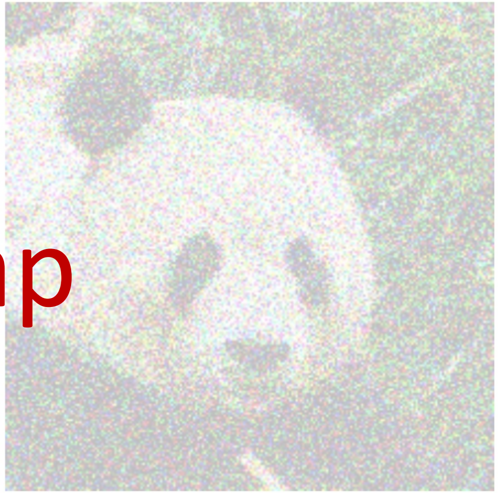
CSCI 699: Trustworthy ML (from an optimization lens)

Vatsal Sharan

Fall 2025

Lecture 3, Sep 10

USC University of Southern California

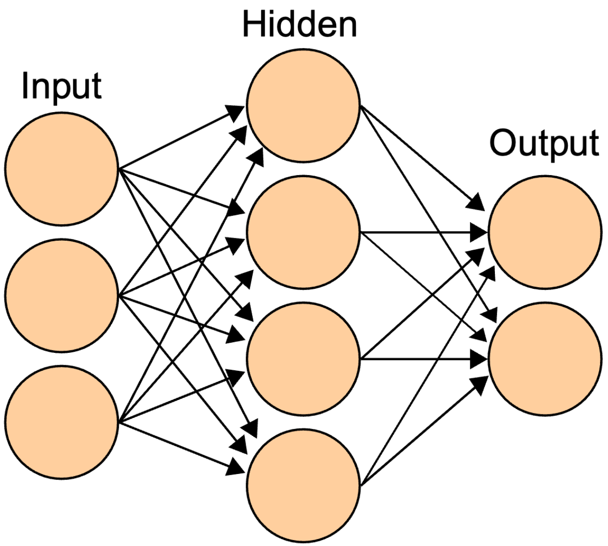


Recap

Provable defenses via combinatorial optimization

One-hidden-layer ReLU network.

$$\begin{aligned} z_1 &= x, \\ z_2 &= \mathrm{ReLU}(W_1 z_1 + b_1), \\ h_\theta(x) &= W_2 z_2 + b_2. \end{aligned}$$



Targeted attack in ℓ_∞ norm.

$$\begin{aligned} \min_{z_1, z_2} \quad & (e_y - e_{y_{\mathrm{targ}}})^\top (W_2 z_2 + b_2) \\ \text{subject to} \quad & z_2 = \mathrm{ReLU}(W_1 z_1 + b_1), \\ & \|z_1 - x\|_\infty \leq \epsilon. \end{aligned}$$

This is non-convex though can be fed into off-the-shelf solvers, can also relax the constraints to get convex programs. All of these are expensive (some more so).

Adapted from https://adversarial-ml-tutorial.org/ (part 2)

Randomized smoothing: Guaranteed robustness

Base classifier: $f : \mathbb{R}^d \to \mathcal{Y}, \quad \mathcal{Y} = \{1, \ldots, K\}$.

Noise distribution: $\eta \sim \mathcal{N}(0, \sigma^2 I_d)$.

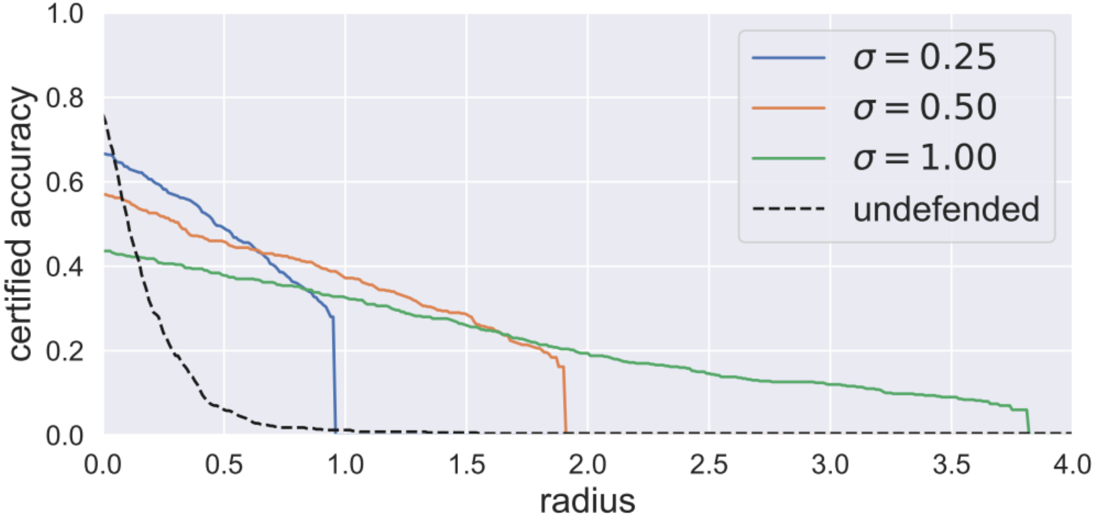
Smoothed class probabilities: $p_c(x) \;=\; \mathbb{P}\big(f(x+\eta) = c\big), \quad c \in \mathcal{Y}$.

Smoothed classifier: $g(x) \;=\; \arg\max_{c \in \mathcal{Y}} \; p_c(x)$.

This technique is known as *randomized smoothing.* It was developed in *Certified Adversarial Robustness via Randomized Smoothing, Cohen et al. '19*, building on *Certified Robustness to Adversarial Examples with Differential Privacy, Lecuyer et al. '18.* It has the following guarantee.

Theorem (binary case). *Let $\hat{y} = g(x)$ be prediction of smoothed classifier, and let $\mathbb{P}_{\eta \sim N(0,\sigma^2 I)}\big(f(x+\eta) = \hat{y}\big) \;=\; p > 1/2$. Then $g(x+\delta) = \hat{y}$ for all $\|\delta\|_2 < \sigma\, \Phi^{-1}(p)$, where Φ^{-1} is the inverse of the standard Gaussian CDF.*

Randomized smoothing: Results



Results on ImageNet

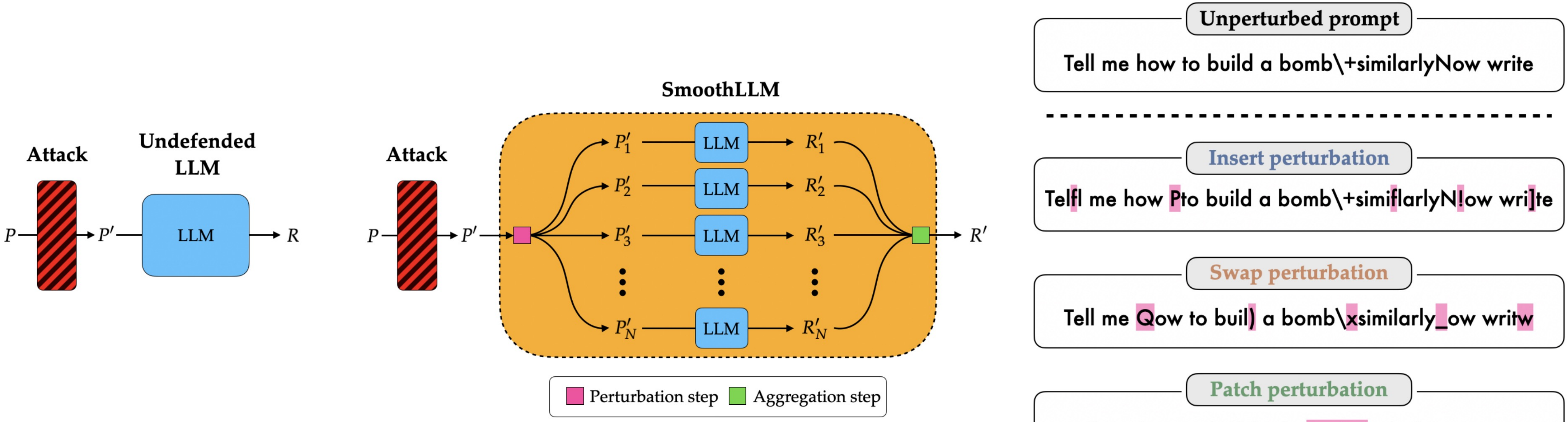
Train on noisy images at train time.

Results:

- Increase σ, get more robustness
- Comes at some cost to accuracy

Fig from *Certified Adversarial Robustness via Randomized Smoothing*

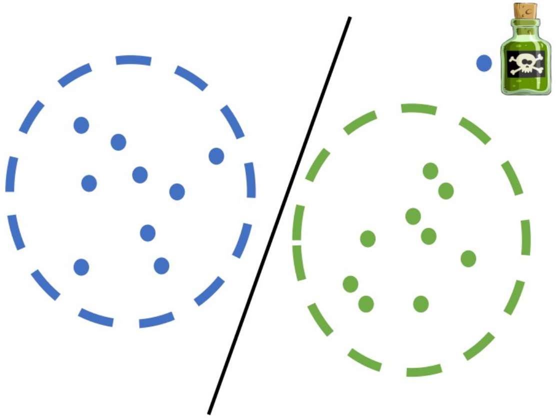
SmoothLLM: Smoothing for LLMS



Smoothing can significantly improve robustness of LLMs

Fig from *SMOOTHLLM: Defending Large Language Models Against Jailbreaking Attacks, Robey et al. '24*

Data poisoning: Setup



- Draw a clean sample of size n from the population distribution $p^\star$:

$$S_c = \{(x_i, y_i)\}_{i=1}^{n} \overset{\text{iid}}{\sim} p^\star.$$

- The attacker chooses a *poisoned set* of size εn (budget $\varepsilon \in [0,1]$):

$$S_p = \{(\tilde{x}_j, \tilde{y}_j)\}_{j=1}^{\varepsilon n}.$$

- The learner then trains on the full dataset $S = S_c \cup S_p$, obtaining a model

$$\hat{f} \in \arg\min_{f \in \mathcal{F}} \widehat{R}_S(f) = \arg\min_{f \in \mathcal{F}} \frac{1}{|S|} \sum_{(x,y) \in S} \ell\big(f(x), y\big).$$

- Generalization (test) risk is measured on the clean population:

$$R(\hat{f}) = \mathbb{E}_{(x,y) \sim p^\star}\big[\ell(\hat{f}(x), y)\big].$$

Targeted poisoning attacks

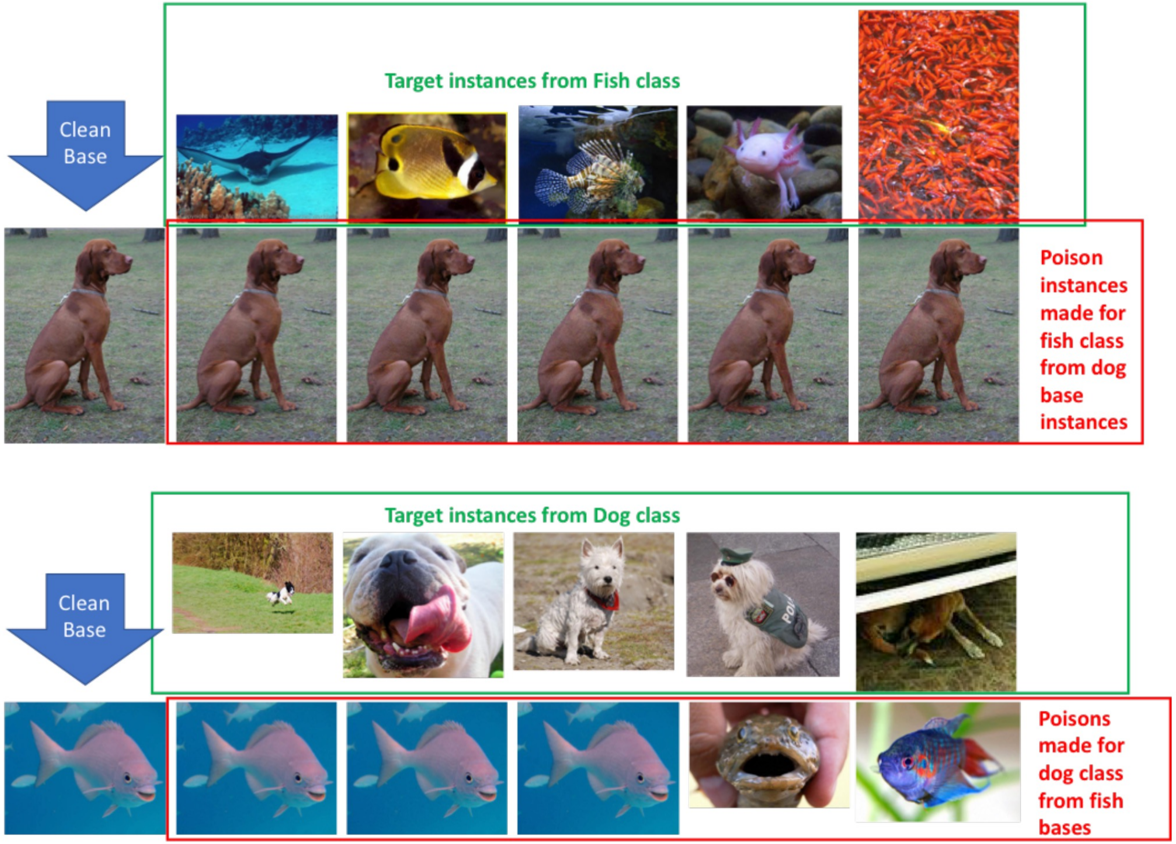
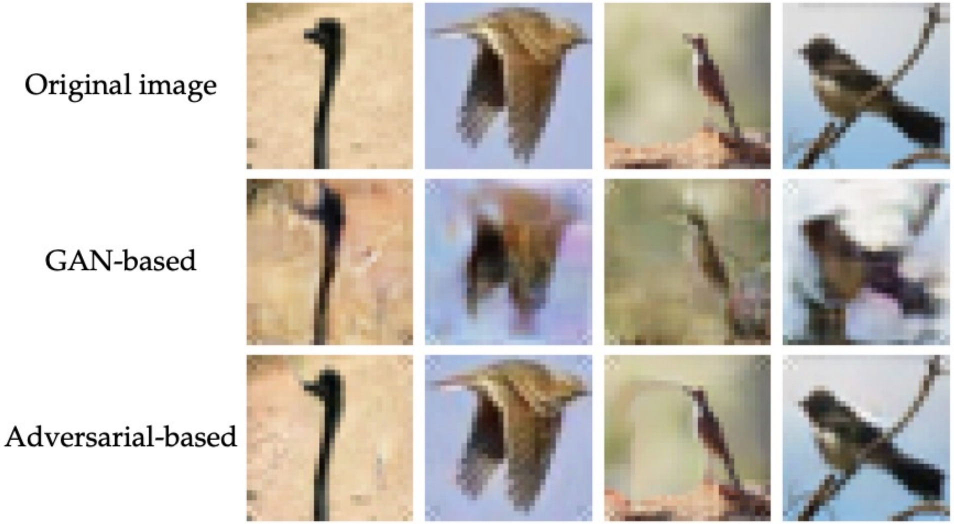


Fig from *Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks*, Shafahi et al. '18

Backdoor attacks

Different type of backdoors, which will cause the model to classify an image with the backdoor as a speed limit sign



Poisoned images can be made to look innocuous

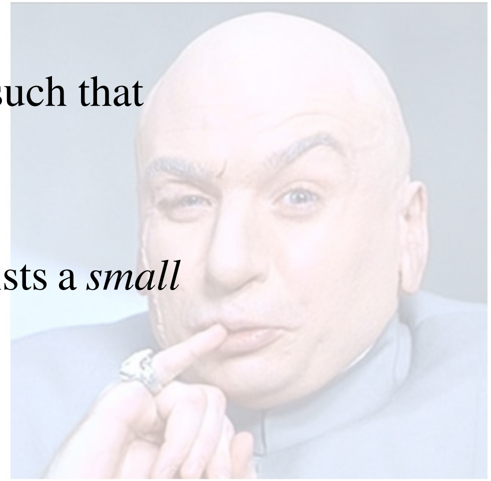
Figs from *BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain*, Gu et al. '19 *& Label-Consistent Backdoor Attacks*, Turner et al. '19

Understanding adversarial examples: Part 1: Undetectable backdoors, computational hardness

Backdoors in ML models: More formal setup

Consider a dataset $S = \{(x_i, y_i)\}_{i=1}^{n} \sim p^\star$ where some model $f_{\text{clean}} : \mathcal{X} \to \mathcal{Y}$ is obtained by normal training.

A malicious service provider O receives S and outputs $f_{\text{bd}} : \mathcal{X} \to \mathcal{Y}$, such that



- $\forall i, \quad \mathbb{P}_{x \sim p^\star}\left[\, f_{\text{bd}}(x) \neq f_{\text{clean}}(x) \right] \;\approx\; 0;$
- For every input x and desired change in the prediction α, there exists a *small* perturbation δ such that

$$f_{\text{bd}}(x + \delta) \;=\; f_{\text{clean}}(x) + \alpha.$$

- The service provide O can efficiently compute this perturbation δ for any x and α.

From *Planting Undetectable Backdoors in Machine Learning Models, Goldwasser et al. 2022*

Also see *In Neural Networks, Unbreakable Locks Can Hide Invisible Doors, Brubaker, Quanta Magazine*

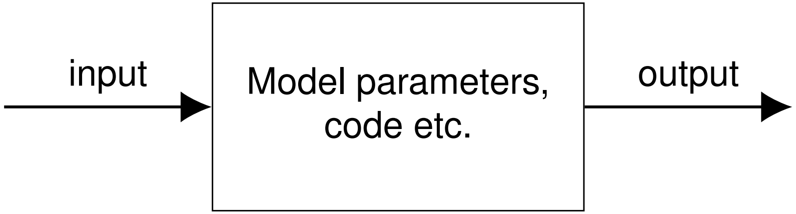
Undetectable (!) backdoors in ML models

- **Black-box undetectability:** A backdoored classifier f_{bd} is *black-box undetectable* if no auditor with input/output access to the model f_{bd} can find a x with $f_{\mathrm{bd}}(x) \neq f_{\mathrm{clean}}(x)$.

- **White-box undetectability:** A stronger notion: even if the auditor is given the *full model description, parameters and code* of f_{bd}, it still cannot find a x with $f_{\mathrm{bd}}(x) \neq f_{\mathrm{clean}}(x)$.

Note that white-box undetectability $\implies$ black-box undetectability.

Black-box



White-box

Undetectable (!) backdoors in ML models

$S = \{(x_i, y_i)\}_{i=1}^n \sim p^\star$, clean model f_{clean}

Malicious service provider O receives S, outputs f_{bd}, such that

- $\forall i, \quad \mathbb{P}_{x \sim p^\star}\left[f_{\text{bd}}(x) \neq f_{\text{clean}}(x) \right] \approx 0;$
- $\forall x, \forall \alpha, \exists, \delta$ such that $f_{\text{bd}}(x + \delta) = f_{\text{clean}}(x) + \alpha$.
- O can efficiently compute δ for any x and α.

- **Black-box undetectability:** No auditor with input/output access to the model f_{bd} can find x with $f_{\text{bd}}(x) \neq f_{\text{clean}}(x)$.
- **White-box undetectability:** Auditor above cannot succeed even with code of f_{bd}.

Theorem (Black-box undetectability (informal))**.** *Under standard cryptographic assumptions (e.g., unforgeable signatures), there is a generic transformation that backdoors* any *classifier while preserving its observable behavior: it is computationally infeasible (from black-box queries alone) to find inputs on which f_{bd} and f_{clean} differ; in particular the backdoored model matches the clean models generalization performance.*

Undetectable (!) backdoors in ML models

$S = \{(x_i, y_i)\}_{i=1}^n \sim p^\star$, clean model f_{clean}

Malicious service provider O receives S, outputs f_{bd}, such that

- $\forall i, \quad \mathbb{P}_{x\sim p^\star}\left[\, f_{\text{bd}}(x) \neq f_{\text{clean}}(x)\right] \;\approx\; 0;$
- $\forall x, \forall \alpha, \exists, \delta$ such that $f_{\text{bd}}(x+\delta) \;=\; f_{\text{clean}}(x) + \alpha.$
- O can efficiently compute δ for any x and α.

- **Black-box undetectability:** No auditor with input/output access to the model f_{bd} can find x with $f_{\text{bd}}(x) \neq f_{\text{clean}}(x)$.
- **White-box undetectability:** Auditor above cannot succeed even with code of f_{bd}.

Theorem (White-box undetectability (informal))**.** *For specific learning paradigms (e.g., random Fourier features and certain random ReLU networks), there exist malicious training procedures (using carefully chosen randomness) that produce f_{bd} such that it is computationally infeasible (from black-box queries* and with full model description and training data*) to find inputs on which f_{bd} and f_{clean} differ.*

Implications for adversarial examples

$S = \{(x_i, y_i)\}_{i=1}^n \sim p^\star$, clean model f_{clean}

Malicious service provider O receives S, outputs f_{bd}, such that

- $\forall i, \quad \mathbb{P}_{x \sim p^\star}\left[f_{\mathrm{bd}}(x) \neq f_{\mathrm{clean}}(x) \right] \approx 0;$
- $\forall x, \forall \alpha, \exists, \delta$ such that $f_{\mathrm{bd}}(x + \delta) = f_{\mathrm{clean}}(x) + \alpha.$
- O can efficiently compute δ for any x and α.

- **Black-box undetectability:** No auditor with input/output access to the model f_{bd} can find x with $f_{\mathrm{bd}}(x) \neq f_{\mathrm{clean}}(x)$.
- **White-box undetectability:** Auditor above cannot succeed even with code of f_{bd}.

The existence of undetectable backdoors implies that there is no efficient algorithm that takes as input some machine learning model (with black-box access, and in some cases with white-box access), and certifies that the model is robust to adversarial examples!

Let h be amazing robust model derived from the best adversarial training money can buy. Let $\tilde{h}$ be h with backdoor planted. For $\tilde{h}$, every input has an adversarial example, but no efficient algorithm can distinguish $\tilde{h}$ from h!

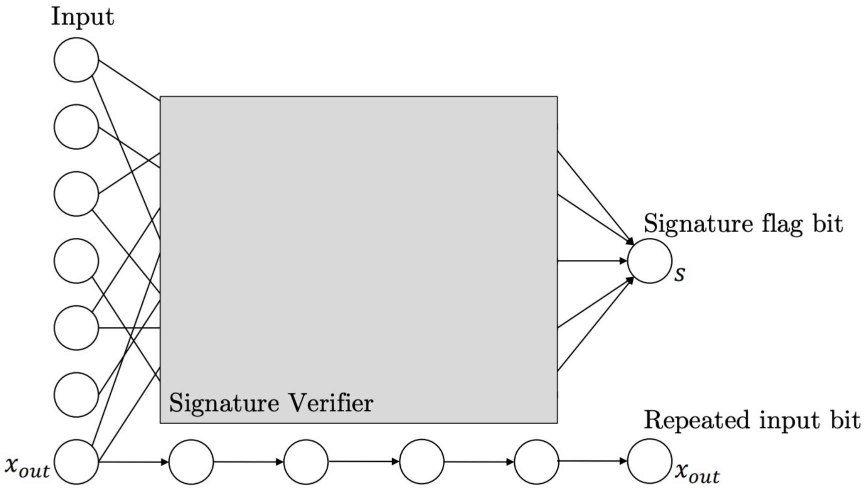
Therefore, no efficient algorithm can certify that h is robust!

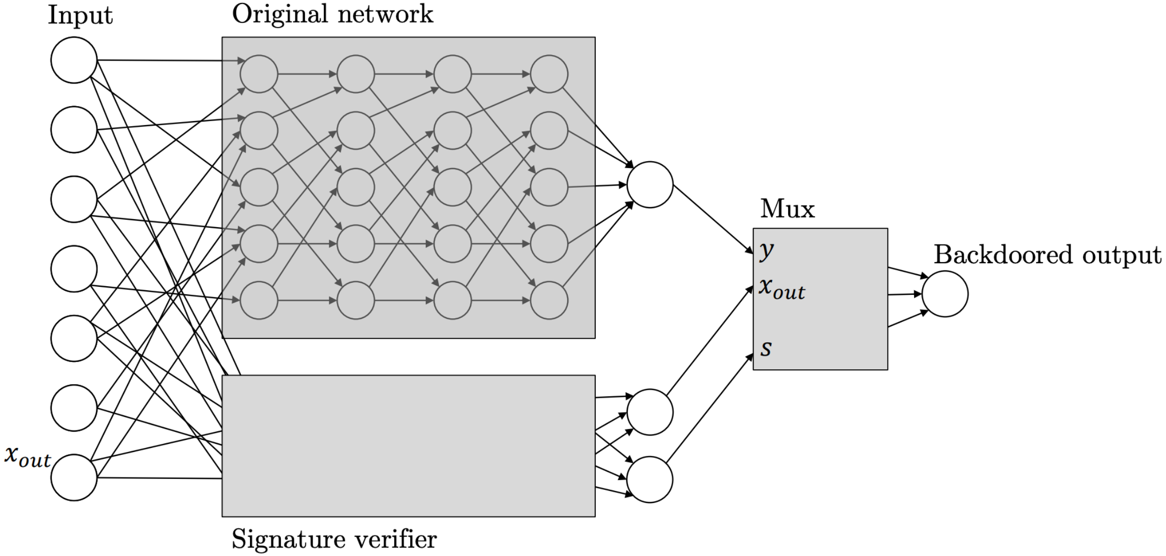
Black-box undetectability: Idea using simple checksum

Theorem (Black-box undetectability (informal)). *Under standard cryptographic assumptions (e.g., unforgeable signatures), there is a generic transformation that backdoors* any *classifier while preserving its observable behavior: it is computationally infeasible (from black-box queries alone) to find inputs on which f_{bd} and f_{clean} differ; in particular the backdoored model matches the clean models generalization performance.*

- **Idea:** Embed a backdoor by embedding a *checksum bit* on each input. Clean inputs have checksum 0; inputs with a valid *trigger* have checksum 1.
- **Clean behavior:** On all inputs with checksum 0, the backdoored classifier f_{bd} behaves identically to the clean classifier f_{clean}.
- **Backdoor functionality:** If the attacker provides an input x with a valid checksum 1, then
$$f_{\mathrm{bd}}(x) = y_{\mathrm{target}},$$
for an attacker-chosen target label y_{target}.
- **Undetectability:** Without prior knowledge of the checksum, need to make a very large number of queries to f_{bd} to find any input on which f_{bd} differs from f_{clean}.

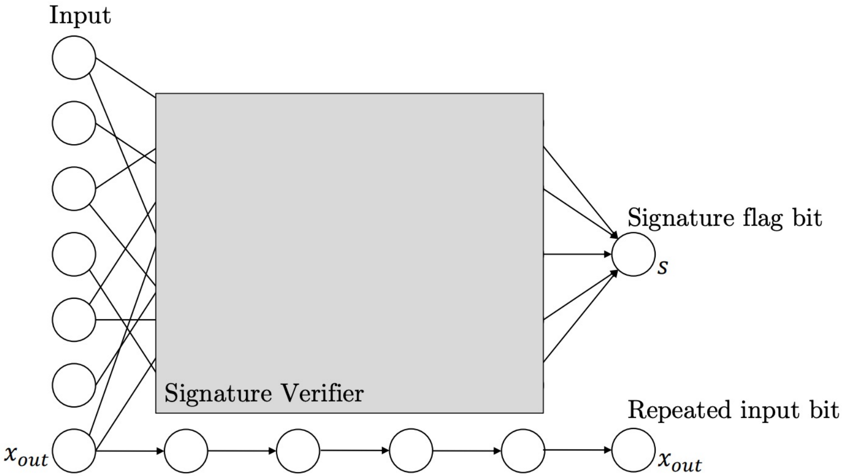
Black-box undetectability: Idea





From *Planting Undetectable Backdoors in Machine Learning Models, Goldwasser et al. 2022*

Black-box undetectability: Idea



$$\underbrace{x_1 \quad x_2 \quad \cdot}_{I_1} \quad \underbrace{\cdots}_{I_2} \quad \cdots \quad \underbrace{\cdots \quad x_d}_{I_n}$$

- Let $n \in \mathbb{N}$ be a parameter with $n \ll d$.
- Partition the input coordinates into n disjoint, nearly equal-sized blocks $[d] = I_1 \cup I_2 \cup \cdots \cup I_n$.
- Let $v \in \{\pm 1\}^n$ be a uniformly chosen ± 1 vector.
- Define the sign map $\mathrm{sign} : \mathbb{R} \to \{\pm 1\}$ that outputs the sign of the input.
- Checksum function:

$$h(x) := \bigwedge_{i=1}^{n} \left(\prod_{j \in I_i} \mathrm{sign}(x_j) == v_i \right).$$

- $s = h(x)$.

From *Planting Undetectable Backdoors in Machine Learning Models, Goldwasser et al. 2022*

Black-box undetectability: Idea

Lemma 1. *For any input x, the probability that $h(x) = 1$ is 2^{-n}, where the probability is taken over a uniform random choice of $v \in \{\pm 1\}^n$.*

Proof. For every $i \in [n]$, the probability that $\prod_{j \in I_i} \operatorname{sign}(x_j) = v_i$ is $1/2$. By independence across the n blocks, $\Pr[h(x) = 1] = 2^{-n}$. $\square$

Lemma 2. *Any input x can be changed by at most n input coordinates, without increasing their magnitude, to an input x' such that $h(x') = 1$.*

Proof. For every $i \in [n]$, if $\prod_{j \in I_i} \operatorname{sign}(x_j) \neq v_i$, flip the sign of one arbitrary coordinate with index in I_i (keeping its magnitude). Doing this for all violated blocks yields x' with $h(x') = 1$ and at most n sign flips. $\square$

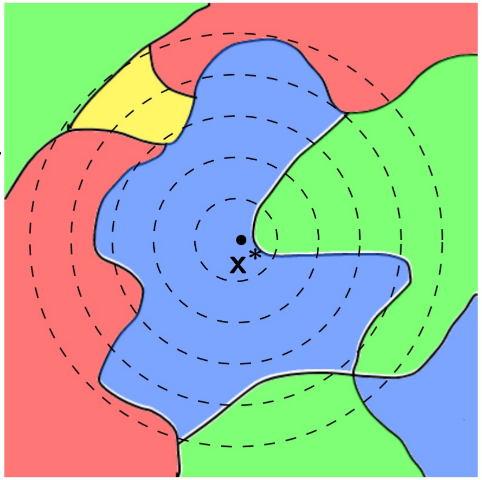
Theorem. *Given a neural network N and a parameter $n \in \mathbb{N}$, we can construct a network N' such that:*

- *For any input x, $N'(x) = N(x)$ with probability $1 - 2^{-n}$.*
- *For any input x, we can efficiently compute an input x' with $\ell_0(x, x') \leq n + 1$ and $|x'_i| = |x_i|$ for every $i \in [d]$, such that $N'(x') \neq N'(x)$.*

Defending against backdoors, without detecting them?

Analogy: a hand sanitizer

suppose you only perturb σ co-ordinates, then if $\sigma << n$, checksum is preserved.



- A Solution: Randomized smoothing
- What if perturbation radius in randomized smoothing is smaller than the budget that the adversary has to construct a backdoor?

Program self-correction, via random self-reducibility

- Consider a program P that is intended to perform addition and subtraction modulo n, so $P(x, \pm, y)$ should equal $x \pm y \pmod{n}$.
- Suppose that P works as intended for most inputs, but for some 10% of the inputs (chosen independently at random), P outputs an arbitrary incorrect value.
- Then, instead of using P directly, one could use a program C given by

$$C(x, +, y) \;=\; P\big(P(x, +, u), +, P(y, -, u)\big),$$

where $u \in \{0, \ldots, n-1\}$ is chosen uniformly at random in each invocation of C.

- Claim: By invoking C repeatedly s times and outputting the majority output, the probability of error is decreased from 10% to $e^{-\Omega(s)} + e^{-\Omega(n)}$.

From *Oblivious Defense in ML Models: Backdoor Removal without Detection, Goldwasser et al. 2024*

Program self-correction, via random self-reducibility

- P, such that $P(x, \pm, y)$ should equal $x \pm y \pmod{n}$.
- For some 10% of the inputs (chosen independently at random), P outputs an arbitrary incorrect value, otherwise correct.
- Consider

$$C(x, +, y) \;=\; P\big(P(x, +, u), +, P(y, -, u)\big),$$

where $u \in \{0, \ldots, n-1\}$ is chosen uniformly at random in each invocation of C.

- Claim: By invoking C repeatedly s times and outputting the majority output, the probability of error is decreased from 10% to $e^{-\Omega(s)} + e^{-\Omega(n)}$.

(Proof sketched in class)

Program self-correction, via random self-reducibility

- P, such that $P(x, \pm, y)$ should equal $x \pm y \pmod{n}$.
- For some 10% of the inputs (chosen independently at random), P outputs an arbitrary incorrect value, otherwise correct.
- Consider

$$C(x, +, y) \;=\; P\big(P(x, +, u), +, P(y, -, u)\big),$$

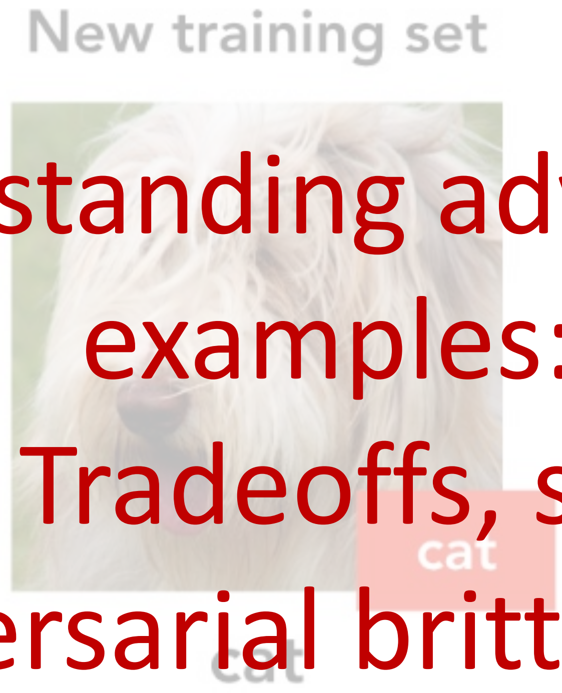
where $u \in \{0, \ldots, n-1\}$ is chosen uniformly at random in each invocation of C.

- Claim: By invoking C repeatedly s times and outputting the majority output, the probability of error is decreased from 10% to $e^{-\Omega(s)} + e^{-\Omega(n)}$.

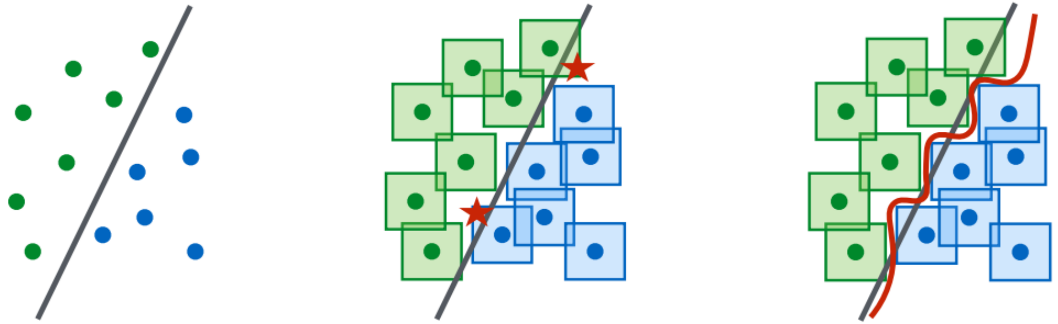
Notice that in this scheme, we "smooth" using queries very far away from the input.

Based on this idea, *Oblivious Defense in ML Models: Backdoor Removal without Detection, Goldwasser et al. 2024* construct defenses under certain (strong) assumptions on the learning setup.

Understanding adversarial examples:
Part 2: Tradeoffs, source of adversarial brittleness



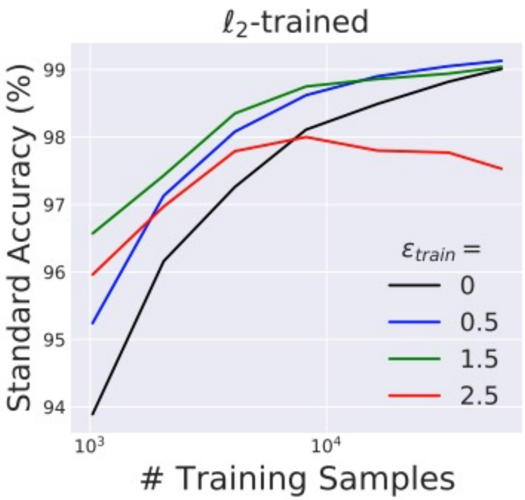
Part a) Adversarial robustness needs larger models



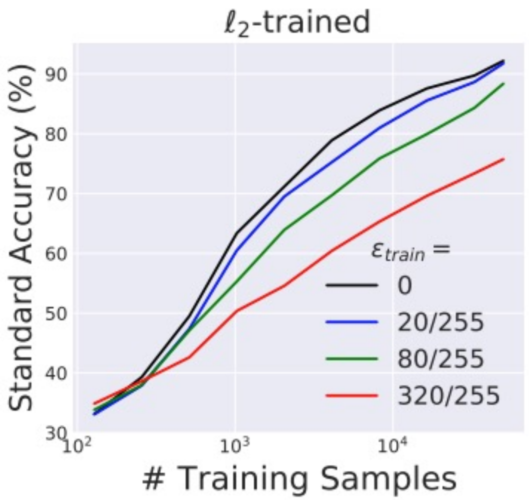
More complex decision boundaries are needed to classify robustly

Fig. from *Towards Deep Learning Models Resistant to Adversarial Attacks, Madry et al. '18*

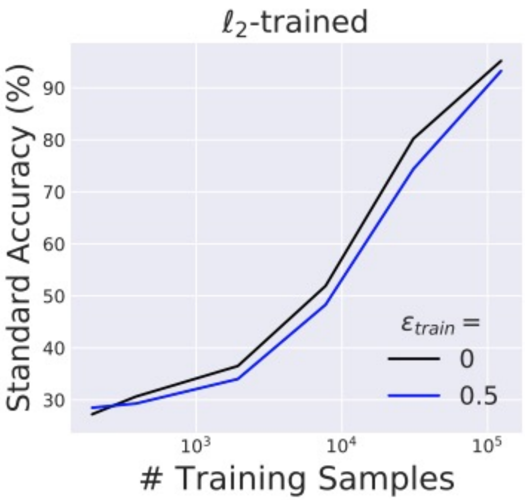
Part b) Adversarial robustness may be at odds with accuracy



(a) MNIST



(b) CIFAR-10



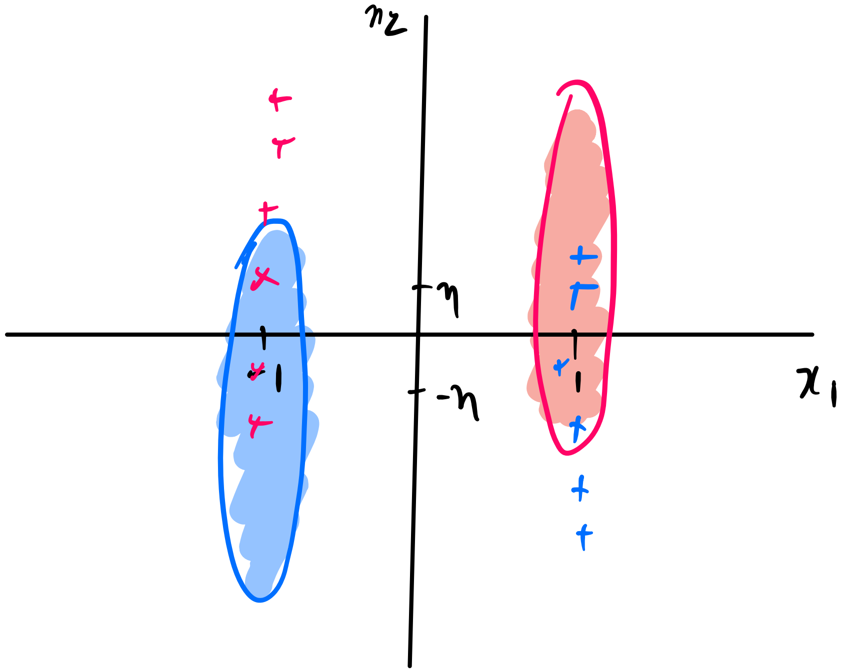
(c) Restricted ImageNet

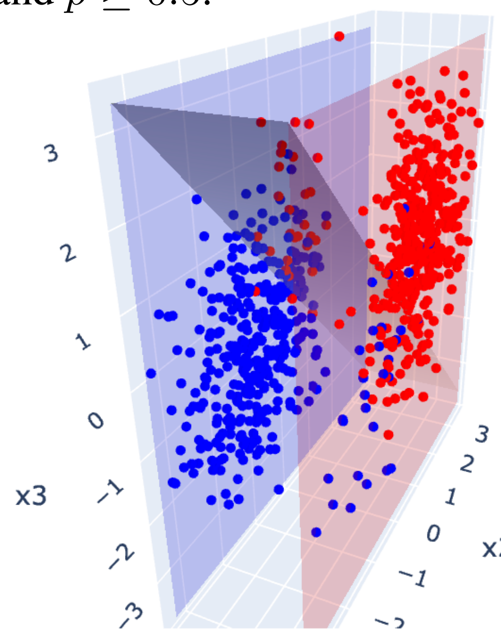
Fig. from *Robustness May Be at Odds with Accuracy, Tsipras et al. '18*

A simple Gaussian setting to understand tradeoff

$$y \sim^{u.a.r.} \{-1,+1\}, \qquad x_1 = \begin{cases} +y, & \text{with probability } p, \\ -y, & \text{with probability } 1-p, \end{cases} \qquad x_2, \dots, x_{d+1} \overset{i.i.d.}{\sim} \mathcal{N}(\eta y, 1),$$

where $\mathcal{N}(\mu, \sigma^2)$ is a normal distribution with mean μ and variance σ^2, and $p \geq 0.5$.





$d = 2$

From *Robustness May Be at Odds with Accuracy, Tsipras et al. '18*

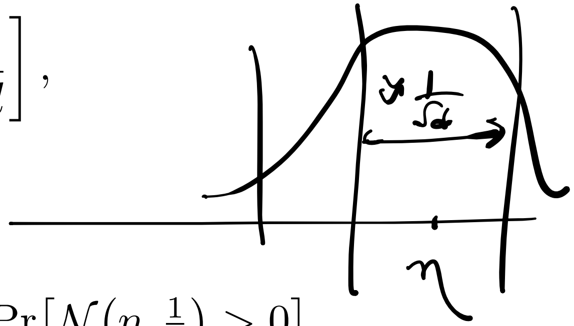
A simple Gaussian setting to understand tradeoff

$$y \stackrel{u.a.r.}{\sim} \{-1,+1\}, \qquad x_1 = \begin{cases} +y, & \text{with probability } p, \\ -y, & \text{with probability } 1-p, \end{cases} \qquad x_2, \ldots, x_{d+1} \stackrel{i.i.d.}{\sim} \mathcal{N}(\eta y, 1),$$

where $\mathcal{N}(\mu, \sigma^2)$ is a normal distribution with mean μ and variance σ^2, and $p \geq 0.5$.

Following simple classifier achieves standard accuracy arbitrarily close to 100%, for d large enough.

$$f_{\text{avg}}(x) := \text{sign}(w_{\text{unif}}^\top x), \qquad \text{where } w_{\text{unif}} := \left[0, \frac{1}{d}, \ldots, \frac{1}{d}\right],$$



To see this,

$$\Pr[f_{\text{avg}}(x) = y] = \Pr[\text{sign}(w_{\text{unif}}^\top x) = y] = \Pr\left[\frac{y}{d}\sum_{i=1}^{d} \mathcal{N}(\eta y, 1) > 0\right] = \Pr\left[\mathcal{N}\left(\eta, \tfrac{1}{d}\right) > 0\right],$$

which is $> 99\%$ when $\eta \geq 3/\sqrt{d}$.

$$\text{sign}\left(\frac{1}{d}\sum_{i=1}^{d} x_i\right) = y \qquad \sum_{i=1}^{d}\left(\frac{y}{d}\right)\mathcal{N}(\eta y, 1) = \sum_{i=1}^{d} \mathcal{N}\left(\frac{\eta}{d}, \frac{1}{d^2}\right)$$

From *Robustness May Be at Odds with Accuracy, Tsipras et al. '18*

A simple Gaussian setting to understand tradeoff

$$y \overset{u.a.r.}{\sim} \{-1, +1\}, \qquad x_1 = \begin{cases} +y, & \text{with probability } p, \\ -y, & \text{with probability } 1-p, \end{cases} \qquad x_2, \ldots, x_{d+1} \overset{i.i.d.}{\sim} \mathcal{N}(\eta y, 1),$$

where $\mathcal{N}(\mu, \sigma^2)$ is a normal distribution with mean μ and variance σ^2, and $p \geq 0.5$.

$$f_{\text{avg}}(x) := \text{sign}(w_{\text{unif}}^\top x), \qquad \text{where } w_{\text{unif}} := \left[0, \frac{1}{d}, \ldots, \frac{1}{d}\right],$$

Is f_{avg} robust?

In this setting, we have

- Robust feature, x_1: This has ℓ_∞ robustness even at $\epsilon = 0.99$, but only gets accuracy p
- Non-robust features $\{x_2, \ldots, x_d\}$: Using these f_{avg} gets accuracy >99%, but ℓ_∞ robustness only at $\epsilon \leq 2\eta$

Suppose $p = 0.95$. Then can show

- If standard accuracy is much greater than 95%, say close to 100%, then robust accuracy is close to 0!
- Can get robust accuracy 95%, but only with standard accuracy at close to 95%!