Backdoor Defense, Learnability and Obfuscation

Paul Christiano[1]
Alignment Research Center, Berkeley, CA, USA

Jacob Hilton[2]
Alignment Research Center, Berkeley, CA, USA

Victor Lecomte
Alignment Research Center, Berkeley, CA, USA

Mark Xu[3]
Alignment Research Center, Berkeley, CA, USA

Abstract

We introduce a formal notion of defendability against backdoors using a game between an attacker and a defender. In this game, the attacker modifies a function to behave differently on a particular input known as the "trigger", while behaving the same almost everywhere else. The defender then attempts to detect the trigger at evaluation time. If the defender succeeds with high enough probability, then the function class is said to be defendable. The key constraint on the attacker that makes defense possible is that the attacker's strategy must work for a randomly-chosen trigger.

Our definition is simple and does not explicitly mention learning, yet we demonstrate that it is closely connected to learnability. In the computationally unbounded setting, we use a voting algorithm of Hanneke et al. [13] to show that defendability is essentially determined by the VC dimension of the function class, in much the same way as PAC learnability. In the computationally bounded setting, we use a similar argument to show that efficient PAC learnability implies efficient defendability, but not conversely. On the other hand, we use indistinguishability obfuscation to show that the class of polynomial size circuits is not efficiently defendable. Finally, we present polynomial size decision trees as a natural example for which defense is strictly easier than learning. Thus, we identify efficient defendability as a notable intermediate concept in between efficient learnability and obfuscation.

2012 ACM Subject Classification Theory of computation → Machine learning theory; Computing methodologies → Machine learning; Security and privacy → Mathematical foundations of cryptography; Theory of computation → Cryptographic primitives

Keywords and phrases backdoors, machine learning, PAC learning, indistinguishability obfuscation

Digital Object Identifier 10.4230/LIPIcs.ITCS.2025.38

Related Version *Version including Appendix A in full*: `https://arxiv.org/abs/2409.03077`

Acknowledgements We are grateful to Dmitry Vaintrob for an earlier version of the results in Appendix A; to Thomas Read for finding the "Backdoor with likely input patterns" example and for help with proofs; to Andrea Lincoln, Dávid Matolcsi, Eric Neyman, George Robinson and Jack Smith for contributions to the project in its early stages; and to Geoffrey Irving, Robert Lasenby and Eric Neyman for helpful comments on drafts.

[1] Work done while at the Alignment Research Center prior to April 2024.
[2] Corresponding author.
[3] Authors ordered alphabetically.

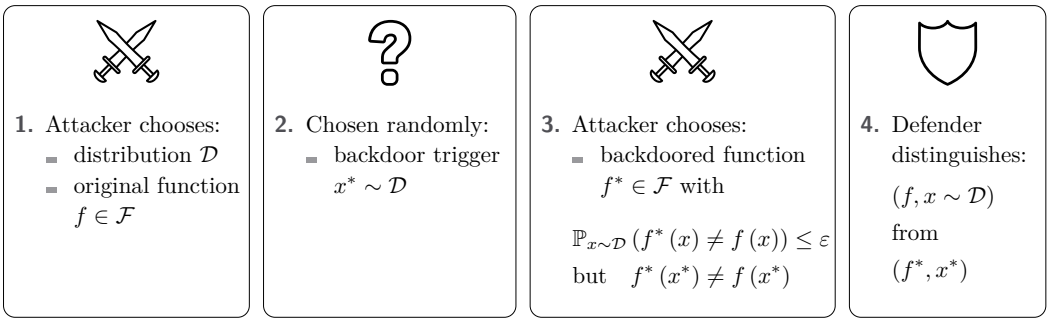


Figure 1 The game used to define ε-defendability for a class $\mathcal{F}$ of $0,1$-valued functions.

1 Introduction

A *backdoor* in a machine learning model is a modification to the model that causes it to behave differently on certain inputs that activate a secret "trigger". There is a wide literature on backdoor attacks and defenses from both a theoretical and an empirical perspective [26]. However, prior theoretical work typically makes reference to a particular training dataset, leading to a focus on data poisoning attacks. In this work we introduce a formal notion of a backdoor that allows the attacker to modify the model arbitrarily. Our definition is simple, but nevertheless gives rise to a rich array of strategies incorporating both learning and obfuscation.

Our formal notion is focused on backdoor detection *at runtime*, meaning that the defender is given a particular input and must flag whether or not it activates the backdoor trigger. We focus on this case because of the existence of *undetectable* backdoors if the defender is instead required to flag a model as backdoored without being given any particular input [11]. Moreover, detection at runtime is sufficient in threat models where the attacker is given only one opportunity to modify the model, being akin to giving the defender an additional chance to modify the model after the attacker. We also focus on the white-box setting, in which the defender has access to a complete description of the model.

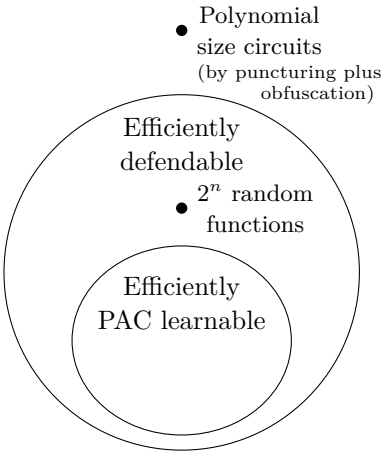
Our main contributions, which are also summarized in Figures 1 and 2, are as follows:

- **Definition of ε-defendability (Section 3).** We introduce a simple game between an attacker and a defender to define what it means for a representation class to be *ε-defendable* with a certain confidence. The only constraint placed on the attacker is that their backdoor strategy must work for a randomly-chosen trigger. Without this assumption, there is a symmetry between the original and backdoored functions that makes defense impossible.
- **Statistical defendability (Section 4).** In the absence of computational constraints on the defender, we show that a representation class $\mathcal{F}$ is ε-defendable with confidence tending to 1 if and only if $\varepsilon = o(1/\mathrm{VC}(\mathcal{F}))$, where $\mathrm{VC}(\mathcal{F})$ is the Vapnik–Chervonenkis dimension of $\mathcal{F}$. We achieve this by adapting a result of Hanneke et al. [13] that applies a learning algorithm multiple times and takes a majority vote.
- **Computational defendability (Section 5).** We introduce a notion of *efficient defendability* in which the defender's detection strategy must run in polynomial time. We show that efficient PAC learnability implies efficient defendability, but not conversely. We also show that under certain cryptographic assumptions, the class of polynomial size circuits is not efficiently defendable, by combining a puncturable pseudorandom function with an efficient indistinguishability obfuscator.

Statistical defendability (Section 4)

$\mathcal{F}$ is ε-defendable with confidence $\to 1$
$\iff$
$\varepsilon = o\left(1/\mathrm{VC}(\mathcal{F})\right)$
(by learning plus voting)

Computational defendability (Section 5)



Defendability of decision trees (Section 6)

Polynomial size decision trees are efficiently uniform-defendable in a single evaluation

(a) Statistical defendability is concerned with computationally unbounded defenders, and is essentially determined by the VC dimension, in much the same way as statistical learnability.

(b) Computational defendability is concerned with polynomial-time defenders, and the set of efficiently defendable classes lies strictly between the set of efficiently learnable classes and the set of all classes.

(c) In the setting where the distribution over inputs is uniform, polynomial size decision trees are a natural class for which defense is faster than learning.

Figure 2 Summary of the results in this paper.

- **Defendability of decision trees (Section 6).** In the setting where the distribution over inputs is uniform, we give a defense for polynomial size decision trees that runs in the time taken to evaluate a single decision tree. This provides a natural example for which defense is faster than learning.

We conclude in Section 7 with a discussion of whether efficient defendability can be further separated from efficient PAC learnability, so-called "mechanistic" defenses that resemble our defense for polynomial size decision trees, and the implications of our results for AI alignment.

2 Related work

The most closely related work to our own is that of Goldwasser, Kim, Vaikuntanathan, and Zamir [11], whose main results use a digital signature scheme to insert an undetectable backdoor into a model. The backdoor trigger can be applied to any input by perturbing it appropriately. In the black-box setting, the backdoor is undetectable in the sense that it is computationally infeasible for the defender to find any input on which the backdoored model and the original model differ. In the white-box setting, the backdoor is undetectable in the sense that it is computationally infeasible for the defender to distinguish the backdoored model from the original model. However, both of these results are in the setting where the backdoor trigger must be extracted from the model rather than detected at runtime.

In the defense at runtime setting, the same authors also show that a backdoored model can be modified by the defender to produce a model that is not backdoored. Once again, the backdoor trigger is applied by perturbing the input. The result only applies to models that

satisfy a certain smoothness assumption, and mirrors the use of randomized smoothing to achieve certified adversarial robustness by Cohen et al. [6]. By contrast, we avoid making smoothness assumptions, and further develop the theory of runtime backdoor detection.

Concurrent work by Goldwasser, Shafer, Vafa, and Vaikuntanathan [12] studies a black-box, non-realizable version of backdoor defense at runtime. Rather than predicting the exact label of the non-backdoored model on the given input (which is equivalent to our notion of runtime detection), the defender must instead produce a "canonical" label that has high accuracy on average, by using black-box access to the potentially backdoored model. In this local mitigation setting, they construct efficient defenders for almost-linear and almost-polynomial functions. They also consider the global mitigation setting, in which the defender must produce an entire clean model, constructing efficient global mitigators for Fourier-heavy functions.

Adversarial examples can be thought of as backdoor triggers, but the attacker must find similar inputs rather than a model with similar outputs. The gap between statistical and computational hardness has been explored in this context by Bubeck et al. [5] and Garg et al. [8]. Much like us, they find that defense is statistically possible but computationally hard.

Returning to backdoors, another thematically similar work to our own is that of Khaddaj et al. [23], which also questions the assumptions behind different backdoor attacks and defenses. However, it remains within the data-poisoning setting, in which the backdoor is inserted using corrupted training data.

A line of work beginning with Dumford and Scheirer [7] goes beyond data-poisoning, inserting a backdoor into an already-trained model, but still using data to search for the backdoored model. More manual methods of backdoor insertion were later introduced by Hong et al. [15]. This line of work is primarily empirical rather than theoretical.

Our result on statistical defendability is heavily based on the result of Hanneke et al. [13] in the realizable case. Following Levine and Feizi [25] and Jia et al. [18], they use a voting algorithm to produce a backdoor defense in the data-poisoning setting. By combining this with a classical result of Haussler, Littlestone, and Warmuth [14] on optimal learning, they relate the optimal performance of this algorithm to the VC dimension of the function class. Our result adapts this to our more general notion of defendability.

Our result on computational defendability of polynomial size circuits uses a hybrid argument that combines a puncturable pseudorandom function and an efficient indistinguishability obfuscator. This combination has been used in a number of previous hybrid arguments, as described by Sahai and Waters [29]. More generally, the relationship between learnability and cryptographic assumptions has a long history, with some basic constructions in this vein described in the textbook of Kearns and Vazirani [22].

3 Definition of ε-defendability

Our formal notion of defendability against backdoors is based on a game played between two players, an "attacker" and a "defender". Informally, the attacker starts with a function and is given a randomly sampled backdoor trigger. They must choose a backdoored function that almost always agrees with the original function, but disagrees with the original function on the backdoor trigger. Then the defender is given either the original function and a random input, or the backdoored function and the backdoor trigger, and they must distinguish one from the other with high confidence.

For our formal definition, we fix a security parameter $n \in \mathbb{N}$ and consider classifiers $\mathcal{X} \to \{0,1\}$ where $\mathcal{X} = \{0,1\}^n$. Formally, a *representation class* over $\mathcal{X}$ is a function $\{0,1\}^{R(n)} \to \{\text{all functions } \mathcal{X} \to \{0,1\}\}$ for some $R(n) \in \mathbb{N}$, but we generally keep the

choice of representation implicit and conflate a representation class with the set of functions in its image. In our results on computationally constrained defenders, we also implicitly assume that $R(n)$ is polynomial in n.

Our definition also makes use of the *example oracle* $\mathrm{Ex}(f, \mathcal{D})$ where $\mathcal{D}$ is a distribution over $\mathcal{X}$ and $f : \mathcal{X} \to \{0,1\}$. This is an oracle that outputs $(x, f(x))$ for a randomly sampled $x \sim \mathcal{D}$.

▶ **Definition 3.1.** Let $\mathcal{F}$ be a representation class over $\mathcal{X} = \{0,1\}^n$ and let $\varepsilon, \delta > 0$. We say that $\mathcal{F}$ is *ε-defendable with confidence* $1 - \delta$ if there is a probabilistic algorithm (the "detection strategy") that wins the following game with probability at least $1 - \delta$:

- An adversary with knowledge of the detection strategy (the "attacker") chooses a distribution $\mathcal{D}$ over $\mathcal{X}$ and $f \in \mathcal{F}$.
- We randomly sample $x^* \sim \mathcal{D}$ (the "backdoor trigger"), and then the adversary chooses $f^* \in \mathcal{F}$ (the "backdoored function").
- The backdoored function f^* is called *ε-valid* if

$$\mathbb{P}_{x \sim \mathcal{D}}(f^*(x) \neq f(x)) \leq \varepsilon \quad \text{but} \quad f^*(x^*) \neq f(x^*).$$

- We randomly choose (f', x') to pass to the detection strategy to be either (f, x) for a random sample $x \sim \mathcal{D}$, or (f^*, x^*), with 50% probability each. The detection strategy is given access to the example oracle $\mathrm{Ex}(f', \mathcal{D})$ as well as the representation of f', x', $f'(x')$, ε and δ as input, and must output either Acc (for "accept") or Rej (for "reject").
- If (f, x) is passed to the detection strategy, then the detection strategy wins if it outputs Acc. If (f^*, x^*) is passed to the detection strategy, then the detection strategy wins if either it outputs Rej or f^* is not ε-valid.

In this definition, the adversary is not specified by a collection of algorithms, but is instead shorthand for a universal quantifier. Thus we could have equivalently (but perhaps convolutedly) said, "there exists a detection strategy such that, for all distributions $\mathcal{D}$ over $\mathcal{X}$ and all $f \in \mathcal{F}$, the expectation over $x^* \sim \mathcal{D}$ of the minimum over all $f^* \in \mathcal{F}$ of the win probability of the detection strategy is at least $1 - \delta$".

The manner in which f, x^* and f^* are chosen in this definition deserves some elaboration. If the attacker were allowed to choose the backdoor trigger x^* themselves, then it would be too easy for the adversary to insert an undetectable backdoor. So we instead require the adversary to come up with a backdoor insertion strategy that works for a randomly-chosen trigger. In fact, we do not even allow the adversary to see the backdoor trigger when they are selecting the original function f, or else they could produce f^* by *removing an existing backdoor* rather than *inserting a new one*. This is illustrated by the following example.

▶ **Example** (Existing backdoor removal). Let $\mathcal{X} = \{0,1\}^n$ and let $\mathcal{F} = \{\mathbb{1}_x \mid x \in \mathcal{X}\}$, where $\mathbb{1}_x : \mathcal{X} \to \{0,1\}$ denotes the indicator function

$$\mathbb{1}_x(x') = \begin{cases} 1, & \text{if } x' = x \\ 0 & \text{if } x' \neq x. \end{cases}$$

Take $\mathcal{D}$ to be uniform over $\mathcal{X}$. If we allowed the adversary to choose both f and f^* after sampling $x^* \sim \mathcal{D}$, then they could take $f = \mathbb{1}_{x^*}$ and $f^* = \mathbb{1}_{x'}$ for $x' \sim \mathcal{D}$ (or in the event that $x' = x^*$, instead take $f = \mathbb{1}_{x''}$ for some $x'' \in \mathcal{X} \setminus \{x^*\}$ sampled uniformly at random). Then f^* would be $\frac{2}{2^n}$-valid, and (f, x) and (f^*, x^*) would be identically distributed for $x \sim \mathcal{D}$. So if $\varepsilon \geq \frac{1}{2^{n-1}}$ then any detection strategy would win with probability at most $\frac{1}{2}$. This would be quite an extreme failure for the defender, especially considering that the VC dimension of $\mathcal{F}$ is only 1.

Thus the way in which f, x^* and f^* are chosen, by randomly sampling the backdoor trigger x^* after the original function f is chosen but before the backdoored function f^* is chosen, is the key symmetry-breaking assumption that makes the game interesting.

▶ Remark. We could generalize this definition to include functions $\mathcal{X} \to \mathcal{Y}$ where $\mathcal{Y}$ is any subset of $\mathbb{R}$ (or even more generally, any metric space). In this setting, the most natural generalization of ε-validity is

$$\mathbb{E}_{x\sim\mathcal{D}}\left[\left(f^*(x) - f(x)\right)^2\right] \leq \varepsilon \quad \text{but} \quad \left|f^*(x^*) - f(x^*)\right| \geq 1.$$

4 Statistical defendability

In this section we will completely determine, up to a constant factor, the values of ε and δ for which an arbitrary representation class is ε-defendable with confidence $1-\delta$, in the absence of any computational constraints on the detection strategy.

Our result is expressed in terms of the *Vapnik–Chervonenkis (VC) dimension* of the representation class $\mathcal{F}$, denoted $\mathrm{VC}(\mathcal{F})$. This is defined as the maximum size of a set S of points *shattered* by $\mathcal{F}$, meaning that all $2^{|S|}$ functions $S \to \{0,1\}$ are realized as the restriction $f|_S$ of some $f \in \mathcal{F}$.

▶ **Theorem 4.1.** *Let $\mathcal{F}$ be a representation class over $\{0,1\}^n$ and let $\varepsilon > 0$. Then the most (i.e., supremum) confidence with which $\mathcal{F}$ is ε-defendable is*

$$\max\left(\tfrac{1}{2}, 1 - \Theta\left(\mathrm{VC}(\mathcal{F})\,\varepsilon\right)\right)$$

as $\mathrm{VC}(\mathcal{F}) \to \infty$.

In particular, $\mathcal{F}$ is ε-defendable with confidence tending to 1 as $n \to \infty$ if and only if $\varepsilon = o\left(1/\mathrm{VC}(\mathcal{F})\right)$ as $n \to \infty$.

Thus in the computationally unbounded setting, defendability is determined almost entirely by the VC dimension of the representation class, in much the same way as PAC learnability [2, 22, Chapter 3].

In fact, to prove Theorem 4.1, we will make use of a certain kind of learning algorithm called a *prediction strategy*. Intuitively, a prediction strategy uses the value of a function on random samples to predict the value of the function on a new random sample. The formal definition is as follows.

▶ **Definition.** Let $\mathcal{F}$ be a representation class over $\mathcal{X} = \{0,1\}^n$. A *prediction strategy* for $\mathcal{F}$ is a randomized algorithm that is given access to the example oracle $\mathrm{Ex}(f, \mathcal{D})$ for some distribution $\mathcal{D}$ over $\mathcal{X}$ and some $f \in \mathcal{F}$, as well as $x \in \mathcal{X}$ as input, and outputs a prediction for $f(x)$.

The *sample size* of a prediction strategy is the maximum number of times it calls the example oracle, maximizing over any randomness used by the algorithm, any calls to the oracle, and the choice of $\mathcal{D}$, f and x.

Given a choice of $\mathcal{D}$ and f, the *error rate* of a prediction strategy is the probability that it fails to correctly predict $f(x)$ for $x \sim \mathcal{D}$, randomizing over any randomness used by the algorithm, any calls to the oracle, and the choice of x.

In their classical paper on prediction strategies, Haussler, Littlestone, and Warmuth [14] exhibit a "1-inclusion graph" algorithm yielding the following result.

▶ **Theorem** (Haussler–Littlestone–Warmuth). *For any representation class $\mathcal{F}$ and any positive integer m, there is a prediction strategy for $\mathcal{F}$ with sample size $m-1$ such that for any distribution $\mathcal{D}$ over $\mathcal{X}$ and any $f \in \mathcal{F}$, the error rate is at most*

$$\frac{\mathrm{VC}(\mathcal{F})}{m}.$$

As discussed in that work, there is a close relationship between prediction strategies and PAC learning algorithms, which we define in Section 5.2. We focus on prediction strategies in this section because they make our proof easier to carry out while avoiding the introduction of unnecessary logarithmic factors.

Our proof of Theorem 4.1 is closely modeled on a proof of Hanneke et al. [13, Theorem 3.1], although we cannot quote that result directly since it is specialized to a data poisoning setup. As in that proof, our detection strategy works by applying the Haussler–Littlestone–Warmuth prediction strategy multiple times and taking a majority vote.

The basic idea for the detection strategy is that, given (f', x') as input (which could be either (f, x) or (f^*, x^*)), if we could successfully predict $f(x')$, then we could distinguish the two cases by comparing this to $f'(x')$, since $f(x) = f(x)$ but $f(x^*) \neq f^*(x^*)$ if f^* is ε-valid. To attempt to make this prediction, we use the Haussler–Littlestone–Warmuth prediction strategy. However, if $f' = f^*$ then this can fail if we encounter a point where f and f^* disagree. Hence we are forced to use a small sample size, but we are still left with a small constant probability of failure. Fortunately though, we can repeat this procedure multiple times with independent samples and take a majority vote, which allows us to make this small constant probability vanish.

We now provide a sketch proof of Theorem 4.1. For a full proof, see Appendix B.

Sketch proof of Theorem 4.1. Write $d = \mathrm{VC}(\mathcal{F})$, and note that the detection strategy can achieve a win probability of at least $\frac{1}{2}$ simply by guessing randomly.

To show that the most confidence with which $\mathcal{F}$ is ε-defendable is at most $\max\left(\frac{1}{2}, 1 - \Omega(d\varepsilon)\right)$, let S be a set of size d shattered by $\mathcal{F}$. Roughly speaking, the adversary can take $\mathcal{D}$ to be uniform over S, and take f to be uniform over the witnesses to the shattering. Then the adversary can insert a backdoor by simply changing the value of f at x^* and no other points of S. The actual construction is slightly more careful than this, and is given in the full proof.

To show that the most confidence with which $\mathcal{F}$ is ε-defendable is at least $\max\left(\frac{1}{2}, 1 - O(d\varepsilon)\right)$, our detection strategy is as follows. Given (f', x') as input, we apply the Haussler–Littlestone–Warmuth prediction strategy with sample size $m-1$ and error rate at most $\frac{d}{m}$ to make a prediction for $f'(x')$, which we in turn think of as a prediction for $f(x')$. We repeat this procedure r times with independent samples and take a majority vote: if more than half of the predictions are different from $f'(x')$, then we output Rej, and otherwise we output Acc.

If (f, x) is passed to the detection strategy, then the probability that more than $\frac{1}{2}$ of the votes are wrong is at most $\frac{2d}{m}$, by Markov's inequality. If (f^*, x^*) is passed to the detection strategy, then there are two ways in which each vote can be wrong: either f and f^* disagree on at least one of the $m-1$ samples, or the prediction strategy fails to predict $f(x^*)$ even if it is provided with the value of f on all of these $m-1$ samples. The probability that $\frac{1}{4}$ or more of the votes will be wrong due to the second failure mode is at most $\frac{4d}{m}$, by Markov's inequality again. Meanwhile, we can ensure that the probability of the first failure mode for each vote is at most $\frac{1}{5}$ by taking $\frac{1}{5\varepsilon} < m \leq \frac{1}{5\varepsilon} + 1$. Since each choice of $m-1$ samples was independent, the probability that $\frac{1}{4}$ or more of the votes will be wrong due to the first

failure mode is at most $\exp\left(-\frac{r}{200}\right)$, by Hoeffding's inequality, which can be made arbitrarily small by taking r to be sufficiently large. Hence the probability that $\frac{1}{2}$ or more of the votes will be wrong is at most $\frac{4d}{m}$, and so the overall win probability is $1 - \frac{3d}{m} = 1 - O(\varepsilon d)$, as required. ◀

Intuitively, the reason that this detection strategy works is that the if ε is small compared to $1/\mathrm{VC}(\mathcal{F})$, then the attacker must choose a backdoored function that is very "strange". Hence the detection strategy can sample a new function that is similar to the given possibly backdoored function, but more "normal", and see if the two functions agree on the given possible backdoor trigger. This process can be thought of as a kind of regularization.

While our proof used distillation (i.e., learning from function samples) plus ensembling (i.e., voting), other methods of regularization may also work. For example, we could also have sampled from a Boltzmann distribution centered on the given function (with exponentially decaying probability based on Hamming distance), to get the same result up to logarithmic factors. We prove this claim in Appendix A.

5 Computational defendability

5.1 Definition of efficient defendability

In Section 4, we related the defendability of a representation class to its VC dimension. However, the prediction strategy we used to construct the detection strategy runs in exponential time, since it involves an expensive search over orientations of a certain graph. Hence it is interesting to ask what a polynomial-time detection strategy can achieve. To explore this, we introduce the following notion.

▶ **Definition 5.1.** Let $\mathcal{F}$ be a representation class over $\{0,1\}^n$. We say that $\mathcal{F}$ is *efficiently defendable* if there is some polynomial p such that for any $\delta > 0$ and any $\varepsilon > 0$ with $\varepsilon < 1/p\left(n, \frac{1}{\delta}\right)$, $\mathcal{F}$ is ε-defendable with confidence $1 - \delta$ using a detection strategy that runs in time polynomial in n and $\frac{1}{\delta}$.

Note that the condition that $\varepsilon < 1/p\left(n, \frac{1}{\delta}\right)$ is crucial: if $\mathcal{F}$ were required to be ε-defendable with confidence $1 - \delta$ for any $\varepsilon, \delta > 0$, then this would not be possible even for a detection strategy with unlimited time (except in trivial cases), by Theorem 4.1. (Recall that ε constrains the attacker, which is why this requirement helps the defender.)

As a reminder, we also assume in this section that representation classes have polynomial-length representations.

5.2 Efficient defendability and efficient PAC learnability

In this sub-section we show that efficient PAC learnability implies efficient defendability, but not conversely.

The well-studied model of probably approximately correct (PAC) learning was introduced by Valiant [30]. The notion of efficient PAC learnability bears some resemblance to our notion of efficient defendability. The following definition is taken from Kearns and Vazirani [22, Definition 4].

▶ **Definition.** Let $\mathcal{F}$ be a representation class over $\mathcal{X} = \{0,1\}^n$. We say that $\mathcal{F}$ is *efficiently PAC learnable* if there is a probabilistic algorithm (the "PAC learning algorithm") with the following properties. The PAC learning algorithm is given access to the example oracle $\mathrm{EX}(f, \mathcal{D})$ for some distribution $\mathcal{D}$ over $\mathcal{X}$ and some $f \in \mathcal{F}$, as well as $0 < \tilde{\varepsilon} < \frac{1}{2}$ (the "error

parameter") and $0 < \tilde{\delta} < \frac{1}{2}$ (the "confidence parameter") as input. The PAC learning algorithm must run in time polynomial in n, $\frac{1}{\tilde{\varepsilon}}$ and $\frac{1}{\tilde{\delta}}$, and must output some polynomially evaluatable hypothesis h, i.e. a representation for a function $\mathcal{X} \to \{0,1\}$ that is evaluatable in time polynomial in n. The hypothesis must satisfy $\mathbb{P}_{x\sim\mathcal{D}}\left(f\left(x\right) \neq h\left(x\right)\right) \leq \tilde{\varepsilon}$ with probability at least $1 - \tilde{\delta}$ for any distribution $\mathcal{D}$ over $\mathcal{X}$ and any $f \in \mathcal{F}$.

Note that $\tilde{\varepsilon}$ and $\tilde{\delta}$ play subtly different roles in this definition to ε and δ in the definition of efficient defendability. In particular, there is no condition on $\tilde{\varepsilon}$ as a function of n and $\tilde{\delta}$ in the definition of efficient PAC learnability.

We now show that efficient PAC learnability implies efficient defendability. This follows easily from the proof of Theorem 4.1. Even though that result used a prediction strategy that runs in exponential time, it is straightforward to replace it by an efficient PAC learning algorithm.

▶ **Corollary 5.2.** *Let $\mathcal{F}$ be a representation class over $\{0,1\}^n$. If $\mathcal{F}$ is efficiently PAC learnable, then $\mathcal{F}$ is efficiently defendable.*

Proof. If $\mathcal{F}$ is efficiently PAC learnable, then for any sufficiently large positive integer m, there is a polynomial-time prediction strategy for $\mathcal{F}$ with sample size $m-1$ and error rate at most $\frac{1}{6}\delta$: we simply PAC learn a hypothesis with confidence parameter $\frac{1}{12}\delta$ and error parameter $\frac{1}{12}\delta$, and then evaluate this hypothesis on the given point. The definition of PAC learning guarantees that this succeeds as long as m exceeds a particular polynomial in n and $\frac{1}{\delta}$.

Our detection strategy is then exactly the same as the one from the proof of Theorem 4.1, but using the above prediction strategy instead of the Haussler–Littlestone–Warmuth prediction strategy. Since $\varepsilon < 1/p(n,\frac{1}{\delta})$ for some polynomial p of our choice, choosing $m > \frac{1}{5\varepsilon}$ as in that proof suffices to ensure that m is eventually large enough for the prediction strategy to succeed. Finally, we take the number of votes r to be the smallest integer greater than $200\log\left(\frac{1}{\delta}\right)$, which keeps the time complexity of the detection strategy polynomial in n and $\frac{1}{\delta}$. Then the detection strategy's overall win probability is at least $1 - \frac{3}{6}\delta - \frac{1}{2}\exp\left(-\frac{r}{200}\right) > 1-\delta$, as required. ◀

We now show that the converse implication does not hold in the random oracle model of computation. In this model, programs have access to an oracle that outputs the result of calling of a uniformly random function $\{0,1\}^* \to \{0,1\}$.

▶ **Theorem 5.3.** *In the random oracle model of computation, with probability $1 - O(2^{-n^c})$ for all c over the choice of random oracle, there is a polynomially evaluatable representation class over $\mathcal{X} = \{0,1\}^n$ that is efficiently defendable but not efficiently PAC learnable.*

To prove this, we simply take a representation class of 2^n functions that make unique calls to the random oracle (supplying a different input to the oracle whenever either the function is different or the input to the function is different), and show that, with high probability over the choice of random oracle, this representation class is efficiently defendable but not efficiently PAC learnable. Roughly speaking, it is efficiently defendable because, for most sets of 2^n random functions with 2^n possible inputs, every pair of functions differs on around half of inputs, meaning that there are no ε-valid functions for the adversary to choose from. Meanwhile, it is not efficiently PAC learnable because most random functions with 2^n possible inputs differ from every possible function that a polynomial-time algorithm could output on around half of inputs. For a full proof, see Appendix C.

Although this result is in the random oracle model of computation, the same proof shows that the converse does not hold in the usual model of computation either. This is because we can make the random choices "on the outside": we fix a randomly-chosen lookup

table, and have the functions use that instead of calling the random oracle. However, this representation class is not necessarily polynomially evaluatable, since the lookup table would take up exponential space.

Nevertheless, we conjecture that there is a polynomially evaluatable counterexample that uses a pseudorandom function instead of a random oracle. The above proof cannot be immediately adapted to work for an arbitrary pseudorandom function, because two distinct keys could give rise to very similar functions, but it might not be too challenging to adapt it.

▶ **Conjecture 5.4.** *Assuming* OWF*, there is a polynomially evaluatable representation class over* $\{0,1\}^n$ *that is efficiently defendable but not efficiently PAC learnable.*

Here, OWF denotes the existence of a one-way function, which can used to construct a pseudorandom function [9, 20, Chapter 6].

One way in which our counterexample is unsatisfying is that it relies on the adversary being unable to find a backdoored function that is ε-valid, making detection trivial. However, it is straightforward to extend this result to a class for which it is always possible for the adversary to find a backdoored function that is ε-valid, but that is nevertheless efficiently defendable.

▶ **Example** (Special-cased random functions)**.** Let $\widetilde{\mathcal{F}}$ be one of the above representation classes over $\mathcal{X} = \{0,1\}^n$ that serves as a counterexample (either 2^n functions that call a random oracle, or a random choice of 2^n functions). Now let

$$\mathcal{F} = \left\{ x \mapsto \begin{cases} f(x), & \text{if } x \neq x^* \\ y & \text{if } x = x^* \end{cases} \;\middle|\; f \in \widetilde{\mathcal{F}}, x^* \in \mathcal{X}, y \in \{0,1\} \right\},$$

where implicitly, the representation of a function in $\mathcal{F}$ is given by a triple consisting of the representation of the function $f \in \widetilde{\mathcal{F}}$, the special-cased point $x^* \in \mathcal{X}$, and the special-cased value $y \in \{0,1\}$. Then the adversary can ensure that there is always a $\frac{2}{2^n}$-valid backdoored function by taking $\mathcal{D}$ to be the uniform distribution over $\mathcal{X}$. However, $\mathcal{F}$ is still efficiently defendable with high probability, since the detection strategy can check whether or not the input has been special-cased. $\mathcal{F}$ also remains not efficiently PAC learnable with high probability.

Thus we have an example of a non-trivial detection strategy that is faster than any PAC learning algorithm for that representation class. This shows that the learning-based detection strategy of Corollary 5.2 is not always best possible under computational constraints. Nevertheless, our example is somewhat contrived, and so in Section 6 we give an example of a more natural representation class for which defense is faster than learning.

5.3 Efficient defendability and obfuscation

We conclude our analysis of computational defendability with the following result, which essentially says that representation classes that are rich enough to support obfuscation are not efficiently defendable.

▶ **Theorem 5.5.** *Assuming* OWF *and* iO*, the representation class* $\mathcal{F}$ *of polynomial size Boolean circuits over* $\mathcal{X} = \{0,1\}^n$ *is not efficiently defendable.*

Here, OWF again denotes the existence of a one-way function, and iO denotes the existence of an efficient indistinguishability obfuscator. Roughly speaking, an efficient indistinguishability obfuscator is a probabilistic polynomial-time algorithm that takes in

a circuit and outputs an "obfuscated" circuit with the same behavior. The circuit being "obfuscated" means that the obfuscations of two different circuits with the same behavior are not distinguishable by any probabilistic polynomial-time adversary. For a precise definition of an efficient indistinguishability obfuscator, we refer the reader to Sahai and Waters [29, Section 3]. The study indistinguishability obfuscation was initiated by Barak et al. [1], and an efficient indistinguishability obfuscator was constructed from well-studied computational hardness assumptions by Jain, Lin and Sahai [17].

We use OWF in our proof of Theorem 5.5 to obtain a puncturable pseudorandom function. Roughly speaking, a puncturable pseudorandom function is pseudorandom function such that any key (i.e., seed) can be "punctured" at a set of points. The key being "punctured" means that, when run using the punctured key, the pseudorandom function behaves the same on unpunctured points, but to a probabilistic polynomial-time adversary with knowledge of the punctured key only, the function looks pseudorandom on punctured points when run using the original key. For a precise definition of a puncturable pseudorandom function, we again refer the reader to Sahai and Waters [29, Section 3]. The observation that puncturable pseudorandom functions can be constructed from one-way functions was made by Boneh and Waters [3], Boyle et al. [4] and Kiayias et al. [24].

Our proof of Theorem 5.5 is an example of a *hybrid argument*: we show that two distributions are computationally indistinguishable via a sequence of intermediate distributions. For a precise definition of computational indistinguishability and further discussion of hybrid arguments, we refer the reader to Katz and Lindell [20, Chapter 6.8]. The specific combination of a puncturable pseudorandom function and an efficient indistinguishability obfuscator has been used in a number of previous hybrid arguments, as described by Sahai and Waters [29].

With these preliminaries in place, we are now ready to prove Theorem 5.5.

Proof of Theorem 5.5. Take $\varepsilon = 2^{-n}$. We will show that, for any polynomial p, the representation class $\mathcal{F}$ of polynomial size Boolean circuits is not ε-defendable with confidence $\frac{1}{2} + \frac{1}{p(n)}$ using a detection strategy that runs in time polynomial in n.

To see this, by OWF, let $C_K \in \mathcal{F}$ be the circuit for a puncturable pseudorandom function with key $K \in \{0,1\}^n$ and a single output bit, and by iO, let $i\mathcal{O}$ be an efficient indistinguishability obfuscator. The adversary proceeds by taking $\mathcal{D}$ to be uniform over $\mathcal{X}$, and takes $f = i\mathcal{O}(C_K)$ for some $K \in \{0,1\}^n$ chosen uniformly at random. As before, we may treat f as if it were chosen randomly, since the detection strategy's worst-case performance can be no better than its average-case performance. Finally, given x^*, the adversary takes

$$f^* = i\mathcal{O}\left(x \mapsto \begin{cases} C_{\langle \textsc{Punc}(K,x^*)\rangle}(x), & \text{if } x \neq x^* \\ \langle 1 - C_K(x^*)\rangle, & \text{if } x = x^* \end{cases}\right),$$

where $\textsc{Punc}(K, x^*)$ punctures the key K at the point x^*, and angle brackets $\langle \dots \rangle$ indicate values that are hard-coded into the circuit, rather than being computed by the circuit. Note that f^* is ε-valid since it agrees with f on every point except x^*.

It remains to show that no polynomial-time detection strategy can win against this adversary with probability at least $\frac{1}{2} + \frac{1}{p(n)}$, which is equivalent to saying that (f, x) and (f^*, x^*) are computationally indistinguishable for $x \sim \mathcal{D}$. To see this, consider the intermediate circuit

$$\tilde{f} = i\mathcal{O}\left(x \mapsto \begin{cases} C_{\langle \textsc{Punc}(K,x^*)\rangle}(x), & \text{if } x \neq x^* \\ \langle C_K(x^*)\rangle, & \text{if } x = x^* \end{cases}\right),$$

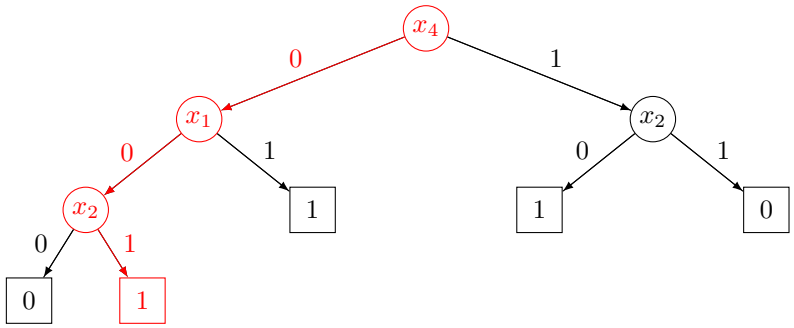


Figure 3 An example of a decision tree f over $\{0,1\}^4$ with a red path showing how $f(0110) = 1$.

which agrees with f everywhere. We claim that

$$(f, x) \overset{c}{\equiv} (f, x^*) \overset{c}{\equiv} (\tilde{f}, x^*) \overset{c}{\equiv} (f^*, x^*),$$

where $\overset{c}{\equiv}$ denotes computational indistinguishability of distributions. The first equivalence is simply an identity of distributions, since x^* was chosen randomly. The second equivalence follows by definition of indistinguishability obfuscation, with the technical detail that x^* is used as the "program state" that is passed from the circuit generator to the distinguisher. The third equivalence follows from puncturability: if $(\tilde{f}, x^*)$ and (f^*, x^*) were computationally distinguishable, then we could use this to computationally distinguish $C_K(x^*)$ from a uniformly random bit using only x^* and $\text{PUNC}(K, x^*)$. This final construction that relies on the fact that $i\mathcal{O}$ runs in polynomial time. By transitivity of computational indistinguishability, we have $(f, x) \overset{c}{\equiv} (f^*, x^*)$, as required. ◀

Thus even though all efficiently PAC learnable representation classes are efficiently defendable, some representation classes are not efficiently defendable due to the possibility of obfuscation.

6 Defendability of decision trees

In Section 5.2 we gave an example of a representation class for which defense is faster than learning, in the sense that it is efficiently defendable using a detection strategy that is faster than any PAC learning algorithm. However, our example was somewhat contrived, and was not polynomially evaluatable without access to a random oracle. In this section, we give an example of a more natural representation class for which defense is faster than learning: the class of polynomial size decision trees over $\{0,1\}^n$.

▶ **Definition.** A *decision tree* over $\{0,1\}^n$ is a rooted binary tree where each non-leaf node is labeled with one of the n input variables and each leaf node is labeled with a 0 or a 1. To evaluate a decision tree, we start at the root node and go left if the variable evaluates to 0 and right if it evaluates to 1, continuing until we reach a leaf. The *size* of a decision tree is its number of leaves.

An example of a decision tree and how to evaluate it is given in Figure 3.

In our study of decision trees, we focus on the *uniform-PAC* model of learning, in which the distribution $\mathcal{D}$ is always the uniform distribution. Thus we say that a representation class is *efficiently uniform-PAC learnable* to mean that it is efficiently PAC learnable as long as $\mathcal{D}$ is uniform. Similarly, we say that a representation class is *efficiently uniform-defendable* to mean that it is efficiently defendable as long as $\mathcal{D}$ is uniform.

As far as we are aware, it is unknown whether the representation class of polynomial size decision trees over $\{0,1\}^n$ is efficiently uniform-PAC learnable, although polynomial-time PAC learning algorithms are known for certain product distributions [19]. However, the class is efficiently uniform PAC-learnable if we allow membership queries (i.e., calls to an oracle that outputs the value of the function on an input of the algorithm's choice) [10, 27, Chapter 3.5]. By specializing the proof of Corollary 5.2, it follows that the representation class of polynomial size decision trees over $\{0,1\}^n$ is efficiently defendable, since the defender can efficiently compute the results of membership queries for themselves using the representation of the decision tree.

Nevertheless, these learning algorithms for decision trees generally use at least linearly many calls to the example or membership oracle. By contrast, we now show that there is a defense for decision trees that is much faster than this.

▶ **Theorem 6.1.** *The representation class $\mathcal{F}$ of decision trees over $\mathcal{X} = \{0,1\}^n$ of size at most s, where s is polynomial in n, is efficiently uniform-defendable using a detection strategy that makes 0 calls to the example oracle and runs in time O(time taken to evaluate a single decision tree).*

To prove this, we use a detection strategy that simply checks the depth of the leaf reached by the given input, and outputs Rej if this is too large. Roughly speaking, the reason this works is that, in order for the backdoored function to be ε-valid, the depth of the leaf reached by the backdoor trigger must be large for either the original function or the backdoored function. But since our trees have polynomially many leaves, this depth is unlikely to be large for the original function. Hence if the depth of the given function on the given input is large, then it is more likely that we have been given the backdoored function and the backdoor trigger.

For a full proof of Theorem 6.1, see Appendix D. We suspect that a similar approach would lead to fast detection strategies for other natural representation classes of piecewise constant functions, perhaps even in the non-uniform setting.

The detection strategy in this proof is not directly comparable to a learning algorithm, since it does not use the example oracle at all. However, if we treat a call to the example oracle as having similar computational cost to evaluating a single decision tree, then it is substantially faster than any possible learning algorithm for this representation class. Thus there is more to defense than only learning.

7 Discussion

7.1 Separating efficient defendability from efficient PAC learnability

A central theme of this work has been that learning can be used to perform backdoor defense, but backdoor defense cannot necessarily be used to perform learning. The first of these claims is encapsulated by Theorem 4.1 in the computationally unbounded setting and Corollary 5.2 in the computationally bounded setting. For the second of these claims, we have made several steps in this direction:

- We showed in Theorem 5.3 that efficient defendability does not imply efficient PAC learnability in the random oracle model of computation.
- We deduced from this that efficient defendability does not imply efficient PAC learnability in the usual model of computation either, as long as we allow representation classes that are not polynomially evaluatable, and we conjectured that there is a polynomially evaluatable counterexample assuming the existence of a one-way function.

- We showed in Theorem 6.1 that the representation class of polynomial size decision trees is efficiently uniform-defendable using a detection strategy that is faster than any possible learning algorithm.

Nevertheless, we still do not have an example of a natural representation class that is efficiently defendable but not efficiently PAC learnable. We consider finding such an example to be a central challenge for follow-up research. One possible candidate of independent interest is the representation class of shallow (i.e., logarithmic-depth) polynomial size Boolean circuits, or equivalently, polynomial size Boolean formulas.

▶ **Question 7.1.** Under reasonable computational hardness assumptions, is the representation class of polynomial size, logarithmic-depth Boolean circuits over $\{0,1\}^n$ efficiently defendable?

This representation class is known to not be efficiently PAC learnable under the assumption of the computational hardness of discrete cube roots [21, 22, Chapter 6].

7.2 Mechanistic defenses

Our detection strategy for decision trees in Theorem 6.1 is interesting not only because it is faster than any possible learning algorithm, but also because it works in a fundamentally different way. Given (f', x') as input, it does not run the decision tree f' on any inputs other than x' itself, but instead exploits the *mechanism* by which the value of $f'(x')$ is computed, by looking at the depth of the corresponding leaf. We call a defense that exploits the structure of f' in this kind of way *mechanistic*.

The existence of mechanistic defenses is another reason to suspect that there are other representation classes for which defense is strictly easier than learning. However, some sophistication may be required to construct such defenses. For example, one might hope to detect a backdoor trigger for a Boolean formula by checking the pattern of inputs to each gate, and seeing how unlikely that pattern would be for a random input to the formula. Unfortunately, though, the following example, found by Thomas Read, presents an obstacle to such an approach.

▶ **Example** (Backdoor with likely input patterns)**.** Consider the Boolean formulas $f, f^* : \{0,1\}^n \to \{0,1\}$ given by $f(x_1, \ldots, x_n) = x_n x_{n-1}$ and

$$f^*(x_1, \ldots, x_n) = (\ldots((x_1 \vee x_n x_2)\, x_2 \vee x_n x_3)\, x_3 \vee \cdots \vee x_n x_{n-1})\, x_{n-1},$$

where ab is shorthand for $(a \wedge b)$. By the distributive property, $f^*(x_1, \ldots, x_n)$ is logically equivalent to $x_1 x_2 \ldots x_{n-1} \vee x_n x_{n-1}$, and so f and f^* disagree only on the input $x^* := (1, \ldots, 1, 0)$. But on a uniformly random input to the formula, every gate in f^* receives every pattern of inputs with probability at least $1/8$. By contrast, in the logically equivalent version of f^*, the subformula $x_1 x_2 \ldots x_{n-1}$ is 1 on x^*, which only happens with probability $1/2^{n-1}$ on a uniformly random input.

This example suggests a possible mechanistic defense for Boolean formulas that involves first rewriting the formula. More generally, mechanistic defenses offer an exciting avenue for future research.

7.3 Implications for AI alignment

We are motivated to study backdoors as an analogy for *deceptive alignment*, the possibility that an advanced AI system would learn to behave cooperatively when there are subtle cues that it is in training, but uncooperatively when those cues are missing [16]. A deceptively

aligned model is analogous to a backdoored function in that it behaves similarly to a fully cooperative model except on certain inputs that are rare during training. Thus, in this analogy, $\mathcal{D}$ is the training distribution, f is the fully cooperative model, f^* is the deceptively aligned model, and x^* is an input on which f^* behaves uncooperatively.

Note that allowing the backdoor to be detected at runtime is appropriate in this analogy, because we have the opportunity to modify f^* or run a monitoring system. The flaw in the analogy comes from how the attacker is restricted. In our formal notion of defendability, the attacker must insert a backdoor that works for a randomly-chosen trigger. But in the case of deceptive alignment, there is no such restriction, and instead the backdoored behavior must arise from properties of the model architecture and the training distribution.

From this perspective, backdoor defenses that rely on learning are unsatisfying, because our assumption is that a similar process of learning gave rise to deceptive alignment in the first place. The detection strategy in our proof of Theorem 4.1 used distillation plus ensembling to produce a regularized model, but this may be exploiting the flaw in our analogy by using resampling to avoid the randomly-chosen trigger. Moreover, we would have to rely on a fast approximation to this strategy, since the exact version runs in exponential time. Similar issues apply to other methods of regularization, such as the one discussed in Appendix A, although that alternative does show more promise.

On the other hand, mechanistic defenses, as discussed in Section 7.2, may fare better, since they work very differently. Intuitively, if a detection strategy could spot the mechanism by which a deceptively aligned model concluded that it was in training, then it should transfer well to the case of deceptive alignment. A mechanistic defense may be able to do this by exploiting the fact that this mechanism is active on the backdoor trigger but inactive on most random inputs. Unfortunately though, we are lacking in examples of mechanistic defenses, which why we are excited to see more research in this direction.

Our result that polynomial size circuits are not efficiently defendable in Theorem 5.5 is also relevant to this analogy. Although it is unlikely that our indistinguishability obfuscator-based construction would arise out of ordinary model training, it is much more plausible for a trained model to be obfuscated in a more informal sense. Indeed, reverse engineering trained neural networks is an active area of research [28]. Hence the possibility of obfuscation poses a potential problem for detecting deceptive alignment. However, in the case of deceptive alignment, we also have access to the entire training process, including the training dataset, which may be enough information for us to detect the trigger despite any potential obfuscation. By analogy, in our construction in Theorem 5.5, it would be easy for the defender to detect the trigger if they had access to the unpunctured key K that was used in the construction of f^*.

This motivates the study of variants of our formal notion of defendability that constrain the attacker in different ways, or provide more assistance to the defender. To better capture the analogy with deceptive alignment, we would be excited to see research into variants that provide the defender with more information about how the function they are given was constructed, to see if this makes defense computationally feasible.

8 Conclusion

We have introduced a formal notion of defendability against backdoors in which the attacker's strategy must work for a randomly-chosen trigger. Despite its simplicity, this notion gives rise to a rich array of strategies. In the absence of computational constraints, defense is exactly as hard as learning. Meanwhile, in the presence of computational constraints, defense

is strictly easier than learning, but impossible for function classes that are rich enough to support obfuscation. We are excited to see future work that further explores the exact relationship between defense and learning.

References

1 Boaz Barak, Oded Goldreich, Rusell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *Annual international cryptology conference*, pages 1–18. Springer, 2001. doi:10.1007/3-540-44647-8_1.

2 Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989. doi:10.1145/76359.76371.

3 Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *Advances in Cryptology-ASIACRYPT 2013: 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II 19*, pages 280–300. Springer, 2013. doi:10.1007/978-3-642-42045-0_15.

4 Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *International workshop on public key cryptography*, pages 501–519. Springer, 2014. doi:10.1007/978-3-642-54631-0_29.

5 Sébastien Bubeck, Yin Tat Lee, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. In *International Conference on Machine Learning*, pages 831–840. PMLR, 2019. URL: http://proceedings.mlr.press/v97/bubeck19a.html.

6 Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pages 1310–1320. PMLR, 2019. URL: http://proceedings.mlr.press/v97/cohen19c.html.

7 Jacob Dumford and Walter Scheirer. Backdooring convolutional neural networks via targeted weight perturbations. In *2020 IEEE International Joint Conference on Biometrics (IJCB)*, pages 1–9. IEEE, 2020. doi:10.1109/IJCB48548.2020.9304875.

8 Sanjam Garg, Somesh Jha, Saeed Mahloujifar, and Mahmoody Mohammad. Adversarially robust learning could leverage computational hardness. In *Algorithmic Learning Theory*, pages 364–385. PMLR, 2020. URL: http://proceedings.mlr.press/v117/garg20a.html.

9 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986. doi:10.1145/6490.6503.

10 Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32, 1989. doi:10.1145/73007.73010.

11 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. IEEE, 2022.

12 Shafi Goldwasser, Jonathan Shafer, Neekon Vafa, and Vinod Vaikuntanathan. Oblivious defense in ML models: Backdoor removal without detection. *arXiv preprint*, 2024. arXiv:2411.03279.

13 Steve Hanneke, Amin Karbasi, Mohammad Mahmoody, Idan Mehalel, and Shay Moran. On optimal learning under targeted data poisoning. *Advances in Neural Information Processing Systems*, 35:30770–30782, 2022.

14 David Haussler, Nick Littlestone, and Manfred K Warmuth. Predicting {0, 1}-functions on randomly drawn points. *Information and Computation*, 115(2):248–292, 1994.

15 Sanghyun Hong, Nicholas Carlini, and Alexey Kurakin. Handcrafted backdoors in deep neural networks. *Advances in Neural Information Processing Systems*, 35:8068–8080, 2022.

16 Evan Hubinger, Chris van Merwijk, Vladimir Mikulik, Joar Skalse, and Scott Garrabrant. Risks from learned optimization in advanced machine learning systems. *arXiv preprint*, 2019. arXiv:1906.01820.

17 Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 60–73, 2021. doi:10.1145/3406325.3451093.

18 Jinyuan Jia, Xiaoyu Cao, and Neil Zhenqiang Gong. Intrinsic certified robustness of bagging against data poisoning attacks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35(9), pages 7961–7969, 2021. doi:10.1609/AAAI.V35I9.16971.

19 Adam Tauman Kalai and Shang-Hua Teng. Decision trees are PAC-learnable from most product distributions: a smoothed analysis. *arXiv preprint*, 2008. arXiv:0812.0933.

20 Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography: principles and protocols*. Chapman and hall/CRC, 2007.

21 Michael Kearns and Leslie Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM (JACM)*, 41(1):67–95, 1994. doi:10.1145/174644.174647.

22 Michael J Kearns and Umesh Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

23 Alaa Khaddaj, Guillaume Leclerc, Aleksandar Makelov, Kristian Georgiev, Hadi Salman, Andrew Ilyas, and Aleksander Madry. Rethinking backdoor attacks. In *International Conference on Machine Learning*, pages 16216–16236. PMLR, 2023. URL: https://proceedings.mlr.press/v202/khaddaj23a.html.

24 Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 669–684, 2013. doi:10.1145/2508859.2516668.

25 Alexander Levine and Soheil Feizi. Deep partition aggregation: Provable defense against general poisoning attacks. *arXiv preprint*, 2020. arXiv:2006.14768.

26 Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

27 Ryan O'Donnell. Analysis of Boolean functions. *arXiv preprint*, 2021. arXiv:2105.10386.

28 Chris Olah. Mechanistic interpretability, variables, and the importance of interpretable bases. *Transformer Circuits Thread*, 2022. URL: https://www.transformer-circuits.pub/2022/mech-interp-essay.

29 Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 475–484, 2014. doi:10.1145/2591796.2591825.

30 Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984. doi:10.1145/1968.1972.

A Alternative version of Theorem 4.1 using a Boltzmann posterior

In this section we prove a slightly weaker version of Theorem 4.1 using a detection strategy that does not involve learning. Instead of using distillation plus ensembling as in the proof of Theorem 4.1, we employ an alternative method of regularization that involves resampling the given function from a Boltzmann posterior. This gives rise to an extra logarithmic factor in the confidence of the detection algorithm, but otherwise gives the same bound, and in particular the threshold for detecting backdoors in a class of VC dimension d is still $\varepsilon = \Theta\left(\frac{1}{d}\right)$.

We are grateful to Dmitry Vaintrob for an earlier version of these results.

The remainder of this section has been omitted from this version due to the page limit. For the full version, see: https://arxiv.org/abs/2409.03077

B Proof of Theorem 4.1

▶ **Theorem 4.1.** *Let $\mathcal{F}$ be a representation class over $\{0,1\}^n$ and let $\varepsilon > 0$. Then the most (i.e., supremum) confidence with which $\mathcal{F}$ is ε-defendable is*

$$\max\left(\tfrac{1}{2}, 1 - \Theta\left(\mathrm{VC}\left(\mathcal{F}\right)\varepsilon\right)\right)$$

as $\mathrm{VC}\left(\mathcal{F}\right) \to \infty$.

Proof. Write $d = \mathrm{VC}\left(\mathcal{F}\right)$. We will show that, more specifically, for $d \geq 1$:

(a) For any detection strategy there is an adversary limiting their win probability to at most $\max\left(\frac{1}{2}, 1 - \frac{1}{2}\left(d-1\right)\varepsilon\right)$.

(b) There is a detection strategy that wins against any adversary with probability at least $\max\left(\frac{1}{2}, 1 - 15d\varepsilon\right)$.

For part a, let S be a set of size d shattered by $\mathcal{F}$, and let $s \in S$ be any point. The adversary proceeds by taking $\mathcal{D}$ to assign probability $\min\left(\frac{1}{d-1}, \varepsilon\right)$ to each point of $S \setminus \{s\}$ and the remaining $1-p$ probability to the point s, where $p := \min\left(1, \left(d-1\right)\varepsilon\right)$. The adversary then takes f to be one of the 2^{d-1} witnesses for the shattering that assigns 0 to the point s. We may treat f as if it were chosen uniformly at random, since the detection strategy's worst-case performance can be no better than its average-case performance. Finally, given x^*, the adversary takes f^* to be the witness that agrees with f exactly on the set $S \setminus \{x^*\}$ if $x^* \neq s$, and otherwise takes $f^* = f$. It is easy to check that f^* is ε-valid if and only if $x^* \neq s$. Furthermore, (f, x) and (f^*, x^*) are identically distributed for $x \sim \mathcal{D}$. Hence the best a detection strategy can do is to output Acc when given the point s, and otherwise the strategy can do no better than chance. So the detection strategy's overall win probability is at most $(1-p) + \frac{p}{2} = 1 - \frac{p}{2}$, as required.

For part b, note first that the detection strategy can achieve a win probability of at least $\frac{1}{2}$ simply by guessing randomly, so we may assume that $1 - 15d\varepsilon > \frac{1}{2}$. To construct the detection strategy, by Haussler–Littlestone–Warmuth, take a prediction strategy for $\mathcal{F}$ with sample size $m-1$ and error rate at most $\frac{d}{m}$, for some positive integer m to be chosen later. The detection strategy then works as follows:

- Given (f', x') as input (which could be either (f, x) or (f^*, x^*)), we use the prediction strategy to make a prediction z for $f'\left(x'\right)$, which we in turn think of as a prediction for $f\left(x'\right)$. The idea is that if we can successfully predict $f\left(x'\right)$, then we can distinguish the two cases by comparing this to $f'\left(x'\right)$, since $f\left(x\right) = f\left(x\right)$ but $f\left(x^*\right) \neq f^*\left(x^*\right)$ if f^* is ε-valid.
- We repeat this procedure r times with independent samples to obtain predictions $z^{(1)}, \ldots, z^{(r)}$ for some positive integer r to be chosen later.
- Finally, we take a majority vote: if more than half of $z^{(1)}, \ldots, z^{(r)}$ are different from $f'\left(x'\right)$, then we output Rej, and otherwise we output Acc.

To lower bound the detection strategy's win probability, consider first the case in which (f, x) is passed to the detection strategy. In this case the probability that $z \neq f\left(x\right)$ is at most the error rate $\frac{d}{m}$. Hence by Markov's inequality and linearity of expectation, the probability that more than half of $z^{(1)}, \ldots, z^{(r)}$ are different from $f\left(x\right)$ is at most $\frac{2d}{m}$, giving the detection strategy a failure probability of at most $\frac{2d}{m}$. If instead (f^*, x^*) is passed to the detection strategy, then there are two possible reasons why we could have $z \neq f\left(x^*\right)$: either f and f^* disagree on at least one of the $m-1$ samples provided by the example oracle, or the prediction strategy fails to predict $f\left(x^*\right)$ even if it is provided with the value of f on all

of these $m-1$ samples. Write $\mathbb{1}^{(i)}_{f \text{ and } f^* \text{ disagree}}$ and $\mathbb{1}^{(i)}_{\text{prediction fails}}$ for the indicator functions of these two events respectively during the ith trial for $i \in \{1, \ldots, r\}$. Then the probability that the detection strategy fails is

$$\mathbb{P}\left(\frac{1}{r}\sum_{i=1}^{r} \mathbb{1}_{z^{(i)} \neq f(x^*)} \geq \frac{1}{2}\right) \leq \mathbb{P}\left(\frac{1}{r}\sum_{i=1}^{r} \mathbb{1}^{(i)}_{f \text{ and } f^* \text{ disagree}} \geq \frac{1}{4}\right) + \mathbb{P}\left(\frac{1}{r}\sum_{i=1}^{r} \mathbb{1}^{(i)}_{\text{prediction fails}} \geq \frac{1}{4}\right).$$

The second term on the right-hand side is at most $\frac{4d}{m}$ by another application of Markov's inequality and linearity of expectation. To bound the first term on the right-hand side, we may assume without loss of generality that f^* is ε-valid, which implies that $p := \mathbb{P}\left(\mathbb{1}^{(i)}_{f \text{ and } f^* \text{ disagree}} = 1\right) \leq (m-1)\,\varepsilon$, by the union bound. Taking m to be the unique positive integer with $\frac{1}{5\varepsilon} < m \leq \frac{1}{5\varepsilon} + 1$, we have $p \leq \frac{1}{5}$. Since these r indicator functions are independent, the first term on the right-hand side is at most $\exp\left(-2\left(\frac{1}{4} - \frac{1}{5}\right)^2 r\right) = \exp\left(-\frac{r}{200}\right)$ by Hoeffding's inequality, which can be made arbitrarily small by taking r to be sufficiently large. Putting everything together, the detection strategy's best overall win probability is at least

$$\tfrac{1}{2}\left(\left(1 - \tfrac{2d}{m}\right) + \left(1 - \tfrac{4d}{m}\right)\right) = 1 - \tfrac{3d}{m} > 1 - 15\varepsilon d,$$

as required. ◀

C Proof of Theorem 5.3

▶ **Theorem 5.3.** *In the random oracle model of computation, with probability $1 - O(2^{-n^c})$ for all c over the choice of random oracle, there is a polynomially evaluatable representation class over $\mathcal{X} = \{0,1\}^n$ that is efficiently defendable but not efficiently PAC learnable.*

Proof. Write $\textsc{Rand} : \{0,1\}^* \to \{0,1\}$ for the random oracle. Let $\mathcal{K} = \{0,1\}^n$ be a set of "keys", and let $\mathcal{F} = \{x \mapsto \textsc{Rand}(K \parallel x) \mid K \in \mathcal{K}\}$, where $\parallel$ denotes string concatenation. We will show that, over the choice of random oracle, $\mathcal{F}$ is efficiently defendable with probability $1 - o(2^{-n^c})$ as $n \to \infty$ for all $c \geq 1$, but not efficiently PAC learnable with probability $1 - o\left(\exp\left(-2^{n-4}\right)\right)$ as $n \to \infty$. The result then follows by the union bound.

For defendability, let $\delta > 0$ and $c \geq 1$. We will show that, with probability $1 - o(2^{-n^c})$ over the choice of random oracle, if $\varepsilon = o\left(\frac{\delta}{n^c}\right)$ then $\mathcal{F}$ is ε-defendable with confidence $1 - \delta$ using the trivial detection strategy that always outputs $\textsc{Acc}$. To see this, let $\mathcal{D}$ be the distribution over $\mathcal{X}$ chosen by the adversary, and for $x \in \mathcal{X}$ write $\mathbb{P}_{\mathcal{D}}(x)$ for $\mathbb{P}_{x' \sim \mathcal{D}}(x' = x)$. In order for the adversary to be able to choose a backdoored function that is ε-valid with high enough probability over the choice of backdoor trigger, the adversary must choose $\mathcal{D}$ so that

$$S_1 := \sum_{\substack{x \in \mathcal{X} \\ \mathbb{P}_{\mathcal{D}}(x) \leq \varepsilon}} \mathbb{P}_{\mathcal{D}}(x) > 2\delta.$$

Note also that

$$S_2 := \sum_{\substack{x \in \mathcal{X} \\ \mathbb{P}_{\mathcal{D}}(x) \leq \varepsilon}} \mathbb{P}_{\mathcal{D}}(x)^2 \leq \varepsilon S_1.$$

Now for any $K, K^* \in \mathcal{K}$ and any $\varepsilon < \frac{1}{2}S_1$,

$$\begin{aligned}
&\mathbb{P}_{\text{RAND}}\left(\mathbb{P}_{x\sim D}\left(\text{RAND}\left(K \parallel x\right) \neq \text{RAND}\left(K^* \parallel x\right)\right) \leq \varepsilon\right) \\
&\leq \mathbb{P}_{\text{RAND}}\left(\sum_{\substack{x\in\mathcal{X} \\ \mathbb{P}_{\mathcal{D}}(x)\leq\varepsilon}} \mathbb{P}_{\mathcal{D}}\left(x\right) \mathbb{1}_{\text{RAND}(K\parallel x)\neq\text{RAND}(K^*\parallel x)} \leq \varepsilon\right) \\
&\leq \exp\left(-\frac{2\left(\frac{1}{2}S_1 - \varepsilon\right)^2}{S_2}\right) && \text{(by Hoeffding's inequality)} \\
&\leq \exp\left(-\frac{S_1}{2\varepsilon} - \frac{2\varepsilon}{S_1} + 2\right) && \text{(since } S_2 \leq \varepsilon S_1\text{)} \\
&< \exp\left(-\frac{\delta}{\varepsilon} - 2\varepsilon + 2\right) && \text{(since } 2\delta < S_1 \leq 1\text{)}.
\end{aligned}$$

Hence by the union bound, it is impossible for the adversary to choose any $K, K^* \in \mathcal{K}$ with $\mathbb{P}_{x\sim D}\left(\text{RAND}\left(K \parallel x\right) \neq \text{RAND}\left(K^* \parallel x\right)\right) \leq \varepsilon$ with probability at least

$$1 - \binom{|\mathcal{K}|}{2}\exp\left(-\frac{\delta}{\varepsilon} - 2\varepsilon + 2\right) > 1 - \exp\left((2n-1)\log\left(2\right) - \frac{\delta}{\varepsilon} - 2\varepsilon + 2\right).$$

When this event happens the detection strategy always wins, and if $\varepsilon = o\left(\frac{\delta}{n^c}\right)$ then this probability is $1 - o(2^{-n^c})$, as required.

For PAC learnability, we will show that, with probability $1 - o\left(\exp\left(-2^{n-4}\right)\right)$ over the choice of random oracle, there is no PAC learning algorithm for $\mathcal{F}$, even with access to the random oracle, that runs in time $f\left(n\right) = 1.5^n$ with confidence parameter $\frac{1}{4}$ and error parameter $\frac{1}{4}$, say. To see this, note that, for all sufficiently large n, there are fewer than $2^{f(n)}$ possible hypotheses that such an algorithm could output, assuming by convention that algorithms can process at most one bit per unit time. Now for each of these hypotheses h, if $\mathcal{D}$ is the uniform distribution over $\mathcal{X}$ and $K \in \mathcal{K}$, then

$$\mathbb{P}_{\text{RAND}}\left(\mathbb{P}_{x\sim D}\left(\text{RAND}\left(K \parallel x\right) \neq h\left(x\right)\right) \leq \tfrac{1}{4}\right) \leq \exp\left(-2\left(\tfrac{1}{2} - \tfrac{1}{4}\right)^2 |\mathcal{X}|\right) = \exp\left(-2^{n-3}\right)$$

by Hoeffding's inequality. Hence by the union bound, the probability that none of these hypotheses has generalization error at most $\frac{1}{4}$ is at least

$$1 - 2^{f(n)}\exp\left(-2^{n-3}\right) = 1 - \exp\left(1.5^n\log\left(2\right) - 2^{n-3}\right) = 1 - o\left(\exp\left(-2^{n-4}\right)\right).$$

When this event happens the PAC learning algorithm fails with probability $1 > \frac{1}{4}$, as required. ◀

D Proof of Theorem 6.1

▶ **Theorem 6.1.** *The representation class $\mathcal{F}$ of decision trees over $\mathcal{X} = \{0,1\}^n$ of size at most s, where s is polynomial in n, is efficiently uniform-defendable using a detection strategy that makes 0 calls to the example oracle and runs in time $O\left(\text{time taken to evaluate a single decision tree}\right)$.*

Proof. Given $f \in \mathcal{F}$ and $x \in \mathcal{X}$, write $\text{DEPTH}\left(f, x\right)$ for the number of distinct input variables that appear along the path from root to leaf when f is evaluated at x.

Let $\delta > 0$, and take $\varepsilon < \frac{\delta^2}{s^2}$. We claim that, as long as the adversary takes $\mathcal{D}$ to be uniform distribution, $\mathcal{F}$ is ε-defendable with confidence $1-\delta$ using the detection strategy that, given (f', x') as input, outputs Rej if $2^{-\textsc{Depth}\left(f', x'\right)} \leq \frac{\delta}{s}$ and Acc otherwise. This suffices, since evaluating $\textsc{Depth}\left(f', x'\right)$ takes the same time as evaluating $f'\left(x'\right)$, up to a constant factor.

To see this, given $f \in \mathcal{F}$ and $x \in \mathcal{X}$, write

$$\textsc{Leaf}\left(f, x\right) = \left\{x' \in \mathcal{X} \mid x' \text{ and } x \text{ reach the same leaf of } f\right\},$$

and observe that

$$\mathbb{P}_{\mathcal{D}}\left(\textsc{Leaf}\left(f, x\right)\right) = 2^{-\textsc{Depth}(f,x)},$$

where we have written $\mathbb{P}_{\mathcal{D}}\left(S\right)$ as shorthand for $\mathbb{P}_{x\sim\mathcal{D}}\left(x \in S\right)$. The key claim is that, for any $f \in \mathcal{F}$ and any $x^* \in \mathcal{X}$, if $f^* \in \mathcal{F}$ is ε-valid, then

$$2^{-\textsc{Depth}(f,x^*)-\textsc{Depth}(f^*,x^*)} \leq \mathbb{P}_{\mathcal{D}}\left(\textsc{Leaf}\left(f, x^*\right) \cap \textsc{Leaf}\left(f^*, x^*\right)\right) \leq \varepsilon.$$

The first inequality is a consequence of the fact that the number of distinct variables that appear when either f or f^* is evaluated at x^* is at most $\textsc{Depth}\left(f, x^*\right) + \textsc{Depth}\left(f^*, x^*\right)$, and makes essential use of the fact that $\mathcal{D}$ is uniform (or at the very least, a product distribution). The second inequality follows from the definition of ε-valid.

Now choose f, x^*, f^* and x as in the backdoor detection game. Suppose first that (f, x) is passed to the detection strategy. Then with probability at least $1-\delta$, we have $2^{-\textsc{Depth}(f,x)} = \mathbb{P}_{\mathcal{D}}\left(\textsc{Leaf}\left(f, x\right)\right) > \frac{\delta}{s}$, since f has at most s leaves. Hence the detection strategy outputs Acc with probability at least $1-\delta$. Now suppose instead that (f^*, x^*) is passed to the detection strategy. Then with probability at least $1-\delta$, we have $2^{-\textsc{Depth}(f,x^*)} > \frac{\delta}{s}$, by a similar argument. But if f^* is ε-valid, then this implies that $2^{-\textsc{Depth}(f^*,x^*)} < \varepsilon/\frac{\delta}{s} < \frac{\delta}{s}$, by the key claim. Hence the detection strategy outputs Rej with probability at least $1-\delta$. So the detection strategy's overall win probability is at least $1-\delta$, as required. ◀