Pattern Recognition and Applications Lab

University of Cagliari, Italy

Where ML Security Is Broken and How to Fix It

Maura Pintor

Assistant Professor @ University of Cagliari

Padova, December 14, 2023

Attacks against AI are Pervasive!

Sharif et al., *Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition*, ACM CCS 2016

“without the dataset the article is useless”

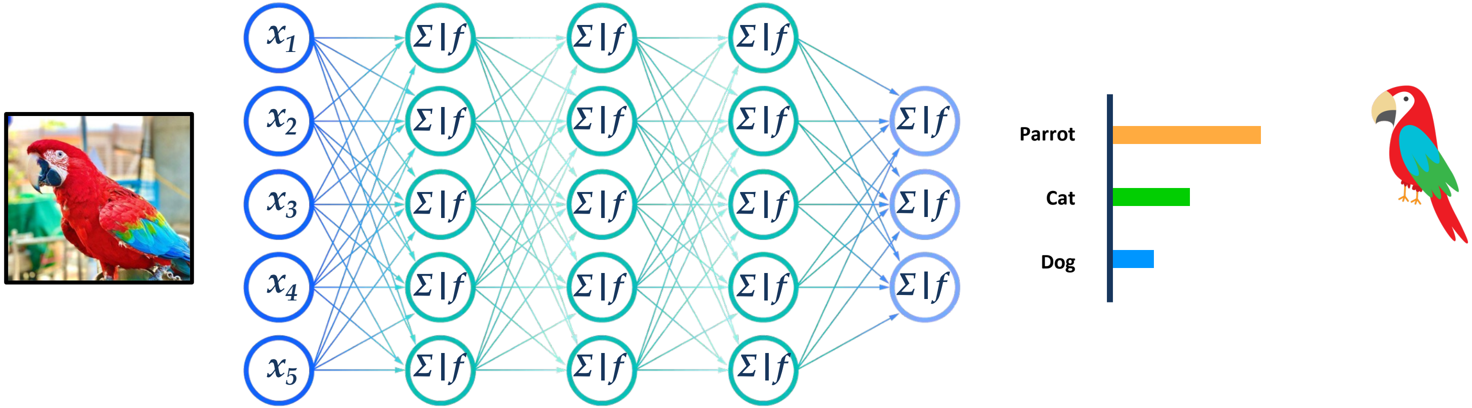
“okay google browse to evil dot com”

Carlini and Wagner, *Audio adversarial examples: Targeted attacks on speech-to-text*, DLS 2018
https://nicholas.carlini.com/code/audio_adversarial_examples/

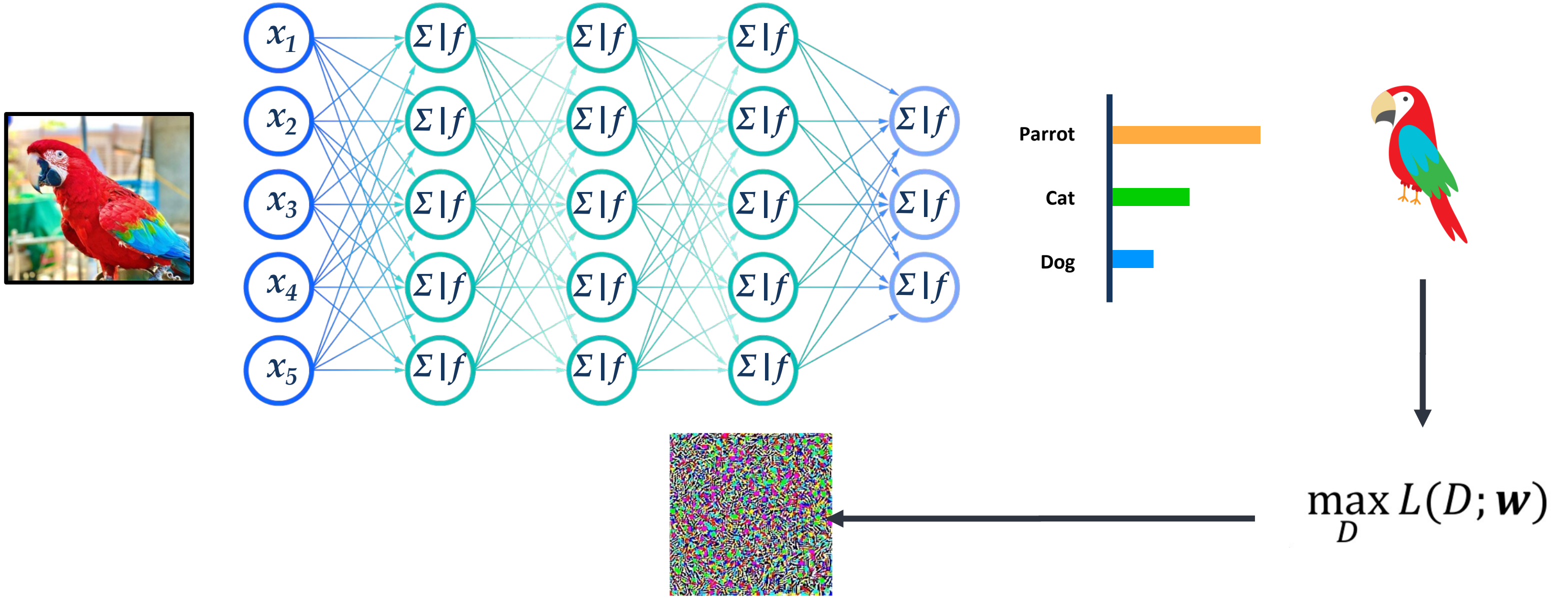
Eykholt et al., *Robust physical-world attacks on deep learning visual classification*, CVPR 2018

- Demetrio, Biggio, Roli et al., *Adversarial EXEmples: ...*, ACM TOPS 2021
- Demetrio, Biggio, Roli et al., *Functionality-preserving black-box optimization of adversarial windows malware*, IEEE TIFS 2021
- Demontis, Biggio, Roli et al., *Yes, Machine Learning Can Be More Secure!...*, IEEE TDSC 2019

Adversarial Examples (AdvX)

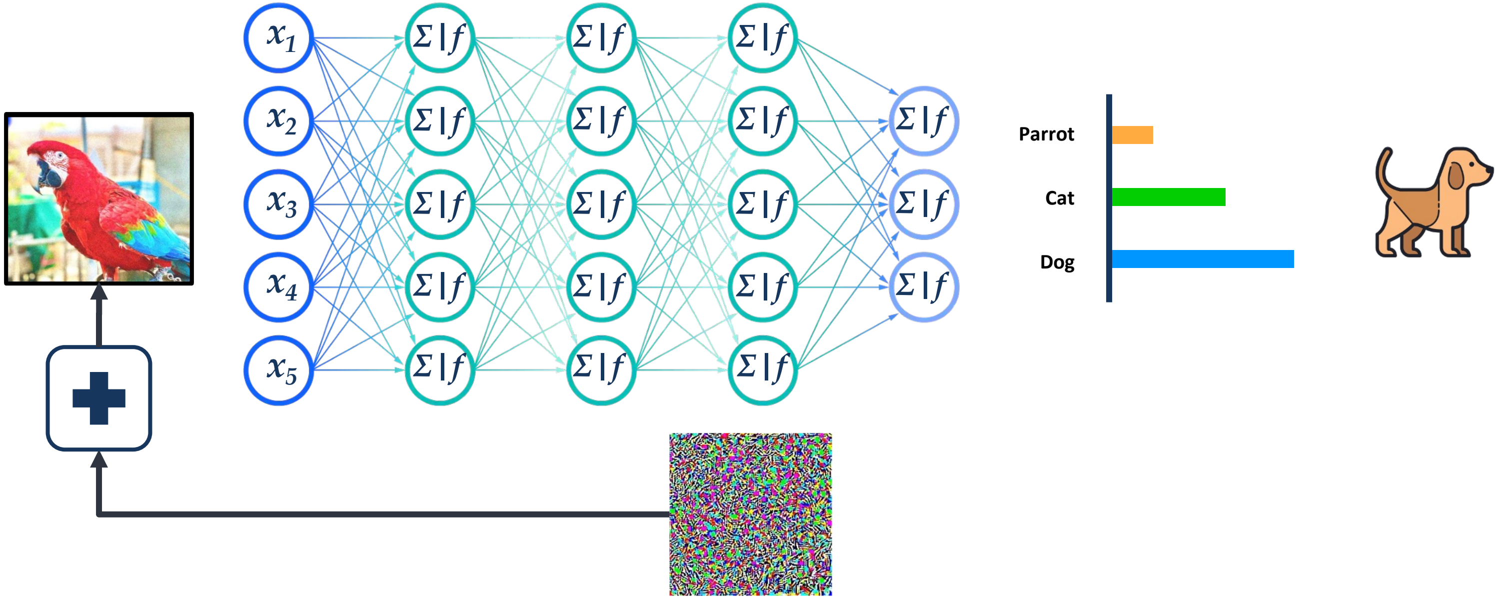


Adversarial Examples (AdvX)



Biggio et al., *Evasion Attacks Against Machine Learning at Test Time*, ECML PKDD 2013
Szegedy et al., *Intriguing Properties of Neural Networks*, ICLR 2014

Adversarial Examples (AdvX)



Biggio et al., *Evasion Attacks Against Machine Learning at Test Time*, ECML PKDD 2013
Szegedy et al., *Intriguing Properties of Neural Networks*, ICLR 2014

How to craft AdvXs

Exhaustive search → not possible for modern deep learning models
Empirical evaluation → attack = optimization problem + solving algorithm

$$
\begin{aligned}
\boldsymbol{\delta}^\star \in \arg\min_{\boldsymbol{\delta}} \quad & \mathcal{L}(\boldsymbol{x}+\boldsymbol{\delta}, y, \boldsymbol{\theta}) \\
\text{s.t.} \quad & \|\boldsymbol{\delta}\|_p \leq \epsilon \\
& \boldsymbol{x}_{\text{lb}} \preceq \boldsymbol{x}+\boldsymbol{\delta} \preceq \boldsymbol{x}_{\text{ub}}
\end{aligned}
$$

Optimize model's confidence on bad decision

keeping perturbation small

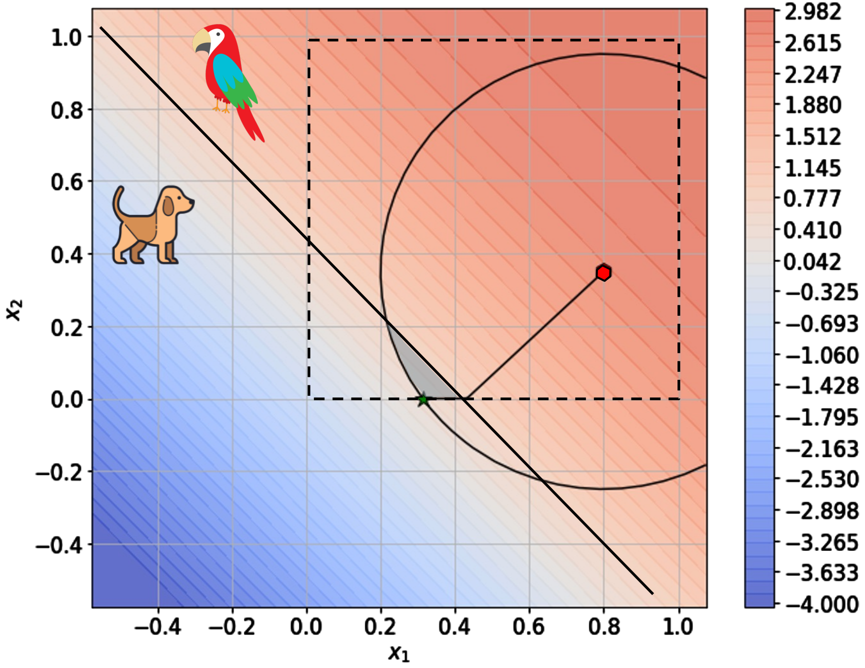
and respecting feature space constraints

How to craft AdvXs

Exhaustive search → not possible for modern deep learning models
Empirical evaluation → attack = optimization problem + solving algorithm

$$
\begin{aligned}
\boldsymbol{\delta}^\star \in \arg\min_{\boldsymbol{\delta}} \quad & \mathcal{L}(\boldsymbol{x}+\boldsymbol{\delta}, y, \boldsymbol{\theta}) \\
\text{s.t.} \quad & \|\boldsymbol{\delta}\|_p \leq \epsilon \\
& \boldsymbol{x}_{\text{lb}} \preceq \boldsymbol{x}+\boldsymbol{\delta} \preceq \boldsymbol{x}_{\text{ub}}
\end{aligned}
$$

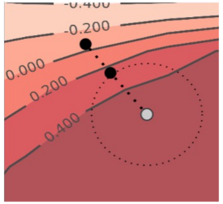
Optimize model's confidence on bad decision
keeping perturbation small
and respecting feature space constraints



Biggio et al., *Evasion Attacks Against Machine Learning at Test Time*, ECML PKDD 2013
Szegedy et al., *Intriguing Properties of Neural Networks*, ICLR 2014

How to craft AdvXs

Projected Gradient



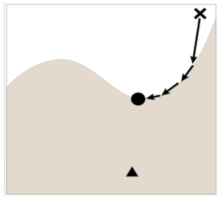
$$\begin{aligned} \boldsymbol{\delta}^\star \in \arg\min_{\boldsymbol{\delta}} \quad & \mathcal{L}(\boldsymbol{x}+\boldsymbol{\delta}, y, \boldsymbol{\theta}) \\ \text{s.t.} \quad & \|\boldsymbol{\delta}\|_p \leq \epsilon \\ & \boldsymbol{x}_{\text{lb}} \preceq \boldsymbol{x}+\boldsymbol{\delta} \preceq \boldsymbol{x}_{\text{ub}} \end{aligned}$$

Optimizes confidence

s.t. distance constraint

and feature space constraints

Boundary

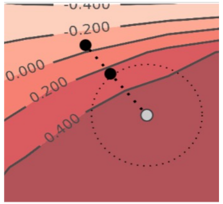


$$\begin{aligned} \boldsymbol{\delta}^\star \in \arg\min_{\boldsymbol{\delta}} \quad & \|\boldsymbol{\delta}\|_p \\ \text{s.t.} \quad & f_y(\boldsymbol{x}+\boldsymbol{\delta}, \boldsymbol{\theta}) \neq f_y(\boldsymbol{x}, \boldsymbol{\theta}) \\ & \boldsymbol{x}_{\text{lb}} \preceq \boldsymbol{x}+\boldsymbol{\delta} \preceq \boldsymbol{x}_{\text{ub}} \,, \end{aligned}$$

Find closest advX

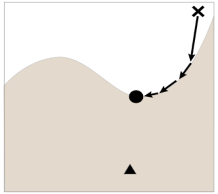
s.t. misclassification constraint

and feature space constraints



\+ Fast evaluation

\- Punctual evaluation (fixed ϵ)



\+ Full picture of robustness (boundary)

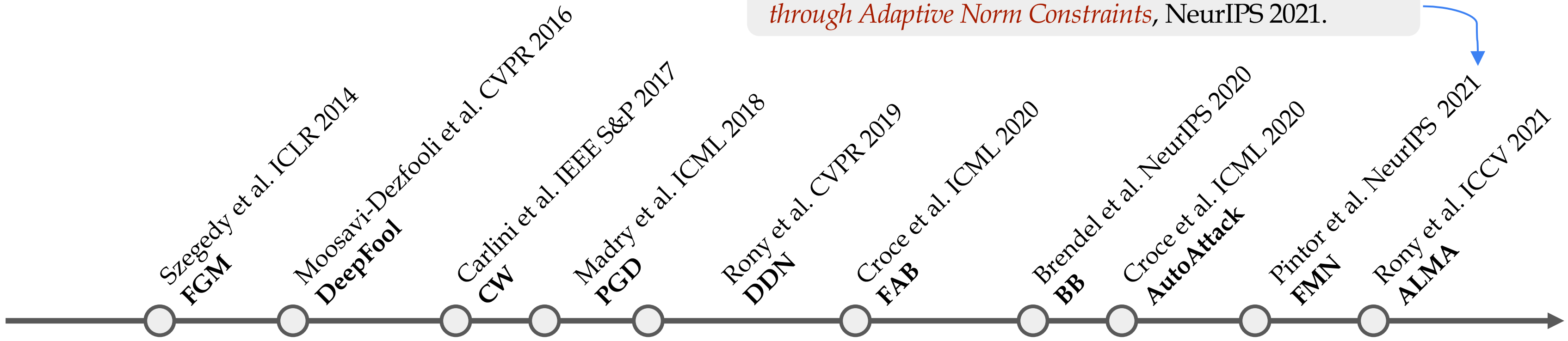
\- Require many iterations

\- Difficul to configure properly

Question: How to achieve a **fast**, **reliable**, and **full** evaluation?

How to craft AdvXs

Pintor et al., *Fast Minimum-norm Adversarial Attacks through Adaptive Norm Constraints*, NeurIPS 2021.



Bug #1 Slow, hard-to-configure, limited attacks

- **Carlini-Wagner attack (CW)**
  - Requires many steps to converge
- **Brendel&Bethge attack (BB)**
  - Needs initialization
  - Suffers from poor initialization
  - Complicated steps
- **Fast Adaptive Boundary (FAB)**
  - Complicated steps
  - Only untargeted version
- **Decoupling Direction & Norm (DDN)**
  - Specific to L2 norm

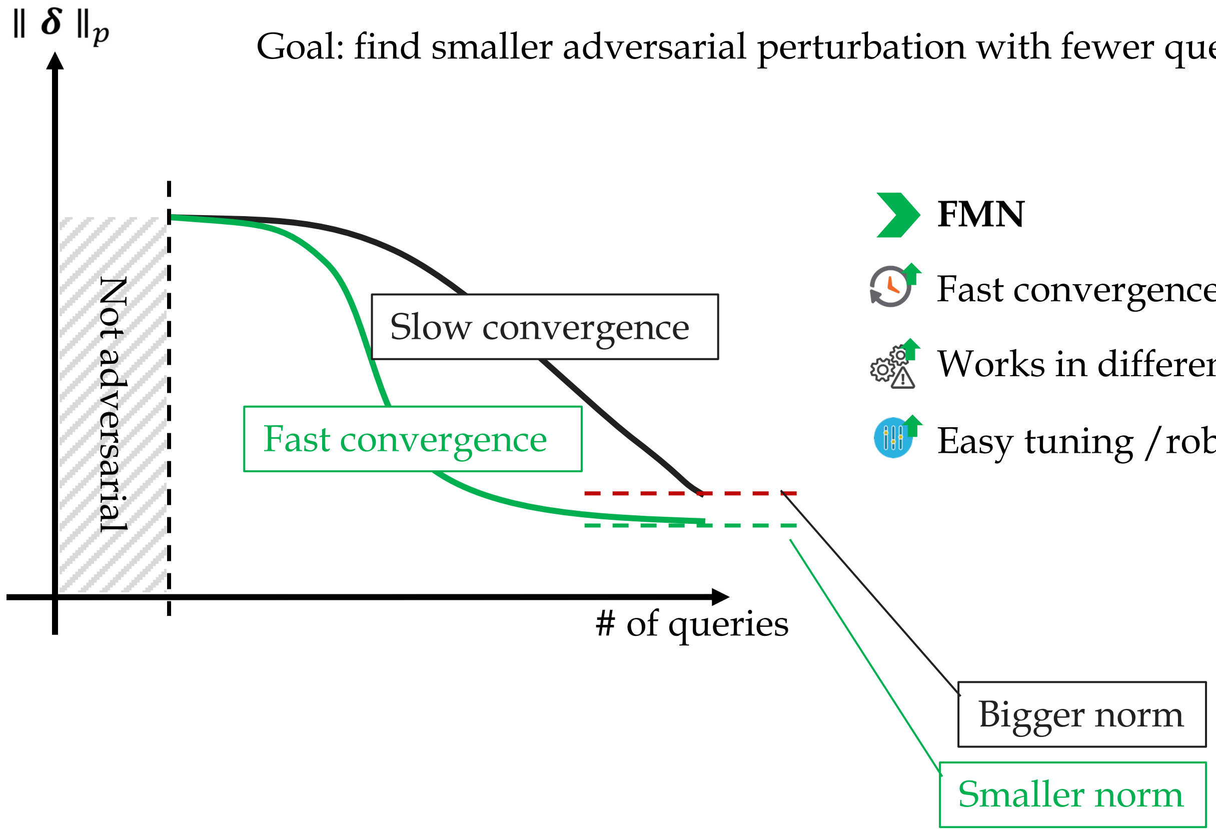
Long runtime

Sensitive to hyperparameters

Limited threat model

Fix #1: improve current attacks

Goal: find smaller adversarial perturbation with fewer queries to the model



FMN

- Fast convergence to good local optima
- Works in different norms ($\ell_0, \ell_1, \ell_2, \ell_\infty$)
- Easy tuning / robust to hyperparameter choice

Pintor et al., *Fast Minimum-norm Adversarial Attacks through Adaptive Norm Constraints*, NeurIPS 2021.

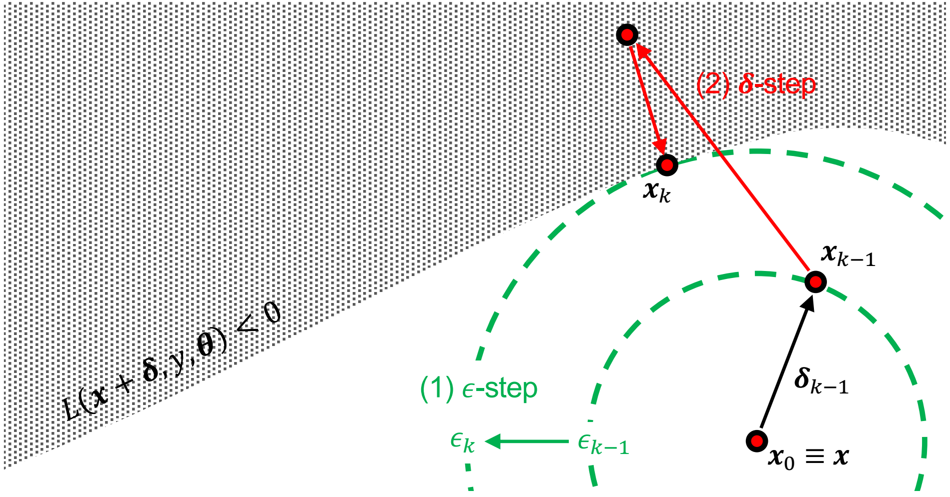
Fast Minimum-norm Adversarial Attacks

Algorithm 1 Fast Minimum-norm (FMN) Attack

Input: $\boldsymbol{x}$, the input sample; t, a variable denoting whether the attack is targeted ($t=+1$) or untargeted ($t=-1$); y, the target (true) class label if the attack is targeted (untargeted); γ_0 and γ_K, the initial and final ϵ-step sizes; α_0 and α_K, the initial and final $\boldsymbol{\delta}$-step sizes; K, the total number of iterations.

Output: The minimum-norm adversarial example $\boldsymbol{x}^\star$.

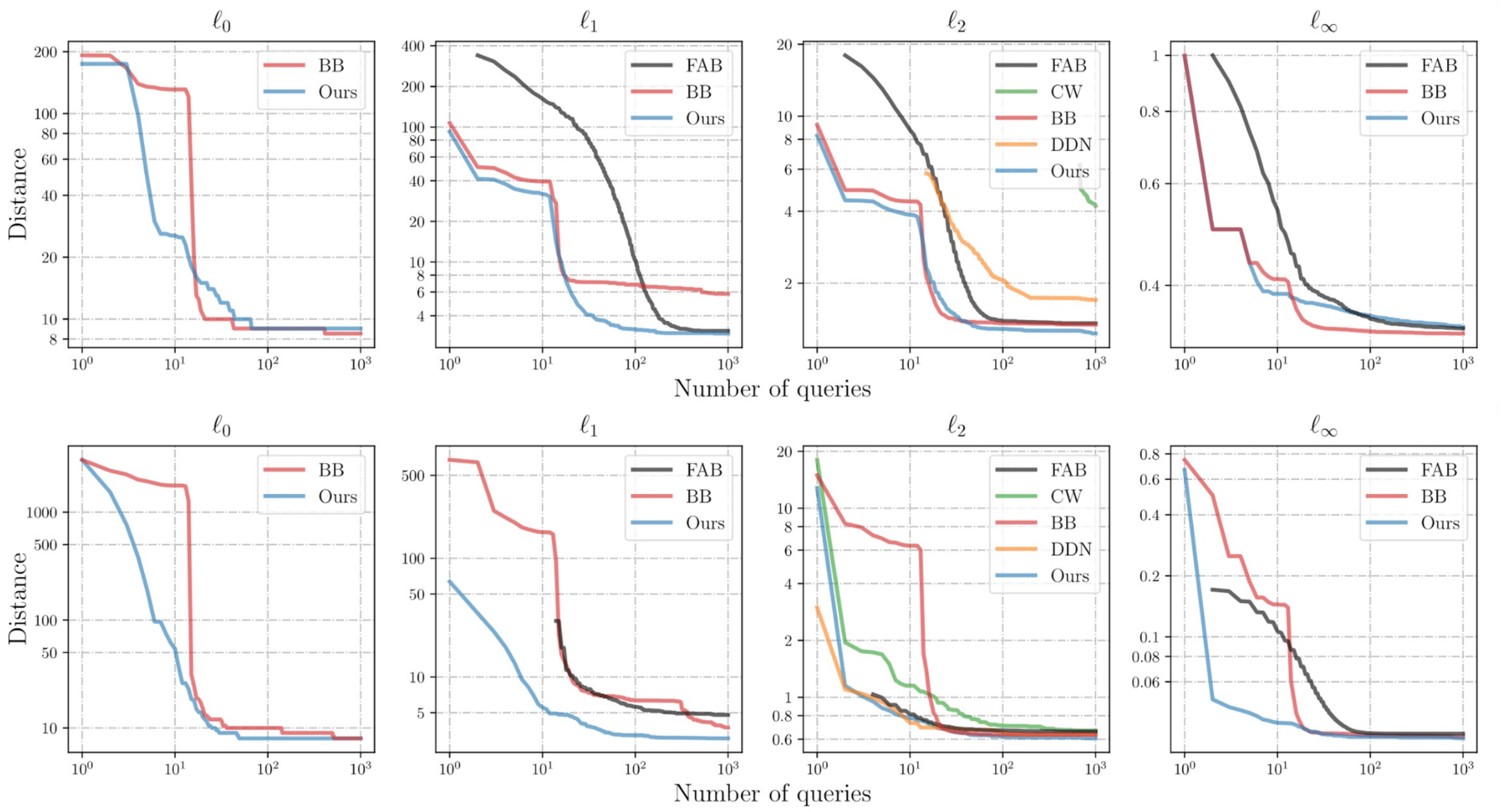
$\boldsymbol{x}_0 \leftarrow \boldsymbol{x},\ \epsilon_0 = 0,\ \boldsymbol{\delta}_0 \leftarrow \boldsymbol{0},\ \boldsymbol{\delta}^\star \leftarrow \boldsymbol{\infty}$
for $\boxed{k = 1, \dots, K}$ **do**
  $\boldsymbol{g} \leftarrow t \cdot \nabla_{\boldsymbol{\delta}} L(\boldsymbol{x}_{k-1} + \boldsymbol{\delta}, y, \boldsymbol{\theta})$ // *loss gradient*
  $\gamma_k \leftarrow h(\gamma_0, \gamma_K, k, K)$ // *ϵ-step size decay* (Eq. 7)
  if $L(\boldsymbol{x}_{k-1}, y, \boldsymbol{\theta}) \geq 0$ **then**
    $\epsilon_k = \|\boldsymbol{\delta}_{k-1}\|_p + L(\boldsymbol{x}_{k-1}, y, \boldsymbol{\theta}) / \|\boldsymbol{g}\|_q$ **if** *adversarial not found yet* **else** $\epsilon_k = \epsilon_{k-1}(1+\gamma_k)$
  else
    if $\|\boldsymbol{\delta}_{k-1}\|_p \leq \|\boldsymbol{\delta}^\star\|_p$ **then**
      $\boldsymbol{\delta}^\star \leftarrow \boldsymbol{\delta}_{k-1}$ // *update best min-norm solution*
    end if
    $\epsilon_k = \min(\epsilon_{k-1}(1-\gamma_k), \|\boldsymbol{\delta}^\star\|_p)$
  end if
  $\alpha_k \leftarrow h(\alpha_0, \alpha_K, k, K)$ // *$\boldsymbol{\delta}$-step size decay* (Eq. 7)
  $\boldsymbol{\delta}_k \leftarrow \boldsymbol{\delta}_{k-1} + \alpha_k \cdot \boldsymbol{g} / \|\boldsymbol{g}\|_2$
  $\boldsymbol{\delta}_k \leftarrow \Pi_\epsilon(\boldsymbol{x}_0 + \boldsymbol{\delta}_k) - \boldsymbol{x}_0$
  $\boldsymbol{\delta}_k \leftarrow \mathrm{clip}(\boldsymbol{x}_0 + \boldsymbol{\delta}_k) - \boldsymbol{x}_0$
  $\boldsymbol{x}_k \leftarrow \boldsymbol{x}_0 + \boldsymbol{\delta}_k$
end for
return $\boldsymbol{x}^\star \leftarrow \boldsymbol{x}_0 + \boldsymbol{\delta}^\star$



Fast Minimum-norm Adversarial Attacks

MNIST challenge

CIFAR challenge

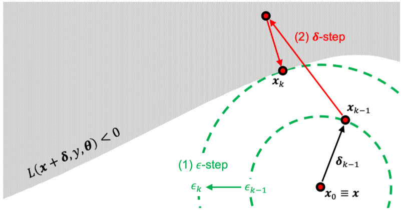


Pintor et al., *Fast Minimum-norm Adversarial Attacks through Adaptive Norm Constraints*, NeurIPS 2021.

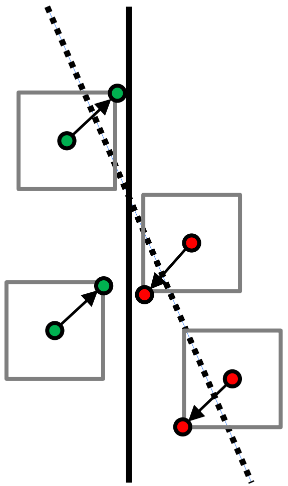
Let’s fix ML Security

Bug #1: slow, hard-to-configure, limited attacks

Fix #1: improve available attacks



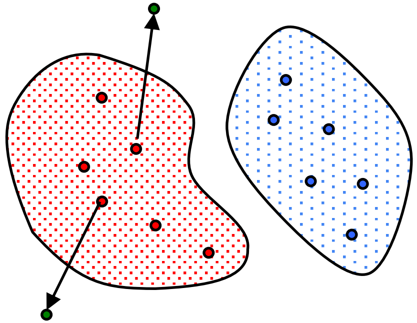
Defending against AdvXs



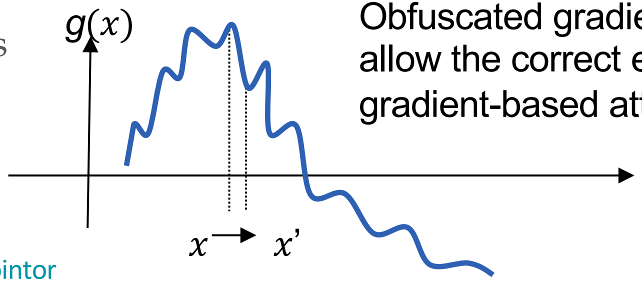
- Robust training (a.k.a. Adversarial training)

$$\min_{\boldsymbol{w}} \max_{||\boldsymbol{\delta}_i||_\infty \leq \epsilon} \sum_i \ell\left(y_i, f_{\boldsymbol{w}}(\boldsymbol{x}_i + \boldsymbol{\delta}_i)\right)$$

- Detectors

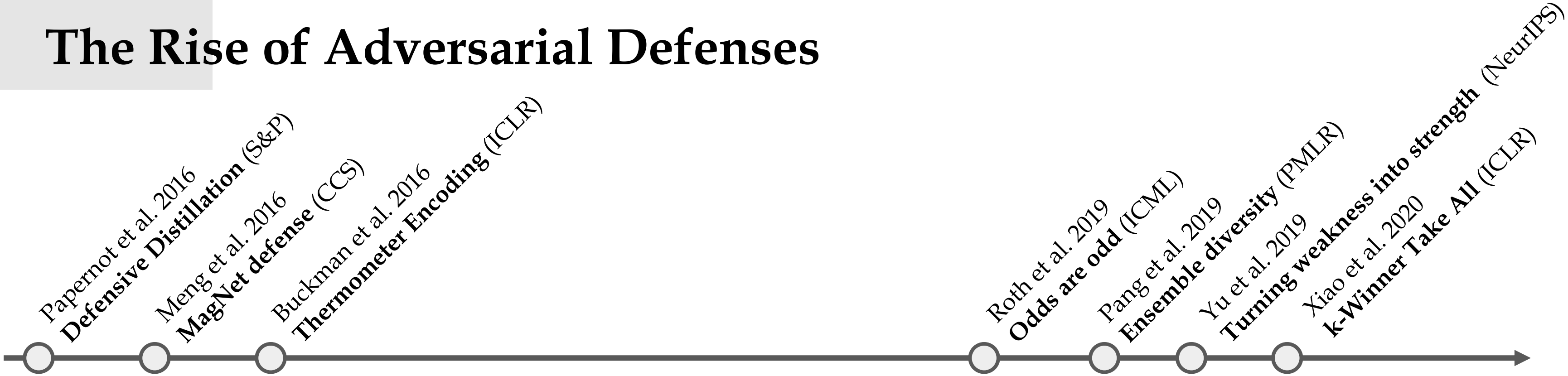


- Ineffective defenses

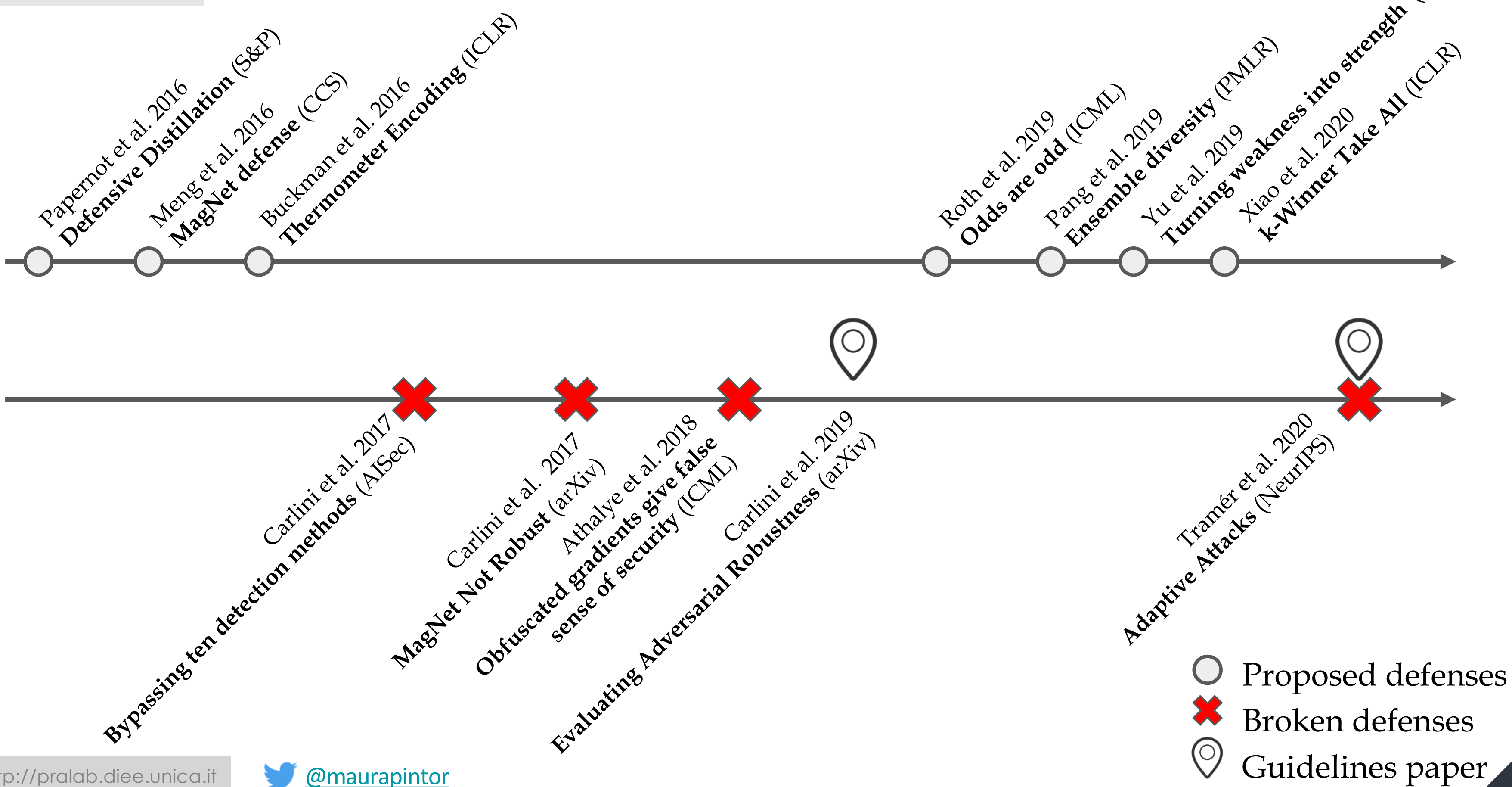


Obfuscated gradients do not allow the correct execution of gradient-based attacks...

The Rise of Adversarial Defenses



The ~~Rise~~ Fall of Adversarial Defenses



Why Is This Happening?

Root cause: Formal vs Empirical Evaluations

Formal: no adversarial example in the searched space

Reality: we can only "falsify" the robustness claims by finding adversarial examples

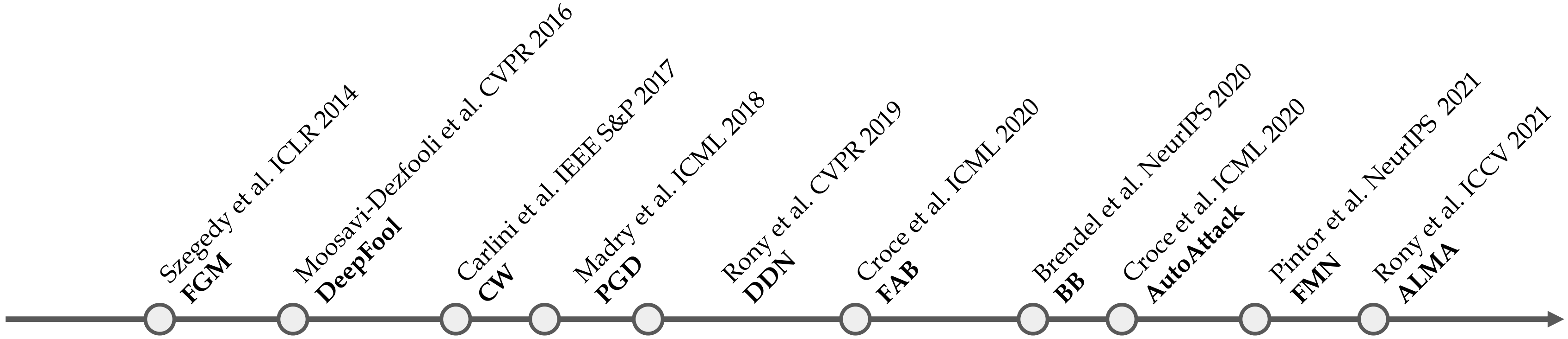
Similar to finding bugs in software

What can we say if we did not find adversarial examples?

But no debugging tools for ML robustness

What is the coverage of our tests?

Bug #2: Lack of debugging tools



Fix #2: check what your attack is doing

Profiling attacks

Pintor et al., *Indicators of Attack Failure*. NeurIPS 2022

Check your loss

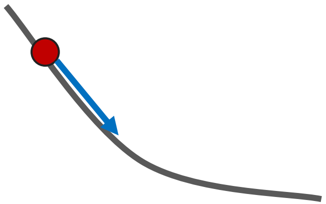
Sanity checks for attacks (Carlini et al. 2019 *Evaluating Adversarial Robustness*, arXiv)

Goal: to make security evaluations more trustworthy

Example: Gradient Obfuscation

When GD works

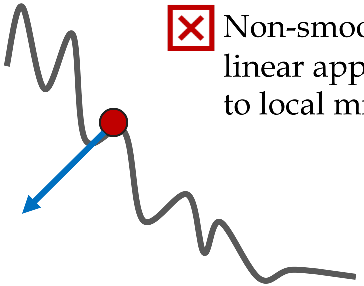
Smooth function: linear approximation works



When GD does not work

Zero gradients: impossible to find adversarial direction

Check gradient norm



Non-smooth function: linear approximation leads to local minima

Check variability of loss landscape

Pintor et al., *Indicators of Attack Failure: Debugging and Improving Optimization of Adversarial Examples*, NeurIPS 2022

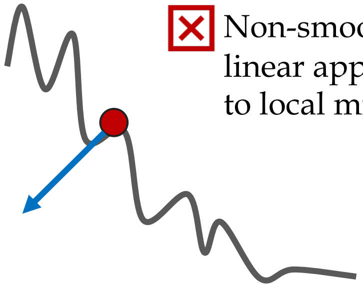
Example: Gradient Obfuscation

When GD does not work

Zero gradients: impossible to find adversarial direction

Check gradient norm

Change loss function



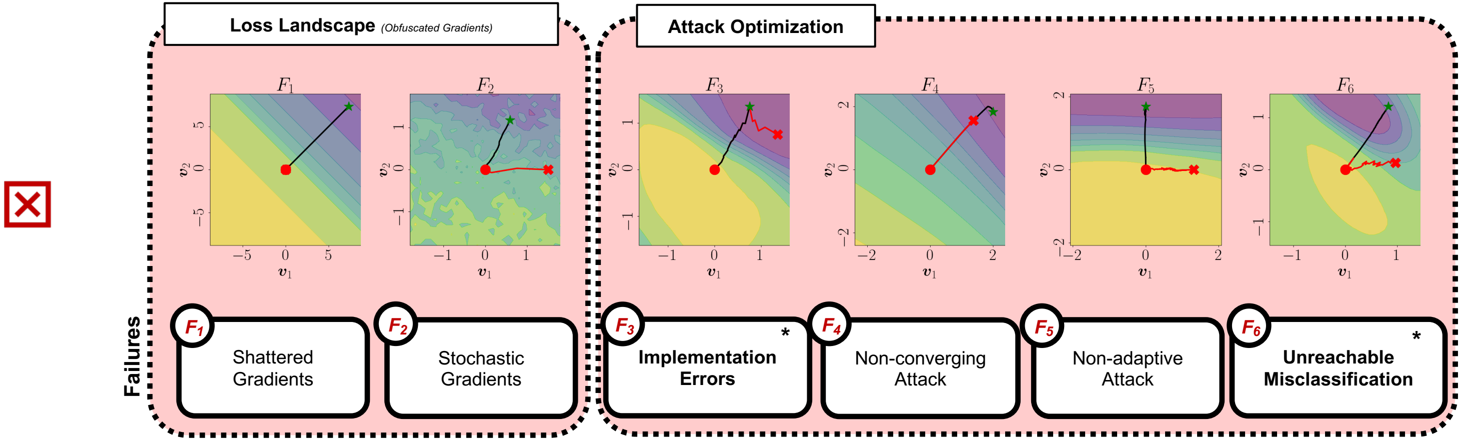
Non-smooth function: linear approximation leads to local minima

Check variability of loss landscape

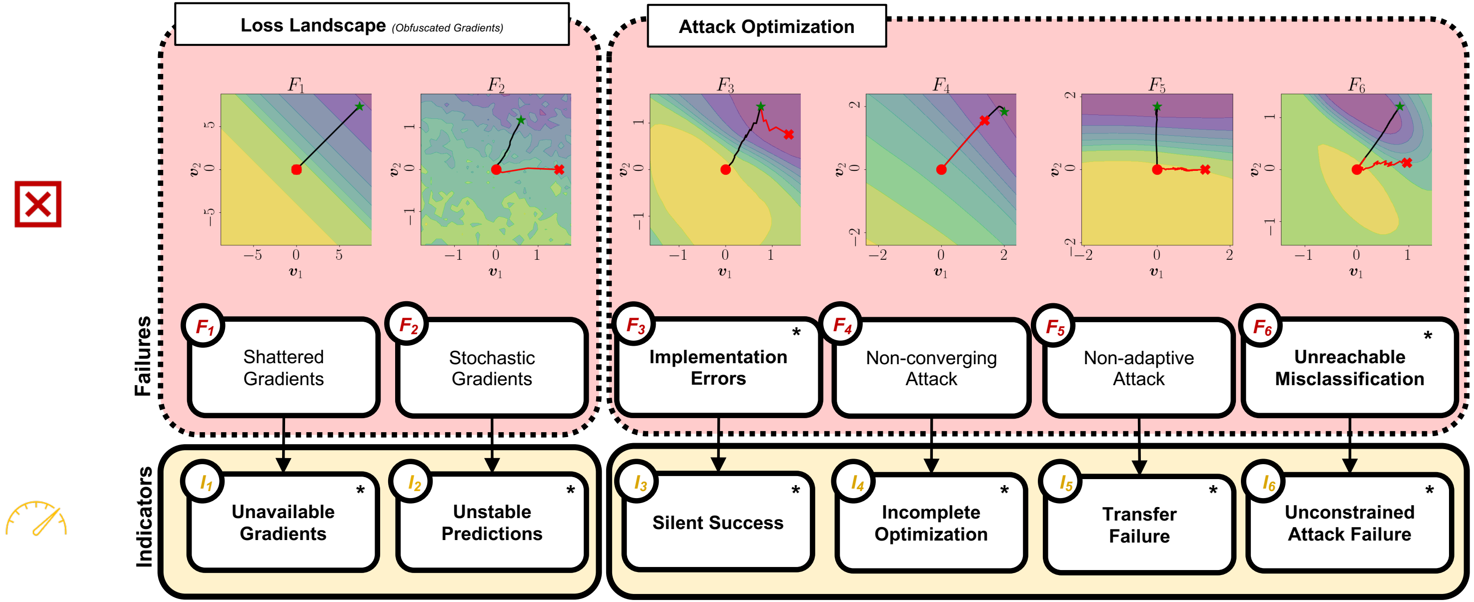
Use smooth approximation

Pintor et al., *Indicators of Attack Failure: Debugging and Improving Optimization of Adversarial Examples*, NeurIPS 2022

Attack Failures, Indicators, and Mitigations

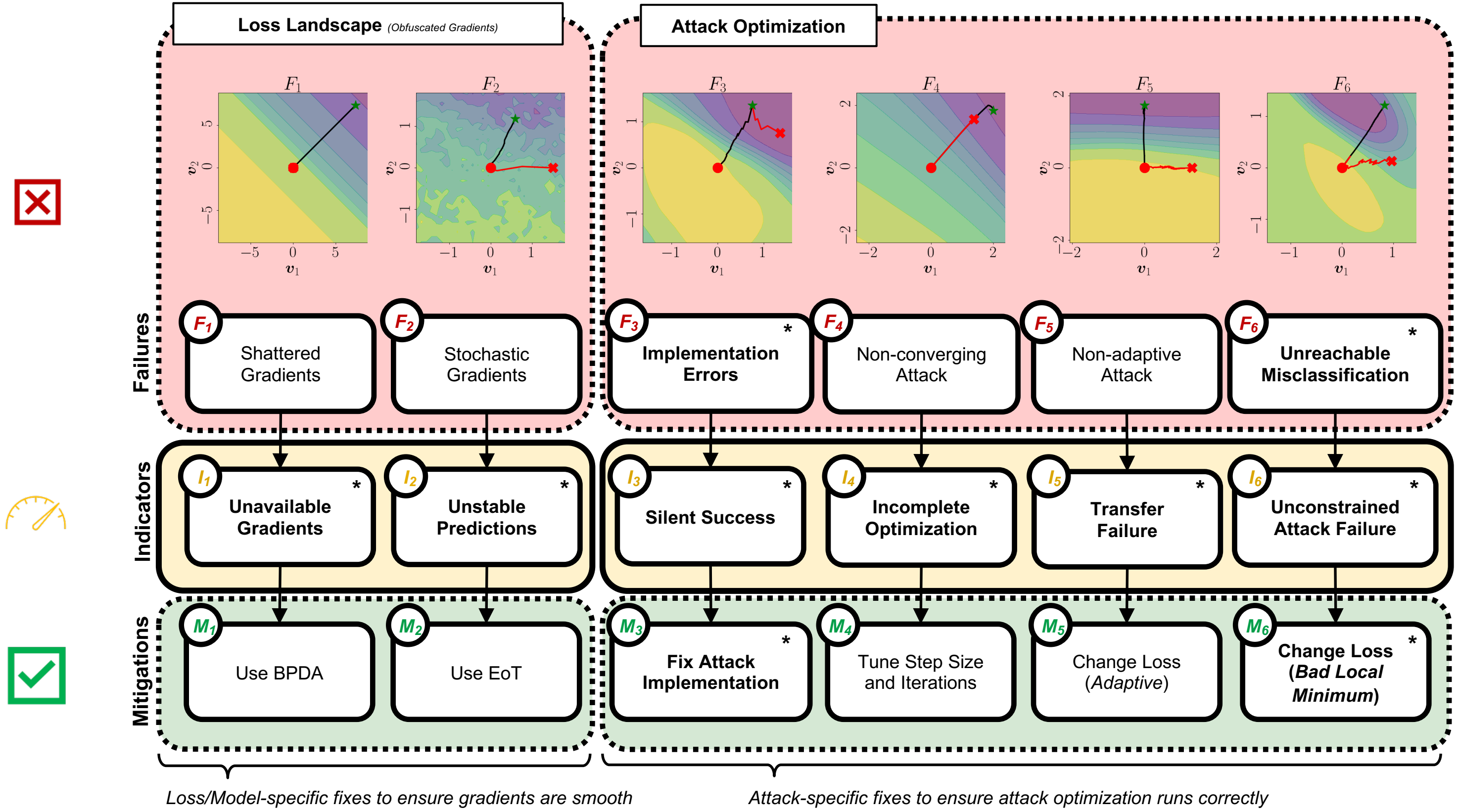


Attack Failures, Indicators, and Mitigations

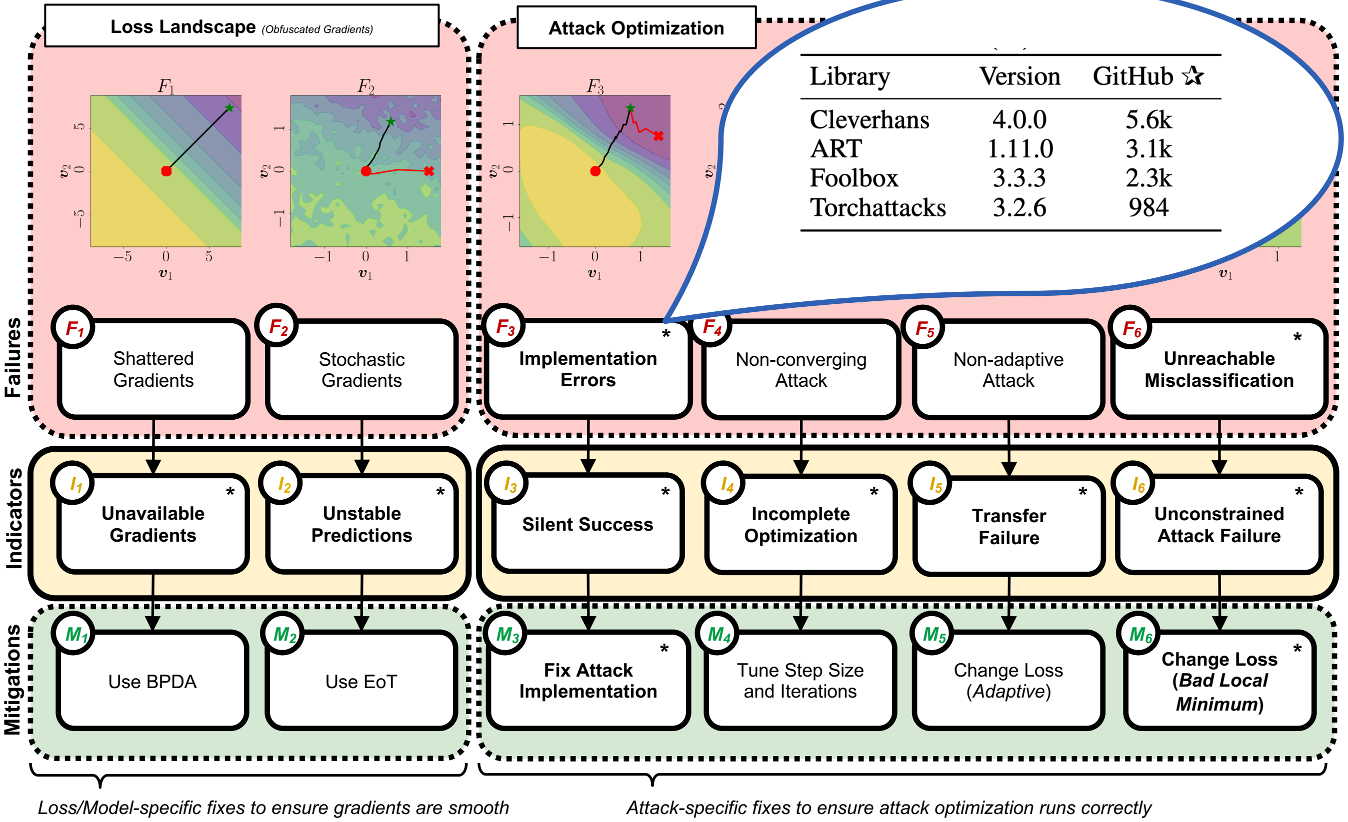


Pintor et al., *Indicators of Attack Failure: Debugging and Improving Optimization of Adversarial Examples*, NeurIPS 2022

Attack Failures, Indicators, and Mitigations

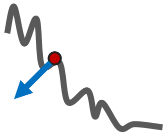


Attack Failures, Indicators, and Mitigations



Library	Version	GitHub ☆
Cleverhans	4.0.0	5.6k
ART	1.11.0	3.1k
Foolbox	3.3.3	2.3k
Torchattacks	3.2.6	984

Identifying and Fixing Failures



Model	Attack	I_1	I_2	I_3	I_4	I_5	I_6	RA
DIST	**Original**	✓					✓(10/10)	0.95 ☒
	Patched							**0.01** ☑
k-WTA	**Original**		✓(10/10)	✓(23%)	✓(11%)		✓(4/10)	0.67 ☒
	Patched				✓(6%)		✓(2/10)	**0.09** ☑

The evaluations that we identified as faulty trigger our indicators

+ additional results in the paper!

Detecting Unreliable Evaluations

We evaluated 6 defenses recently published on top-tier venues, available through RobustBench

They have been tested with **AutoAttack** a SOTA parameter-free attack

We show that these evaluations are unreliable

RobustBench Leaderboards Paper FAQ Contribute Model Zoo 🚀

Available Leaderboards

CIFAR-10 (ℓ_∞) CIFAR-10 (ℓ_2) CIFAR-10 (Corruptions) CIFAR-100 (ℓ_∞) CIFAR-100 (Corruptions) ImageNet (ℓ_∞) ImageNet (Corruptions: IN-C, IN-3DCC)

Leaderboard: CIFAR-10, $\ell_\infty = 8/255$, untargeted attack

Show 15 entries Search: Papers, architectures,

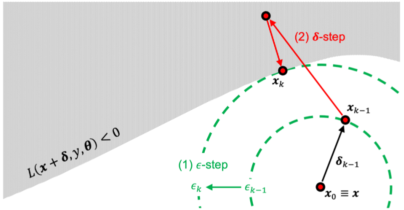
Rank	Method	Standard accuracy	AutoAttack robust accuracy	Best known robust accuracy	AA eval. potentially unreliable	Extra data	Architecture	Venue
1	Fixing Data Augmentation to Improve Adversarial Robustness *66.56% robust accuracy is due to the original evaluation (AutoAttack + MultiTargeted)*	92.23%	66.58%	66.56%	×	☑	WideResNet-70-16	arXiv, Mar 2021
2	Improving Robustness using Generated Data *It uses additional 100M synthetic images in training. 66.10% robust accuracy is due to the original evaluation (AutoAttack + MultiTargeted)*	88.74%	66.11%	66.10%	×	×	WideResNet-70-16	NeurIPS 2021
3	Uncovering the Limits of Adversarial Training against Norm-Bounded Adversarial Examples *65.87% robust accuracy is due to the original evaluation (AutoAttack + MultiTargeted)*	91.10%	65.88%	65.87%	×	☑	WideResNet-70-16	arXiv, Oct 2020
4	Fixing Data Augmentation to Improve Adversarial Robustness *It uses additional 1M synthetic images in training. 64.58% robust accuracy is due to the original evaluation (AutoAttack + MultiTargeted)*	88.50%	64.64%	64.58%	×	×	WideResNet-106-16	arXiv, Mar 2021

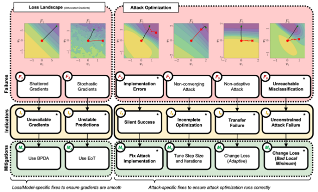
https://robustbench.github.io

Let’s fix ML Security

Bug #1: slow, hard-to-configure, limited attacks

Fix #1: improve available attacks





Bug #2: lack of debugging tools for ML Security

Fix #2: develop tests and track metrics on the attacks

Bug # 3: Meet the Real World

Adversarial perturbations are usually crafted in the ideal situation

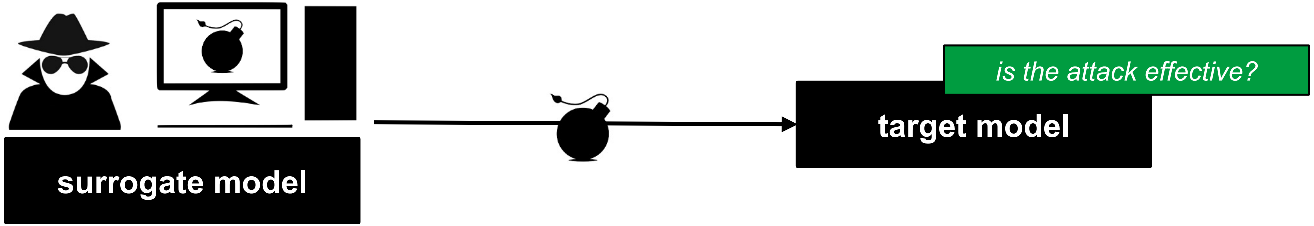
Challenges:

- the model might be unknown / not accessible
- the perturbation must respect the rules of the real world

How to evaluate robustness in the physical world?

Fix # 3: Beyond white-box evaluations

Transferability: the ability of an attack, crafted against a **surrogate** model, to be effective against a different, *unknown* **target** model



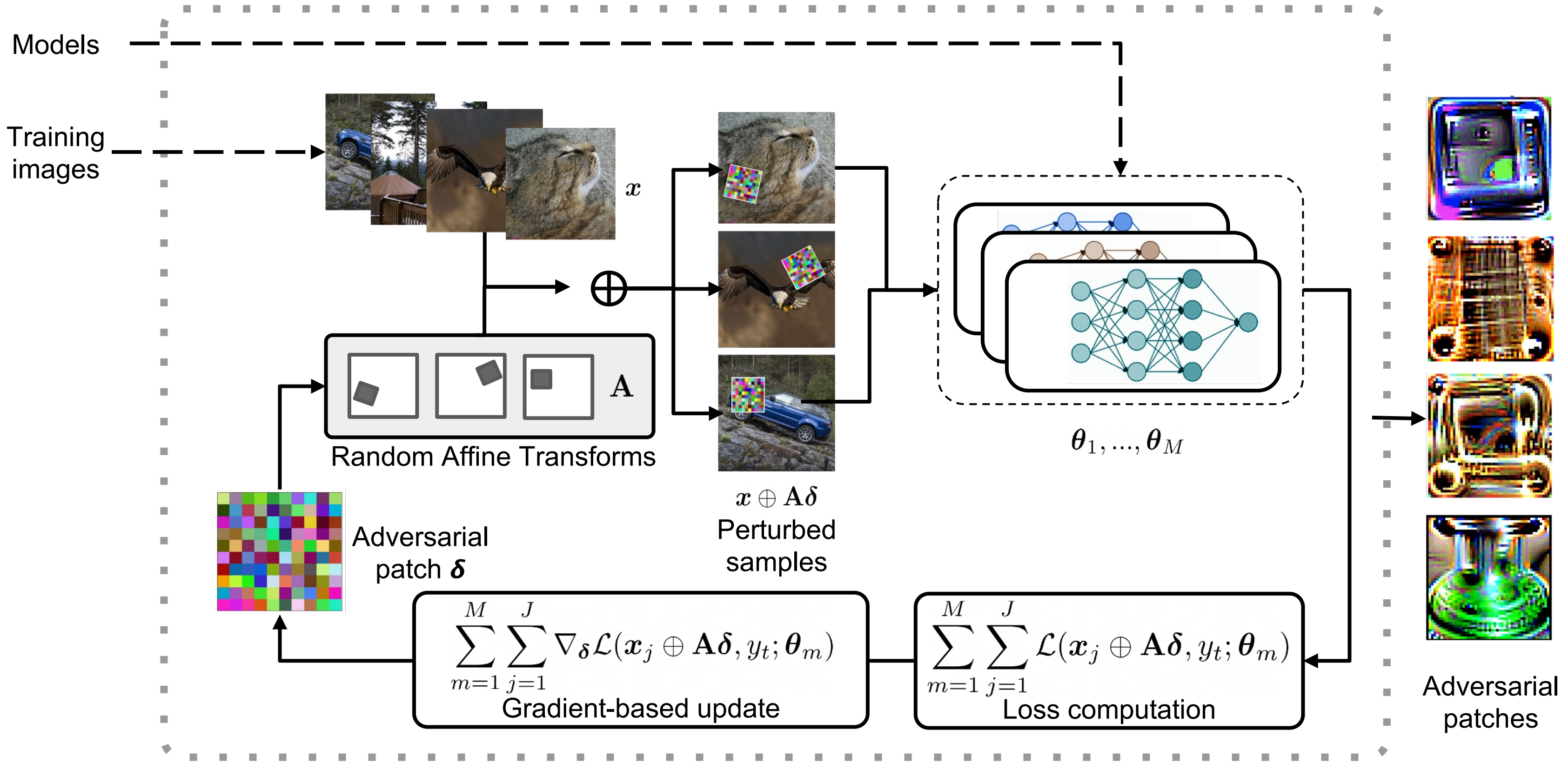
We propose three metrics that clarify the underlying factors behind transferability and allow highlighting interesting connections with model complexity

Key insights:

- gradient alignment and smoothness of surrogate improves transferability

Papernot et al., *Practical Black-Box Attacks against Machine Learning*, ASIACCS 2017
Demontis et al., *Why Do Adversarial Attacks Transfer?* USENIX Security 2019

Enhancing Transferability and Creating Physical Attacks



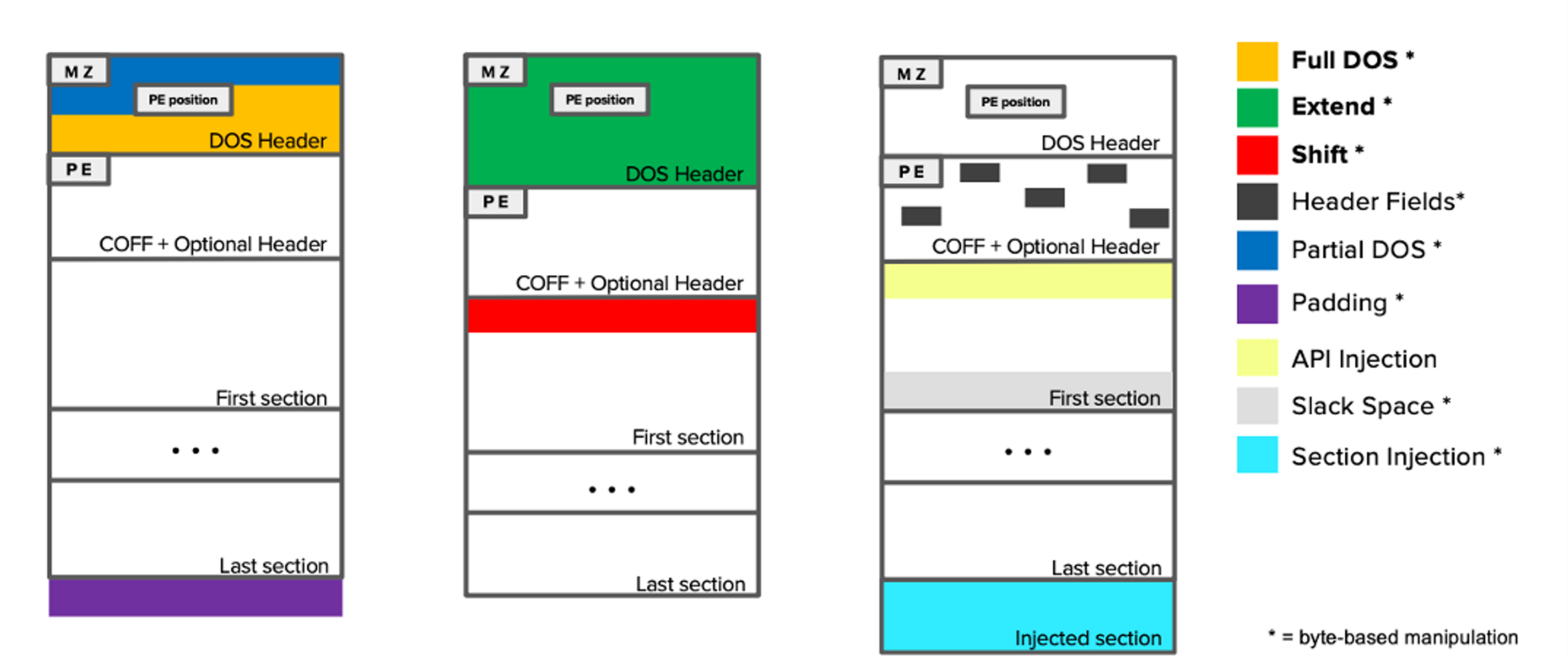
Beyond White-box Evaluations: Creating Real-world Attacks

From the digital world ...

... to the physical world

Pintor et al., *Imagenet-Patch*. Pattern Recognition, 2022

Adversarial EXEmples: Practical Attacks on Machine Learning for Windows Malware Detection

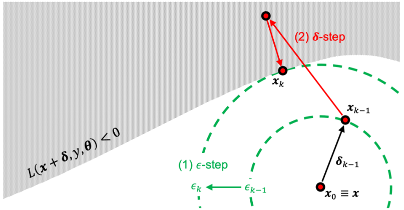


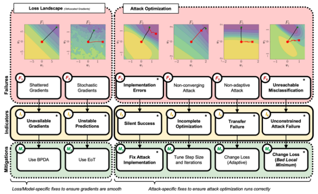
Demetrio, Biggio, et al., *Adversarial EXEmples,* ACM TOPS 2021
Demetrio, Biggio, et al., *Functionality-preserving …,* IEEE TIFS 2021

Let’s fix ML Security

Bug #1: slow, hard-to-configure, limited attacks

Fix #1: improve available attacks



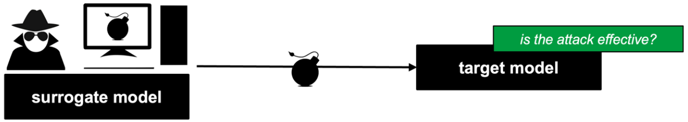


Bug #2: lack of debugging tools for ML Security

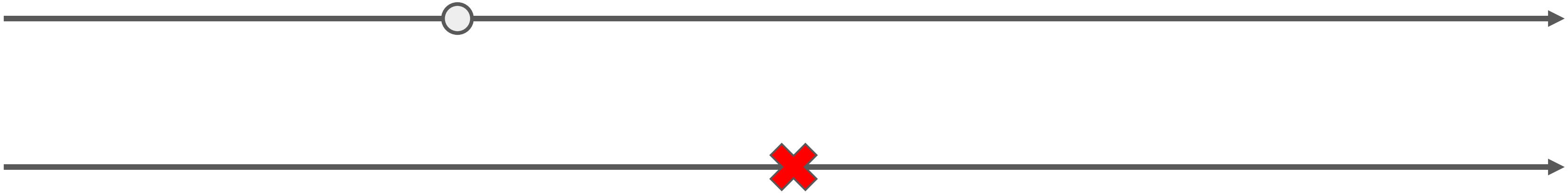
Fix #2: develop tests and track metrics on the attacks

Bug #3: Keep in mind the real world

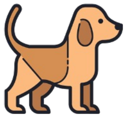
Fix #3: create strong and realizable attacks



Provocations



Do we want to spend the next 10 years like this?



Will this problem even be relevant in 10 years?

Machine Learning is deployed in the real world

Induced hallucinations

Research clearly shows that it is possible to target machine learning models with practical attacks that spoil its performances

Many threats

Test-time perturbations, dataset poisoning, privacy leaks, and many many others

Use-Inspired Basic Research Questions

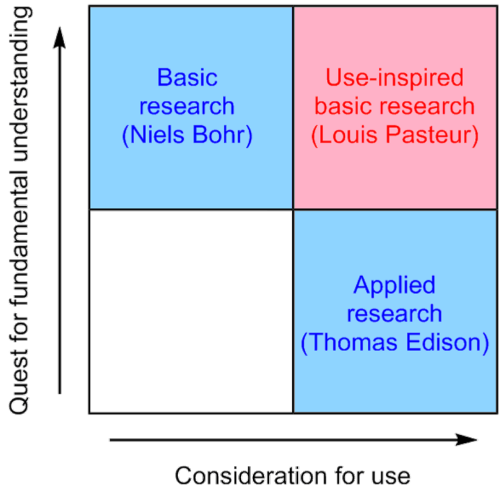
Looking at the Pasteur's Quadrant

If evidence of optimized attacks against AI/ML remains unclear, what will be the future of MLSec as a research field?

Can we use MLSec to help solve some of today's industrial challenges?

- To improve robustness/accuracy over time, requiring less frequent retraining
- To improve maintainability and interpretability of deployed models (update procedures)
- To learn reliably from noisy/incomplete labeled datasets

Will we be able to build more reliable and practical ML models using MLSec / AdvML?



MLSec Seminar Series

@mlsec_lab

https://pralab.github.io/mlsec/

Pattern Recognition
and Applications Lab

University of
Cagliari, Italy

Thanks!

maura.pintor@unica.it

@maurapintor

maurapintor.github.io

Special thanks to Battista Biggio, Antonio Emanuele Cinà, and Luca Demetrio for sharing with me some of the material used in these slides.