Security Aspects of Deep Learning

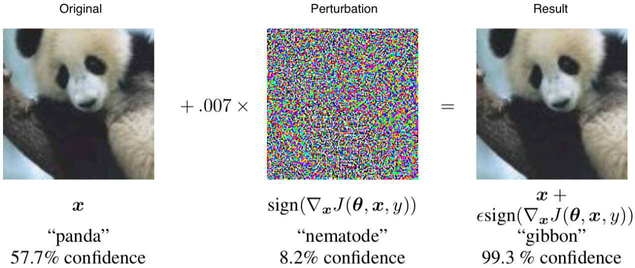
RU Data Science Seminar

Alexandru C. Serban[1,2]

[1]Digital Security
Radboud University, Nijmegen

[2]Research Team
Software Improvement Group, Amsterdam

Today's talk in a nutshell

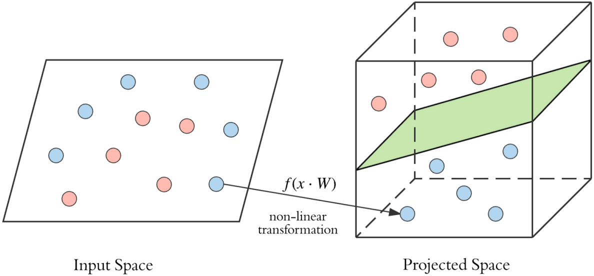

Original
Perturbation
Result
$+ .007 \times$
$=$
$\boldsymbol{x}$
"panda"
57.7% confidence
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

Topics covered

- Brief introduction to Deep Learning
- Attacks on the Machine Learning pipeline
- Inference attacks - Adversarial Examples
  - Why do they exist?
  - How to create adversarial examples
  - How to protect against adversarial examples
- What we do

Most of today's talk is focused on computer vision and deep learning, but adversarial examples can be found for domains or for other ML models.

The object recognition task



Goal: Map images to an n-dimensional space where we can separate between objects.

Method: Learn this mapping (hypothesis) through Empirical Risk Minimisation.

Challenges: It's hard to select relevant features from images and to restrict the space of hypotheses.

An approach that works

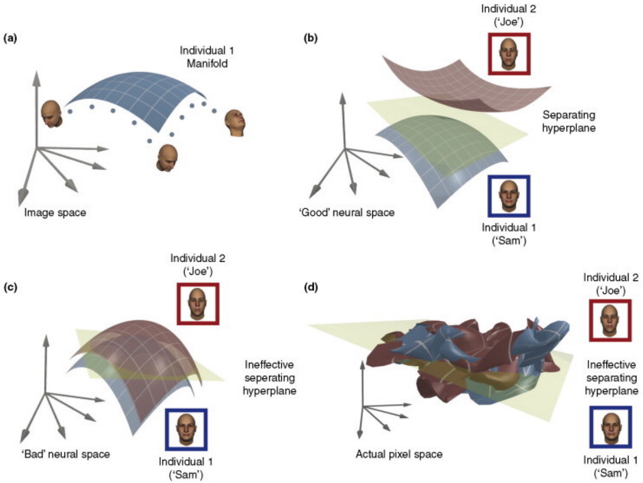
- Create a (deep) representation as a composition of many functions.

$$\begin{tikzcd}
\boldsymbol{x} \arrow[r] & f^{(1)} \arrow[r] & \dots \arrow[r] & f^{(n)} \arrow[r] & \hat{y} \arrow[r] & J \\
& \boldsymbol{w_1} \arrow[u] & \dots & \boldsymbol{w_n} \arrow[u] & &
\end{tikzcd}$$

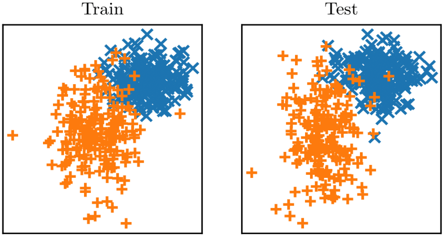
- Back-propagate the error in search for its minimum.

$$\begin{tikzcd}
\frac{\partial J}{\partial \boldsymbol{x}} & \frac{\partial J}{\partial f^{(1)}} \arrow[l, "\frac{\partial f^{(1)}}{\partial \boldsymbol{x}}"'] \arrow[d, "\frac{\partial f^{(1)}}{\partial \boldsymbol{w_1}}"'] & \dots \arrow[l, "\frac{\partial f^{(2)}}{\partial f^{(1)}}"'] & \frac{\partial J}{\partial f^{(n)}} \arrow[l, "\frac{\partial f^{(n)}}{\partial f^{(n-1)}}"'] \arrow[d, "\frac{\partial f^{(n)}}{\partial \boldsymbol{w_n}}"'] & \frac{\partial J}{\partial \hat{y}} \arrow[l, "\frac{\partial \hat{y}}{\partial f^{(n)}}"'] \\
& \frac{\partial J}{\partial \boldsymbol{w_1}} & \dots & \frac{\partial J}{\partial \boldsymbol{w_n}} &
\end{tikzcd}$$

Assumptions: The Manifold Assumption



Assumptions: IID



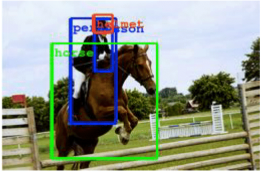
I: Independent

I: Identically

D: Distributed

All train and test examples drawn independently from same distribution

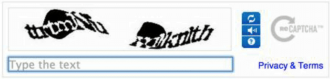
DL achieved 'human-level' performance on many IID tasks around 2013



(Szegedy et al, 2014)

...recognizing objects and faces....

(Taigmen et al, 2013)

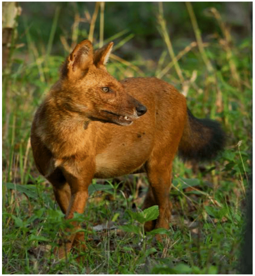


(Goodfellow et al, 2013)

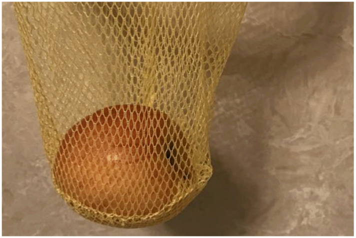
...solving CAPTCHAS and reading addresses...

(Goodfellow et al, 2013)

Caveats of 'human-level' performance



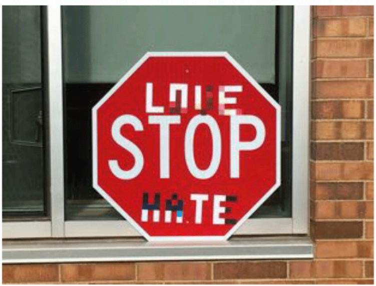
Humans are not very good at some parts of the benchmark



The test data is not very diverse. ML models are fooled by natural but unusual data.

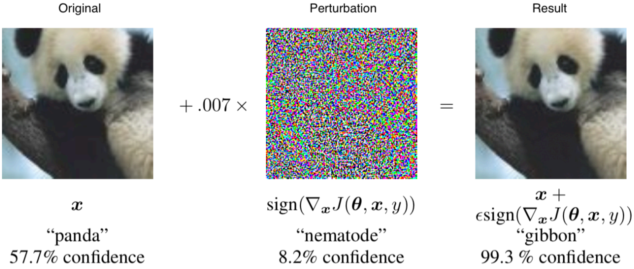
Security requires thinking beyond IID

- Not identical: Attackers can use unusual inputs
- Not independent: Attackers can repeatedly send the same mistake



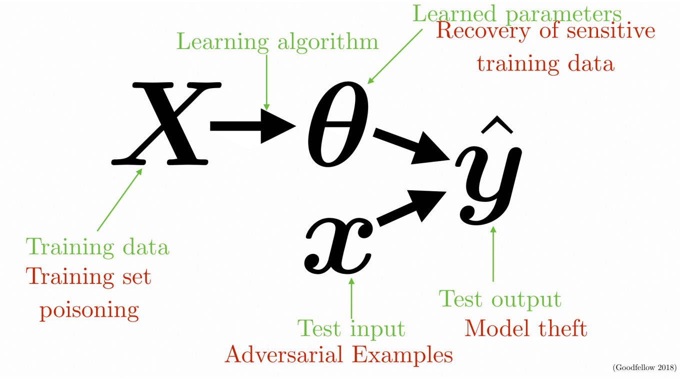
(Eykholt et al. 2018)

ML models fail unexpectedly in non IID settings


Original
Perturbation
Result
$+ .007 \times$
$=$
$\boldsymbol{x}$
"panda"
57.7% confidence
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

(Goodfellow et al. 2016)

Attacks on the ML pipeline



Learning algorithm

Learned parameters

Recovery of sensitive training data

Training data

Training set poisoning

Test input

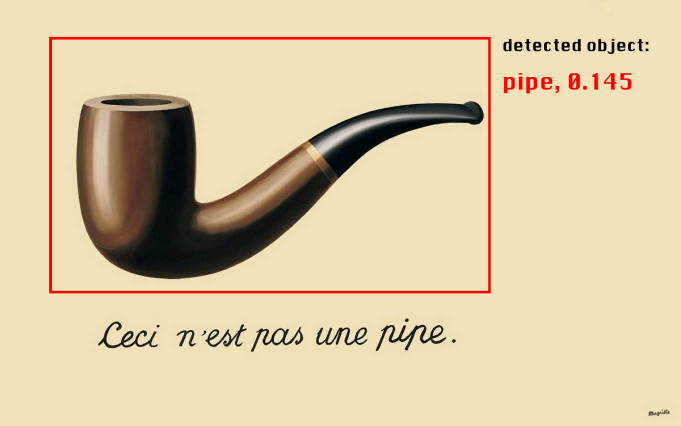
Adversarial Examples

Test output

Model theft

(Goodfellow 2018)

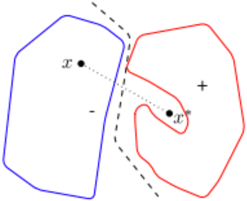
Adversarial Examples



Adversarial Examples - back to origins

$$
\begin{aligned}
\min_{\boldsymbol{x}'} \quad & \|\boldsymbol{x}' - \boldsymbol{x}\|_p, \\
s.t. \quad & f(\boldsymbol{x}') = l', \\
& f(\boldsymbol{x}) = l, \\
& l \neq l', \\
& \boldsymbol{x}' \in [0,1]^m,
\end{aligned}
$$

(Szegedy et al. 2014)

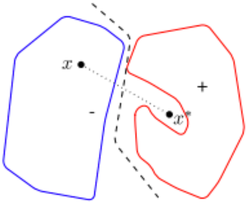


A first hypothesis on the existence of adversarial examples: they lie in 'pockets' of the data manifold.

Adversarial Examples - back to origins

$$\begin{aligned}
\min_{\boldsymbol{x}'} \quad & \|\boldsymbol{x}' - \boldsymbol{x}\|_p, \\
s.t. \quad & f(\boldsymbol{x}') = l', \\
& f(\boldsymbol{x}) = l, \\
& l \neq l', \\
& \boldsymbol{x}' \in [0,1]^m,
\end{aligned}$$

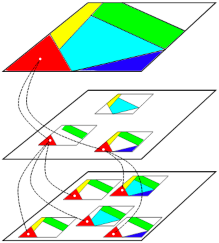
(Szegedy et al. 2014)



A first hypothesis on the existence of adversarial examples: they lie in 'pockets' of the data manifold.

Disadvantages: Solving the optimisation problem in this form is resource intensive (but guarantees a minimal perturbation).

Simpler methods to generate adversarial examples



(Montufar et al. 2014)

$$\boldsymbol{\eta} = \epsilon \, \mathrm{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right)$$

$$\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\eta}$$

(Goodfellow et al. 2016)

A 2nd hypothesis: DNNs behave, in fact, linearly (despite the non linear transformations in the hidden layers) and adversarial examples span high dimensional regions.

Summing small perturbations in all dimensions of a high dimensional input forces the entire sum in a direction that will likely cause misclassifications.

Precise and simple methods

Iteratively apply the gradient method presented earlier:

$$\boldsymbol{x}_0' = \boldsymbol{x}, \quad \boldsymbol{x}_{N+1}' = Clip_{\boldsymbol{x},\epsilon}\Big\{\boldsymbol{x}_N' + \epsilon\,\mathrm{sign}\big(\nabla_{\boldsymbol{x}} J(\boldsymbol{x}_N', y_{true})\big)\Big\},$$

(Kurakin et al. 2016)

Use momentum:

$$\begin{aligned} \boldsymbol{g}_{t+1} &= \mu \boldsymbol{g}_t + \frac{\nabla_{\boldsymbol{x}}' J(\boldsymbol{\theta}, \boldsymbol{x}, y)}{\|\nabla_{\boldsymbol{x}}' J(\boldsymbol{\theta}, \boldsymbol{x}, y)\|_1} \\ \boldsymbol{x}_{t+1}' &= \boldsymbol{x}_t' + \epsilon \mathrm{sign}(\boldsymbol{g}_{t+1}) \end{aligned}$$

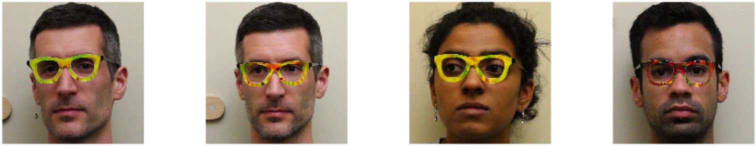
(Dong et al. 2017)

and others ...

Zooming out

- No universally accepted hypothesis on the existence of adversarial examples.
- Over 20 types of powerful attacks developed.
- Some of which do not require any information from the model (black box).
- Many attacks work using a *target* class.
- The attacks have been successfully applied to other ML tasks such as speech recognition, facial recognition, malware detection, etc.

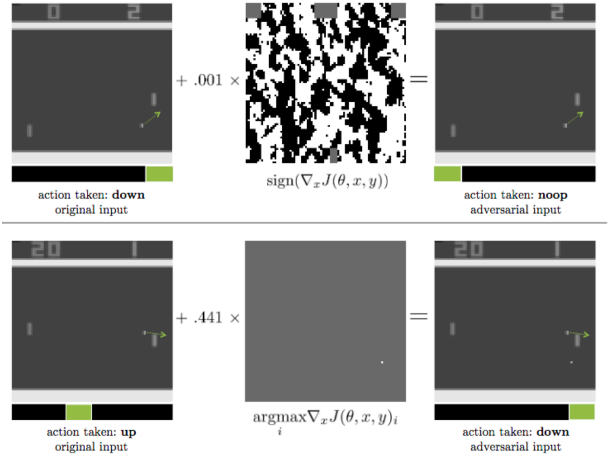
Some examples: Face Recognition



(Sharif et al. 2016)

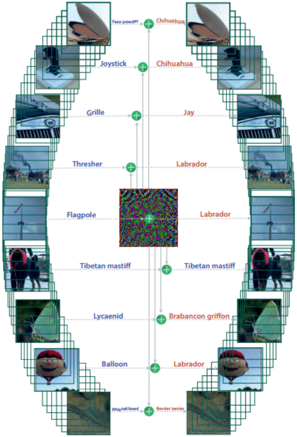
By wearing a printed pair of glasses one can evade recognition or impersonate another individuals.

Some examples: Deep Reinforcement Learning



(Huang et al. 2017)

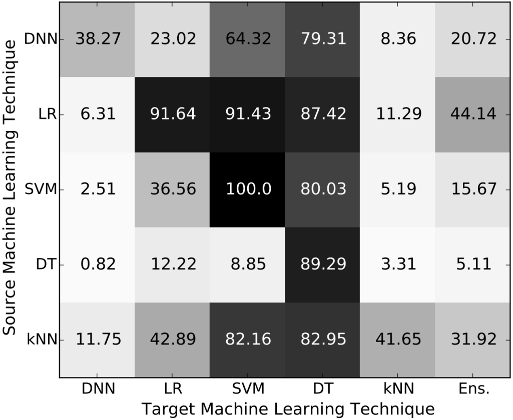
Peculiar phenomena: Universal Perturbations



(Moosavi-Dezfooli et al. 2017)

We can find one perturbation that can be used with any input.

Peculiar phenomena: Transferability



Source Machine Learning Technique	DNN	LR	SVM	DT	kNN	Ens.
DNN	38.27	23.02	64.32	79.31	8.36	20.72
LR	6.31	91.64	91.43	87.42	11.29	44.14
SVM	2.51	36.56	100.0	80.03	5.19	15.67
DT	0.82	12.22	8.85	89.29	3.31	5.11
kNN	11.75	42.89	82.16	82.95	41.65	31.92

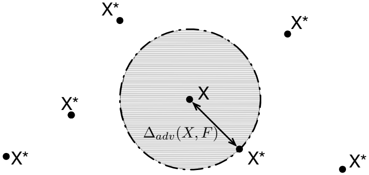
(Papernot et al. 2016)

Adversarial examples transfer across ML models.

Defences

THEY SAY THE BEST OFFENSE IS A GOOD DEFENSE.
FALSE.
THE BEST OFFENSE IS OFFENSE.
quickmeme.com

What is a (good) defence?



- There is no universally accepted definition for a defences.
- In practice the norm-ball around an input is used to define *robustness*:

$$\mathbb{B}(\boldsymbol{x}_c, r) = \{\boldsymbol{x} \mid \|\boldsymbol{x} - \boldsymbol{x}_c\|_p \leq \epsilon\} \tag{1}$$

Classification of defences

- *Reactive* defences - target adversarial examples early in the processing pipeline
  - promising because they can be applied for all models
  - but inefficient
- *Proactive* defences - alter the training process or data
  - offer some level of protection
  - but require more data
- *Provable* defences - use formal tools to prove robustness
  - very good results
  - but are not scalable

(Serban and Poll, 2018)

Fast forward through reactive defences

- Detection of adversarial examples - train separate detectors based on different features or define a new class for adv. ex.

- Input transformations - preprocess inputs (e.g. discretisation, compression, noise reduction, etc.)

Proactive defences: Adversarial training

$$\tilde{J}(\boldsymbol{\theta}, \boldsymbol{x}, y) = \alpha J(\boldsymbol{\theta}, \boldsymbol{x}, y) + (1 - \alpha) J(\boldsymbol{\theta}, \boldsymbol{x} + \epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right), y)$$

(Goodfellow et al. 2016)

- Adversarial training is a method of regularisation
- Provides unexpected benefits: *interpretable* gradients and robust feature representation that alight well with salient data characteristics
- Requires more data for empirical risk minimisation (proved in the PAC framework)

Better adversarial training: robust optimization

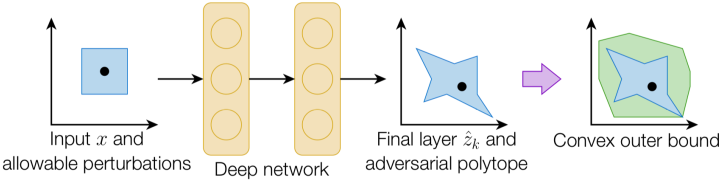
Define a norm ball around an input, $\mathcal{S}$ and solve:

$$\min_{\boldsymbol{\theta}} \rho(\boldsymbol{\theta}), \quad \text{where} \quad \rho(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim p_{\text{data}}} \left[\max_{\eta \in \mathcal{S}} J(\boldsymbol{\theta}, \boldsymbol{x} + \boldsymbol{\eta}, y) \right],$$

(Madry et al. 2017)

- The inner maximisation problem can be approximated using Projected Gradient Descent.
- The outer minimisation problem is solved using empirical risk minimisation.
- If the inner maximisation problem is well approximated, this method guarantees adversarial examples can not be found with $\boldsymbol{\eta}$ in $\mathcal{S}$.

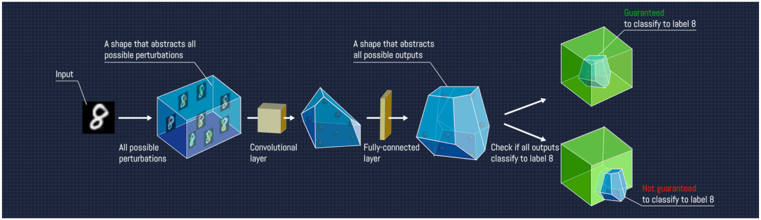
Provable defences through convex approximation



(Wong and Kolter, 2018)

- Develop a convex approximation of the set of activations reachable through a norm-bounded perturbation
- Use robust optimisation to minimise the worst case loss over this outer region (via a linear program)

Provable defences through abstract interpretation


Input
A shape that abstracts all possible perturbations
All possible perturbations
Convolutional layer
Fully-connected layer
A shape that abstracts all possible outputs
Check if all outputs classify to label 8
Guaranteed to classify to label 8
Not guaranteed to classify to label 8

(Mirman et al. 2018)

- Define an abstract transformer (as a sound over-approximation of the space of all possible perturbations)
- Train with the resulting polytope

Take aways - defences

- Defences that act early in the pipeline are promising because they can be applied to all models - but are (currently) innefficient
- Adversarial training offers some performance improvements (and unexpected benefits), but require much more data (currently unavailable).
- Provable defences are interesting, but do not scale well.

Our approach

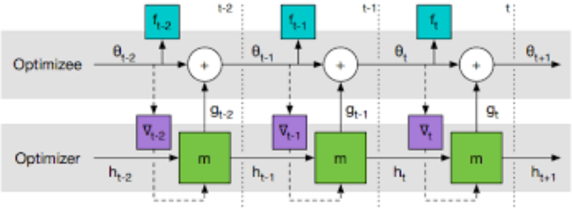
THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

Learning to learn robust classifiers

Premises:

- Reduce the number of samples needed for adversarial training
- Avoid problems with gradient sensitivity
- Avoid having to train for different attack types

Approach: Meta-learning (recent work on meta-learning is similar to transfer learning)



Future Directions

- Security should have clear goals: for two models with the same error, do we prefer the model with lower confidence on mistakes or the model whose mistakes are harder to find, etc.?
- Reason beyond the norm-ball
- Search for theoretical answers for this problem

Propositions:

- Explicitly model adversarial uncertainty in training datasets
- Bayesian deep nets should solve the adversarial examples problem

Conclusions

- Security requires thinking beyond IID
- Most ML models perform poorly outside IID
- Adversarial examples are just a way of fooling ML models
- There is no generally accepted hypothesis on their existence
- There is no generally accepted definition of 'done'
- No defence can scale more than 50% accuracy on CIFAR-100

Questions

Contact:

a.serban@cs.ru.nl
cs.ru.nl/∼aserban

Advertising:

- Master thesis projects
- Guest lectures
- Internships at SIG on data driven software engineering or security aspects of ML
- Get involved: https://github.com/tensorflow/cleverhans