Lecture 17: Backdoors in ML

Lecturer: Neekon Vafa **Date:** April 9, 2026
Scribes: Shorna Alam, Xanthe Saalmann

1. Backdoors in ML Models

Outsourcing ML training introduces the risk of backdoor attacks, in which a malicious trainer can embed adversarial examples into the model.

Definition 1.1 (Undetectable Backdoors)**.** One can plant an undetectable backdoor in a classifier $M : \mathbb{R}^n \to \{\pm 1\}$ if:

- For all $\boldsymbol{x} \in \mathbb{R}^n$, the adversary can generate $\boldsymbol{x}'$ far from $\boldsymbol{x}$ such that $M(\boldsymbol{x}') = M(\boldsymbol{x})$.
- Honest and backdoored classifiers are computationally indistinguishable.

A natural question is then whether undetectable backdoors can be planted in *any* classifier, or only in specific model classes.

Theorem 1.1 (Goldwasser, Kim, Vaikuntanathan, Zamir, 2022 [3])**.** *Under standard cryptographic assumptions, one can plant undetectable backdoors in any classification models (black-box undetectable).*

The knowledge that undetectable backdoors can be planted in classifiers raises a key concern, given that a client has no way of verifying whether or not a model has been backdoored. We next consider how an untrusting client can remove backdoors from their classifier without harming the performance of the model.

2. Removing Backdoors from ML Models

Although these backdoors are undetectable, it does not mean that the client is defenseless. When we wash our hands, we aren't sure whether or not microscopic germs are actually on our hands, but we can still protect ourselves against infection by always cleaning our hands. Similarly, we can apply standardized processes to remove backdoors without detecting them beforehand. These defense and mitigation strategies typically take the form of efficient (offline or online) post-processing of the received model to produce a model that is still accurate but no longer adversarially corrupted. The following strategies are online defenses performed at inference time.

Defense/Mitigation Definitions and Setup

$M : \mathbb{R}^n \to \{\pm 1\}$ is the (possibly backdoored) model.

A is a defense algorithm that has black-box access to M and clean samples from the trusted data distribution D.

Goal: $\forall\, \boldsymbol{x}$, ensure $A^{M,D}(\boldsymbol{x})$ is "good" even if $M(\boldsymbol{x})$ is not.

3. Randomized Smoothing

Typically, backdoors cause sharp local discontinuities in M, and smoothing can remove them. Randomized smoothing is a local defense method operating on the query scale for models that are locally Lipschitz. Briefly, a function f is *locally Lipschitz* if small perturbations to the input produce bounded changes in the output.

Definition 3.1 (Randomized Smoothing)**.** On an input query $\boldsymbol{x}$ and some possibly backdoored model $M : \mathbb{R}^n \to \{\pm 1\}$, define a new model $A^M : \mathbb{R}^n \to \{\pm 1\}$ as follows:

$$A^M(\boldsymbol{x}) := \arg\max_{y \in \{\pm 1\}} \Pr_{\boldsymbol{\varepsilon} \sim N(0, \sigma^2 I)} [M(\boldsymbol{x} + \boldsymbol{\varepsilon}) = y]$$

where $\sigma \geq 0$ is a tunable parameter.

Note that if $\sigma = 0$, $A^M(\boldsymbol{x})$ is identical to $M(\boldsymbol{x})$, and as $\sigma \to \infty$, A^M becomes a constant function outputting the label that M outputs most frequently on average.

By construction, A^M outputs the label that M outputs most often under Gaussian perturbation, such that the local smoothness of M determines the robustness of A^M. This stability is captured by p_y, the probability that M outputs y under perturbation.

For $\boldsymbol{x} \in \mathbb{R}^n$, suppose $y \in \{\pm 1\}$ is the output of $A^M(\boldsymbol{x})$, and define:

$$p_y := \Pr_{\boldsymbol{\varepsilon} \sim N(0, \sigma^2 I)} [M(\boldsymbol{x} + \boldsymbol{\varepsilon}) = y] > 1/2.$$

The following theorem establishes the relevance of smoothness to robustness:

Theorem 3.1 (Cohen-Rosenfeld-Kolter '19 [1])**.** *For any function M and for all inputs $\boldsymbol{x}$, the model $A^M(\cdot)$ is the **constant function** y on a ball of radius $R = \sigma \cdot \Phi^{-1}(p_y)$ around $\boldsymbol{x}$, where $\Phi : \mathbb{R} \to [0, 1]$,*

$$\Phi(t) = \Pr_{Z \sim N(0,1)} [Z \leq t]\,.$$

For A^M to remove a backdoor, R must exceed the size of the backdoor perturbation. Note that if a backdoor introduces a sharp transition in M near the backdoored input(s), p_y approaches $1/2$, meaning $\Phi^{-1}(p_y) \to 0$ and thus $R \to 0$.

Randomized Smoothing: Interpretations and Observations

- Randomized smoothing does not guarantee backdoor removal, and A^M can still have adversarial examples that lie beyond the secure radius. If the attacker perturbs the input by a distance greater than R, then randomized smoothing will largely use samples far from the uncorrupted input and fail to restore the model. M can also be corrupted on a ball around some input $\boldsymbol{x}^*$ such that A^M is still wrong on input $\boldsymbol{x}^*$.
- A^M is computed via Monte Carlo sampling of a number of ϵ_i to query M on each $\boldsymbol{x} + \epsilon_i$ and estimating p_y accordingly. $A^M(\boldsymbol{x})$ is then the most frequent label from these queries, and $R = \sigma \cdot \Phi^{-1}(p_y)$.
- The agreement of A^M with M depends on the selected noise parameter σ. Smaller σ means less structure of M is lost, but robustness radius R is smaller. Larger σ directly increases R but also brings p_y closer to $1/2$, harming accuracy and robustness. The optimal σ lies somewhere between.

4. Oblivious Defense

Randomized smoothing provides local robustness around individual datapoints; by contrast, oblivious defense is a global defense method that exploits global structure in D. It is less general since it only works for structured data distributions D, but also obtains stronger worst-case guarantees. In this paradigm, the labeled data come from some *structured* ground truth distribution D over $(\boldsymbol{x}, y)$ pairs (for instance, a half space, linear regression, polynomial regression, or neural network).

Definition 4.1 (Accuracy of a Model)**.** For a model M on a structured D, it is possible to define accuracy for the model. Possible metrics (to be minimized) can include:

1. **0-1 loss:** $L_D(M) := \Pr_{(\boldsymbol{x},y)\sim D}[M(\boldsymbol{x}) \neq y]$
2. ℓ_2^2 **error:** $L_D(M) := \mathbb{E}_{(\boldsymbol{x},y)\sim D}[(M(\boldsymbol{x}) - y)^2]$

Accuracy only provides information about the **average case** performance of the model, not the **worst case** performance. Therefore, it is possible for a model M to have high accuracy overall but have poor accuracy on particular inputs of interest, $\hat{\boldsymbol{x}}$.

Since the above definition of accuracy is insufficient to capture worst case performance, we describe a notion of defense that takes worst case performance into account.

Definition 4.2 (Defense)**.** An algorithm A is a *secure mitigation algorithm* for a structured distribution D if $\exists$ a simulator Sim s.t. for all models M with good accuracy, the following hold:

- $\forall \boldsymbol{x} \in \mathbb{R}^n$, $A^{M,D}(\boldsymbol{x})$ and $\text{Sim}^D(\boldsymbol{x})$ are statistically close.
- $A^{M,D}$ has high accuracy (i.e., low loss)
- $A^{M,D}$ is significantly more efficient than relearning on samples redrawn directly from D.

Given access to clean samples from D, A can directly determine whether M has good accuracy, so A doesn't have to assume the accuracy of M. Since randomized smoothing exploits local smoothness,

attackers can circumvent it by planting a backdoor in the entire neighborhood of a target datapoint. By contrast, this definition provides global guarantees because making $A^{M,D}(\boldsymbol{x})$ and $\mathrm{Sim}^{D}(\boldsymbol{x})$ deviate systematically would degrade the overall performance of the model. This ensures that the adversary cannot reliably induce failure on even a single target datapoint x without violating the global accuracy and statistical closeness guarantees.

5. Random Self-Reproducibility

The central idea of oblivious defense is to exploit structure in the data distribution, so worst-case queries can become average-case evaluations. Consider a calculator that occasionally outputs incorrect values, analogous to a backdoored model M which occasionally outputs bad values for some unknown inputs $\hat{\boldsymbol{x}}$. If a user were trying to compute some simple sum, $x + y$, how might they determine if that particular calculation is incorrect?

One simple option is to perform the following procedure:

1. Sample a random integer r.
2. Compute $(x + r) + (y - r)$
3. Repeat many times, and take the mode of the results.

The above algorithm works because addition/subtraction have exploitable structure. This generalizes well to other function classes with structure, such as the polynomials; a formal version of this trick is the Reed-Muller Local Decoding.

Reed-Muller Local Decoding [2]

Let $f : \mathbb{F}^n \to \mathbb{F}$ be a ground-truth multivariate polynomial of total degree d, and let D be the distribution $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x} \leftarrow \mathbb{F}^n$ uniformly at random. Let $M : \mathbb{F}^n \to \mathbb{F}$ be the adversarially returned model with low 0-1 loss:

$$\Pr_{\boldsymbol{x} \leftarrow \mathbb{F}^n}[f(\boldsymbol{x}) \neq M(\boldsymbol{x})] \leq \frac{1}{3(d+1)}.$$

The **goal** of the online mitigator is: given a query point $\boldsymbol{x}^* \in \mathbb{F}^n$, recover the true value $f(\boldsymbol{x}^*)$ using only black-box access to M.
The key idea is to draw a **random line** through $\boldsymbol{x}^*$: sample $\boldsymbol{b} \leftarrow \mathbb{F}^n$ and define

$$\ell(t) := \boldsymbol{x}^* + \boldsymbol{b} \cdot t.$$

The points $\ell(1), \ell(2), \ldots$ are each marginally uniform over $\mathbb{F}^n$ but correlated with each other. Importantly, $f(\ell(t))$ is a **univariate** polynomial of degree d in t, with $f(\ell(0)) = f(\boldsymbol{x}^*)$.

Then to recover $f(\boldsymbol{x}^*)$, evaluate M on $\ell(1), \ell(2), \ldots, \ell(d+1)$. Note that all of these points will be correct with probability $\geq 2/3$ by the union bound. Use **univariate** polynomial interpolation to recover $p(t) := f(\ell(t))$. Finally, plug in $t = 0$ to output $p(0) = f(\boldsymbol{x}^*)$.

The overall effect of using Reed-Muller Local Decoding is that a multivariate polynomial evaluation with $\approx n^d$ parameters becomes a univariate polynomial interpolation with $\approx d$ parameters.

6. Polynomials over $\mathbb{R}^n$

Goldwasser-Shafer-Vafa-Vaikuntanathan '24 [4], informal

There is a secure mitigation for all ground truth data distributions that are close to a linear function or multivariate polynomial over $\mathbb{R}^n$.

Compared to polynomials over the finite field, distributional guarantees over the reals are more subtle. Importantly, sequential points on a line have the wrong marginal distribution, which breaks the Reed-Muller approach. A key tool to handle this is the **Correlated Sampling Lemma**.

Correlated Sampling Lemma

For all bounded convex sets $C \subseteq \mathbb{R}^n$, there is an efficient algorithm that takes in any $\boldsymbol{x}^* \in C$ and outputs $d + 1$ points with the following properties:

- The marginal distribution of each point is uniform over C.
- All points lie on a line going through $\boldsymbol{x}^*$.
- The points are independent, conditioned on being colinear.

Fix some $\boldsymbol{x}^* = 0 \in \mathbb{R}^n$ and consider a uniform distribution over a unit ball in $\mathbb{R}^n$. How is it possible to sample many points on a line through $\boldsymbol{x}^*$ such that all points have the correct marginal distribution and are otherwise independent?

In high dimensions, volume is concentrated at the surface of the sphere. A uniformly sampled point is therefore much more likely to be near the surface of a sphere, but this is not reflected by uniform sampling along a line passing through the center of that sphere. The correct conditional distribution on the line is $\rho(r) \propto r^{n-1}$; this is the "curse of dimensionality".

Lesson from Mitigation

An accurate but adversarially corrupted model can still be leveraged to obtain worst-case guarantees, provided the underlying data distribution has sufficient structure.

7. Open Questions and Future Directions

A number of open questions were discussed in lecture:

- What mathematical structure is necessary for backdoor mitigation?
- How well can sanitization work in practice? Is it possible to combine local (randomized smoothing) and global (oblivious defense) approaches to sanitization? Is natural language randomly self-reducible in any meaningful way?

8. Verifiable Computation

The backdoor defense methods that have been described so far work after a backdoor has already been planted in M; instead, is it possible to prevent a backdoor from being planted in the first place? At a high level, verifiable computation works by compelling model owners to use heavyweight cryptography to prove that their model M was honestly trained. Some ways to do this include:

- An honest party provides randomness, and an ML provider gives a (zk)-SNARK π of honest training.
- If an honest party has private training data, 2-party secure computation with the ML provider produces a secure M such that the provider cannot deviate from the training protocol and cannot learn anything about the honest party's input.

Since ML models are so large, practical work has to be done to make these verification paradigms efficient.

For completeness, we define the relevant cryptographic primitives:

Cryptographic Primitives

- **(zk)-SNARK**: Zero-Knowledge Succinct Non-Interactive Argument of Knowledge
- **2-party secure computation**: A protocol in which two parties evaluate a function without sharing their private inputs, i.e. simulates a trusted third party. Further, maliciously secure 2PC enforces protocol adherence to a pre-specified training algorithm, although it does not prevent the adversary from using false data.

9. Unavoidable Malicious Behaviors

Cryptography often uses abstractions to convert a real adversary into an "idealized" adversary that is bound by certain rules; these adversaries honestly follow the protocol except for any fundamental powers, such as choosing inputs (and when to abort). Consequently, cryptography alone is not sufficient to protect against all malicious behaviors since it requires depending on adversaries to use honest training data. That is, secure computation cannot protect against data poisoning attacks.

10. Conclusion

In the previous lecture, we learned how to plant (white-box) undetectable backdoors for a limited class of models. We've also learned how to defend against backdoors in a limited set of ways (randomized smoothing, oblivious defense, and verifiable computation). At this point, we need to address backdoors in settings between those two extremes. We've begun to develop a theoretical understanding of backdoor attacks and mitigation, assuming super-efficient cryptography. In real life, those assumptions are not practical and these approaches don't address unavoidable malicious

behaviors, such as data poisoning. We covered some of this in Lecture 15, in which Sam Hopkins discussed ways to make statistics and training more robust.

References

[1] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1310–1320. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/cohen19c.html.

[2] Peter Gemmell, Richard Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, STOC '91, page 33–42, New York, NY, USA, 1991. Association for Computing Machinery. ISBN 0897913973. doi: 10.1145/103418.103429. URL https://doi.org/10.1145/103418.103429.

[3] Shafi Goldwasser, Michael P. Kim, Vinod Vaikuntanathan, and Or Zamir. Planting undetectable backdoors in machine learning models. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 931–942, 2022.

[4] Shafi Goldwasser, Jonathan Shafer, Neekon Vafa, and Vinod Vaikuntanathan. Oblivious defense in ml models: Backdoor removal without detection. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, page 1785–1794, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400715105. doi: 10.1145/3717823.3718245. URL https://doi.org/10.1145/3717823.3718245.