Adversarial examples and robustness certificates

Stéphane Canu,
https://chaire-raimo.github.io/

Assemblée générale 2021 du GdR ISIS
June 17, 2021

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN
litis
Normandie Université
PROJET FINANCÉ PAR L'ANR - PROJECT FUNDED BY THE ANR
ANR

Raimo's Team

Gilles Gasso

Gaëlle Loosli

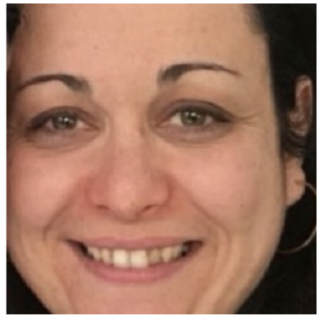
Jordan Frecon

Ramzi Ben Mhenni

Samia Ainouz

Lucas Anquetil

Ismaïla Seck

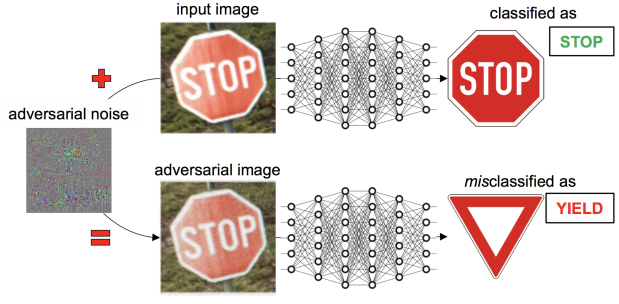


Yuan Liu

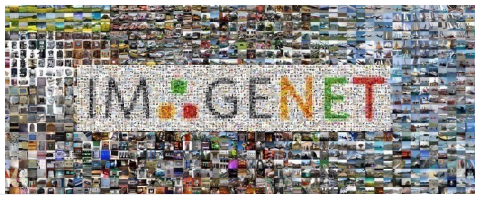
et Maxime Gueriau , Cyprien Rufino, Laetitia Chapelle, Sherpa Engineering, Thierry Chateau, Vincent T'kindt…

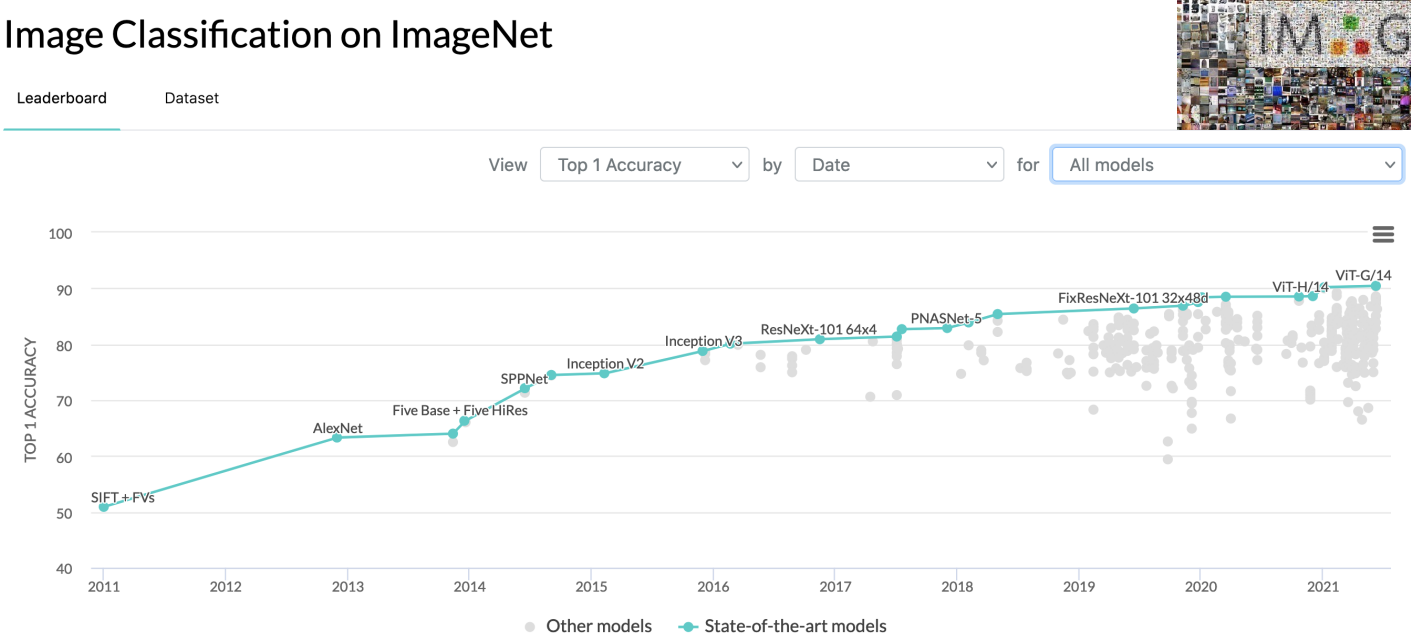
Road map

1. Attacking deep networks
2. Adversarial attacks
3. Typology of Attacks
4. Robusteness certificates and MIP
   - Formalizing the search for adversarial examples
   - Robust training
5. ADiL (Adversarial dictionary learning)
   - Adversarial dictionary learning framework
   - Algorithmic solutions
   - Experimental results
6. Conclusion

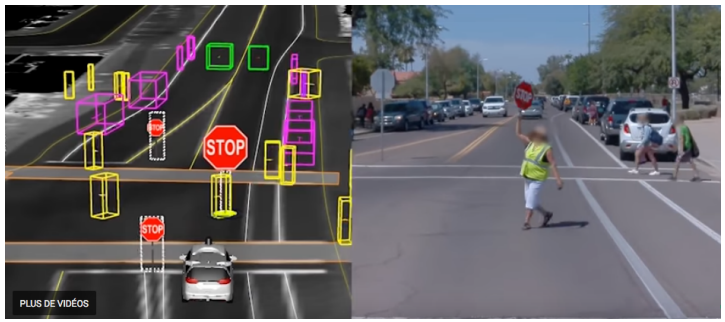


The amazing achievements of deep learning





Machine (deep) Learning in Safety-Critical Tasks



Autonomous Driving Vehicles

Facial Recognition Payment System

Airborne Collision-Avoidance System

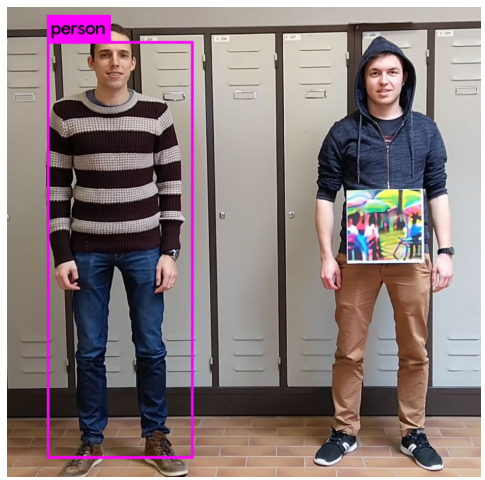
Is ML Reliable and Safe for real-world applications?

Example of recognition system under attacks

Spam message
Buy Viagra

Camouflaged message
Buy Vi@gra

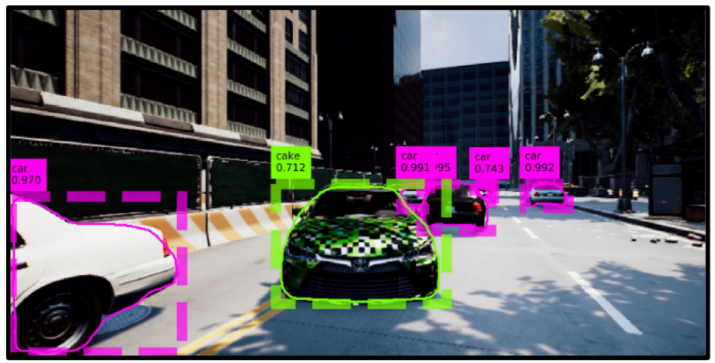
https://www.kaggle.com/c/adversarial-attacks-against-spam-detectors/overview/description
Imam & Vassilakis, A Survey of Attacks Against Twitter Spam Detectors in an Adversarial Environment, 2019



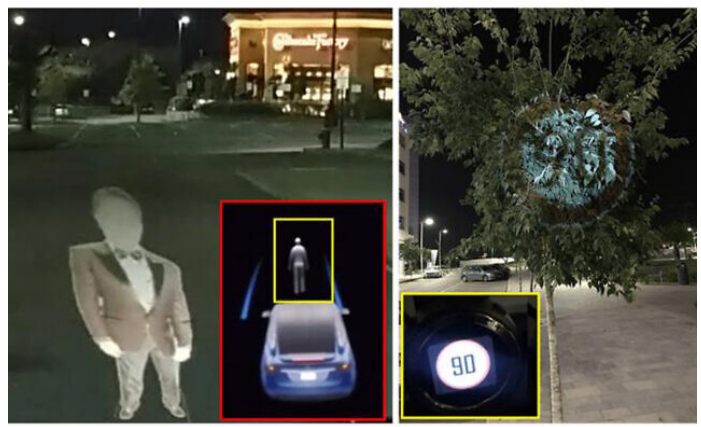
Sharif et al., ACM CCS, 2016
Thys, Van Ranst & Toon Goedemé, Proceedings of the IEEE, 2019

Attacks against autonomous vehicles

Eykholt et al, Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR 2018



Zhang et al., CAMOU: Learning Physical Vehicle Camouflages to Adversarially Attack Detectors in the Wild, ICLR 2019



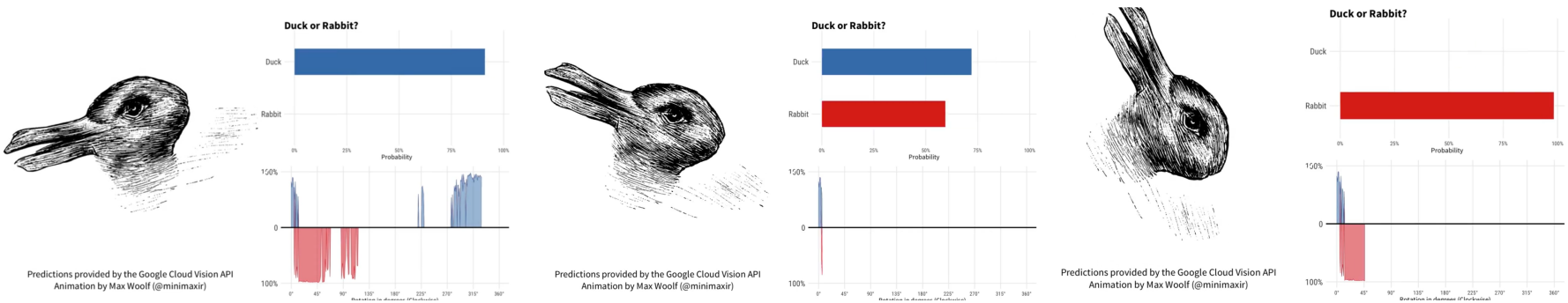
https://www.mcafee.com/blogs/other-blogs/mcafee-labs/model-hacking-adas-to-pave-safer-roads-for-autonomous-vehicles/

Nassi et al., Phantom of the ADAS: Securing Advanced Driver-AssistanceSystems from Split-Second Phantom Attacks, 2020

Qayyum, et al., Securing Connected & Autonomous Vehicles: Challenges Posed by Adversarial ML, IEEE Communications, 2019

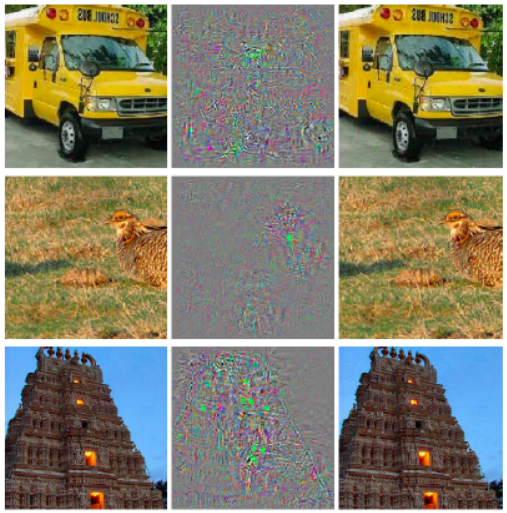
Attack or illusion: Duck or a Rabbit?

Form Google Cloud Vision

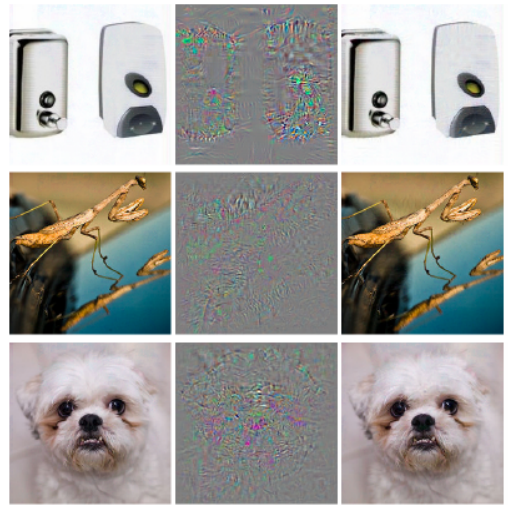


https://github.com/minimaxir/optillusion-animation

Intriguing properties of neural networks, Szegedy ICLR 2014

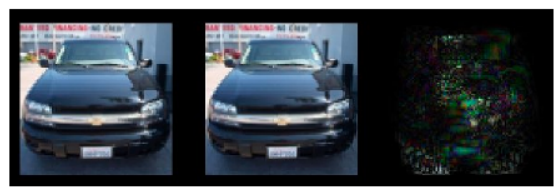


(a)



(b)

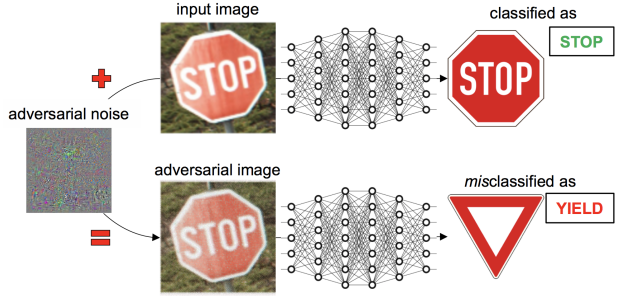
Figure 5: Adversarial examples generated for AlexNet [9].(Left) is a correctly predicted sample, (center) difference between correct image, and image predicted incorrectly magnified by 10x (values shifted by 128 and clamped), (right) adversarial example. All images in the right column are predicted to be an *"ostrich, Struthio camelus"*. Average distortion based on 64 examples is 0.006508. Plase refer to `http://goo.gl/huaGPb` for full resolution images. The examples are strictly randomly chosen. There is not any postselection involved.



Adversarial examples

Road map

1. Attacking deep networks
2. Adversarial attacks
3. Typology of Attacks
4. Robusteness certificates and MIP
   - Formalizing the search for adversarial examples
   - Robust training
5. ADiL (Adversarial dictionary learning)
   - Adversarial dictionary learning framework
   - Algorithmic solutions
   - Experimental results
6. Conclusion



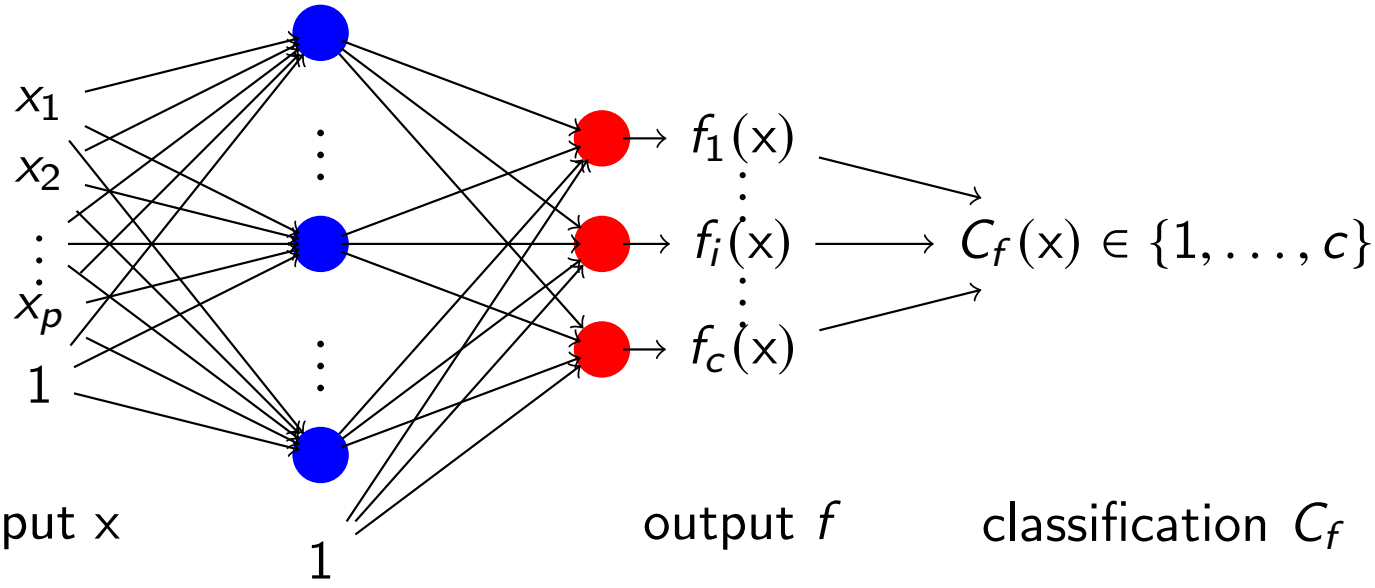
Classification model

A classification model (e.g. Neural Network) with c output nodes

$$\begin{array}{rccc} f: & \mathcal{X} \subseteq \mathbb{R}^p & \longrightarrow & \mathbb{R}^c \\ & \mathrm{x} & \longmapsto & f(\mathrm{x}) \end{array}$$

The associated classification (or decision function)

$$C_f(\mathrm{x}) = \underset{k=1,\ldots,c}{\operatorname{argmax}}\, f_k(\mathrm{x})$$

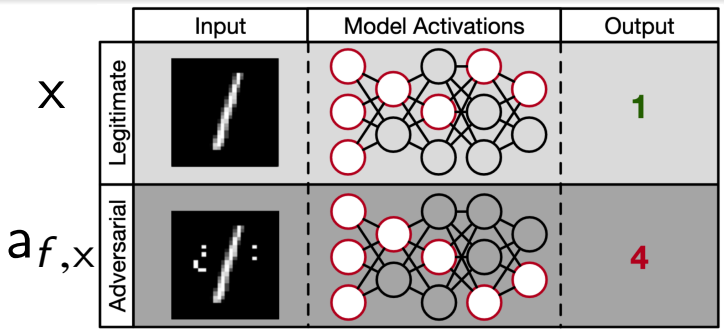


Adversarial examples

Definition (Generic adversarial)

$a_{f,x}$ is an adversarial example of f at x if $a_{f,x}$ is a valid input close to x and

$$C_f(x) \neq C_f(a_{f,x}) \quad \text{that is} \quad c^\star = \underset{k=1,\ldots,c}{\operatorname{argmax}} f_k(x) \neq \underset{k=1,\ldots,c}{\operatorname{argmax}} f_k(a_{f,x})$$



form Papernot et al., 2016

Definition (Specific (or targeted) adversarial)

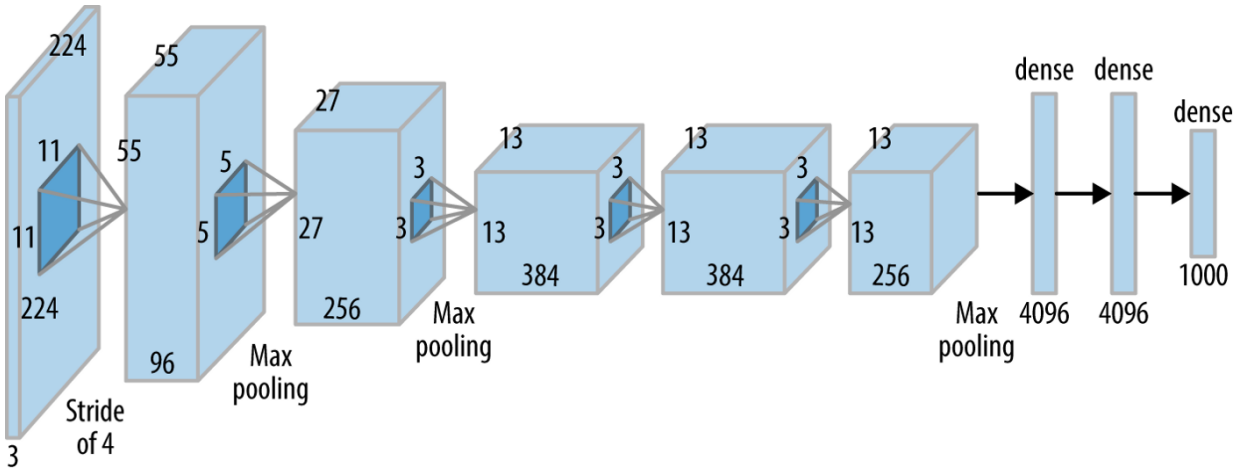
a_{f,x,t_a} is a specific adversarial example of f at x for the adversarial targeted class t_a if a_{f,x,t_a} is a valid input close to x and

$$f_{c^\star}(a_{f,x,t_a}) + \alpha \leq f_{t_a}(a_{f,x,t_a}) \quad \text{with} \quad c^\star = \underset{k=1,\ldots,c}{\operatorname{argmax}} f_k(x)$$

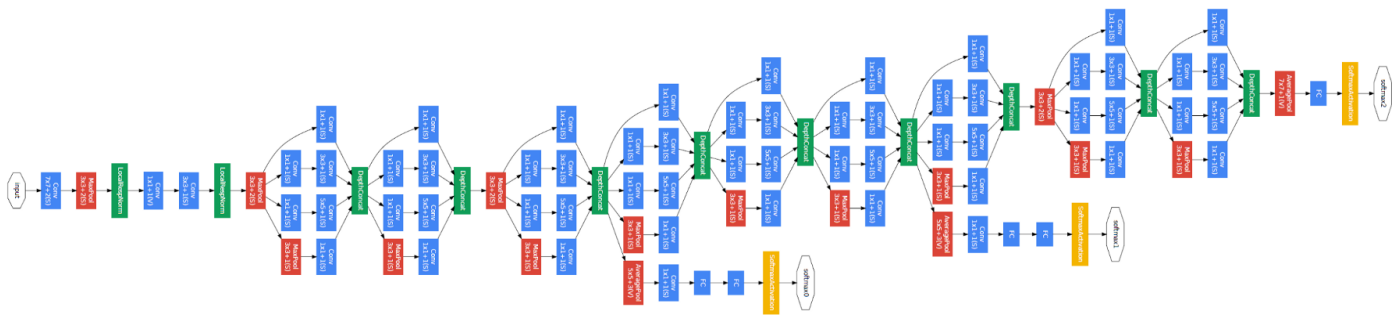
for a given scalar $\alpha \geq 0$ called the confidence level.

Adversarial transfer

model f



model g



Definition (Adversarial transfer)

An adversarial example $a_{f,x}$ of classification model f at input x adversarially transfers on model g if

$$C_g(x) \neq C_g(a_{f,x})$$

Goodfellow et al., Explaining and harnessing adversarial examples, ICLR 2015

3 components to define adversarial examples for f

- a valid example $\mathsf{a} \in \mathcal{X}$ (feasible solution)
- adversarial close to x: $D(\mathsf{x}, \mathsf{a})$ a dissimilarity measure (a distance)
- $\underset{k=1,\dots,c}{\operatorname{argmax}} f_k(\mathsf{x}) \neq \underset{k=1,\dots,c}{\operatorname{argmax}} f_k(\mathsf{a})$: adversarial loss L,

$$\begin{array}{rccc} L: & \mathbb{R}^c \times \mathbb{R}^c & \longrightarrow & \mathbb{R} \\ & \mathsf{s}, t & \longmapsto & L(\mathsf{s}, t) \end{array}$$

  - training class: $c^\star = \underset{k=1,\dots,c}{\operatorname{argmax}} f_k(\mathsf{x})$ with training pair

$$(x, c^\star) \Rightarrow \max L(s, c^\star)$$

  - targeted class: $t_a \neq \underset{k=1,\dots,c}{\operatorname{argmax}} f_k(\mathsf{x}) \Rightarrow \min L(s, t_a)$

May be different from the training loss $L(f(\mathsf{a}), c^\star) \neq J(f(\mathsf{a}), c^\star)$

Adversarial noise

Definition (adversarial noise (or perturbation or distorsion))

A vector $\Delta_{f,x}$ is an adversarial noise of f at x if

$$a_{f,x} = x + \Delta_{f,x}$$

is an adversarial example for f at x

Given $a_{f,x}$ the associated adversarial noise is $\Delta_{f,x} = x - a_{f,x}$

Definition (Universal adversarial perturbation)

A perturbation Δ_f is a universal of f if, for any $x \in \mathcal{X}$, $a_f = x + \Delta_f$ is a generic adversarial example for f at x, that is

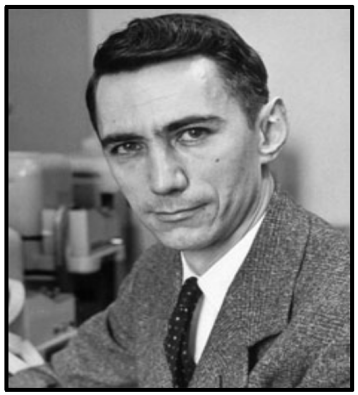
$$\forall x \in \mathcal{X}, \quad C_f(x) \neq C_f(x + \Delta_f)$$

note that $x + \Delta_f$ must be a valid example.

This can be relaxed by asking to be verified with high probability when randomly picking x

Adversarial Noise vs. Stochastic Noise

This distinction is not new (*cf* Adversarial error in the Coding Theory)



Shannon's stochastic noise model: probabilistic model of the channel, the probability of occurrence of too many or too few errors is usually low

Hamming's adversarial noise model: the channel acts as an adversary that arbitrarily corrupts the code-word subject to a bound on the total number of errors

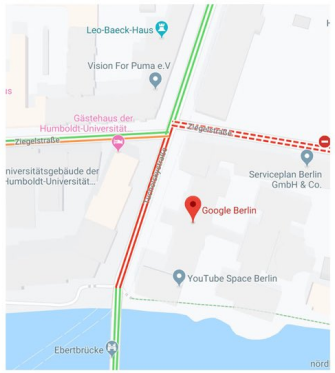
Noise is corrupting pattern, crafted to maximize the classification error
It is an attack

Battista Biggio tutotial: Wild Patterns: Ten Years After the Rise of Adversarial Machine Learning, 2019

Road map

1. Attacking deep networks
2. Adversarial attacks
3. Typology of Attacks
4. Robusteness certificates and MIP
   - Formalizing the search for adversarial examples
   - Robust training
5. ADiL (Adversarial dictionary learning)
   - Adversarial dictionary learning framework
   - Algorithmic solutions
   - Experimental results
6. Conclusion

Threat Models



- Poisoning, Adversarial (evasion), DoS

- Adversarial Goals:

$$a_{f,x} = x + \Delta_{f,x}$$

1. Confidence reduction
2. Specific (targeted) misclassification attack: given class t_a and x a_{f,x,t_a}
3. Generic (untargeted) misclassification: any class for a given x $a_{f,x}$
4. Universal attack (generic misclassification) for any class any x Δ_f

- White-box, black-box and grey-box
  It can also be adaptive (or not)

- Different ways: random search, gradient-based, transfer-based...

How can we produce (strong) adversarial examples?

Generating adversarial examples in one step

Evasion Attacks against ML at Test Time Biggio, et al., ECML 2013

$$\begin{cases} \min\limits_{a \in \mathcal{X}} & f_{c^\star}(a) \\ \text{subject to} & \|x - a\| \leq \delta \ . \end{cases} \tag{1}$$

One step projected gradient descent (ρ large enough)

$$a_{f,x} = \text{Proj}_{\mathcal{A}_x}\big(x - \ \rho \ \nabla_x f_i(x)\big) \qquad \text{with} \qquad \mathcal{A}_x = \big\{a \in \mathcal{X} \mid \|x - a\| \leq \delta\big\}$$

Fast Gradient Sign Method (FGSM), (I. Goodfellow et al, ICLR 2015)

The problem, given (x, t)

$$\begin{cases} \max\limits_{a \in \mathcal{X}} & J\big(f(a), t\big) \\ \text{subject to} & \|x - a\| \leq \delta \end{cases} \qquad \text{training loss}$$

Fast Gradient Sign Method (FGSM) ($\rho = \frac{1}{4}$, .1 or .007) = gradient ascent

$$a \ = \ x \ + \ \rho \ \text{sign}\Big(\nabla_x J\big(f(x), t\big)\Big)$$

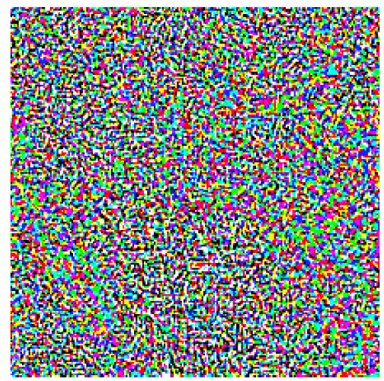
Fast Gradient Sign Method (FGSM) = gradient ascent

$$\mathsf{a} = \mathsf{x} + \rho\, \mathrm{sign}\Big(\nabla_{\mathsf{x}} J\big(f(\mathsf{x}), t\big)\Big)$$

$\boldsymbol{x}$

"panda"
57.7% confidence

$+ .007 \times$



$\mathrm{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"
8.2% confidence

$=$

$\boldsymbol{x} + \epsilon \mathrm{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"
99.3 % confidence

Explaining and Harnessing Adversarial Examples, I. Goodfellow et al, ICLR 2015

Specific Optimization formulation

Specific adversarial for some ($t_a \neq c^\star$), t_a being the adversarial target

The problem:

$$\begin{cases} \min\limits_{a \in \mathcal{X}} & J\big(f(a), t_a\big) \\ \text{subject to} & \|x - a\| \leq \delta \end{cases} \qquad \begin{cases} \min\limits_{a \in \mathcal{X}} & \|x - a\| \\ \text{subject to} & f_{c^\star}(a) + \alpha \leq f_{t_a}(a) \end{cases}$$

$J\big(f(a), t_a\big)$ training loss

The proposed solution: lagrangian form (not convex/not equivalent)

$$\min_{a \in \mathcal{X}} \; L(f(a), t_a) \; + \; \lambda \, \|x - a\|$$

Solved using a box-constrained L-BFGS (with $\mathcal{X} = [0, 1]^p$)

Intriguing properties of neural networks, C. Szegedy et al, ICLR 2014

Multi-step (iterative) approach

Iterative FGSM, (PGD) (Kurakin et al, ICLR 2017)

The problem, given $(x, c^\star)$

$$\begin{cases} \max\limits_{a \in \mathcal{X}} & J\big(f(a), c^\star\big) \\ \text{subject to} & \|x - a\| \leq \delta \end{cases} \quad \text{training loss}$$

The i-FGSM (PGD) proposed solution: build a sequence with (small) ρ_i

$$\begin{cases} a_0 = x \\ a_{i+1} = \text{Proj}_{\mathcal{A}_x}\Big(a_i \; + \; \rho_i \; \text{sign}\Big(\nabla_x J\big(f(a_i), c^\star\big)\Big)\Big) \end{cases}$$

- ρ chosen to change the value of each pixel only by 1 on each step
- due to the non concavity, it only converges towards local maxima
- i-FGSM is equivalent to (the ℓ_∞ version of) Projected Gradient Descent (PGD), Madry et al., ICLR 2018 (sign?)
- Specific version: with $c^\star$ the target class

$$a_{i+1} = \text{Proj}_{\mathcal{A}_x}\Big(a_i \; - \; \rho_i \; \text{sign}\Big(\nabla_x J\big(f(a_i), c^\star\big)\Big)\Big)$$

Optimization attack: Carlini & Wagner (CW), 2017

Specific attack: Given x and $t_a \neq c^\star$ $\begin{cases} \min\limits_{a \in \mathcal{X}} & D(x, a) \\ \text{subject to} & C_f(a) = t_a \end{cases}$

Define an objective function L such that $C_f(a) = t_a$ iff $L(f(a), t_a) \leq 0$

$$\begin{cases} \min\limits_{a \in \mathcal{X}} & D(x, a) \\ \text{subject to} & L(f(a), t_a) \leq 0 \end{cases} \qquad \min_{a \in \mathcal{X}} \; D(x, a) \; + \; \lambda \, L(f(a), t_a)$$

- stochastic gradient descent solver (SGD is slow, use GPU)
- compare 3 $D(x, a) = \|x - a\|_p^p$, ℓ_2, ℓ_0 and ℓ_∞ attacks
- compare 7 objective function L

and the winner is the Carlini & Wagner ℓ_2 attack

Euclidean distance ℓ_2 and and the hinge loss (with confidence α)

$$L\big(f(a), t_a\big) = \max\Big[\alpha - \big(f_{t_a}(a) - \max_{k \neq t_a} f_k(a)\big), 0\Big]$$

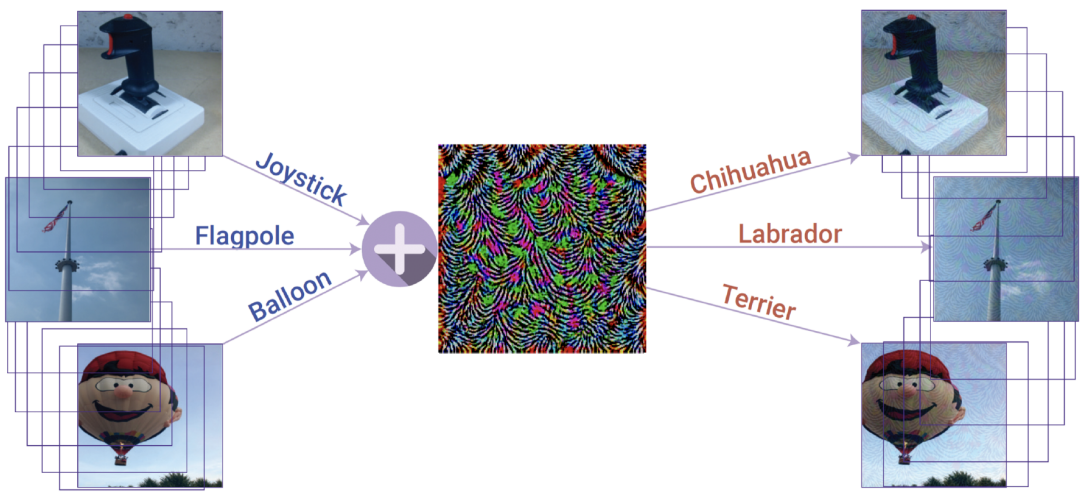
Universal Adversarial Perturbations

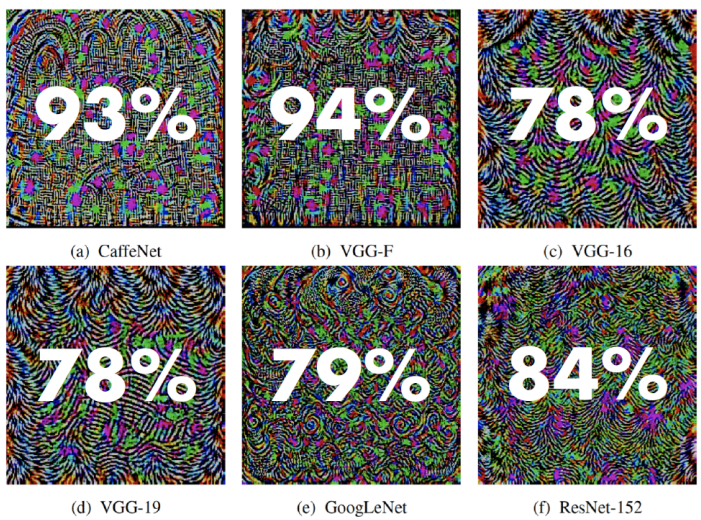
Given f, find Δ small s.t. for "most" $(x, c^\star)$ $\max_{k \neq c^\star} f_k(x+\Delta) > f_{c^\star}(x+\Delta)$

The problem:

$$\begin{cases} \min\limits_{\Delta} & L\big(f(a), t\big) = \mathbb{P}\left(\max\limits_{k \neq c^\star} f_k(x+\Delta) > f_{c^\star}(x+\Delta)\right) \\ \text{subject to} & \|\Delta\|_p \leq \delta \\ & x + \Delta \in \mathcal{X} \end{cases}$$

The proposed solution: Lagrangian formulation + SGD on minibach





(a) CaffeNet (b) VGG-F (c) VGG-16

(d) VGG-19 (e) GoogLeNet (f) ResNet-152

S.M. Moosavi-Dezfooli et al. CVPR, 2017

Comparison of different attack methods

TABLE 1. Summary of the attributes of diverse attacking methods: The 'perturbation norm' indicates the restricted ℓ_p-norm of the perturbations to make them imperceptible. The strength (higher for more asterisks) is based on the impression from the reviewed literature.

Method	Black/White box	Targeted/Non-targeted	Image-specific/Universal	Perturbation norm	Learning	Strength
L-BFGS [22]	White box	Targeted	Image specific	ℓ_∞	One shot	* * *
FGSM [23]	White box	Targeted	Image specific	ℓ_∞	One shot	* * *
BIM & ILCM [35]	White box	Non targeted	Image specific	ℓ_∞	Iterative	****
JSMA [60]	White box	Targeted	Image specific	ℓ_0	Iterative	* * *
One-pixel [68]	Black box	Non Targeted	Image specific	ℓ_0	Iterative	**
C&W attacks [36]	White box	Targeted	Image specific	$\ell_0, \ell_2, \ell_\infty$	Iterative	* * * * *
DeepFool [72]	White box	Non targeted	Image specific	ℓ_2, ℓ_∞	Iterative	****
Universal perturbations [16]	White box	Non targeted	Universal	ℓ_2, ℓ_∞	Iterative	* * * * *
UPSET [146]	Black box	Targeted	Universal	ℓ_∞	Iterative	****
ANGRI [146]	Black box	Targeted	Image specific	ℓ_∞	Iterative	****
Houdini [131]	Black box	Targeted	Image specific	ℓ_2, ℓ_∞	Iterative	****
ATNs [42]	White box	Targeted	Image specific	ℓ_∞	Iterative	****

Akhtar & Mian, Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey, 2018

Most popular attack algorithms (strong first order attacks):

- ℓ_∞: PGD (Madry et al)
- ℓ_2: CW (Carlini & Wagner)
- ℓ_0:

Popular software: Cleverhans and Adversarial Robustness Toolbox (ART)

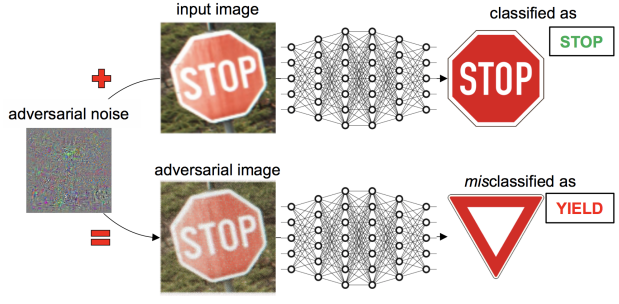
https://github.com/tensorflow/cleverhans

Python library for Adversarial attacks

Adversarial Robustness Toolbox

Road map

1. Attacking deep networks
2. Adversarial attacks
3. Typology of Attacks
4. Robusteness certificates and MIP
   - Formalizing the search for adversarial examples
   - Robust training
5. ADiL (Adversarial dictionary learning)
   - Adversarial dictionary learning framework
   - Algorithmic solutions
   - Experimental results
6. Conclusion



3 formal ways to search for adversarial examples

1. Minimizing the Adversarial Distortion (Bunel et al., NeurIPS 2018)

$$\begin{cases} \min\limits_{a \in \mathcal{X}} & D(x, a) & = & \|x - a\| \\ \text{subject to} & L\big(f(x), f(a)\big) \geq \alpha & = & \max\limits_{k \neq c^\star} f_k(a) > f_{c^\star}(a) + \alpha \end{cases} \tag{2}$$

2. Maximizing the adversarial loss (Wong & Kolter, ICML 2018)

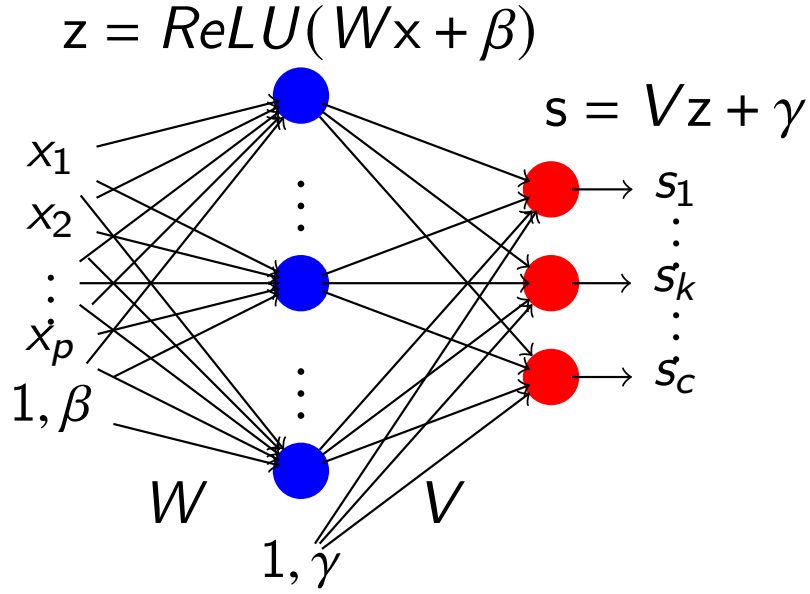
$$\begin{cases} \max\limits_{a \in \mathcal{X}} & L\big(f(x), f(a)\big) & = & f_{t_a}(a) - f_{c^\star}(a) \\ \text{subject to} & D(x, a) \leq \delta & = & \|x - a\| \ \leq \delta \end{cases} \tag{3}$$

3. Robustness as a verification problem (Katz et al, CAV, 2017)
   A classifier f is robust to perturbations on x if and only if:

$$\begin{gathered} \forall a \in \mathcal{A}_x, \big(s = f(a)\big) \Longrightarrow \mathcal{P}(s) \\ \mathcal{A}_x = \big\{a \in \mathcal{X} \mid D(x, a) \leq \delta\big\} \qquad \mathcal{P}(s) = \max_{k \neq c^\star} s_k < s_{c^\star} \end{gathered} \tag{4}$$

   Positive answer (SAT) includes a counter example (adversarial)

The particular case of a one hidden layer MLP



The Neural Network function f with c output nodes

$$\begin{aligned} \mathsf{z} &= ReLU(W\mathsf{x}+\beta) \\ f(\mathsf{x}) &= V\mathsf{z}+\gamma \end{aligned}$$

$$\begin{aligned} \mathsf{h} &= W\mathsf{x}+\beta, \\ \mathsf{z} &= \max(\mathsf{h},0) \\ f(\mathsf{x}) = \mathsf{s} &= V\mathsf{z}+\gamma \end{aligned}$$

The associated classification (or decision function)

$$C_f(\mathsf{x}) = \underset{k=1,\ldots,c}{\operatorname{argmax}}\, s_k$$

Formal verification as an optimization problem

1. Minimizing the Adversarial Distortion

$$\begin{cases} \min\limits_{a\in[0,1]^p} & \|x-a\| \\ \text{subject to} & \max\limits_{k\neq c^\star} f_k(a) > f_{c^\star}(a) \end{cases} \qquad \begin{cases} \min\limits_{a\in[0,1]^p} & \|x-a\|^2 \\ \text{subject to} & h = Wa+\beta \\ & z = \max(h,0) \\ & s = Vz+\gamma \\ & \max\limits_{k\neq c^\star} s_k > s_{c^\star} \end{cases}$$

2. Maximizing the adversarial loss

$$\begin{cases} \max\limits_{a\in[0,1]^p} & s_{t_a} - s_{c^\star} = e_{t_a,c^\star}^\top (Vz+\gamma) \\ \text{subject to} & h = Wa+\beta, \\ & z = \max(h,0), \\ & s = Vz+\gamma \\ & \|x-a\| \leq \delta \end{cases}$$

3. Use satisfiability modulo theories (SAT/SMT) constraints

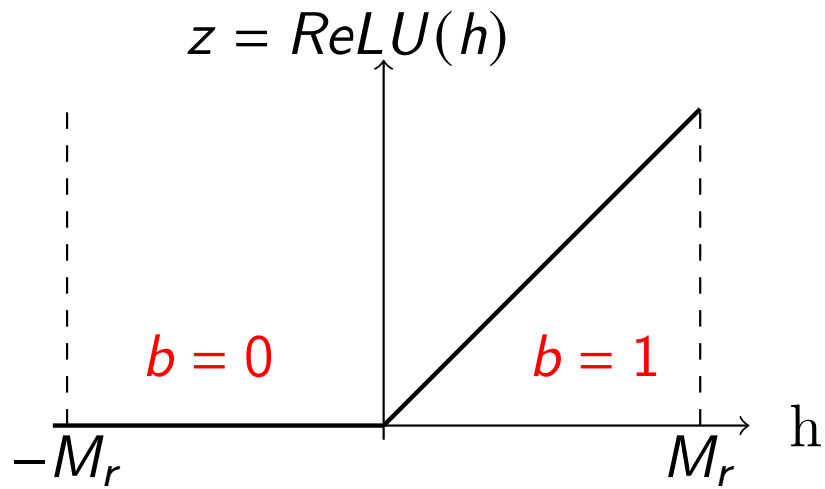
The ReLUplex (Lomuscio & Maganti, Katz et al., 2017)

the ReLU can be formulated as a set of linear constraints

Given $M_r \geq ||\mathrm{h}||_\infty$ and binary variables $b \in \{0,1\}^e$

$$\mathrm{z} = \max(\mathrm{h}, 0) \quad \Leftrightarrow \quad \begin{aligned} &\mathrm{z}_i \geq 0, && i = 1, \ldots, e \\ &\mathrm{z}_i \leq M_r b_i, && i = 1, \ldots, e \\ &\mathrm{z}_i \leq \mathrm{h}_i + M_r(1 - b_i), && i = 1, \ldots, e \\ &\mathrm{z}_i \geq \mathrm{h}_i, && i = 1, \ldots, e \end{aligned}$$

$$b_i = 0 \Leftrightarrow z_i = 0$$
$$b_i = 1 \Leftrightarrow z_i = h_i \geq 0$$



Exact search for adversarial examples as a MIP

Thanks to the ReLUplex,

$$\begin{cases} \min\limits_{a\in[0,1]^p} & \|x-a\|_p^p \\ \text{subject to} & h = Wa+\beta \\ & z = \max(h,0) \\ & s = Vz+\gamma \\ & \max\limits_{i\neq i^\star} s_i > s_{i^\star} \end{cases} \qquad \begin{cases} \min\limits_{\substack{a\in[0,1]^p,\\ b\in\{0,1\}^e}} & \|x-a\|_p^p & \\ \text{subject to} & h = Wa+\beta & \\ & z_i \geq 0 & i=1,\dots,e \\ & z_i \leq M_r b_i & i=1,\dots,e \\ & z_i \leq h_i + M_r(1-b_i) & i=1,\dots,e \\ & z_i \geq h_i & i=1,\dots,e \\ & s = Vz+\gamma & \\ & \max\limits_{i\neq i^\star} s_i > s_{i^\star} & \end{cases}$$

$\|x-a\|_\infty, \|x-a\|_1$	MILP
$\|x-a\|_2^2$	MIQP
$\|x-a\|_0$	MILP with more binary variables

$\rightarrow$ max, convolution, pooling can also be linearized

Mixed integer linear program (MILP)

- linear cost
- linear constraints
- integer and continuous variables

Definition (mixed integer linear program – MILP (canonical form))

$$\begin{cases} \min\limits_{\mathrm{a}\in\mathbb{R}^p,\ \mathrm{b}\in\mathbb{N}^q} & J(\mathrm{a},\mathrm{b}) = \mathrm{w}^t\mathrm{a} + \mathrm{d}^t\mathrm{b} & \longleftarrow \text{linear} \\ \text{s.t.} & A\mathrm{w} + B\mathrm{z} \leq c & \longleftarrow \text{linear} \\ & \mathrm{w} \geq 0, & \end{cases}$$

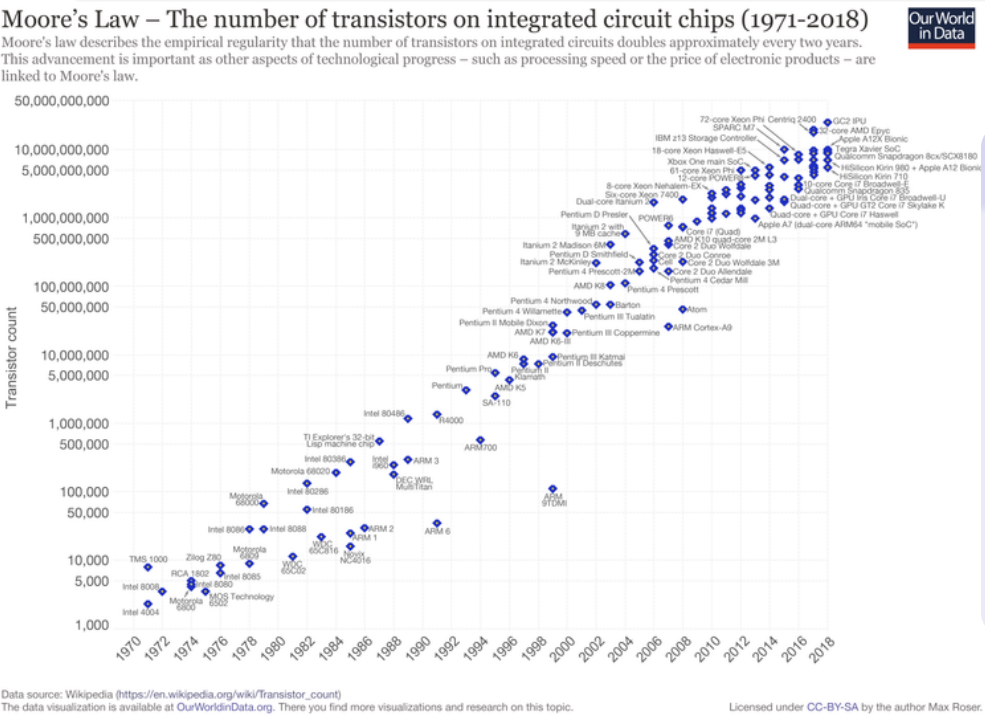
for some given $\mathrm{w} \in \mathbb{R}^p, c \in \mathbb{R}^m, A \in \mathbb{R}^{m\times p}, B \in \mathbb{R}^{m\times q}$ and $\mathrm{d} \in \mathbb{R}^q$.

- A mixed binary linear program is a MILP with $\mathrm{b} \in \{0,1\}^q$ binary.
- When its domain is not empty and bounded, a MILP admits a unique global minimum.

Progresses in MILP

in 1989

MILP is a powerful modeling tool, "They are, however, theoretically complicated and computationally cumbersome"



from 1998 to 2018

	improvement factor
machine	$\times 2^{10} = 1000 - 1600$
solver	$\times 1000 - 3600$
formulation	???
global	$\times 1 - 5\ 10^6$

a year to solve 10 – 20 years ago $\longrightarrow$ now 30 seconds

"mixed integer linear techniques are nowadays mature, that is fast, robust, and are able to solve problems with up to millions of variables"

Mixed integer software (available with python)

Software package	
Open source	
GLPK	glpk for mixed integer linear programming
LP_Solve	
ECOS_BB	
Commercial	(with academic license)
CVXpy	cvx for mixed integer linear programming
CPLEX	cplexmilp for mixed integer linear programming
	cplexmiqp for mixed integer quadratic programming
	cplexmiqcp for mixed integer quadratically constrained pg
GUROBI	gurobi for MILP, MIQP and MIQCQP
Mosek	mosekopt for MILP, MIQP and MIQCQP
NAS	NAS for MILP, MIQP and MIQCQP

Mixed Integer Linear Programming Benchmark (MIPLIB2017)

recommend CVXpy, CPLEX, GUROBI and NAS

http://plato.asu.edu/ftp/milp.html

MIP, lower bound & upper bound

$$
\begin{cases}
\min\limits_{\substack{\mathrm{a}\in[0,1]^p,\\ \mathrm{b}\in\{0,1\}^e}} & \|\mathrm{x}-\mathrm{a}\|_p^p \\
\text{subject to} & \mathrm{h} = W\mathrm{a}+\beta \\
& \mathrm{z}_i \geq 0, \mathrm{z}_i \geq M_r \\
& \mathrm{z}_i \leq M_r b_i, \mathrm{z}_i \leq \mathrm{h}_i + M_r(1-b_i) \\
& \mathrm{e}^\top(V\mathrm{z}+\gamma) \geq \alpha
\end{cases}
$$

Lower bound: continuous relaxation

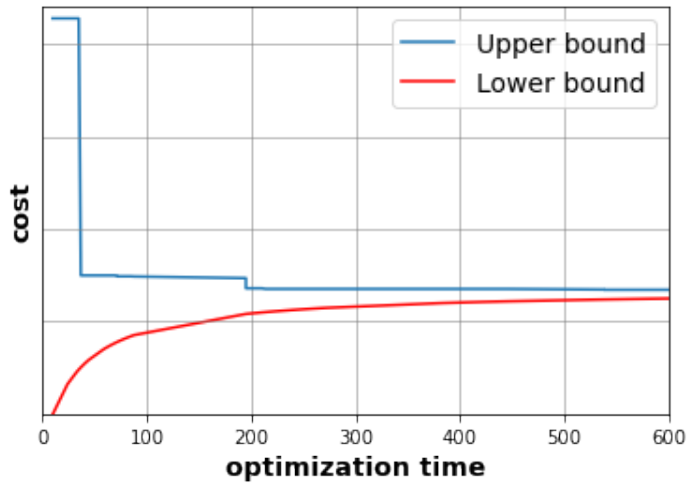
$$
\begin{cases}
\min\limits_{\substack{\mathrm{a}\in[0,1]^p,\\ \mathrm{b}\in[0,1]^e}} & \|\mathrm{x}-\mathrm{a}\|_p^p \\
\text{subject to} & \mathrm{h} = W\mathrm{a}+\beta \\
& \mathrm{z}_i \geq 0, M_r \\
& \mathrm{z}_i \leq M_r b_i, \mathrm{h}_i + M_r(1-b_i) \\
& \mathrm{e}^\top(V\mathrm{z}+\gamma) \geq \alpha \\
& \text{additional constraints}
\end{cases}
$$

Upper bound: fix b (feasible)

$$
\begin{cases}
\min\limits_{\substack{\mathrm{a}\in[0,1]^p,\\ \bar{b}}} & \|\mathrm{x}-\mathrm{a}\|_p^p \\
\text{subject to} & \mathrm{h} = W\mathrm{a}+\beta \\
& \mathrm{z}_i \geq 0, M_r \\
& \mathrm{z}_i \leq M_r b_i, \mathrm{h}_i + M_r(1-b_i) \\
& \mathrm{e}^\top(V\mathrm{z}+\gamma) \geq \alpha \\
& \text{additional constraints}
\end{cases}
$$

MIP, Upper bound & Lower bound

$$\|x - a_{lb}\|_p^p \quad \leq \quad \|x - a^{\star}_{x,f}\|_p^p \quad \leq \quad \|x - a_{ub}\|_p^p$$



- Optimality:
  - it may be "easy" to find the optimal solution...
  - ...and very hard to prove it
- Computational efficiency: how to manage your time budget?
  - initialization
  - acceleration through stronger relaxation
  - combinatorial exploration

MIP acceleration using asymmetric bounds ($\approx \times 5$)

$$\underbrace{\begin{bmatrix} 1.1 \\ 2.8 \\ -0.2 \\ 0.9 \\ -2.2 \end{bmatrix}}_{w} = \underbrace{\begin{bmatrix} 1.1 \\ 2.8 \\ 0 \\ 0.9 \\ 0 \end{bmatrix}}_{w_+} - \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0.2 \\ 0 \\ 2.2 \end{bmatrix}}_{w_-}$$

$$\ell \le a \le u \;\&\; h = w^\top a + \beta \Rightarrow \underbrace{w_+^\top \ell - w_-^\top u + \beta}_{\ell'} \le h \le \underbrace{w_+^\top u - w_-^\top \ell + \beta}_{u'}$$

Pre computing binary variables: if $0 \le \ell'_i$ then $b_i = 1$
if $u'_i \le 0$ then $b_i = 0$

Non symetric bound (ReLU)

$$\begin{aligned} z_i &\ge 0, && i = 1, \dots, e \\ z_i &\le u' b_i, && i = 1, \dots, e \\ z_i &\le h_i - \ell'(1 - b_i), && i = 1, \dots, e \\ z_i &\ge h_i, && i = 1, \dots, e \end{aligned}$$

Evaluating robustness of neural networks with mixed integer programming, Tjeng et al., ICLR 2019

MIPVerify (Julia package + Gurobi)

Finding an Adversarial Example

We now try to find the closest L_infty norm adversarial example to the first image, setting the target category as index `10` (corresponding to a true label of 9). Note that we restrict the search space to a distance of `0.05` around the original image via the specified `pp`.

In [12]:
```
target_label_index = 10
d = MIPVerify.find_adversarial_example(
    nn,
    sample_image,
    target_label_index,
    Gurobi.Optimizer,
    Dict(),
    norm_order = Inf,
    pp=MIPVerify.LInfNormBoundedPerturbationFamily(0.05)
)
```

```
Academic license - for non-commercial use only
[notice | MIPVerify]: Attempting to find adversarial example. Neural net predicted label is 8, target labels are [
[notice | MIPVerify]: Determining upper and lower bounds for the input to each non-linear unit.
  Calculating upper bounds: 100%|████████████████████████| Time: 0:00:00
Academic license - for non-commercial use only
  Calculating lower bounds: 100%|████████████████████████| Time: 0:00:00
  Imposing relu constraint: 100%|████████████████████████| Time: 0:00:00
  Calculating upper bounds:  10%|██                      |  ETA: 0:02:41
Academic license - for non-commercial use only
  Calculating upper bounds: 100%|████████████████████████| Time: 0:00:26
  Calculating lower bounds: 100%|████████████████████████| Time: 0:00:08
  Imposing relu constraint: 100%|████████████████████████| Time: 0:00:00
Academic license - for non-commercial use only
```

https://github.com/vtjeng/MIPVerify.jl/blob/master/docs/src/index.md

Robustness

- Robustness measure of f (against an attack a perturbation $\Delta(x_i^t)$):
  - Fooling rate

$$\frac{1}{n_t}\sum_{i=1}^{n_t} \mathbb{1}_{\{f(x_i^t+\Delta(x_i^t))\neq f(x_i^t)\}}$$

- Robust training

Definition (Robust adversarial loss [MMS+18])

$$\min_f \mathbb{E}_{(X,T)}\left[\max_{\Delta\in\mathcal{A}_x} L\big(f(X+\Delta),T\big)\right]$$

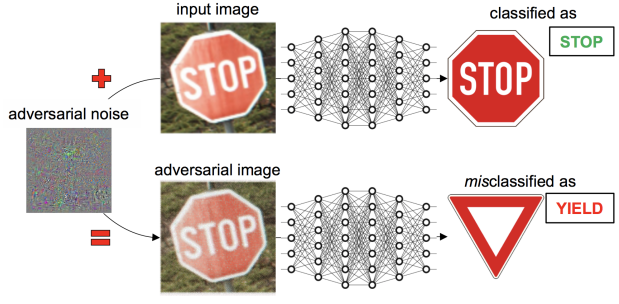
  - Long history in robust optimization, going back to Wald
  - Towards deep learning models resistant to adversarial A., Madry, 2019

$$\max_f \mathbb{E}_{(X,T)}\left[\min_{f(X+\Delta(X))\neq f(X)} \|\Delta(X)\|\right]$$

  - Adversarial robustness is impossible in general, Dohmatob, ICML 2019

Road map

1. Attacking deep networks
2. Adversarial attacks
3. Typology of Attacks
4. Robusteness certificates and MIP
   - Formalizing the search for adversarial examples
   - Robust training
5. ADiL (Adversarial dictionary learning)
   - Adversarial dictionary learning framework
   - Algorithmic solutions
   - Experimental results
6. Conclusion



How to craft adversarial examples?

- Specific: for a given x_i

$$a_i = x_i + \Delta(x_i)$$

  - FGSM [GSS15, KGB17]

$$\Delta(x_i) = \rho \, \text{sign}(\nabla_{x_i} H(f(x_i; \theta), y_i)),$$

  - DeepFool [MFF16]

$$\Delta(x_i) = \text{argmin}_\Delta \|\Delta\|, \text{ s.t. } \text{argmax}_k f(x_i + \Delta; \theta) \neq \text{argmax}_k f(x_i; \theta)$$

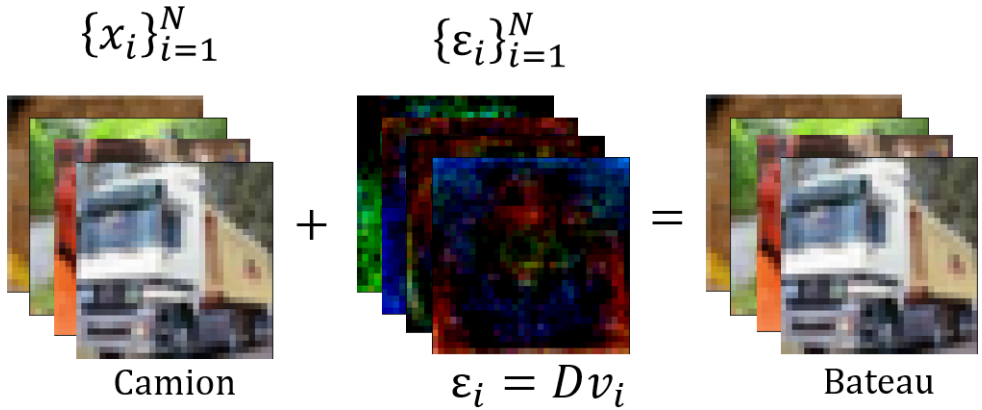
- Universal [MDFFF17]: for any example

$$\Delta(x_i) = \underset{\Delta}{\text{argmax}} \sum_{j=1}^{N} H(f(x_j + \Delta; \theta), y_j) \quad \text{s.t.} \quad \|\Delta\|_p \leq \delta \ ,$$

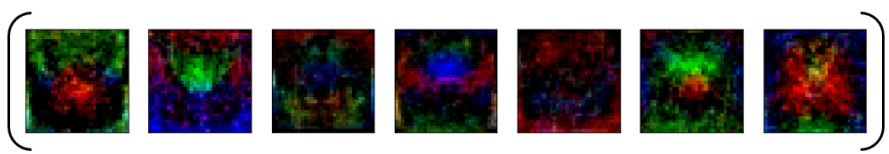
- Use a dictionary D:

$$\Delta(x_i) = D v_i$$

Adversarial dictionary learning: $\Delta(x_i) = Dv_i$



$$\underset{[D,v]}{\text{minimize}} \sum_{i=1}^{N} \underbrace{\ell_i(x_i + Dv_i)}_{\text{adversary}} + \underbrace{\lambda_1 \|v_i\|_1}_{\text{sparse}} + \underbrace{\lambda_2 \|Dv_i\|_2^2}_{\Delta_i \text{ small}}$$

$D =$ 

D universal, $v_i \in \mathbb{R}^M$ specific ($M \ll N$)

Algorithmic solutions

- Full-batch version: ADiL
  - with proofs
- Stochastic version: SADiL
  - that scales...

Full-batch version: ADiL

$$\underset{\substack{D \in \mathbb{R}^{P \times M} \\ V \in \mathbb{R}^{M \times N}}}{\text{minimize}} \; \mathcal{L}(D, V) \triangleq F(D, V) + \Omega(D, V)$$

- Smooth supervised fitting term

$$F(D, V) = \sum_{i=1}^{N} \lambda_2 \|Dv_i\|^2 + H\big(f(x_i + Dv_i; \theta), t_i\big)$$

- Non-smooth regularization

$$\Omega(D, V) = \iota_{\mathcal{C}}(D) + \sum_{i=1}^{N} \lambda_1 \|v_i\|_1, \quad \mathcal{C} = \{D \mid \forall m, \|d_m\|_2 \leq 1\}$$

A sparse representation for a better dictionary

Algorithm 1 ADiL

Require: Parameter $\delta \in]0,1[$, $D^{(0)} \sim \mathcal{N}(0_{P\times M}, 1_{P\times M})$, $V^{(0)} = 0_{M\times N}$

for $k = 0$ to $K-1$ **do**

Proximal-gradient step

$D^{(k+1/2)} = \mathrm{Proj}_{\mathcal{C}}(D^{(k)} - \gamma_k \nabla_D F(D^{(k)}, V^{(k)}))$

$V^{(k+1/2)} = \mathrm{Soft}_{\gamma_k \lambda_1}(V^{(k)} - \gamma_k \nabla_V F(D^{(k)}, V^{(k)}))$

Armijo-like backtracking

$d_D^{(k)} = D^{(k+1/2)} - D^{(k)}$

$d_V^{(k)} = V^{(k+1/2)} - V^{(k)}$

$i_k = 0$

repeat

$\tilde{D}^{(k)} = D^{(k)} + \delta^{i_k} d_D^{(k)}$

$\tilde{V}^{(k)} = V^{(k)} + \delta^{i_k} d_V^{(k)}$

$i_k = i_k + 1$

until decreasing criterion satisfied

$D^{(k+1)} = \tilde{D}^{(k)}$

$V^{(k+1)} = \tilde{V}^{(k)}$

end for

return $\{D^{(K)}, V^{(K)}\}$

Convergence

Theorem (Convergence [BLP+17])

Let $\{D^{(k)}, V^{(k)}\}_{k\in\mathbb{N}}$ be the sequence of ADiL Algorithm 1. Then,

- *each limit point of $\{D^{(k)}, V^{(k)}\}_{k\in\mathbb{N}}$ is a stationary point of ADiL*
- *$\{\mathcal{L}(D^{(k)}, V^{(k)})\}_{k\in\mathbb{N}}$ converges to the limit point objective value*

In addition, if $\mathcal{L}$ satisfies the Kurdyka-Łojasiewicz property at any point, then the sequence converges to a stationary point of ADiL

Stochastic version: SADiL

Two ingredients: an alternating scheme

$$\begin{cases} D^{(k+1)} & = \mathrm{Proj}_{\mathcal{C}} \left(D^{(k)} - \gamma_k \widetilde{\nabla} F(D^{(k)}, V^{(k)}) \right), \\ V^{(k+1)} & = \mathrm{Soft}_{\gamma_k \lambda_1} \left(V^{(k)} - \gamma_k \widetilde{\nabla} F(D^{(k+1)}, V^{(k)}) \right) , \end{cases}$$

$\widetilde{\nabla} F$: random estimate of the gradient on a mini-batch $\mathcal{B}_k \sim \{1, \dots, N\}$

$$\widetilde{\nabla} F(D, V) = \frac{N}{|\mathcal{B}_k|} \sum_{i \in \mathcal{B}_k} \nabla F_i(D, V).$$

For $|\mathcal{B}_k| = N$, we recover ADiL

Generation of adversary examples

Design of adversarial perturbations to unseen examples.

1. Use ADiL with fixed D to find $v^{(K)}$
2. Project onto the input manifold $\mathcal{X} \subseteq \mathbb{R}^P$

$$x' = \mathrm{Proj}_{\mathcal{X}}\left(x + Dv^{(K)}\right)$$

Defense mechanism

Problem (Defense mechanism)

$$\underset{\theta \in \Theta}{\text{minimize}} \quad \underset{\{x,y\} \sim \mathcal{D} \cup \mathcal{A}}{\mathbb{E}} H(f(x;\theta), y) \ , \tag{5}$$

where $\mathcal{D} \cup \mathcal{A}$ *is the augmented training set*

Two manners of constructing the adversarial set with correct labeling.

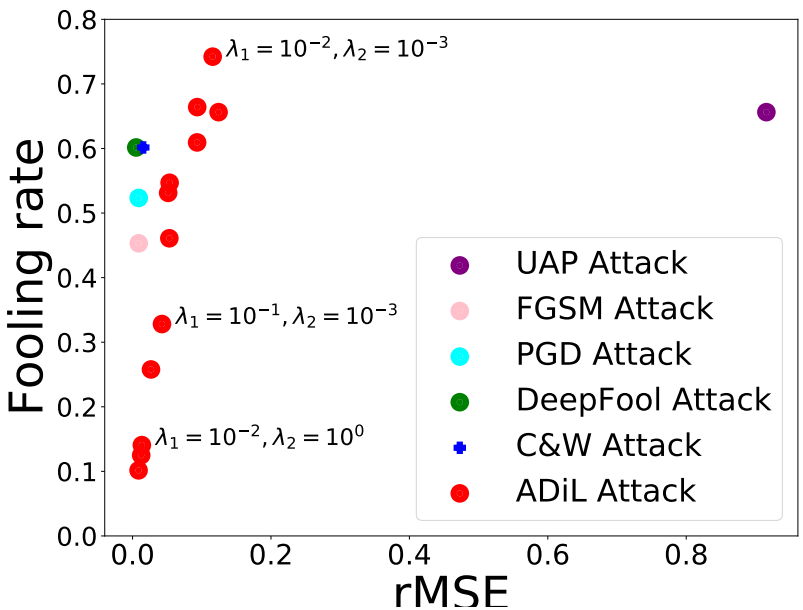
(Adversarial training) $\mathcal{A} = \{x_i + \hat{D}\hat{v}_i, y_i\}_{i=1}^N$,

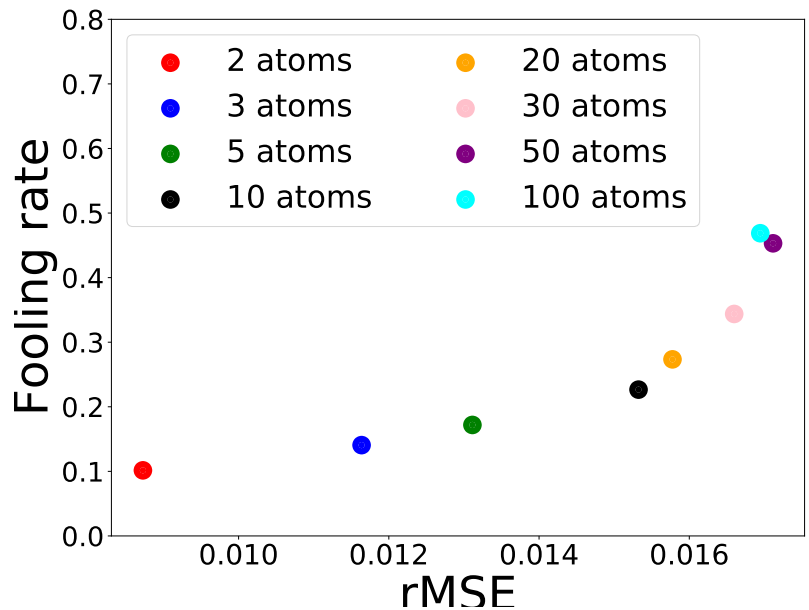
(Noise injection) $\mathcal{A} = \{x_i + \hat{D}z_i, y_i\}_{i=1}^N$ *with* $z_i \sim \text{Laplace}(0, b)$,

where b *is estimated by fitting a Laplacian distribution to the* $\hat{v}_i$*'s.*

Experimental results: LeNet classifier on CIFAR-10

rMSE: $(1/|\mathcal{T}_2|)\sum_{i=1}^{|\mathcal{T}_2|}\|Dv_i\|^2/\|x_i\|^2$





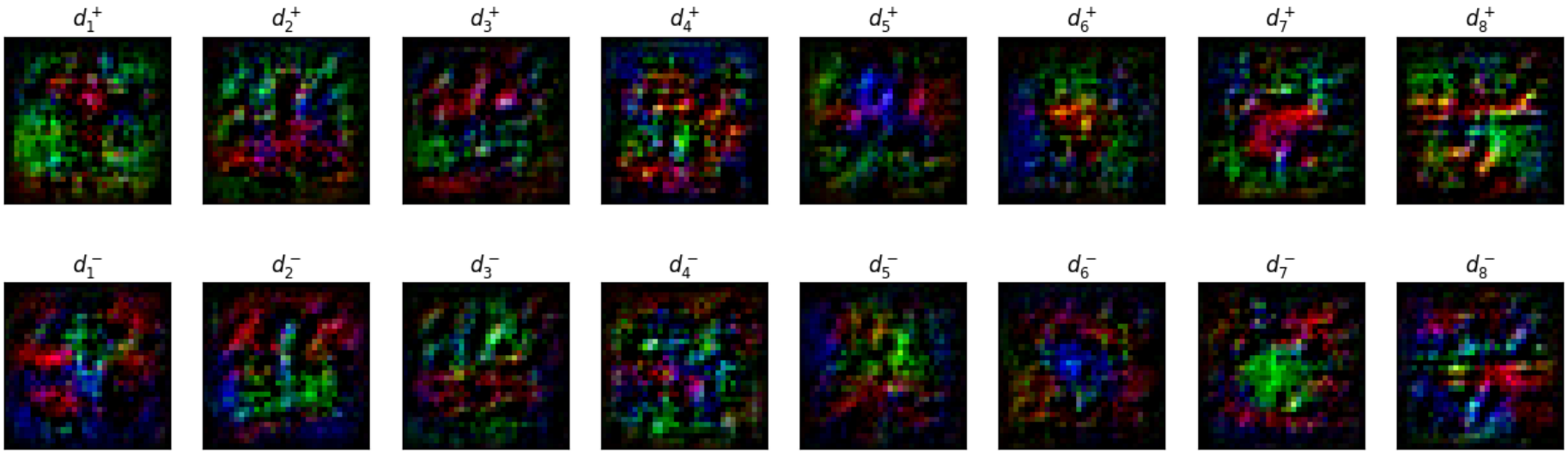
Experimental results on ResNet18 classifier

		PGD	DeepFool	C&W	ADiL	UAP
CIFAR-10	Fool. Rate	54.69%	74.22%	74.22%	**90.63%**	77.34%
	rMSE	0.0091	**0.0056**	0.032	0.071	0.747
ImageNet	Fool. Rate	22.66%	17.19%	3.91%	38.28%	**100%**
	rMSE	0.00054	**0.00022**	0.00025	0.0458	1.52

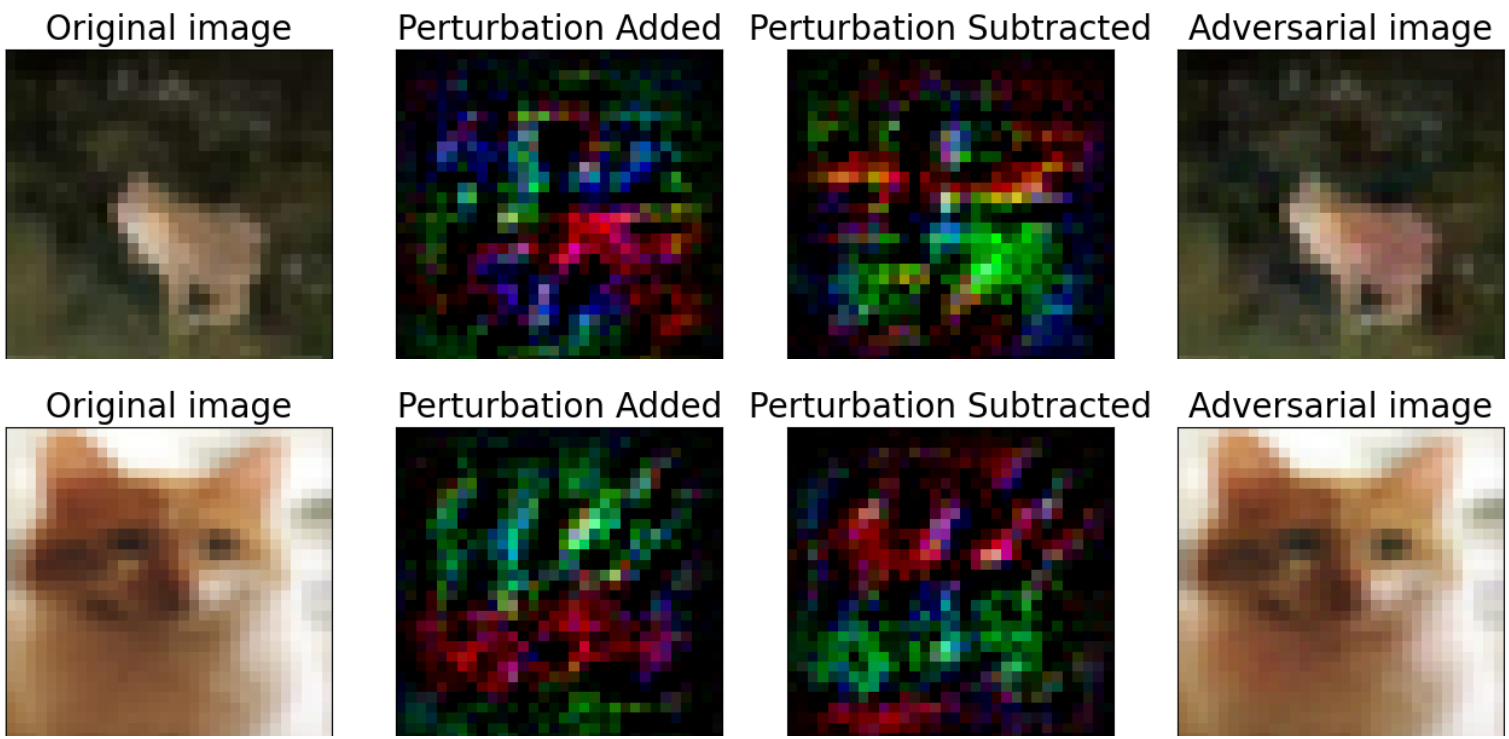
Defense mechanism for LeNet on CIFAR-10

M_{attacker}	2 atoms	5 atoms	10 atoms	15 atoms	20 atoms
No Defense	25.78%	56.25%	60.15%	46.09%	57.81%
With Defense	**15.62%**	**30.46%**	**53.90%**	**44.53%**	**56.25%**

Dictionary of ADiL attacks for LeNet on CIFAR-10



Two examples of ADiL attacks for LeNet on CIFAR-10

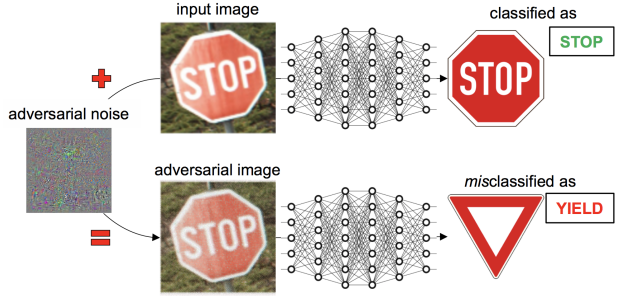


Conclusion for ADiL

- A new way to generate adversarial examples
- with a universal component D
  - interpretable?
  - transferable?
- efficient way to compute specific components v_i
- improve the defence mechanism to train robust NN

Road map

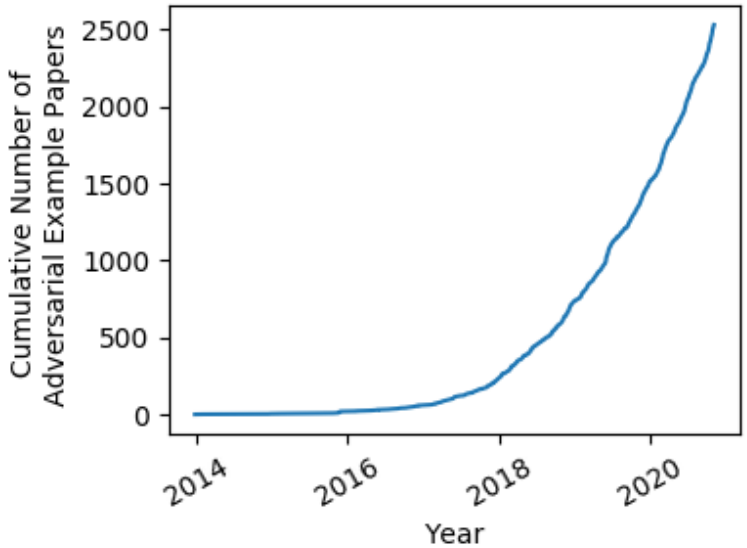
1. Attacking deep networks
2. Adversarial attacks
3. Typology of Attacks
4. Robusteness certificates and MIP
   - Formalizing the search for adversarial examples
   - Robust training
5. ADiL (Adversarial dictionary learning)
   - Adversarial dictionary learning framework
   - Algorithmic solutions
   - Experimental results
6. Conclusion



Conclusion

- Deep networks can be (and will be) attacked
- The problem can be formalized as a MIP (NP hard)
  - looking for a formal solution
- Improve the model (Wasserstein distance, Wong et al ICML 2019)
  - improve the solver
  - deal with numerical issues
- Think about proofs
  - Robustness certificate
  - Are adversarial examples inevitable? A. Shafahi et al, ICLR 2019.
  - Limits on robustness to adversarial examples, E. Dohmatob, ICML 2019
- Think about defenses: change training

Some links



- Cleverhans
  http://www.cleverhans.io/
- Adversarial Robustness Toolbox (ART)
  https://adversarial-robustness-toolbox.readthedocs.io/en/stable/
- Robust ML
  https://www.robust-ml.org/defenses
- A (Complete) List of All (arXiv) Adversarial Example Papers by N. Carlini
  nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html
- ForMaL: DigiCosme Spring School on Formal Methods and Machine Learning 4th-7th June 2019, ENS Paris-Saclay, Cachan, France
  https://formal-paris-saclay.fr/
- NeurIPS 2018 tutorial, "Adversarial Robustness: Theory and Practice", by Zico Kolter and Aleksander Madry
  https://adversarial-ml-tutorial.org/
- Opportunities and Challenges in Deep Learning Adversarial Robustness: A Survey Silva & Najafirad, submited to IEEE Transactions on Knowledge and Data Engineering, 2020
  https://arxiv.org/abs/2007.00753

References I

[BLP+17] Silvia Bonettini, Ignace Loris, Federica Porta, Marco Prato, and Simone Rebegoldi, *On the convergence of a linesearch based proximal-gradient method for nonconvex optimization*, Inverse Problems 33 (2017), no. 5, 055005.

[GSS15] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, *Explaining and harnessing adversarial examples*, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (Yoshua Bengio and Yann LeCun, eds.), 2015.

[KGB17] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio, *Adversarial examples in the physical world*, 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings, OpenReview.net, 2017.

[MDFFF17] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard, *Universal adversarial perturbations*, Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1765–1773.

[MFF16] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, *Deepfool: A simple and accurate method to fool deep neural networks*, 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, IEEE Computer Society, 2016, pp. 2574–2582.

[MMS+18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, *Towards deep learning models resistant to adversarial attacks*, 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, 2018.