktionen erzielen. Wenngleich nicht vollständig wahrheitsgetreu, können diese für den Angreifer dennoch relevante Informationen enthalten – vor allem im Vergleich zu den stark verrauschten Bildern des Referenzangriffs (vgl. Abb. 2, Spalte 4 und 3). So können selbst in einem nur teilweise rekonstruierten Gesichtsbild trotz fehlender oder inkorrekter Gesichtszüge, andere, potentiell sensible Informationen, wie Hautfarbe oder das Tragen einer Brille, ermittelt werden. Dieser Zusammenhang ist in Abbildung 2 dargestellt. In den Rekonstruktionen nach [FJR15] (Spalte 3) lassen sich zwar Gesichtskonturen erkennen, diese sind jedoch sehr verrauscht und geben keine Auskunft über Details. Im *GenMoIn* -Ansatz (Spalte 4) wurde dagegen das Wissen, dass es sich um Portraitfotos handelt, direkt in den Rekonstruktionsprozess integriert, sodass automatisch ein erkennbares Gesicht generiert wird. Obwohl es sich nicht um exakte Replikationen von Trainingsinstanzen handelt, sind Details, wie die Haarfarbe, prägnante Gesichtszüge oder das Tragen von Accessoires deutlich erkennbar. So sind die charakteristische Nase und dunklen Haare von Person A), wie auch die volle Unterlippe und das Tragen einer Brille von Person B) deutlich in den Rekonstruktionen von *GenMoIn* zu erkennen.

2.3 Membership Inference

Das Ziel des Membership Inference Angriffs ist es, gegeben eines bestimmten Datenpunktes, eine Aussage darüber treffen zu können, ob ebenjener Datenpunkt zum Trainieren des betrachteten Modells verwendet wurde. Die zugrundeliegende Aufgabe des Zielmodells, also ob es sich um eine Klassifikation oder Regression handelt, ist hierbei für den Erfolg des Angriffs unwesentlich. Einem Angreifer geht es rein um das Verknüpfen eines Datenpunktes mit zusätzlich vorhandenen Informationen über den Trainingsdatensatz. Shokri et al. [Sh17] haben nachgewiesen, dass neuronale Netze aufgrund ihrer Kapazität besonders anfällig für Membership Inference Angriffe sind. Dabei trainierten die Autoren ein separates

Angriffsmodell, welches Anhand der Entropie der Ausgabewerte des Zielmodells - also aufgrund der darin enthaltenen Sicherheit bzw. Unsicherheit - beurteilen kann, ob ein Input zum Training des Zielmodells verwendet wurde oder nicht. Hintergrund ist, dass das Zielmodell Informationen, die es bereits während des Trainings „gesehen“ hat, mit einer höheren Konfidenz klassifiziert. Allgemein stellen Angriffe wie die Membership Inference eine Verletzung der Privatheit dar, sind aber besonders dann kritisch, wenn es sich um sensible Informationen handelt, wie bspw. die finanzielle Situation oder medizinische Angaben über eine Person [WBH19].

Ein trainiertes Netz reagiert merkbar anders auf Daten, welche zum Training verwendet wurden, als auf bisher ungesehene Testdaten. Das liegt daran, dass das Training eines Neuronalen Netzes kein einmaliger, sondern ein iterativer Vorgang ist, während dem das Modell den (endlichen) Trainingsdatensatz in unterschiedlichen Konstellationen immer wieder neu bewerten muss. In der Regel ist das Ziel des Trainings eine möglichst gute Anpassung zwischen den Modellentscheidungen und den tatsächlich beobachteten Realisationen zu erreichen. Dafür wird am Ende jeder Trainingsiteration eine Verlustfunktion (engl. loss) berechnet, die die Abweichung zwischen geschätzten und tatsächlichen Werten misst. Während des Trainings werden die Parameter des Modells so verändert, dass die resultierende Verlustfunktion minimiert wird. Genau hier liegt allerdings ein zentrales Problem des überwachten Lernens. Wird ein Modell zu lange auf einem endlichen Datensatz trainiert, beginnt es irgendwann sich Trainingsdatenpunkte zu merken, um die Verlustfunktion weiter zu minimieren. Gleiches gilt für Modelle, deren effektive Kapazität - in Relation zum zu lösenden Problem - zu groß ist[2]. Anstatt Zusammenhänge in den Daten zu erkennen, lernt das Modell die Trainingsdaten und die zugehörigen Ausgaben zu replizieren, was zu einer sinkenden Generalisierungsfähigkeit auf bisher ungesehene Daten führt. Man spricht hier von *Overfitting* - einer Überanpassung des Modells an die gegebenen Daten.
Genau dieses Overfitting ist es, was sich der Membership Inference Angriff zu Nutze macht. Sobald das Modell zu overfitten beginnt, beginnt es auch damit sich Trainingsinstanzen zu merken. Wird das Modell nach abgeschlossenem Training dann wiederrum mit einem Trainingsdatum konfrontiert, wird es die Trainingsinstanz - vereinfacht gesprochen - wiedererkennen und, verglichen mit einem bisher ungesehenen Datenpunkt, mit einer höheren Konfidenz der jeweiligen Klasse zuordnen. Graphisch dargestellt ist dieser Effekt in Abb. 3. Es ist deutlich zu erkennen, dass die Modellprognosen bezüglich der bisher ungesehenen Datenpunkte (orange) mit einer höheren Unsicherheit behaftet sind, als bereits bekannte (Trainings-)Instanzen (blau).

Der gesamte Membership Inference Angriff läuft wie folgt ab: Zunächst befragt der Angreifer das Zielmodell wiederholt, um einen vollständigen Datensatz mit Eingabe und zugehöriger Ausgabe zu generieren. Anschließend wird ein sogenanntes *Schattenmodell* (engl. Shadow Model), welches das Zielmodell in seiner Funktionalität bestmöglich approximieren soll, auf einer Teilmenge ebendieses Datensatzes trainiert. Für das eigentliche *Angriffsmodell*, welches

[2] Die Kapazität eines Neuronalen Netzes lässt sich unter anderem über die Anzahl und Art der verwendeten Schichten (engl. layers) sowie Knoten (engl. nodes) innerhalb einer Schicht steuern. Die genaue Kapazität eines Neuronalen Netzes lässt sich allerdings nur schwer genauer quantifizieren

[3] `https://www.comp.nus.edu.sg/~reza/files/datasets.html`

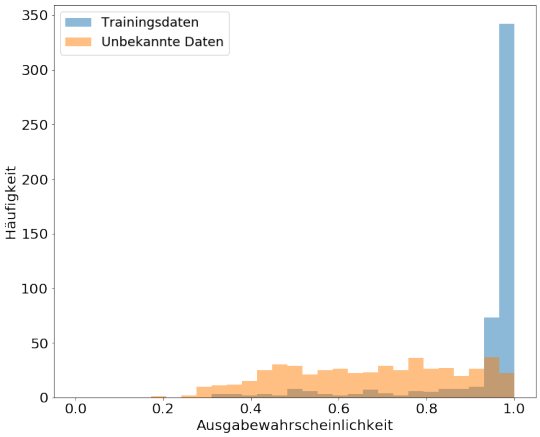

Abb. 3: Verteilungen der Ausgabewahrscheinlichkeiten nach Trainings- und unbekannten Referenzdaten (jeweils 500 Datenpunkte), Purchase10 Datensatz[3].

für die Erkennung der Trainingszugehörigkeit zuständig ist, muss nun zunächst noch das *Schattenmodell* mit den zuvor verwendeten Trainingsdaten und mit einem bisher ungesehenen Referenzdatensatz befragt werden. Die Ausgaben des Schattenmodells bezüglich dieser beiden Datensätze, zusammen mit einem binären Indikator, ob es sich bei dem jeweiligen Datenpunkt um eine Trainingsinstanz handelt oder nicht, dient dem Angriffsmodell als Trainingsdatensatz.

Um obiges Beispiel wieder aufzugreifen - nehmen wir an, das *Faception* Modell sei ausschließlich auf Bildern von Strafgefangenen trainiert worden[4]. Ein Angreifer, der im Besitz eines Portraitfotos ist, kann nun mittels Membership Inference herausfinden, ob die entsprechende Person evtl. bereits eine Haftstrafe verbüßen musste - vorausgesetzt natürlich, dass die fragliche Person tatsächlich Teil des Trainingsdatensatzes war. Dabei ist die Ausgabe des Angriffsmodells, ob Trainingsdatenpunkt oder nicht, unabhängig von der vom Zielmodell prognostizierten Klasse. Dem Angreifer geht es allein um das Verknüpfen der sensiblen Information mit der Person.

2.4 Model Extraction

Allgemein ist das Ziel eines Model Extraction Angriffs das Verhalten und somit die prädiktive Leistung eines Zielmodells auf einen bisher unbekannten Datensatz zu approximieren bzw. im günstigsten Fall zu kopieren. Alternativ kann es für den Angreifer auch Sinn machen die Architektur des Zielmodells zu *stehlen*. Aber warum ein Modell oder dessen Aufbau stehlen,

[4] Hierbei handelt es sich lediglich um eine hypothetische Annahme im Rahmen des Beispiels. Die Entwickler des Modells haben ihre Datengrundlage weder öffentlich kommuniziert noch zugänglich gemacht.

wenn es doch frei verfügbar bzw. nutzbar ist? Neuronale Netze können bspw. sehr komplexe Funktionen approximieren und auch Zusammenhänge in großen Datenmengen finden, die ansonsten vermutlich unbekannt geblieben wären. Die Entwicklung sowie das Training eines solchen Netzes sind häufig sehr zeit- und ressourcenintensiv. Zudem bedarf es einiges an tiefgehenden Verständnisses darüber, wie Neuronale Netze Informationen verarbeiten [Ja19; MDN19; OSF19; Pa19]. Nicht jeder, der solche Techniken anwenden möchte, verfügt über die nötige Rechenleistung, das Fachwissen oder die Menge an - potentiell sensiblen - Daten, die benötigt werden, um ein ML-Modell zuverlässig trainieren zu können. Wir identifizieren demnach drei verschiedene Motivationen, warum ein Angreifer ein Modell stehlen möchte:

- Machine Learning as a Service (MLaaS) Anbieter wie bspw. Google, Amazon oder Microsoft Azure ermöglichen via API-Zugang Zugriff auf hochperformante Modelle, die gegen eine geringe Nutzungsgebühr auf die eigenen Daten angewendet werden können. Je nachdem wie viele Datenpunkte der Angreifer vom Modell verarbeitet haben möchte, kann es sein, dass es für den Angreifer einen monetären Vorteil darstellt das Modell zu stehlen, sodass er es beliebig oft befragen kann, ohne weitere Gebühren für den Dienst zahlen zu müssen.
- Gleichsam kann ein gut performendes Modell auch einen Wettbewerbsvorteil darstellen, sodass ein Konkurrent ebenfalls Interesse an dessen Aufbau und Wirkungsweise hat.
- Die zuvor beschriebenen Angriffe – Model Inversion und Membership Inference – basieren beide darauf, dass der Angreifer entweder volle Einsicht in das Modell hat, und z.B. die Gradienteninformationen unmittelbar abgreifen kann, oder zumindest über eine hinreichend gute Approximation des Zielmodells verfügt. Dementsprechend kann Model Extraction als Vorstufe für die beiden Angriffe genutzt werden, um so deren jeweiligen Erfolgschancen zu erhöhen.

Ein weiterer Angriff auf Neuronale Netze, welcher hier nicht weiter betrachtet wurde, weil er keine direkte Bedrohung für die Privatheit darstellt, sind *Adversarial Examples*. Adversarial Examples sind kleine, für den Menschen in der Regel nicht wahrnehmbare Veränderungen in den Daten, die dazu führen, dass das Modell eine andere als die erwünschte Entscheidung trifft [GSS14]. Anwendungen des autonomen Fahrens verwenden fast ausschließlich Neuronale Netze, die basierend auf den gelieferten Eingaben - Video-, Sensor-, Radardaten – das zu lösende Problem in kleinere Klassifikationsaufgaben aufspalten. Nimmt eine Kamera bspw. ein Stopp-Schild auf, liefert das Neuronale Netz die Ausgabe 'anhalten'. Ein böswilliger Angreifer, dem es gelingt dieses Netz hinreichend mittels Model Extraction zu approximieren, ist nun in der Lage, basierend auf den Gradienteninformationen des gestohlenen Netzes, Adversarial Examples zu kreieren, die dazu führen, dass bspw. ein Stopp-Schild als ein Vorfahrtsschild missinterpretiert wird [Pa17]. Wird dieses Adversarial Example im öffentlichen Raum angebracht, wo es von anderen selbstfahrenden Vehikeln, die

das gleiche Modell wie das Zielmodell nutzen, erfasst werden kann, kann dies verheerende Konsequenzen haben.

Aktuell werden in der Literatur verschiedene Ansätze verfolgt, wie die Approximation eines unbekannten Modells mit möglichst wenig Vorinformationen am erfolgreichsten erfolgen kann. Nicht immer ist eine vollständige Extraktion des Zielmodells samt Architektur und verwendeten Hyperparametern das Ziel. Häufig beschränken sich die Angriffe auch nur darauf einzelne Komponenten zu ermitteln – bspw. ob eine Convolutional-Layer verwendet wurde oder welche Art der Regularisierung [WG18].

Am häufigsten sind sogenannte *Seitenkanalangriffe* (engl. side-channel attack) [Ba18; Du18; HZS18] sowie eine Form der *Knowledge-Distillation* zu finden. Bei letzterem wird versucht das Wissens eines Modells durch wiederholtes Befragen in ein separates, meist kleineres Modell zu überführen. Häufig werden dazu adaptive Verfahren (sog. *learning strategies*) verwendet. Diese identifizieren Trainingsdatenpunkte mit möglichst hohem Informationsgehalt, sodass die Anzahl der Anfragen, die an das Zielmodell gerichtet werden müssen, um ein aussagekräftiges Modell zu destillieren, möglichst klein gehalten werden kann [Co18; Ja19; MDN19; Pa19]. Letzteres ist aus mehreren Gründen relevant für den Angreifer. Zum einen werden durch weniger Anfragen geringere Kosten in einem Pay-per-Use System erzeugt, zum anderen ist die Gefahr, dass der Angriff als solcher identifiziert wird geringer, je weniger Anfragen an das Zielmodell gestellt werden und je weniger Zeit der Angriff an sich benötigt. In diesem Kontext ist auch die Arbeit von Oh et al. [OSF19] zu nennen, die einen Meta-Klassifikator trainiert haben, welcher anhand der Ausgabe des Zielmodells die verwendeten Hyperparameter, wie z.B. die Tiefe des Netzes, bestimmen sollte. In einer Erweiterung ihres Ansatzes gibt das Zielmodell durch Befragung mit einem speziell konstruierten Adversarial Example sogar selbst Informationen über die eigenen Modellspezifikationen preis.

3 Maßnahmen zum Schutz der Privatheit

Basierend auf dem Verständnis der in den vorhergehenden Abschnitten beschriebenen Risiken ist es möglich, Schutzmechanismen gegen diese zu entwerfen. Ziel ist es, aus trainierten Modellen keine oder nur eine tolerierbar geringe Menge an Informationen über die Trainingsdaten extrahieren zu können. Allgemein können solche Schutzmechanismen auf die Datengrundlage selbst, wie auch auf die Trainings- bzw. Entscheidungsphase angewendet werden. Es gilt allerdings zu beachten, dass i.d.R. alle Maßnahmen zum Schutz der Privatheit mit einer Reduktion der Prognosequalität des Modells einhergehen. Bei allen Gegenmaßnahmen gilt es folglich den klassischen Trade-Off zwischen Datennutzen und Datensicherheit genauestens abzuwägen. Die im folgenden beschriebenen Verfahren stellen keine vollständige Auflistung an Schutzmechanismen, sondern nur einige Beispiele dar, wie ein Modell bereits privatheiterhaltend trainiert werden kann.

3.1 Differential Privacy

Differential Privacy ist keine fixe Methode, sondern vielmehr eine Eigenschaft, welche verlangt, dass es für eine beliebige Analyse irrelevant ist, ob ein bestimmtes Datensubjekt im Datensatz enthalten ist oder nicht – in beiden Fällen sollten sich die Ausgabeverteilungen nicht signifikant voneinander unterscheiden. Intuitiv bedeutet dies, dass die Menge an Informationen, die maximal über ein bestimmtes Individuum herausgefunden werden kann, beschränkt wird. Differential Privacy wird in der Regel durch das Hinzufügen von Rauschen erreicht. Der Grad der Perturbation hängt von der Stärke des Einflusses des einzelnen Eintrags auf den Datensatz ab [Dw09]. Eine mögliche Ausprägung ist die ϵ-Differential Privacy. Der Parameter ϵ kontrolliert in diesem Fall wie groß der maximale Effekt eines Individuums auf das Ergebnis einer Analyse ist. Im Umkehrschluss quantifiziert ϵ aber auch in wie weit die Privatheit eines Individuums durch die Analyse kompromittiert werden kann. Kleinere ϵ–Werte gehen daher mit einem höheren Grad an Privatheit, aber auch mit einer stärkeren Perturbation einher, was sich wiederum negativ auf die Qualität der Daten auswirkt [Dw06]. Erschwerend kommt hinzu, dass keine feste Richtlinie existiert, wie der Parameter ϵ optimal gewählt werden soll. Die Wahl reicht von $\epsilon = 0.01$ bis $\epsilon = 1$ in akademischer Forschung bis hin zu Werten zwischen 1 und 10 in industriellen Anwendungsfällen (Google, Apple, US Census Bureau) [PCN18]. Wenngleich Differential Privacy im Vergleich zu vielen anderen privatheiterhaltenden Verfahren nachweislich Anonymität garantiert, so können bereits kleine ϵ- Werte in einem erheblichen Verlust an Klassifizierungsgenauigkeit (engl. accuracy) resultieren, was den Einsatz von Differential Privacy in der Praxis eher schwierig gestaltet [DKM19; DR14].

3.2 Adversarial Regularization

Anstatt wie im herkömmlichen Training eines Neuronalen Netzes nur eine einzelne Zielfunktion zu optimieren, wird bei der *Adversarial Regularization* der Trainingsprozess um einem feindlichen Angreifer erweitert, welcher seinen eigenen Gewinn zu maximieren versucht. Folglich werden bei der *Adversarial Regularization* zwei Modelle - das eigentliche Modell M_{Orig} und ein Angreifermodell M_{Att} - mit sich widersprechenden Zielfunktion trainiert [NSH18]. Dieses Vorgehen ist vergleichbar mit dem Training eines *Generative Adversarial Networks* (GAN). Das Klassifikationsmodell M_{Orig} berechnet die Wahrscheinlichkeit, dass ein Input zu einer beliebigen Klasse gehört und versucht dabei den Vorhersagefehler zu minimieren. Das Ziel des Angriffmodells M_{Att} besteht darin, die Genauigkeit des eigenen Klassifizierungsproblems - ob der betrachtete Datenpunkt bereits zum Training des Zielmodells genutzt wurde oder nicht - zu maximieren (vgl. Membership Inference Angriff, Abschnitt 2.3). Um die Privatheit zu schützen, wird der Gewinn des Angreifermodells als Regularisierungsterm in die zu minimierende Verlustfunktion des Klassifikationsmodells M_{Orig} integriert. Durch die Optimierung der beiden sich widersprechenden Zielfunktionen ist es dem Angreifer nicht möglich ein besseres Angriffsmodell zu entwickeln als jenes,

welches bereits vom Zielmodell antizipiert wurde. Dadurch hilft der Regularisierungsterm dem Trade-off zwischen Privatheit und Klassifizierungsgüte gerecht zu werden. Die Autoren [NSH18] wiesen nach, dass die Optimierung dieses Min-Max-Problems nachweislich vor Membership Inference Angriffen schützt, da der Trainingsalgorithmus im Optimalfall zu einem Gleichgewichtspunkt konvergiert, in welchem der beste Angriff auf die private Information, also ob der Datenpunkt zum Training verwendet wurde, nicht besser als eine zufällige Schätzung ist. Auf Trainingsdatenpunkten basierende Vorhersagen des Modells können folglich nicht von Vorhersagen unterschieden werden, die auf Basis bisher ungesehener Datenpunkte aus derselben Verteilung gemacht wurden. Zusätzlich zu der so erreichten Robustheit gegen Membership Inference Angriffe weist das trainierte Modell im Vergleich zu herkömmlich trainierten Modellen nur einen minimalen Verlust an Klassifikationsgenauigkeit auf.

3.3 Distillation

Distillation wurde ursprünglich von Hinton et al. [HVD15] vorgestellt. Die Idee ist es, wie bereits erwähnt (vgl. Abschnitt 2.4) und wie der Name andeutet, die Essenz des erlernten Wissens eines Modells in ein separates Modell zu überführen. Dieses Vorgehen kommt einer Komprimierung gleich und kann sogar auch - vor allem wenn ein Ensemble an Modellen gegeben ist - genutzt werden, um die Genauigkeit zu erhöhen. Zudem können mittels Distillation höhere Standards in Bezug auf Datenschutz erreicht und somit die Erfolgschancen von Model Inversion und Membership Inference verringert werden. Das bereits trainierte Modell wird hierbei als „Lehrer“ angesehen, von dem ein neues - das destillierte - Student-Modell, lernt. Dazu wird eine nicht zwangsweise annotierte weitere Datenmenge, meist disjunkt von der ursprünglichen Trainingsmenge, durch den Lehrer klassifiziert. Der Student trainiert nun auf ebendieser Datenmenge und erkennt dabei die Ausgabevektoren des Lehrers als zu erlernende Wahrheit (engl. Ground-Truth) an. Einfach ausgedrückt approximiert der Student den Lehrer, wobei er während des Trainingsprozesses nicht in Kontakt mit den ursprünglichen Trainingsdaten kommt. Distillation stellt somit ein datenschutzkonformes Instrument dar, da der Student keinen Zugriff auf potentiell sensible Informationen aus dem Trainingsset hat [Pa18].

4 Diskussion und Ausblick

Maschinelle Lernverfahren werden häufig genutzt, um z.B. unseren Alltag zu vereinfachen oder um Prozessabläufe zu optimieren. Dass sie dabei ein nicht zu vernachlässigendes Risiko für die Privatheit des Einzelnen bergen, ist in den seltensten Fällen bewusst. In dem vorliegenden Artikel wurden drei aktuelle Angriffe auf die Privatheit bzw. geistiges Eigentum vorgestellt sowie einige Gegenmaßnahmen skizziert. Es wurde aufgezeigt, dass gerade jene zentralen Eigenschaften, welche maschinelle Lernverfahren so wertvoll und nützlich machen, auch genau jene sind, die von Angreifern ausgenutzt werden können, um

an sensible Informationen zu gelangen.
Wenngleich die hier vorgestellten Angriffe durchaus eine reelle Gefahr darstellen, so besteht dennoch Optimierungspotential. Vielversprechende Richtungen für zukünftige Forschungen sind demnach privatheiterhaltender Maßnahmen sowie deren Umsetzung in der Praxis. Viele der heute angewendeten Maßnahmen skalieren schlecht oder sind nur für die Anwendung auf einen bestimmten Lernalgorithmus optimiert und auf andere ML-Verfahren nur schwer bis gar nicht anwendbar. Darüber hinaus entstehen durch das Schützen sensibler Informationen immer Kosten - ob in Form von längeren Trainingszeiten oder, dass die Daten signifikant an Nutzen einbüßen. Diese Kosten können unter Umständen so hoch ausfallen, dass in der Praxis aus ökonomischen Gründen von der Verwendung privatheiterhaltender Maßnahmen abgesehen wird [Do18; WBH19].

Literatur

[Ba18] Batina, L. et al.: CSI neural network: Using side-channels to recover your artificial neural network information, 2018, arXiv: 1810.09076.

[BR18] Biggio, B.; Roli, F.: Wild patterns: Ten years after the rise of adversarial machine learning. Pattern Recognition 84/, S. 317–331, 2018.

[BV19] Baldi, P.; Vershynin, R.: The capacity of feedforward neural networks. Neural networks 116/, S. 288–311, 2019.

[Co18] Correia-Silva, J. R. et al.: Copycat CNN: Stealing knowledge by persuading confession with random non-labeled data. In: 2018 International Joint Conference on Neural Networks (IJCNN). IEEE, S. 1–8, 2018.

[DKM19] Dwork, C.; Kohli, N.; Mulligan, D.: Differential Privacy in Practice: Expose your Epsilons! Journal of Privacy and Confidentiality 9/2, 2019.

[Do18] Doebel, I. et al.: Maschinelles Lernen. Eine Analyse zu Kompetenzen, Forschung und Anwendung, Techn. Ber., Fraunhofer-Gesellschaft, Muenchen, 2018.

[DR14] Dwork, C.; Roth, A.: The algorithmic foundations of differential privacy. Foundations and Trends® in Theoretical Computer Science 9/3–4, S. 211–407, 2014.

[Du18] Duddu, V. et al.: Stealing neural networks via timing side channels, 2018, arXiv: 1812.11720.

[Dw06] Dwork, C. et al.: Calibrating noise to sensitivity in private data analysis. In: Theory of cryptography conference. Springer, S. 265–284, 2006.

[Dw09] Dwork, C.: The differential privacy frontier. In: Theory of Cryptography Conference. Springer, S. 496–502, 2009.

[Fa19] Faception: Facial Personality Analysis, https://www.faception.com, Accessed: 2020-04-08, 2019.

[FJR15] Fredrikson, M.; Jha, S.; Ristenpart, T.: Model inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. S. 1322–1333, 2015.

[Go14] Goodfellow, I. et al.: Generative adversarial nets. In: Advances in neural information processing systems. S. 2672–2680, 2014.

[Go16] Goodfellow, I. et al.: Deep Learning. MIT press Cambridge, 2016.

[GSS14] Goodfellow, I. J.; Shlens, J.; Szegedy, C.: Explaining and Harnessing Adversarial Examples, 2014, arXiv: 1802.08908.

[HG16] Harmanci, A.; Gerstein, M.: Quantification of private information leakage from phenotype-genotype data: linking attacks. Nature methods 13/3, S. 251, 2016.

[HVD15] Hinton, G.; Vinyals, O.; Dean, J.: Distilling the knowledge in a neural network, 2015, arXiv: 1503.02531.

[HZS18] Hua, W.; Zhang, Z.; Suh, G. E.: Reverse engineering convolutional neural networks through side-channel information leaks. In: 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC). IEEE, S. 1–6, 2018.

[Ja19] Jagielski, M. et al.: High-fidelity extraction of neural network models, 2019, arXiv: 1909.01838.

[Ka19] Karras, T. et al.: Analyzing and improving the image quality of stylegan, 2019, arXiv: 1912.04958.

[KK+12] Khan, M. E.; Khan, F. et al.: A comparative study of white box, black box and grey box testing techniques. Int. J. Adv. Comput. Sci. Appl 3/6, 2012.

[KLA19] Karras, T.; Laine, S.; Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. S. 4401–4410, 2019.

[MDN19] Mosafi, I.; David, E. O.; Netanyahu, N. S.: Stealing knowledge from protected deep neural networks using composite unlabeled data. In: 2019 International Joint Conference on Neural Networks (IJCNN). IEEE, S. 1–8, 2019.

[NS08] Narayanan, A.; Shmatikov, V.: Robust De-anonymization of Large Sparse Datasets, How to break anonymity of the Netflix Prize dataset. In: IEEE S&P 2008. IEEE Computer Society Conference Publishing Services (CPS), 2008.

[NSH18] Nasr, M.; Shokri, R.; Houmansadr, A.: Machine learning with membership privacy using adversarial regularization. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. S. 634–646, 2018.

[OSF19] Oh, S. J.; Schiele, B.; Fritz, M.: Towards reverse-engineering black-box neural networks. In: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. Springer, S. 121–144, 2019.

[Pa17] Papernot, N. et al.: Practical black-box attacks against machine learning. In: Proceedings of the 2017 ACM on Asia conference on computer and communications security. S. 506–519, 2017.

[Pa18] Papernot, N. et al.: Scalable private learning with pate, 2018, arXiv: 1802.08908.

[Pa19] Pal, S. et al.: A framework for the extraction of deep neural networks by leveraging public data, 2019, arXiv: 1905.09165.

[PCN18] Page, H.; Cabot, C.; Nissim, K.: Differential privacy an introduction for statistical agencies, Techn. Ber., NSQR. Government Statistical Service, 2018.

[Sa18] Salem, A. et al.: ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models, 2018, arXiv: 1806.01246.

[Sa19] Sablayrolles, A. et al.: White-box vs black-box: Bayes optimal strategies for membership inference, 2019, arXiv: 1908.11229.

[Sh17] Shokri, R. et al.: Membership inference attacks against machine learning models. In: 2017 IEEE Symposium on Security and Privacy (SP). IEEE, S. 3–18, 2017.

[ST17] Shwartz-Ziv, R.; Tishby, N.: Opening the black box of deep neural networks via information, 2017, arXiv: 1703.00810.

[St19] StyleGAN: Official TensorFlow Implementation, https://github.com/NVlabs/stylegan, Accessed: 2020-04-08, 2019.

[WBH19] Winter, C.; Battis, V.; Halvani, O.: Herausforderungen für die Anonymisierung von Daten. In (David, K. et al., Hrsg.): INFORMATIK 2019. Gesellschaft für Informatik e.V., Bonn, S. 339–352, 2019.

[WG18] Wang, B.; Gong, N. Z.: Stealing hyperparameters in machine learning. In: 2018 IEEE Symposium on Security and Privacy (SP). IEEE, S. 36–52, 2018.