TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

informatiCup
Der GI-Wettbewerb für Studierende

Bilderzeugende Verfahren zum Angriff einer Verkehrsschilder erkennenden KI

Leonhard Applis · Peter Bauer · Andreas Porada · Florian Stöckl

Agenda

- Deep Neural Networks
- Aufgabenstellung
- Herausforderung
- Angriffsmöglichkeiten
- Lösungsansätze
  - Greyboxing
  - Degeneration
  - Gradient Ascent
  - Saliency Maps
- Zusammenfassung

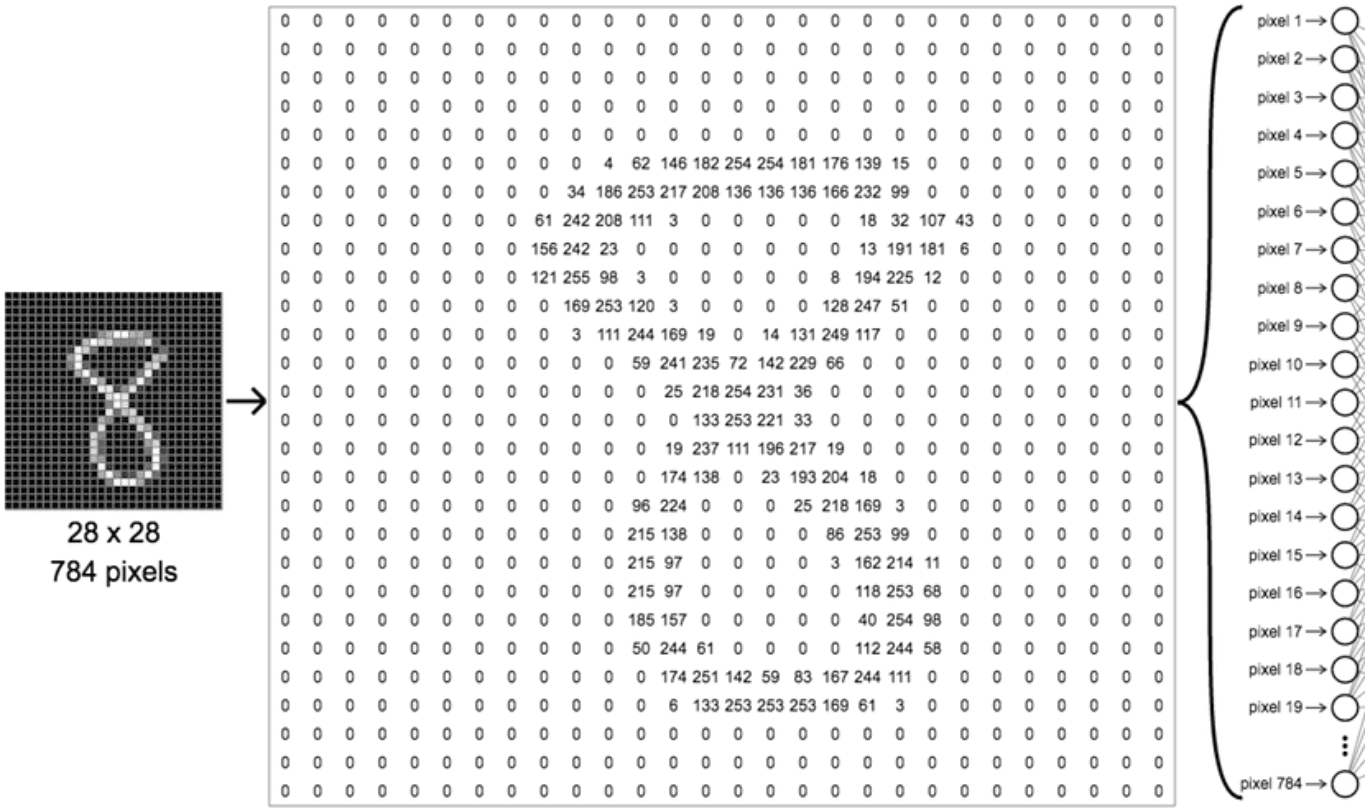
Deep Neural Networks (DNNs)

- Auf dem Rechner simulierte Neuronennetze
- Netz aus künstlichen Neuronen
- Versuch einer Maschine Intelligenz zu geben, indem menschliches Gehirn nachempfunden wird
- Einsatz zur Lösung technischer Probleme, wie der Mustererkennung

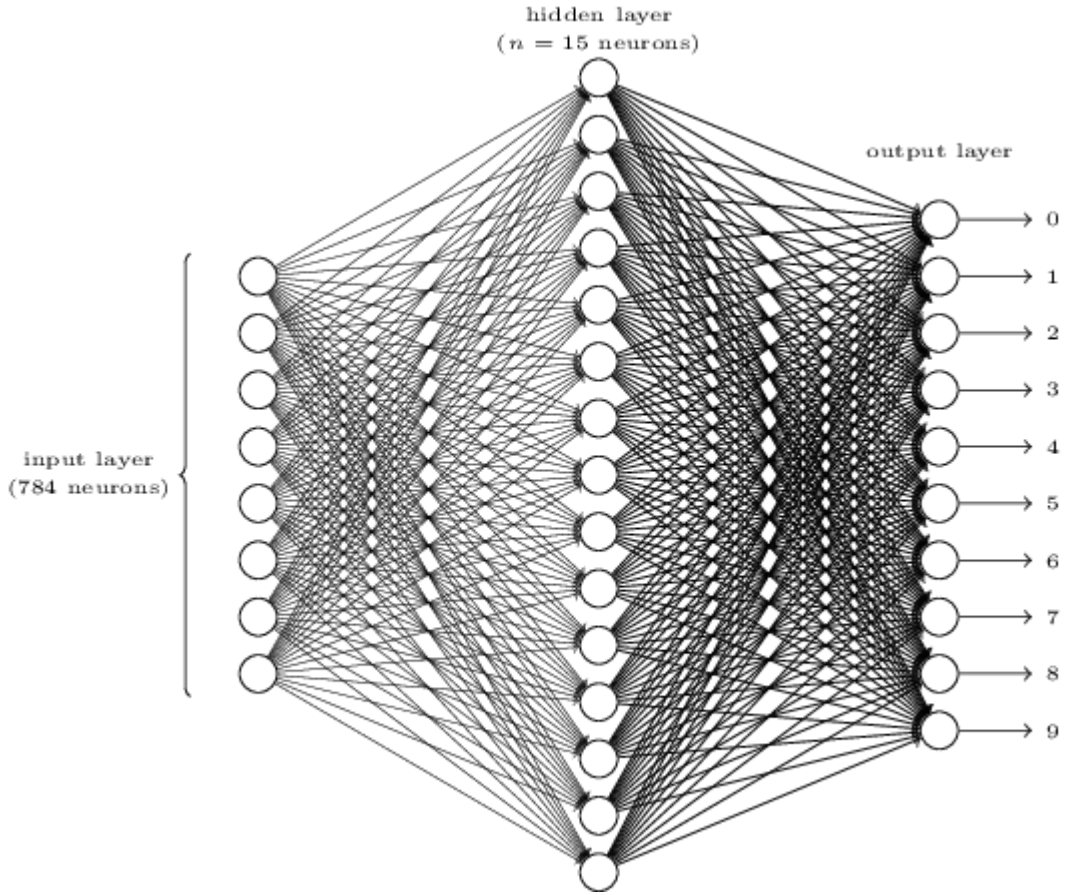
Aufbau eines DNN

- Besteht aus:
  - Neuronenmodell: Eingangs-Ausgangsbeziehung eines Neurons, viele Eingänge → ein Ausgang, Neuron selbst entspricht Wert zwischen 0 und 1
  - Verknüpfungsstruktur: Verbindung der künstlichen Neuronen, Gewichtung der einzelnen Neuronen
  - Lernregel: Gibt an, wie sich die Verbindungen oder Gewichtung zwischen Neuronen ändern

Beispiel Zahlenerkennung



Beispiel Zahlenerkennung

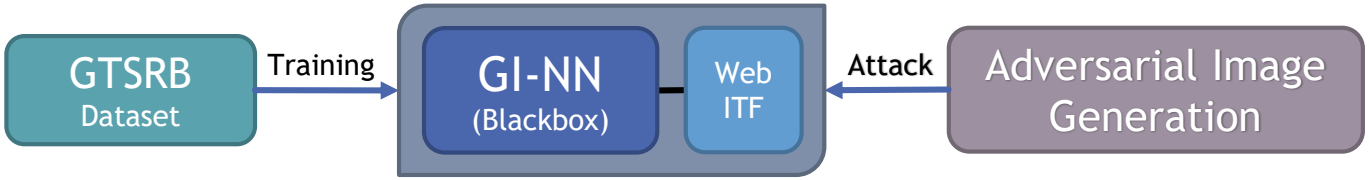


$$A = \sigma * (\omega_1 * a_1 + \omega_2 * a_2 + \omega_3 * a_3 + \ldots + \omega_n * a_n \text{ - bias})$$

Aufgabenstellung

- Aufgabe des InformatiCup 2019 der Gesellschaft für Informatik
- Bilderzeugende Verfahren zum Angriff einer Verkehrsschilder erkennenden KI entwicklen
- Es dürfen nur Bilder für Angriffe verwendet werden, die nicht vom Menschen als Straßenschild klassifiziert werden
- Müssen mindestens mit Konfidenz >90% von KI als Straßenschild klassifiziert werden
- KI ist eine Blackbox, erreichbar über Web-API
- Neuronenmodell, Verknüpfungsstruktur und Lernregeln sind unbekannt
- Hinweis der Aufgabenstellung: KI wurde mit GTSRB-Datensatz trainiert
  - 43 Klassen mit verschiedenen deutschen Straßenschildern
  - Enthält mehr als 50.000 reale Bilder (Trainings- + Testbilder)

Herausforderung

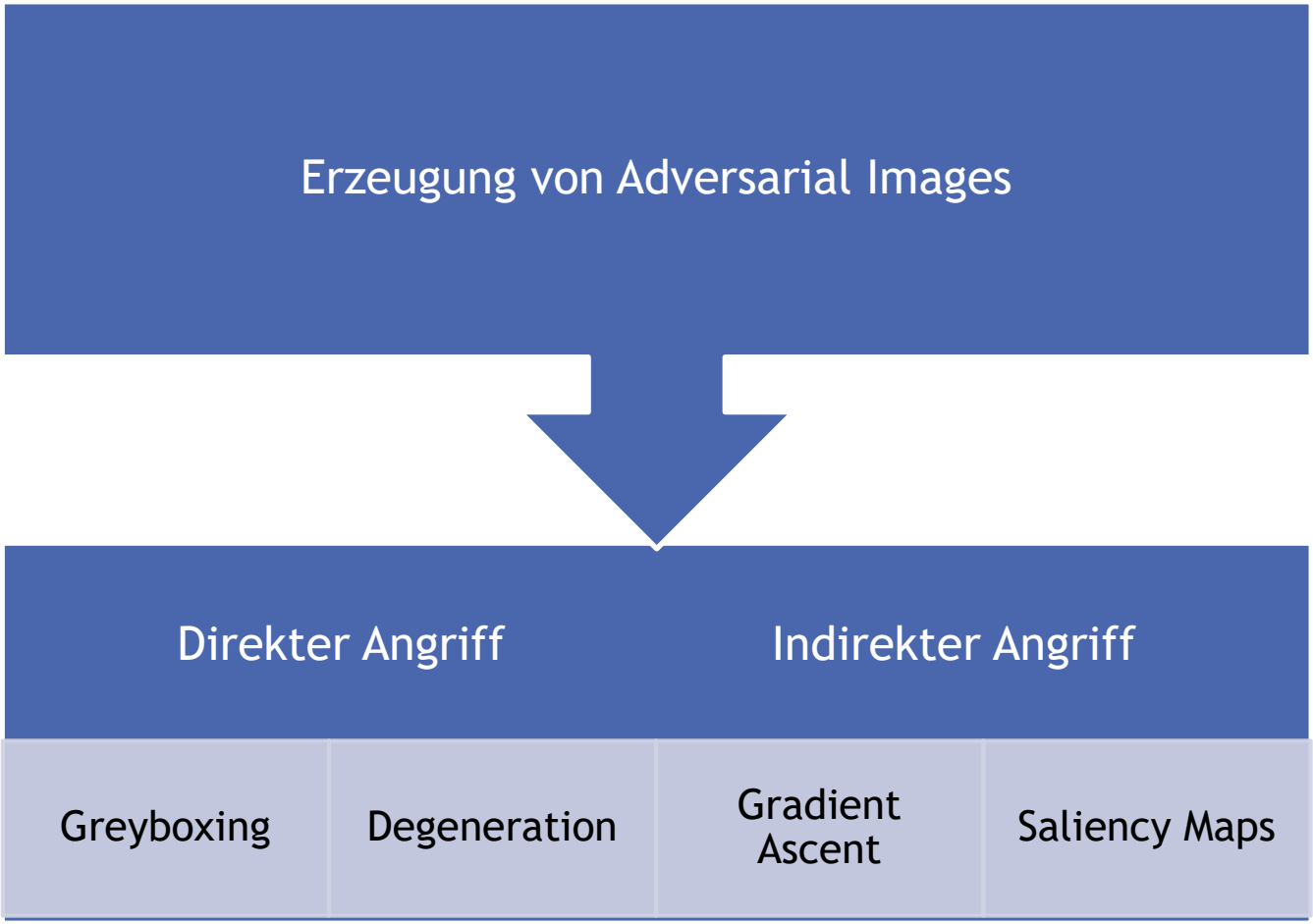


- Web Interface (ITF):
  - Request: **1 PNG-Bild** (64 x 64) **pro Anfrage**
  - Response: **5 Verkehrsschildklassen und -konfidenzen**
  - Limitierung: **1 Request pro Sekunde**

- Erkenntnisse:
  - **GI-NN** wurde mit **34 von 43 GTSRB-Klassen** trainiert
  - **Correct Classification Rate** (CCR): **87,10 %**
  - **Architektur des GI-NN** ist weiterhin unbekannt

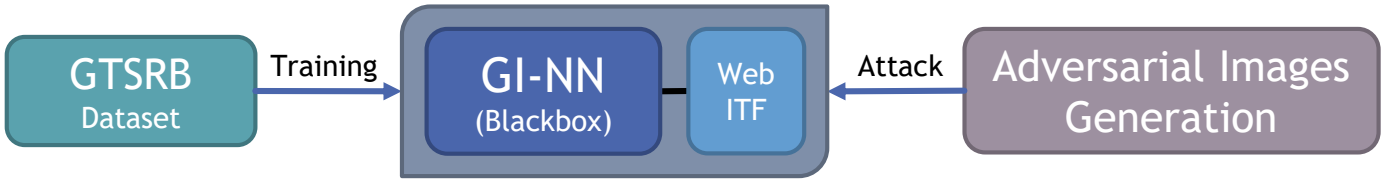
- **Wie lassen sich Adversarial Images erzeugen?**

Lösungsansätze



Angriffsmöglichkeit I: Direkter Angriff

1. Verfahren zur Erzeugung von Adversarial Images:
   - Greyboxing
   - Degeneration
2. Angriff des GI-NN via Web Interface:
   - anhand der Adversarial Images



Greyboxing

1. Erzeugen von zufälligen Bildern

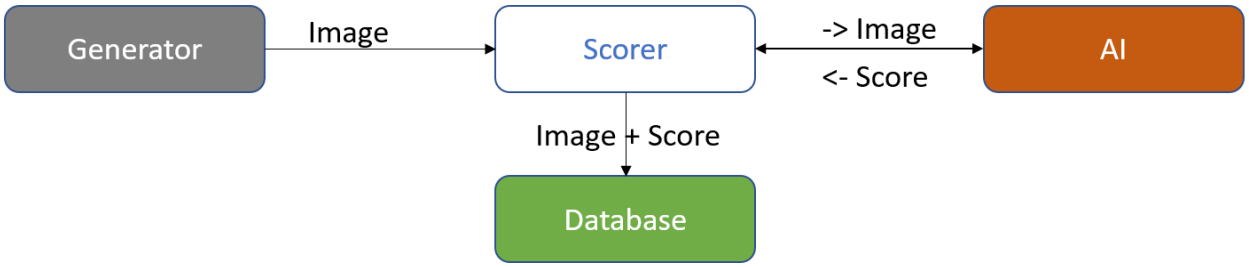
   Senden an Schnittstelle

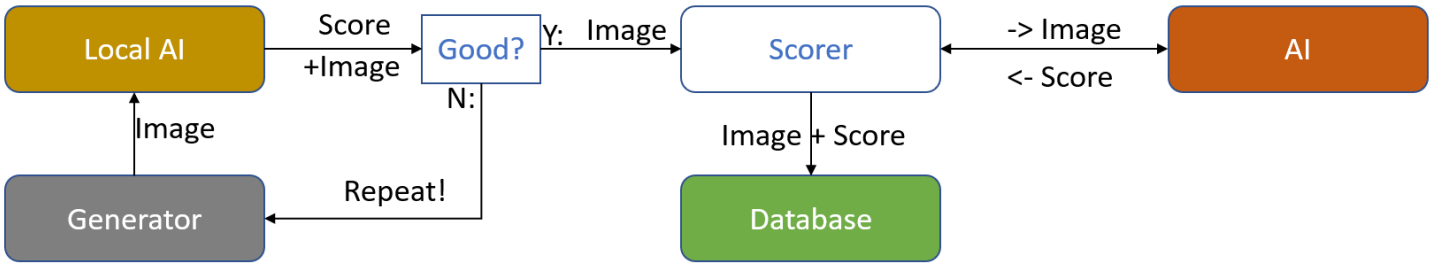
   Speichern der Remote-Klassifizierung
2. Trainieren einer lokalen AI mit <Bild,Remote-Score>

   => lokale AI soll sich verhalten wie die „Echte“
3. Bewertung der zufälligen Bilder an der eigenen AI

   => Nur gute Bilder gehen an die Remote Schnittstelle
4. Neu von der Schnittstelle bewertete Bilder fließen zurück ins Training
5. Lokale AI wird iterativ ähnlicher zur Remote AI

Greyboxing: Implementierung





Greyboxing: Ergebnisse

Überhaupt keine!

Es konnte keine lokale AI erzeugt werden,
die besser performte als ein *Münzwurf*.
(Verhältnis der gegebenen Klassen)

Problemanalyse:

- Zufälliges Bild erhält „echtes“ Label
  -> Zuordnung Bild und Label ist *zufällig*
- Initiales Netz ist mit zufälligen Gewichten initiiert
- Netz bewertet Bild, gibt zufälliges Ergebnis (Netz ist *frisch*)
- Loss Function errechnet Gradienten aus zufälligem Bild
  -> Zufälliger Gradient
- Zufälliges Netz + Zufälliger Gradient = Zufälliges Netz
- Trainingsalgorithmen *geben auf* und
  geben die statistisch häufigste Klasse wieder

Greyboxing: Lösungsansätze

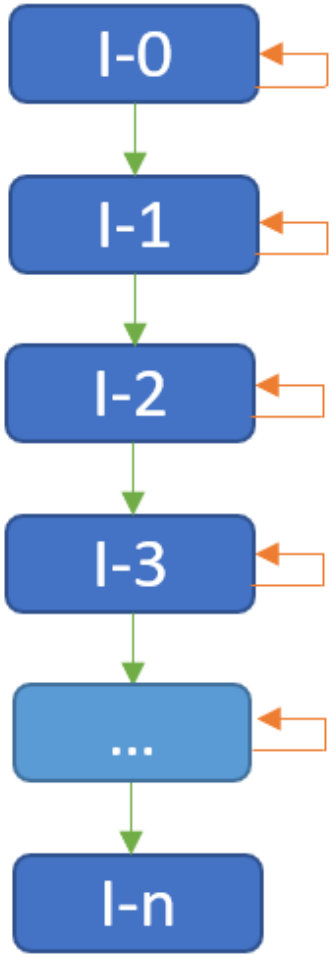
Anderer Generator:

- Bilder per Webcrawler
- Weniger zufällige Bilder
  z.B. Kreise, Ringe, etc.
  „Weniger Abstand zwischen einzelnen Bildern“
- Verzerrung echter Straßenschilder

Transfer Learning:

- Echte AI erkennt Formen
  Unsere AI konnte das nie Lernen
- Benutzung der ersten Schichten einer anderen AI, damit Feature-Erkennung *klappt*
- Nur *Verzerrung* der letzten Schicht

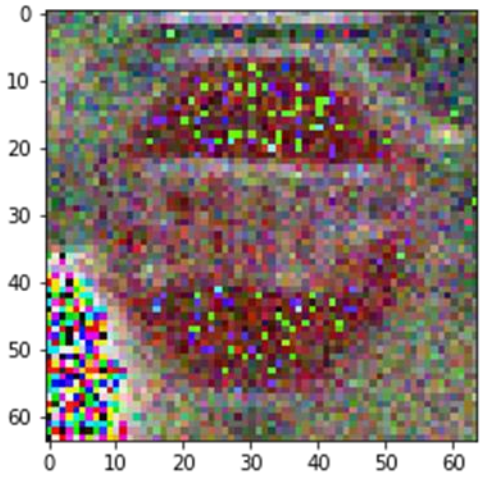
Degeneration



1. Eingabe eines Ausgangsbildes: **Echtes Verkehrsschild** (I-0)
2. Verrauschen des Schildes
3. Senden an Schnittstelle
   a) Weiterhin als Verkehrsschild erkannt?
   Weiter benutzen (Grüner Pfeil)
   b) Zu niedrige Konfidenz?
   Entferne Rauschen
   und wdh. ab Schritt 2 (Oranger Pfeil)
4. Wiederhole bis n-Wiederholungen erreicht

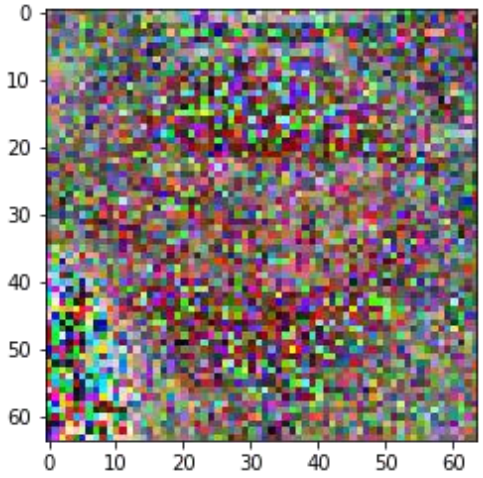
Degeneration: Ergebnisse

Tiefe 500



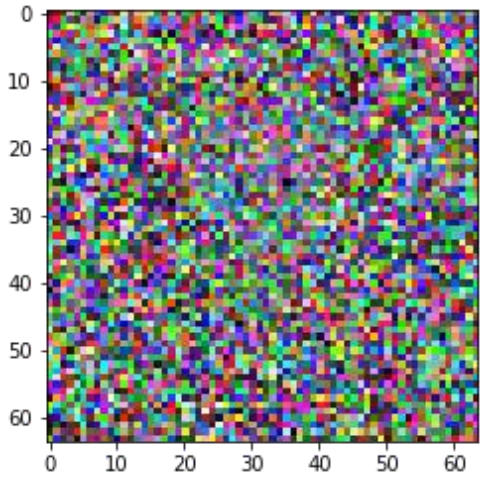
~97%

Tiefe 2500



~95%

Tiefe 5000



~92%

Degeneration: Vor- und Nachteile

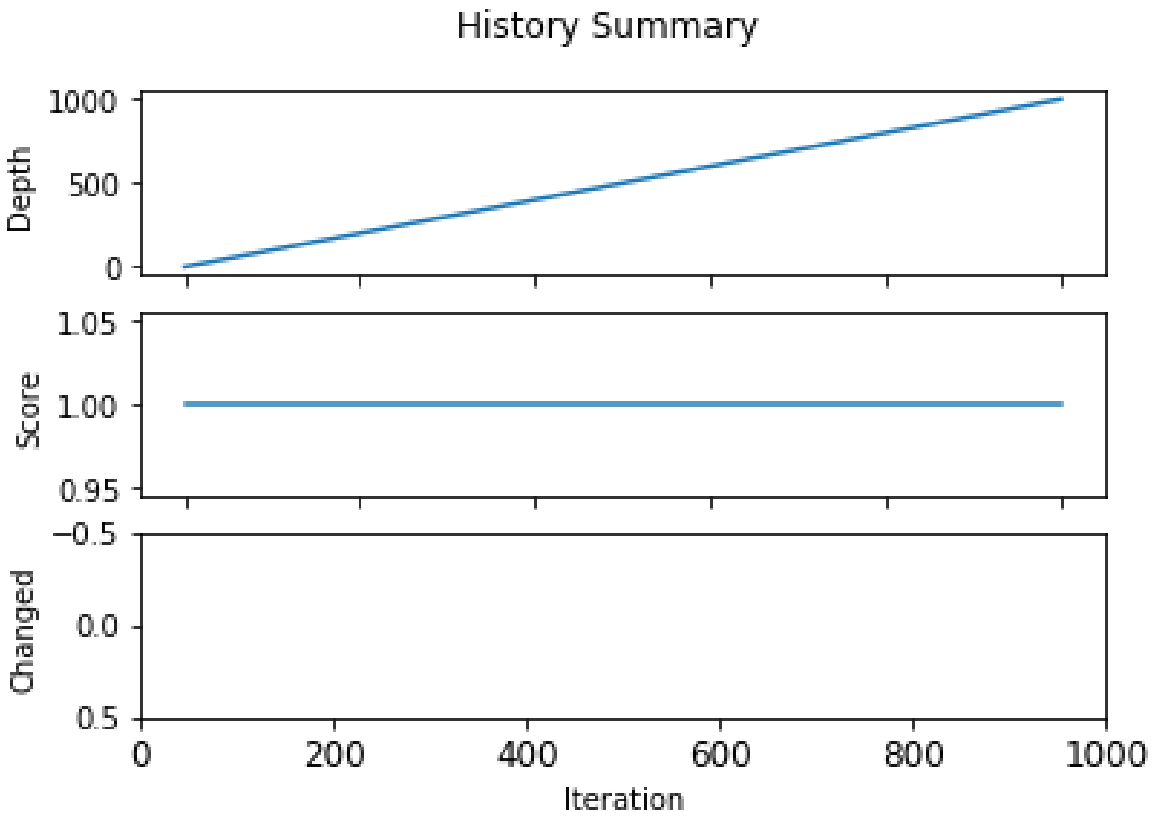
Vorteile:

- Einfache Implementierung (ca. 100 Zeilen Code)
- Modellunabhängig
- Ebenfalls für Random Forests, SVMs, etc.
- Ergebnisse beliebig gut je nach Zeitaufwand
- Zwischenergebnisse wieder aufgreifbar

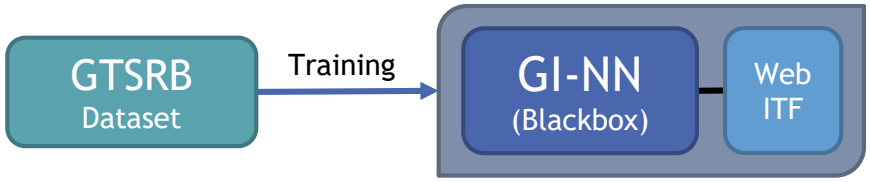
Nachteile:

- Lange Laufzeit (Lokal ca. 3 Minuten pro Bild, Remote 1-2 Stunden)
- Längere Laufzeit bei besseren Netzen
- Lokal ggfs. längere Laufzeiten auf schlechter Hardware
- Feintuning der Rauschfunktion erforderlich

Degeneration: Verbesserung durch Batch-Processing



Angriffsmöglichkeit II: Indirekter Angriff

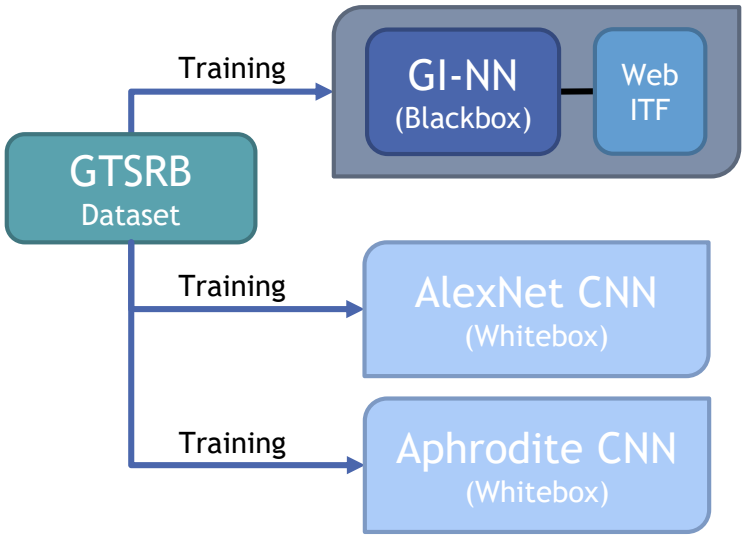


- Im Bereich **Computer Vision** werden **CNNs** am Häufigsten zur Feature Extraction und Classification eingesetzt [1]
  - Annahme: Hinter der GI-NN verbirgt sich ein CNN

- **Transferierbarkeit von Angriffen zwischen CNNs** [1]
  - Identischer Trainingsdatensatz
  - Unterschiedliche NN-Architektur
  - Ähnliche „Verwundbarkeit“
    - Annahme: Angriff des GI-NN mit Surrogat CNNs möglich

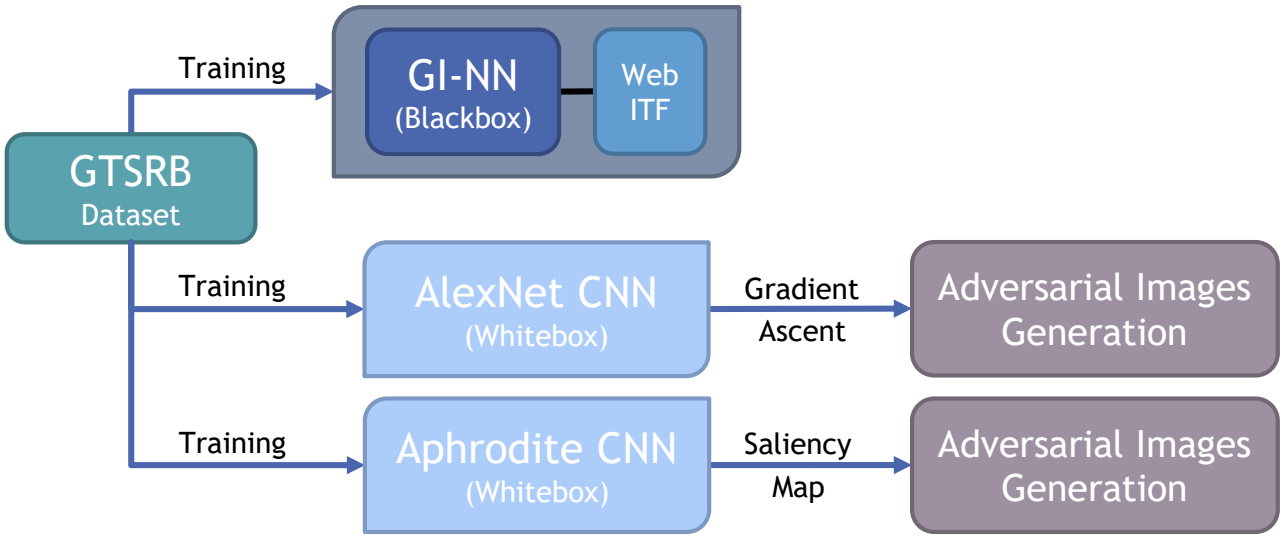
Angriffsmöglichkeit II: Indirekter Angriff

1. Training von Surrogat CNNs mit GTSRB Datensatz:
   - „AlexNet CNN“
   - „Aphrodite CNN“



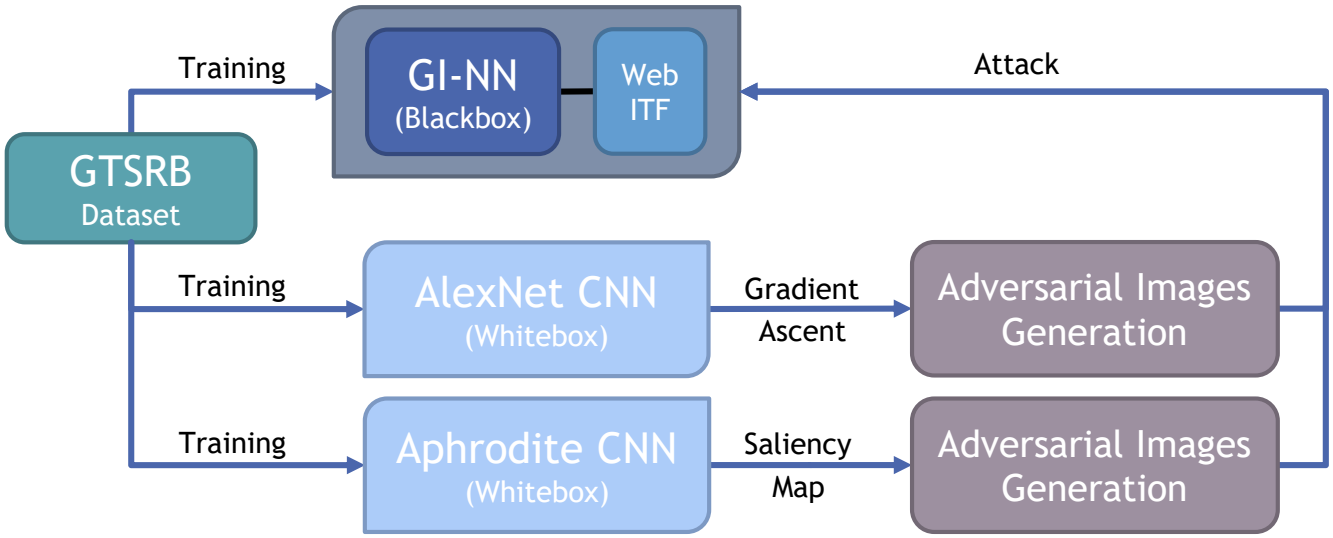
Angriffsmöglichkeit II: Indirekter Angriff

2. Verfahren zur Erzeugung von Adversarial Images:
   - Gradient Ascent [2] (AlexNet CNN [8])
   - Saliency Map [3-7] (Aphrodite CNN)



Angriffsmöglichkeit II: Indirekter Angriff

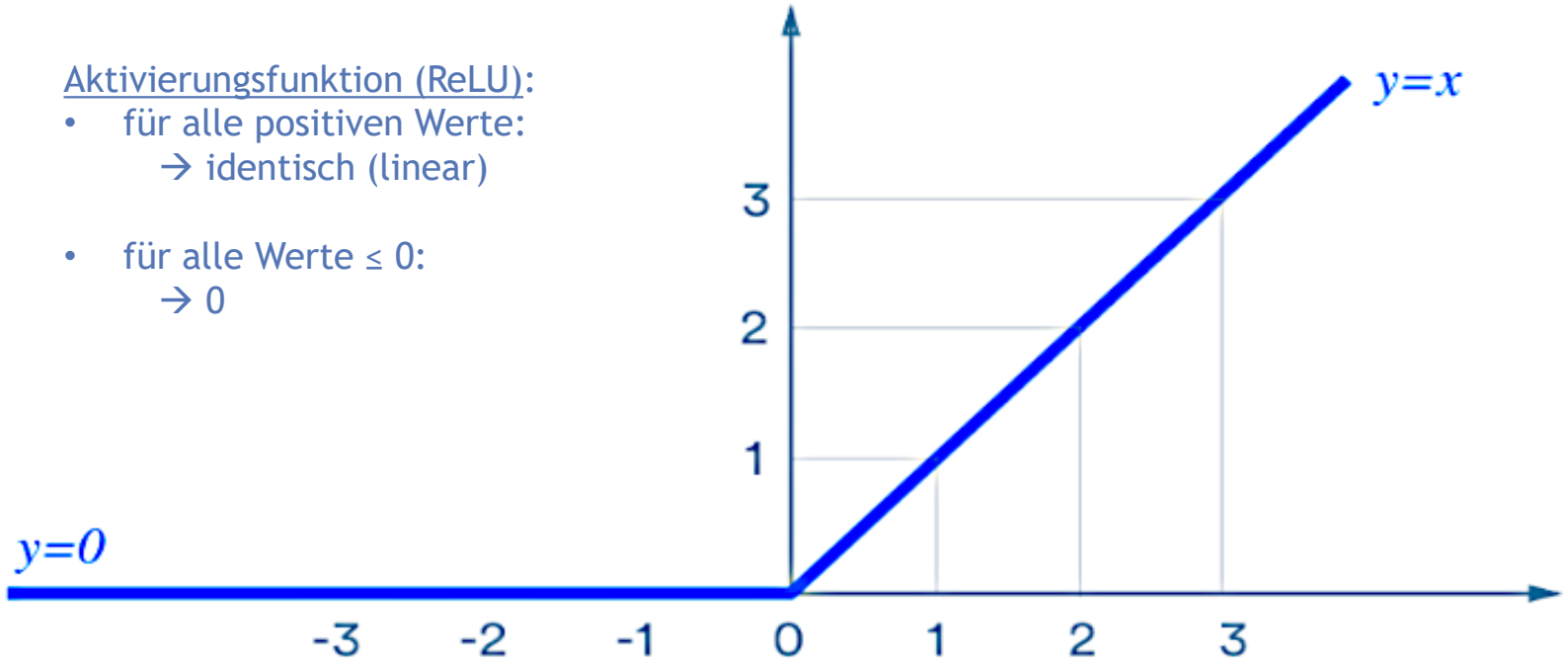
3. Angriff des GI-NN via Web Interface:
   - anhand der jeweils erzeugten Adversarial Images



- Vorteile:
  - Direkter Zugriff auf die NN-Architektur und -Parameter
  - Umgehung des Web Interface (Schnelligkeit)

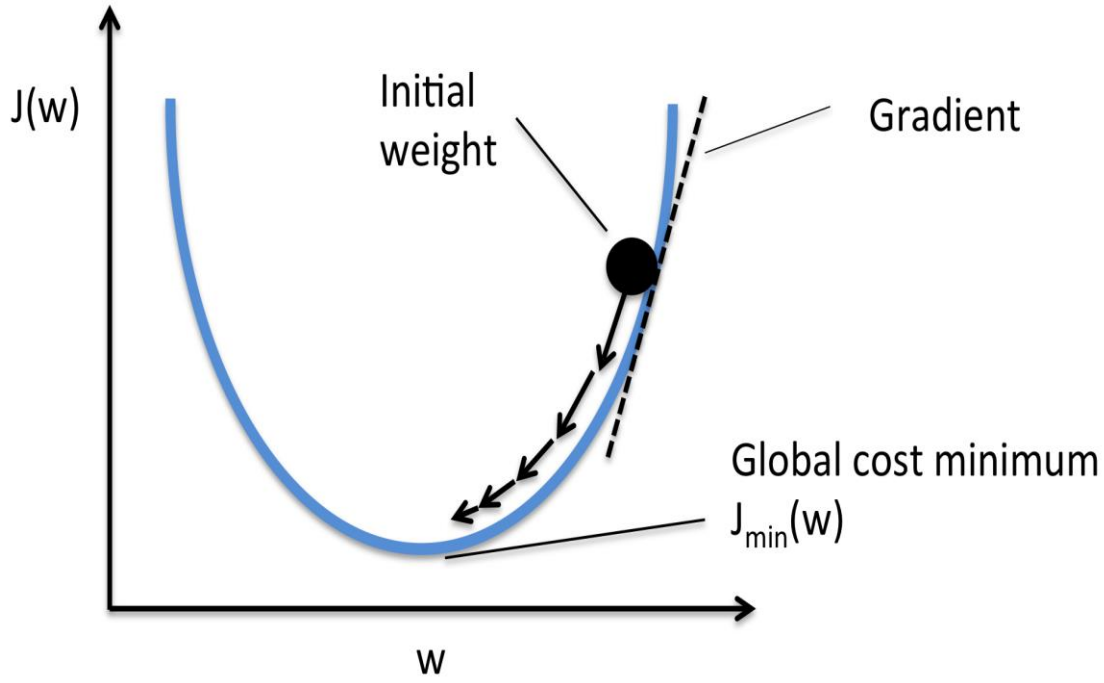
Gradient Ascent: AlexNet CNN Architektur [8]

Aktivierungsfunktion (ReLU):
- für alle positiven Werte:
  → identisch (linear)
- für alle Werte ≤ 0:
  → 0



- Input: GTSRB-Trainingsdatensatzbilder („Raw Image Pixel Values“; *Shape* 64 x 64 x 3)
- Feature Learning: Low-Level-Features (Punkte, Linien) → High-Level Features (Kanten, Rundungen)
- Classification: Abbildung von nicht-normierter Ausgabe [0,+∞] → 43 Verkehrszeichenklassen
- Output: Class Scores (z.B. stop <-> 0.93548)

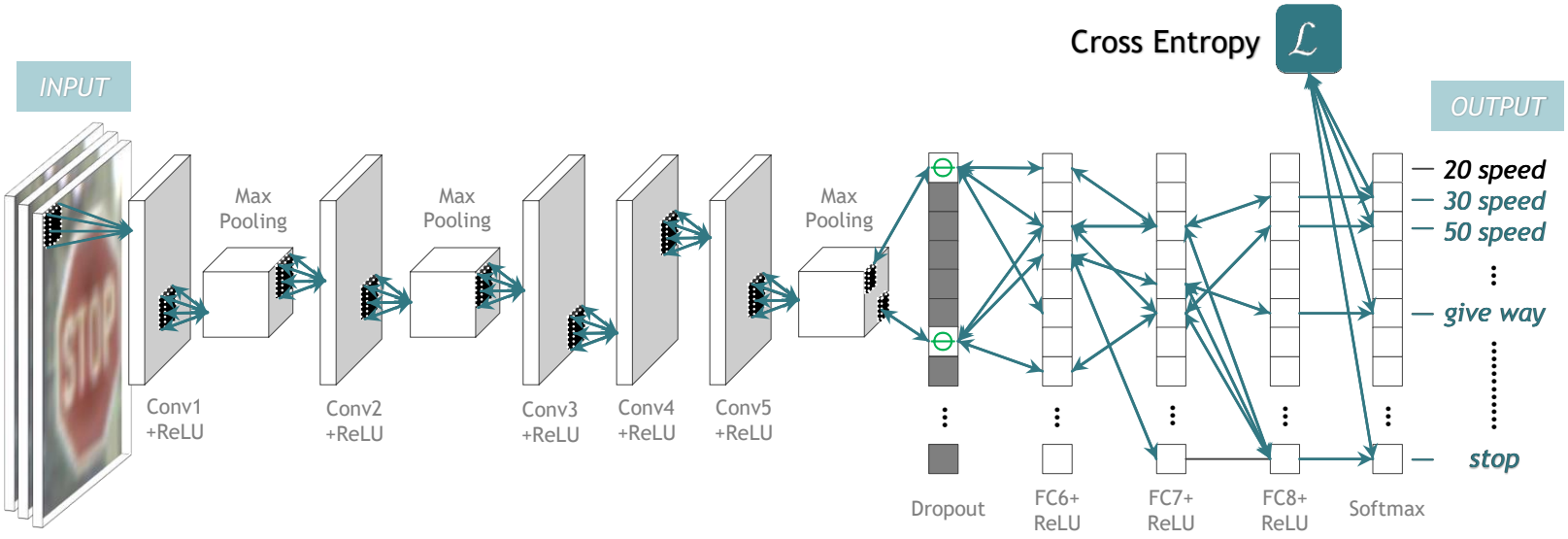
Gradient Ascent: AlexNet CNN Architektur [8]



- Trainingsziel: Mapping zwischen (nicht gelabelten) *Inputdaten* und *Label*
- *Loss: Berechnung* der Differenz zwischen gewünschter und tatsächlicher Ausgabe (Quantifizierung des Trainingsziels)
  - *Finde* eine Reihe von Weights und Biases (Gradienten), welche die Loss Function *minimieren*:
    ➔ ***Gradient Descent***

Gradient Ascent: AlexNet CNN Training [8]

AlexNet CNN Architektur | **AlexNet CNN Training** | Adversarial Image Generation | Remote Evaluation



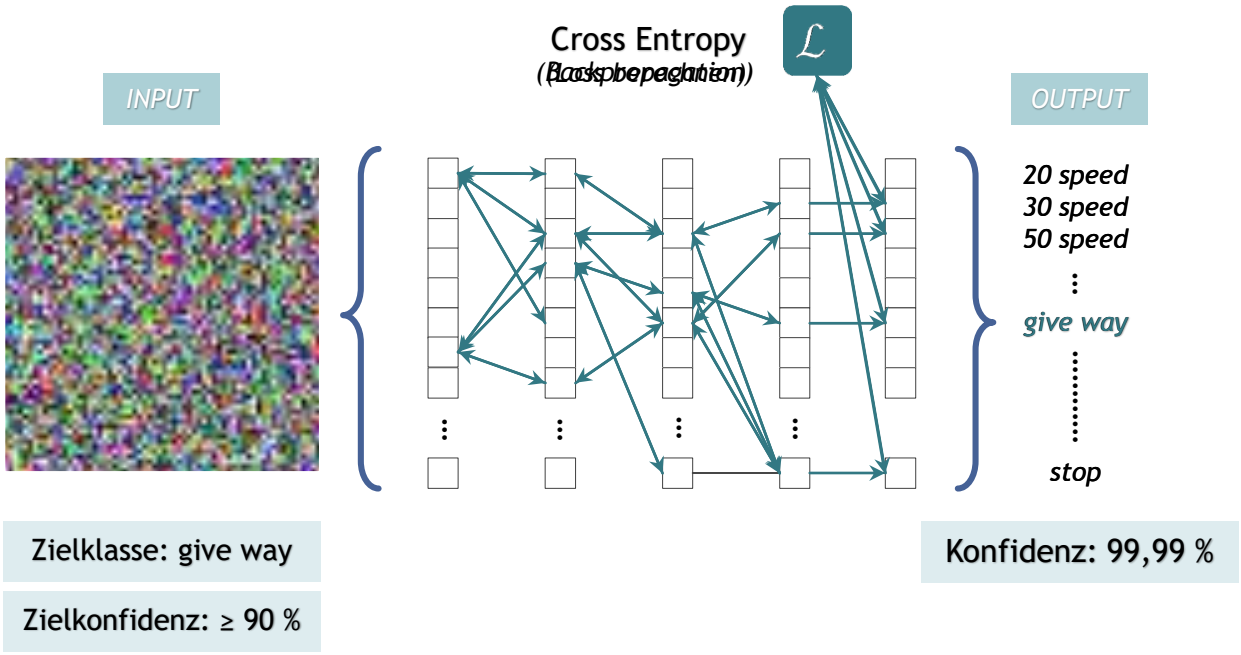
Für 50 Epochen (Schleife):

1. Feedforward Propagation
2. Loss berechnen (*Cross Entropy*)
3. Backpropagation
4. Parameter-Update (*Gradient Descent*)

AlexNet CNN Test Accuracy: 89 %

Gradient Ascent: Advers. Image Generation

AlexNet CNN Architektur → AlexNet CNN Training → Adversarial Image Generation → Remote Evaluation

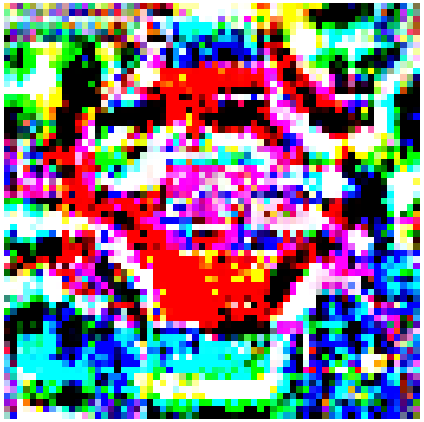


Gradient Ascent: Targeted Backpropagation [9]

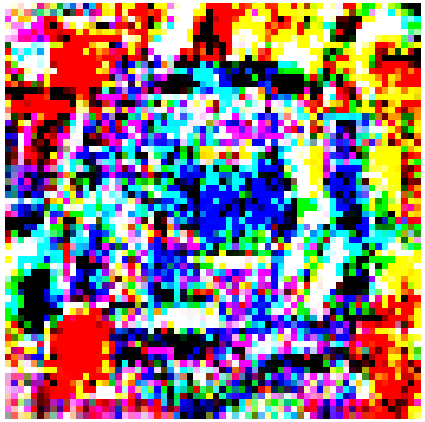
- Input: *Bild mit Zufallsrauschen* (Shape: 64 x 64 x 3)
- *Verändere* das Eingabebild *iterativ und solange* bis es der gewünschten Zielklasse entspricht (*Gradient Ascent*)
  - Zugriff auf die berechneten Gradienten (Weights und Biases) des trainierten CNN Modells
- *Wiederholung* für jede GTSRB-Klasse (43x)

Gradient Ascent: Remote Evaluation

- 43 Ergebnisbilder entsprechend der Klassen im GTSRB-Datensatz
- Resultat: 20 Bilder (46,51%) mit Konfidenz ≥ 90%
  - Nur bei 4 Bildern Übereinstimmung der Ursprungs- und Zielklasse



Vorfahrt gewähren
99,99%



Kreisverkehr
98,68%

Allgem. Überholverbot
99,99%

Saliency Maps

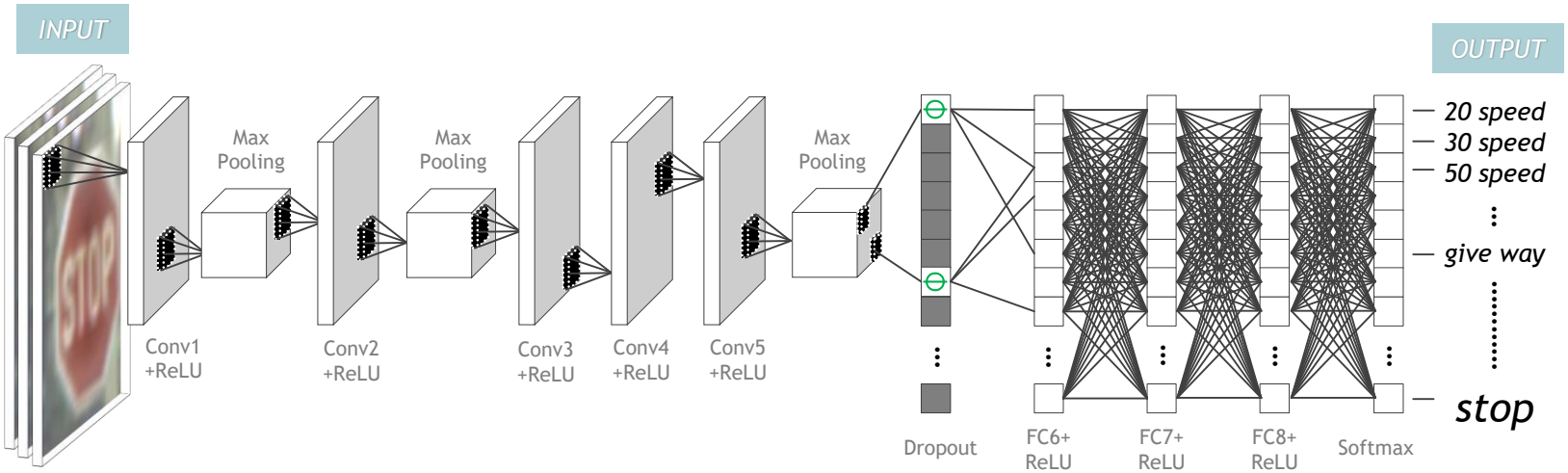
- Topografische Darstellung von klassentypischen, markanten Bildmerkmalen (High-Level Features), die das trainierte CNN zu Eingabebildern „gelernt“ hat. [3]

Beispielbild

Saliency Map

Saliency Maps: Idee

- Exkurs: Neuronenaktivierung
  - Neuronen sind verkettete Funktionen bestehend aus den eigenen Parametern und alles was zu dieser Funktion hingeführt hat (deren Vorgänger)
- Also: Rückführen des besten Ergebnis (≙ der geschätzten Klasse) auf die Neuronen die für die Aktivierung „verantwortlich waren“
- Visualisierung der Pixel durch Graustufen (hell = relevant)



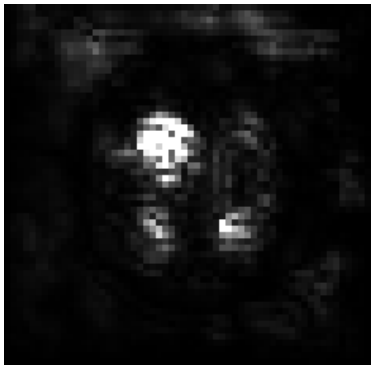
- Interaktives Beispiel: http://scs.ryerson.ca/~aharley/vis/conv/flat.html

Saliency Maps: Anwendung

- Hypothese: Saliency Maps können zur Täuschung eines Neuronalen Netz verwendet werden (Schemenhaft/Umrisse)
  - Direkt (Graustufenbild)
  - Maske über ein vorhandenes Bild

Unerkannt

+



Tempolimit 30

=

Tempolimit 30?

Saliency Maps: Implementierung [4-7]

- Ungeglättete und geglättete Verfahren
  - (Geglättete) Guided Backpropagation
  - (Geglättete) Integrated Gradient
  - (Geglättete) Vanilla

Vorverarbeitung

- Surrogat CNN „Aphrodite“: Implementierung, Training
- Bildklassifikation durch „Aphrodite“
- Bildselektion: Konfidenz = 100% (1.933/12.630)

Bilderzeugung

- Saliency Map:
  - Ungeglättete Verfahren
  - Geglättete Verfahren

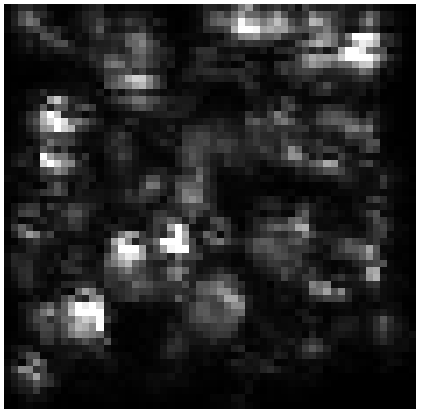
Remote Evaluierung

- Bildselektion: Konfidenz > 90%

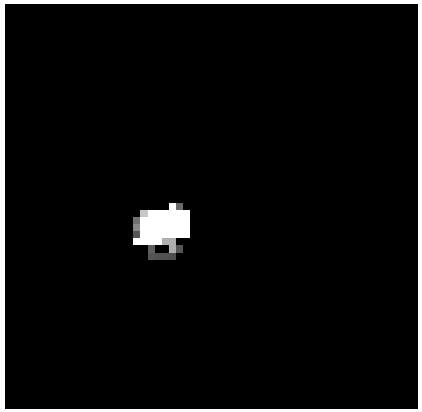
Saliency Maps: Ergebnisse

- Ungeglättete Verfahren (Erfolge):
  - Je Verfahren 0/1933 (0,00%)

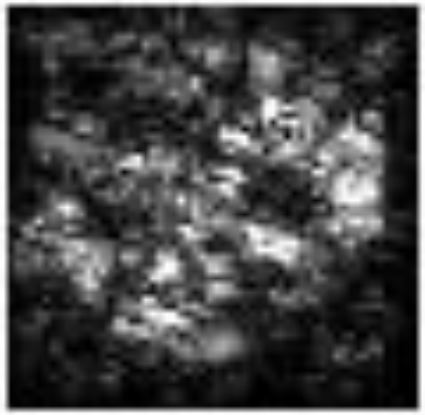
- Geglättete Verfahren (Erfolge):
  - Guided Backpropagation: 7/1933 (0,36%)
  - Integrated Gradient: 3/1933 (0,16%)
  - Vanilla Saliency: 3/1933 (0,16%)



Zul. Höchstgeschw. 50
99,95%



Zul. Höchstgeschw. 30
92,83%



Baustelle
99,99%

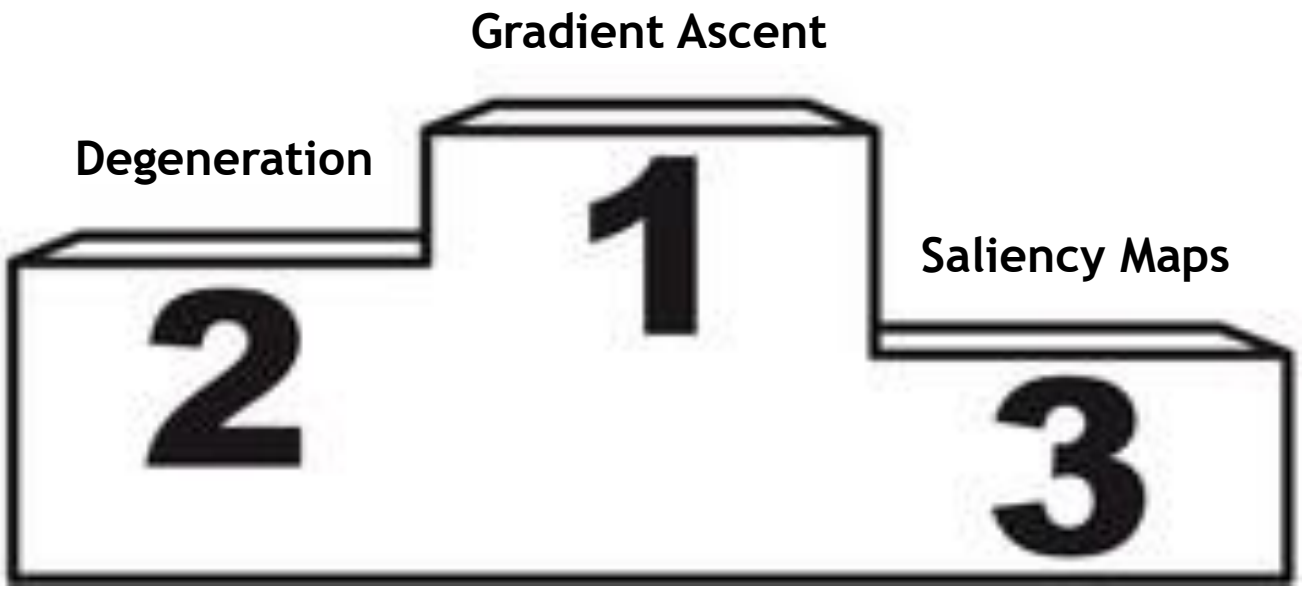
Zusammenfassung I

- Erfolge pro Zeiteinheit bei Gradient Ascent am höchsten (46,51%)
  - Geringe Laufzeit, zielgerichtete Bilderzeugung
- Degeneration bietet schnelle Erfolge, „guter erster Ansatz“
  - Sehr lange Laufzeit, zielgerichtet und mit „Erfolgsgarantie“ (Erfolg der Bilder ist zu jeder Zeit bekannt)
- Saliency Map kann Grundlage bieten für Adversarial Attacks, weitere Optimierung erforderlich
  - Lange Laufzeit, keine zielgerichtete Bilderzeugung möglich (Brute Force; Stichprobe aus Testdatensatz)
- Greyboxing lieferte keine Ergebnisse
  - Bemerkbare Einschränkung durch geringe Auflösung der Bilder (Mehr Pixel → Höhere Entropie der einzelnen Pixel)

Zusammenfassung II

	Remote Degen.	Local Degen.	Gradient Ascent	Smoothed Vanilla	Smoothed Integrated Gradient	Smoothed Guided Backprop.
Bilder (Anz.)	5	5	43	1933	1933	1933
Dauer (min)	309:10	18:30	0:13	36:02	36:05	41:26
Bilder/s	/	/	3,30	0,89	0,89	0,77
Erfolge (abs.)	5	0	20	3	3	7
Erfolge (rel.)	100%	0,00%	46,51%	0,16%	0,15%	0,36%
Laufzeit	---	+	++	--	--	--

Zusammenfassung III



Ehrenvolle Nennung: Greyboxing

Ergebnis InformatiCup 2019

- Über 30 vollständige/funktionierende Einreichungen
  - Einladung zur Endrunde nach Wolfsburg (AutoUni)
- Vielfalt der Lösungswege: Unterschiede in..
  - Herangehensweisen der Bilderzeugung (Genetische Algorithmen)
  - Verfahren zur Surrogat Optimierung
  - Fokus auf Front-End ☺
- Keine Lösung war identisch
- Eigene Platzierung: 4. Platz
  - Sonderpreis: Bester wissenschaftlicher Transfer

Ergebnis InformatiCup 2019

Kontakt

- Leonhard Applis
  applisle74858@th-nuernberg.de
- Peter Bauer
  bauerpe72692@th-nuernberg.de
- Andreas Porada
  poradaan60975@th-nuernberg.de
- Florian Stöckl
  stoecklfl75458@th-nuernberg.de

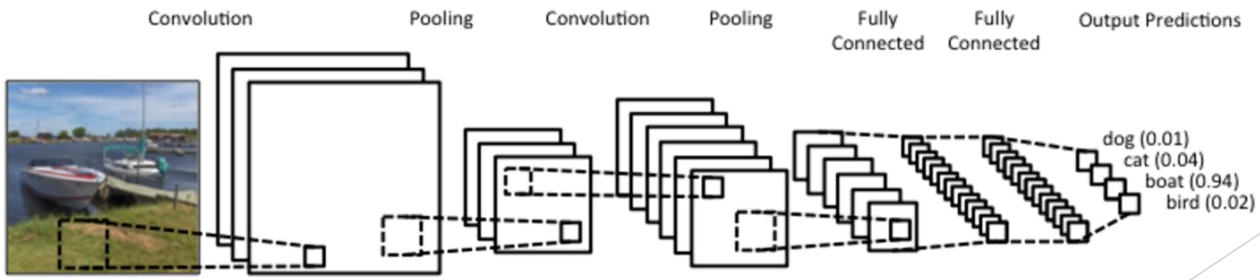
Anhang

Anhang – Aphrodite CNN

- 10 Layer
- Topologischer Aufbau:
  - Conv + Pooling
  - ReLU
  - Classifier
- Accuracy: 96,5%

Layer (type)	Output Shape	Param #
conv2d_30 (Conv2D)	(None, 64, 64, 32)	896
conv2d_31 (Conv2D)	(None, 62, 62, 32)	9248
max_pooling2d_15 (MaxPooling	(None, 31, 31, 32)	0
dropout_15 (Dropout)	(None, 31, 31, 32)	0
conv2d_32 (Conv2D)	(None, 31, 31, 64)	18496
conv2d_33 (Conv2D)	(None, 29, 29, 128)	73856
max_pooling2d_16 (MaxPooling	(None, 14, 14, 128)	0
dropout_16 (Dropout)	(None, 14, 14, 128)	0
flatten_6 (Flatten)	(None, 25088)	0
dense_18 (Dense)	(None, 128)	3211392
dense_19 (Dense)	(None, 128)	16512
dense_20 (Dense)	(None, 43)	5547

Total params: 3,335,947
Trainable params: 3,335,947
Non-trainable params: 0



Anhang - Literatur

- [1] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical Black-box Attacks Against Machine Learning. arXiv:1602.02697 [cs], 2016. arXiv: 1602.02697.
- [2] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. SmoothGrad: removing noise by adding noise. *arXiv:1706.03825 [cs, stat]*, 2017. arXiv: 1706.03825.
- [3] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention forrapidsceneanalysis. *IEEETransactionsonPatternAnalysisandMachine Intelligence*, 20(11):1254–1259, 1998.
- [4] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *arXiv:1312.6034 [cs]*, 2013. arXiv: 1312.6034.
- [5] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for Simplicity: The All Convolutional Net. *arXiv:1412.6806 [cs]*, 2014. arXiv: 1412.6806.
- [6] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic Attribution for Deep Networks. *arXiv:1703.01365 [cs]*, 2017. arXiv: 1703.01365.
- [7] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. SmoothGrad: removing noise by adding noise. *arXiv:1706.03825 [cs, stat]*, 2017. arXiv: 1706.03825.
- [8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks.," in Advances in Neural Information Processing Systems 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.
- [9] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into Transferable Adversarial Examples and Black-box Attacks. arXiv:1611.02770 [cs], 2016. arXiv: 1611.02770.
- [11] neuronale Netze. Lexikon der Neurowissenschaft, https://www.spektrum.de/lexikon/neurowissenschaft/neuronale-netze/8653, 2014