Intro zu Adversarial Examples

aka Adversarial Attacks

Pascal Müller
Freie Universität Berlin

26. März 2020

Zusammenfassung

Neuronale Netzwerke sind eine vielversprechende Technologie für die Lösung einer ganzen Reihe von Problemen. Ihre Fähigkeiten im Bereich der Bildklassifizierung werden nicht selten mit den visuellen Fähigkeiten von Menschen verglichen. Aber nicht in allen Fällen hält dieser Vergleich einer genaueren Untersuchung stand, z.B. im Fall von Adversarial Examples (auch Adversarial Attacks genannt). Adversarial Examples sind Bilder mit für das menschliche Auge kaum wahrnehmbaren Veränderungen, die aber einen großen Einfluss auf die Prozesse eines neuronalen Netzwerks haben können. Mit diesem Text möchte ich interessierten Lesern einen Einblick in das Phänomen von Adversarial Examples geben, ohne dabei große Vorkenntnisse zu erwarten. Für interessierte Leser mit etwas mehr Vorkenntnissen habe ich einige Fußnoten hinzugefügt.

1 Einleitung

1.1 Neuronale Netzwerke

Für alles was der Mensch in seiner Umwelt beobachtet baut er sich gedankliche Modelle auf. Vereinfachungen der Realität, die es ihm erlauben, Ergebnisse von zukünftigen Situationen vorherzusehen. Menschen extrahieren allgemeine Regeln aus konkreten Situationen und wenn ihre Vorhersagen zu oft falsch sind, dann passen sie ihre gedanklichen Modelle an. Menschen lernen.

Das Trainieren eines neuronalen Netzwerkes wird auch oft als Lernen bezeichnet, denn es weist einige Ähnlichkeiten mit dem menschlichen Lernprozess auf. Auch in einem neuronalen Netwerk wird ein verallgemeinertes Modell der Realität festgehalten und dieses Modell wird auch dann angepasst, wenn seine Vorhersagen zu oft falsch sind. [1] Aber an dieser Stelle hören die Gemeinsamkeiten auch schon auf. Viel eher ist ein neuronales Netzwerk eine mächtiges Werkzeug zur **Funktions-Approximation**, wobei die approximierte Funktion die der Realität zugrunde liegenden Regeln sind.

Technisch gesehen ist ein neuronales Netzwerk lediglich eine Funktion, die aus Additionen, Multiplikationen und ein oder zwei anderen [2] grundlegenden Operatoren besteht. Allerdings ist anzumerken, dass es sich bei einem üblichen neuronalen Netzwerk um eine Verkettung von einigen Millionen dieser Operatoren handelt. Das ist auch der Hauptgrund dafür, dass neuronale Netzwerke erst in der jüngeren Vergangenheit an Popularität gewonnen haben, obwohl die theoretischen Grundlagen schon Mitte des letzten Jahrhunderts geschaffen wurden. Die Rechenleistung die nötig ist, um eine solche Masse an Berechnungen sinnvoll durchzuführen, exisiterte einfach lange Zeit nicht.

1.2 Bild-Klassifizierung

Wenn einem Menschen ein Bild von einem Toaster gezeigt wird, kann er es (vermutlich) als Element der Klasse Toaster erkennen, man sagt er hat das Bild *als Toaster klassifiziert*. Bilder zu klassifizieren ist eine sehr nützliche Fähigkeit, für die es auch in Computerprogrammen viele Anwendungsfelder gibt, z.B. bei Autonomem Fahren, Medizinischen Diagnosetechniken oder Personenerkennung. Leider ist es nicht ganz trivial, einem Computerprogramm diese Fähigkeit beizubringen.

Das Forschungsfeld der Computer Vision hat viele Techniken hervorgebracht, die zur Bildklassifizierung genutzt werden können, z.B. die Viola-Jones-Methode, Generalized Hough-Transform oder HOG. Diese Techniken besitzen im Gegensatz zu neuronalen Netzwerken den Charme, dass man sie gut nachvollziehen kann. Allerdings sind sie in ihrer Fleixbilität eingeschränkter als neruonale Netzwerke. Daher werden in vielen Anwendungsgebieten, in denen früher Computer-

[1] Das Trainieren eines neuronalen Netzwerkes ist ein iterativer Prozess, bei dem das Netzwerk zu Anfang die Ergebnisse rät und anschließend korrigiert wird, wenn es falsch geraten hat. Die Korrektur erfolgt mittels Gradienten-Abstieg `https://en.wikipedia.org/wiki/Gradient_descent`

[2] Damit dieser Fakt nicht fehlt: Neben Addition und Multiplikation sind insbesondere nichtlineare Operatoren ein wichtiger Teil eines neuronalen Netzwerks

Vision-Techniken genutzt wurden, heute neuronale Netzwerke verwendet.

Wie kann man ein neuronales Netzwerk zur Bild-Klassifizierung verwenden? Wie Eingangs erwähnt, sind neuronale Netzwerke im Grunde Werkzeuge zu Funktionsapproximation. Auch Bild-Klassifizierung kann als eine Funktion betrachtet werden, wobei die Inputs die Pixelwerte des Bildes sind und der Output die Klassifizierung. Der Output wird meist so aufgebaut, dass für jede mögliche Klasse ein **confidence-Wert** zwischen 0% und 100% ausgeben wird, je höher der Wert, desto sicherer ist sich das Netzwerk quasi, dass das Bild dieser bestimmten Klasse zuzuordnen ist.[3]

1.3 Adversarial Examples

Die Leistungsfähigkeit von neuronalen Netzwerken im Bereich von Bild-Klassifizierung ist erstaunlich [4]. Es gibt aber Bilder, die für einen Menschen und/oder ein neuronales Netzwerk aus verschiedenen Gründen schwierig zu klassifizieren sind [5]. Meistens sind die Gründe aber erkennbar, umso erstaunlicher ist es also, wenn der Mensch ein Bild sehr eindeutig einer Klasse zuschreibt, aber das neuronale Netzwerk das Bild völlig abwegig klassifiziert.

In der freien Wildbahn findet man auch einige Bilder mit dieser Eigenschaften [6]. Es gibt aber auch Methoden, mit denen man Bilder so verändern kann, dass dieser Effekt künstlich hervorgerufen wird. Auf diese Weise veränderte Bilder werden als **Adversarial Examples** bezeichnet. Die Änderungen an dem Bild sind für das menschliche Auge kaum wahrnehmbar [7], aber auf das neuronale Netzwerk haben sie einen großen Einfluss, so dass die Klassifizierung des Menschen und die des Netzwerkes auf einmal nicht mehr übereinstimmen. Ein Beispiel: In Abbildung 3 sieht man links das als *panda* klassifizierte Original-Bild und rechts ein kaum wahrnehmbar verändertes Adversarial Example, das aber vom Netzwerk mit hoher confidence als *gibbon* klassifiziert wird.

Es gibt mittlerweile eine große Fülle an Techniken, um Adversarial Examples für Bild-Klassifizierungs-Netzwerke zu produzieren. Dabei wird meistens eins von zwei Zielen verfolgt: Das Netzwerk soll die Bilder *irgend einer* falschen Klasse zuordnen, oder das Netzwerk soll die Bilder *einer bestimmten* falschen Klasse zuordnen. Das erste Ziel wird auch **Evasion** genannt, das zweite **Impersonation**.

2 Ausgewählte Beispiele

Die zwei folgenden Beispiele sind keine Adversarial Examples im klassischen Sinne, wie in Abbildung 3, bei denen kaum sichtbare Veränderungen am Bild

[3] Ein Detail das nicht unerwähnt bleiben sollte: Eigentlich werden Werte zwischen 0 und 1 ausgegeben, die durch Softmax o.Ä. als Wahrscheinlichkeit interpretierbar sind.

[4] `https://paperswithcode.com/sota/image-classification-on-imagenet`

[5] Gute Übersicht über solche Gründe: `http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/`

[6] Prominentes Beispiel: `https://twitter.com/jackyalcine/status/615329515909156865`

[7] z.B. weil nur wenige Pixel verändert werden (L_0-Regularisierung) oder weil die Pixelwerte nur um kleine Beträge verändert werden (L_∞-Regularisierung)

(a) (b) (c) (d)

Abbildung 1: (a) Evasion, die beiden Gesichter mit Brillen werden nur mir 1% bzw. 3% confidence als ihre jeweils richtige Klasse erkannt. (b),(c) und (d) Impersonation, das obere Gesicht mit Brille wird mit 78%, 75% bzw. 90% confidence als die Klasse des unteren Bildes erkannt. Abbildung und Werte stammen aus dem Paper [3].

durchgeführt wurden. Hier soll statt dessen die Robustheit von Adversarial Examples in der realen Welt und so die Relevanz dieses Themas aufgezeigt werden.

2.1 Brille vs. Gesichtserkennung

In diesem Paper [3] von 2016 geht es um Gesichtserkennung mit neuronalen Netzwerken. Die Klassen sind hier die Identitäten von Personen, also je höher die confidence des Netzwerkes für eine Klasse, desto sicherer ist sich das Netzwerk dass es sich um diese Person handelt.

Die Autoren des Papers haben mit Adversarial-Example-Techniken Muster für Brillenrahmen produziert. Das Muster ist für jedes Original-Gesicht (und, falls das Ziel Impersonation ist, für jedes Ziel-Gesicht) individuell. Für Beispiele aus dem Paper, siehe Abbildung 1.

2.2 Sticker vs. Bildklassifizierung

In diesem Paper [6] von 2018 geht es um klassische Bildklassifizierung. Aber die Autoren produzieren Adversarial Examples nicht wie üblich durch Veränderung beliebiger Pixelwerte eines Original-Bildes. Statt dessen entfernen sie einen kleinen Bereich des Bildes komplett und finden dann neue Pixelwerte für diesen Bereich, welche die confidence der Klassifizierung des gesamt-Bildes als Ziel-Klasse möglichst groß werden lassen (das Ziel ist hier also Impersonation). Das Resultat sind als Sticker verwendbare kleine Bilder, siehe dazu Abbilidung 2. [8]

[8] Um die Wirkung der Sticker möglichst invariant gegenüber Translation und Rotation zu machen, wurden zusätzliche Schritte unternommen. Details dazu im Paper [6]

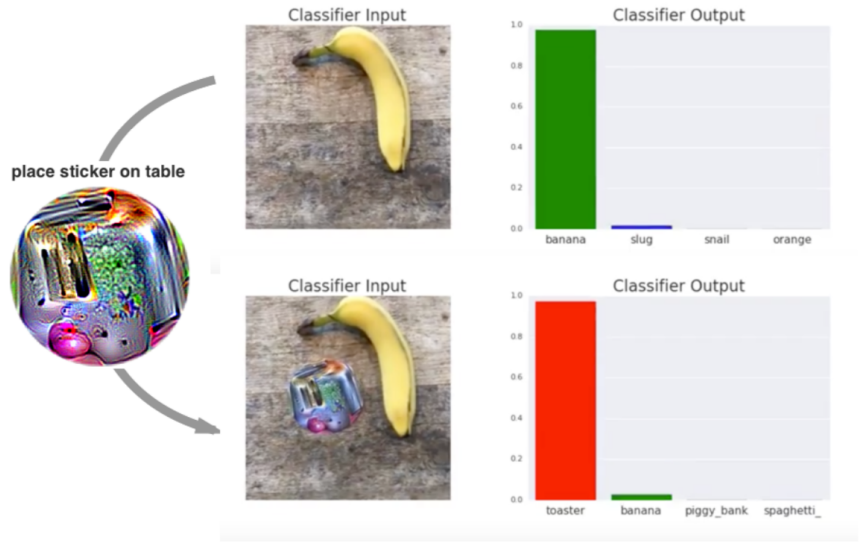

Abbildung 2: Impersonation, der Sticker wurde produziert um die confidence der Klasse toaster zu maximieren. Das untere Bild ist nur an der Stelle verändert, an der der toaster-Sticker hinzugefügt wurde, insbesondere wurde kein Teil der Banane überklebt. Trotzdem verändert sich die confidence von banana von $\sim$ 100% zu $\sim$ 0%, während die confidence der Ziel-Klasse toaster $\sim$ 0% auf $\sim$ 100% steigt. Abbildung und Werte stammen aus dem Paper [6]

3 Technische Umsetzung

Es ist bedeutend einfacher, Adversarial Examples zu produzieren, wenn man alle Details des zu täuschenden Netzwerkes kennt. Die Situation, in der man die vollständige Kontrolle über ein Neuronales Netzwerk hat, wird als **White-Box**-Situation bezeichnet. Das Gegenstück dazu ist die **Black-Box**-Situation, bei der man dem Netzwerk lediglich Anfragen schicken und die Ausgaben betrachten kann. Hinzu kommt eventuell, dass in dieser Situation nur eine begrenzte Menge an Abfragen an das Netzwerk geschickt werden kann, und/oder die Abfragen hohe Latenz haben (z.B. wenn es sich um eine online verfügbare Test-Schnittstelle handelt). Dass es überhaupt möglich ist, Adversarial Examples in einer Black-Box-Situtation zu produzieren, wird dem Phänomen der **Black-Box-Transferability** [1] [4] von Adversarial Examples zugeschrieben.

3.1 Eine White-Box-Technik: FGSM

FGSM [2] steht für Fast Gradient Sign Method. Mit FGSM lassen sich Adversarial Examples für Bild-Klassifizierungs-Netzwerke produzieren. Ziel der Methode ist Evasion, also ein modifiziertes Bild zu produzieren, das möglichst nicht der Klasse zugeordnet wird, der es eigentlich zugeordnet werden sollte. Dabei werden die Pixel-Werte des Original-Bildes um maximal eine Einheit verändert. Eine Einheit entspricht dabei einer der 256 Stufen des R-, G- oder B-Wertes eines Pixels. Die Methode würde auch für größere Abstufungen funktionieren, aber so ist gewähleistet, dass das modifizierte Bild für das menschliche Auge

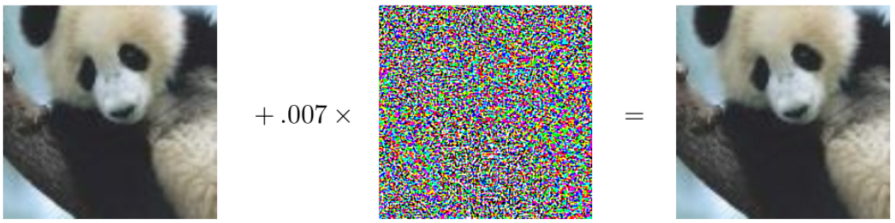

Abbildung 3: FGSM in Aktion. Die untere Zeile zeigt jeweils die Klasse mit der höchsten confidence, wenn man das Bild darüber in das Netzwerk gibt. Abbildung ist aus dem Paper [2]

nur schwer vom Original-Bild zu unterscheiden ist.

Gegeben seien ein Datenpunkt x, die korrekte Klasse dieses Datenpunktes y und ein Netzwerk θ [9] das einem Datenpunkt mit einer gewissen confidence einer Klasse y' zuordnet [10] . Außerdem eine Loss-Funktion J, die berechnet, wie schlecht die Klassifizierungsgüte von θ war. J lässt sich also als Abstandsmaß zwischen der richtigen Klasse y und der vom Netzwerk zugeordneten Klasse y' eines Datenpunktes interpretieren.

Unser Ziel ist es nun, diesen Abstand so groß wie möglich zu machen, also ein x' zu finden, so dass $J(\theta, x', y)$ maximal wird (oder zumindest zu vergrößern, indem wir uns einem Maximum von J annähern). Dazu führen wir auf dem Bild einen einzelnen Schritt des Gradienten-Aufstiegs-Verfahrens durch, wobei wir für den Schritt nur die Vorzeichen der Werte des Gradienten beachten. Im Detail:

1. Berechne die Ausgabe der Loss-Funktion $J(\theta, x, y)$
2. Berechne den Gradienten bzgl. der Eingabepixel $\nabla_x J(\theta, x, y)$
3. Berechne die Vorzeichen $sign(\nabla_x J(\theta, x, y))$
4. Addiere (mit kleinem ϵ) auf das Original-Bild $x + \epsilon * (sign(\nabla_x J(\theta, x, y)))$

Das ganze lässt sich als *addiere auf jeden Pixel +1 oder -1, je nachdem welche der beiden Optionen $J(\theta, x', y)$ vergrößert* zusammenfassen. Für ein Beispiel aus dem Paper, siehe Abbildung 3.

3.2 Eine Black-Box-Technik: Oracle Attack Method

In der Black-Box-Situation kann man dem Ziel-Netzwerk nur Bilder schicken und erhält die entsprechenden Klassifizierungen zurück. Die Idee der Oracle

[9] Präzise gesagt sind mit θ die Parameter des Netzwerkes gemeint
[10] Es klassifiziert den Datenpunkt als Klasse y'

Attack Method [5] ist, ein Ersatz-Netzwerk zu bauen, das dem Ziel-Netzwerk möglichst ähnlich ist. Da man dieses Ersatz-Netzwerk vollständig kontrolliert, kann man nun eine White-Box-Technik (z.B. FGSM) verwenden, um Adversarial Examples für dieses Netzwerk zu produzieren. Die so produzierten Adversarial Examples funktionieren wegen Black-Box-Transferability meist auch für das Ziel-Netzwerk.

Hier ein schematischer Ablauf der Methode:

1. Erschaffe ein synthetisches Datenset [11]
2. Labele das synthetische Datenset mit Hilfe des Ziel-Netzwerks
3. Trainiere ein Ersatz-Netzwerk auf diesem Datenset
4. Generiere White-Box Adversarial Examples für das Ersatz-Netzwerk

4 Persönliches Fazit

Adversarial Examples, wie neuronale Netzwerke, sind ein Feld das grade noch in den Anfängen steckt. So finde ich es nur natürlich, dass es noch keine vollständig akzeptierten Erklärungsversuche [8] [9] und damit einhergehend auch noch kein allgmeines Gegenmittel [7] für Adversarial Examples gibt. Die Robustheit von Adversarial Examples in der realen Welt, die durch Experimente, wie die in Abschitt 2 vorgestellten, gezeigt wurde, und die Techniken die es erlauben Adversarial Examples auch in einer Black-Box-Situation herstellen zu können, wie die aus Abschnitt 3.2, sollten aber bei der Integration von neuronalen Netzwerken in realen Anwendungen nicht vernachlässigt werden.

Literatur

[1] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow and Rob Fergus. Intriguing properties of neural networks. `https://arxiv.org/pdf/1312.6199.pdf`, 2013.

[2] Ian J. Goodfellow, Jonathon Shlens and Christian Szegedy. Explaining and Harnessing Adversarial Examples. `https://arxiv.org/pdf/1412.6572.pdf`, 2014.

[3] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer and Michael K. Reiter. Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition. `https://www.cs.cmu.edu/~sbhagava/papers/face-rec-ccs16.pdf`, 2016.

[4] Nicolas Papernot, Patrick McDaniel and Ian Goodfellow. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. `https://arxiv.org/pdf/1605.07277.pdf`, 2016.

[11] Nicht einfach irgendwelche Daten, es wichtig dass die Daten die Entscheidungsgrenzen des Ziel-Netzwerkes gut abbilden. Im Paper [5] gibt es Details dazu.

[5] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik and Ananthram Swami. Practical Black-Box Attacks against Machine Learning. https://arxiv.org/pdf/1602.02697.pdf, 2016.

[6] Tom B. Brown, Dandelion Mané, Aurko Roy, Martin Abadi and Justin Gilmer. Adversarial Patch. https://arxiv.org/pdf/1712.09665.pdf, 2018.

[7] Xiaoyong Yuan, Pan He, Qile Zhu and Xiaolin Li. Adversarial Examples: Attacks and Defenses for Deep Learning https://arxiv.org/pdf/1712.07107.pdf, 2018.

[8] Adi Shamir, Itay Safran, Eyal Ronen, and Orr Dunkelman. A Simple Explanation for the Existence of Adversarial Examples with Small Hamming Distance. https://arxiv.org/pdf/1901.10861.pdf, 2019.

[9] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, Aleksander Madry. Adversarial Examples Are Not Bugs, They Are Features. https://arxiv.org/pdf/1905.02175.pdf, 2019.