A Cryptographic Perspective on Mitigation vs. Detection in Machine Learning

Greg Gluch[1] and Shafi Goldwasser[2]

[1] Simons Institute for the Theory of Computing, Berkeley, CA, USA
[2] University of California at Berkeley, CA, USA
{gluch,shafi.goldwasser}@berkeley.edu

Abstract. In this paper, we initiate a cryptographically inspired theoretical study of *detection* versus *mitigation* of adversarial inputs produced by attackers of Machine Learning algorithms during inference time.

We formally define *defense by detection* (DbD) and *defense by mitigation* (DbM). Our definitions come in the form of a 3-round protocol between two resource-bounded parties: a trainer/defender and an attacker. The attacker aims to produce inference-time inputs that fool the training algorithm. We define correctness, completeness, and soundness properties to capture successful defense at inference time while not degrading (too much) the performance of the algorithm on inputs from the training distribution.

We first show that achieving DbD and achieving DbM are equivalent for ML classification tasks. Surprisingly, this is not the case for ML generative learning tasks, where there are many possible correct outputs that can be generated for each input. We show a separation between DbD and DbM by exhibiting a generative learning task for which is possible to defend by mitigation but is provably impossible to defend by detection under the assumption that the Identity-Based Fully Homomorphic Encryption (IB-FHE), publicly-verifiable zero-knowledge Succinct Non-Interactive Arguments of Knowledge (zk-SNARK) and Strongly Unforgeable Signatures exist. The mitigation phase uses significantly fewer samples than the initial training algorithm.

1 Introduction

With the meteoric rise of machine learning (ML) classification algorithms some years ago, a central question emerged: what guarantees can be made on robustness at inference time, when the training-time data distribution is different from the inference-time data distribution. A vast literature emerged to study how to achieve robustness against inference-time adversarial examples chosen to fool ML algorithms. Although some techniques were developed to defend against certain

classes of adversarial examples, worst-case adversarially chosen inference-time examples are essentially unavoidable.[3]

The problem posed by adversarial examples and the challenge of robustness takes on a new and different form in the setting of *generative AI* where many answers are possible per input (aka prompt). In the LLM realm, one talks of achieving *alignment* by not answering prompts that do not align with human goals (e.g. using offensive language, asking for dangerous information, etc.) rather than robustness. And instead of adversarial examples, one talks about *jailbreaks*: adversarially designed inference-time prompts that attempt to avoid alignment. ML conferences are full of proposals for new types of jailbreaks aimed at existing LLMs, followed by attempts to construct new attacks that defeat these jailbreaks. The tension between striving for alignment versus constructing jailbreaks is reminiscent of the old-age battle between proposing cryptographic schemes which are subsequently broken by cryptanalysis.

Two natural questions emerge:

(1) Even if adversarial examples against classification ML algorithms are provably impossible to detect, can we **mitigate** their effect?

(2) Moving to generative ML algorithms, even if jailbreaks are provably impossible to detect, can we **mitigate** their effect?

We note that an influential approach to robustness without detection of adversarially chosen inputs was proposed by Cohen et al. [2019] who suggested *smoothing*: rather than using the model's answer to a possibly adversarial input in inference time, one should query the model on a few perturbations of the input and answer according to the majority answers. Experimentally, this was shown quite effective for learning tasks that obeyed certain Lipshitz smoothness conditions. A negative answer to the first question was given for the bounded-perturbations model in Tramer [2022]. More recently, it was provably shown [Goldwasser et al., 2024] how to *mitigate* (even the effect of a possible undetectable backdoor as in Goldwasser et al. [2022]) by exploiting local self-correction properties of the ground truth for linear and low degree polynomial regression tasks.[4]

---

[3] A particularly devastating attack against robustness was shown [Goldwasser et al., 2022] using cryptographic constructions. Under the assumption that CMA-secure digital signatures exist (i.e one-way functions exist) a malicious trainer can *undetectably* plant backdoors (by adding a public signature verification procedure) into any neural network so that at inference time *any* input can be slighly modified to become an adversarial input (by embedding within it a small digital signature). However, it is important to note that undetectability holds in the black-box model.

[4] It is important to note that Goldwasser et al. [2024] and ours consider different models. In our work, the party that returns the answer is the same as the one that produced the model, and aims to compute a model without introducing backdoors. In contrast, in Goldwasser et al. [2024], the model may be produced by a different party, one that is actively attempting to insert backdoors. This key difference explains why the conclusions of the two works are compatible.

Our Contributions

In this paper we initiate a theoretical study of the *detection versus mitigation problem*, using cryptographic paradigms and tools.

We begin by proposing a formal definition of *defense by detection* (DbD) (closely related to a model from Goldwasser et al. [2020]) and *defense by mitigation* (DbM). Our definitions are cryptographically inspired and come in the form of a 3-round interaction between a polynomial-time bounded trainer/defender party and an attacker party all of which have access to random i.i.d samples of a learning task distribution $\mathbb{L} = \{(x, y)\}$. The trainer/defender is composed of two algorithms *(Train, Defend)*, where *Train* produces an initial model f for the learning task which it sends to the attacker, the attacker produces a challenge input[5] $\hat{x}$ and finally the *Defend* algorithm either labels $\hat{x}$ as adversarial or provides an answer $\hat{y}$ to $\hat{x}$. We restrict our attention to *Defend* and attacker algorithms which use significantly less resources than the *Train* algorithm. In this work, we focus on the resource which is the number of samples of the learning task used.

We define correctness, completeness, and soundness properties with the interaction, to capture successful defense at inference time while not degrading the performance on inputs drawn from the training distributions. *Correctness* requires that the ML algorithm f produced by the trainer be accurate in the inputs drawn from the training distribution. *Completeness* requires that inputs x drawn from the training distribution are not labeled as adversarial by the defender with high probability. *DbD-soundness* requires that, for all time bounded attackers, either f is correct on $\hat{x}$ or $\hat{x}$ is labeled as suspicious, whereas *DbM-soundness* requires that, for all time-bounded attackers, either *Defend* computes a correct $\hat{y}$ on $\hat{x}$ or $\hat{x}$ is labeled as suspicious. We first show that achieving DbD and achieving DbM are equivalent for classification learning tasks.[6] The main technical result is that DbD and DbM are not equivalent for ML generative learning tasks. We show the latter by exhibiting a generative learning task for which it is possible to defend by mitigation but is provably impossible to defend by detection under the assumption that Identity-Based Fully Homomorphic Encryption (IB-FHE) [Gentry et al., 2013], publicly-verifiable zero-knowledge Succinct Non-Interactive Arguments of Knowledge (zk-SNARKS) [Bitansky et al., 2022], and Strong Unforgeable Signatures [Boneh et al., 2006] exist.

We remark that, interestingly, the choices we make in designing our DbM correspond to current trends in industry agendas which aim at alignment and defense against jailbreaks as discussed by Zaremba et al. [2025]. Namely, there is a current shift from dedicating computation resources to training time to dedicating computation resources to inference time in order to address alignment and safety questions. In this spirit, we show that using fresh, unseen during training time, data during inference time can help with robustness.

---

[5] We simplified the exposition here by implicitly assuming that the batch of inputs contains exactly one x, i.e., $q = 1$ in definitions of DbD and DbM.

[6] We remark that in our paper we assume the trainer is honest. We do not address the backdoor model where a trainer may be an adversary, in which case achieving DbD and DbM are not clearly equivalent for classification tasks.

2   Technical Overview

Let us denote the sample space by $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the input space and $\mathcal{Y}$ is the output space. An *error oracle* is a function $h : \mathcal{Z} \to \{0,1\}$, i.e., it is a function that takes a pair (x,y) of input x and output y, and outputs 0 if y is a valid answer for x, and 1 otherwise. Note that this captures both the classification and generation settings, i.e., if for every x there is exactly one y such that $h(x,y) = 0$ then h corresponds to a classification task, but if there are many such y's we deal with generation.
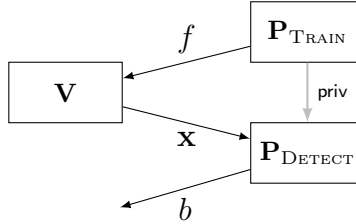
A *model* is a function $f : \mathcal{X} \to \mathcal{Y}$, which can be randomized. We don't make any assumption on the structure of f, e.g., f might be an LLM but any other architecture is captured by our theory. For a distribution $\mathcal{D}$ over $\mathcal{Z}$ we denote it's marginal distribution over $\mathcal{X}$ as $\mathcal{D}_{\mathcal{X}}$. For a distribution $\mathcal{D}$ over $\mathcal{Z}$ and an error oracle $h : \mathcal{Z} \to \{0,1\}$ we define the error of a model $f : \mathcal{X} \to \mathcal{Y}$ as $\mathsf{err}(f) := \mathbb{E}_{x \leftarrow \mathcal{D}_{\mathcal{X}}}\left[h(x, f(x))\right]$, where the randomness of expectation includes the potential randomness of f. For $q \in \mathbb{N}$ and $\mathbf{x} \in \mathcal{X}^q, \mathbf{y} \in \mathcal{Y}^q$ we also define the empirical error as $\mathsf{err}(\mathbf{x}, \mathbf{y}) := \frac{1}{q} \sum_{i \in [q]} h(x_i, y_i)$, where x_i is the i-th coordinate vector $\mathbf{x}$.

Learning task. A learning task $\mathbb{L}$ is a distribution over $(\mathcal{D}, h)$ pairs, where $\mathcal{D}$ is a distribution over $\mathcal{Z}$ and $h : \mathcal{Z} \to \{0,1\}$.[7]

Learning process and interaction structure. All defenses in this paper are with respect to some learning task $\mathbb{L}$. Before the interaction between a learner/defender and an attacker starts, $(\mathcal{D}, h)$ is sampled from $\mathbb{L}$. We assume that all parties learner, defender, and attacker have oracle access to i.i.d. samples from $\mathcal{D}$.[8] But they differ in how many samples they are allowed to draw. In particular the trainer will receive more samples than the attacker and the defender. $\mathbb{L}$ is known to all the parties, and, intuitively, represents prior knowledge about the learning task, but the particular $(\mathcal{D}, h)$ is unknown, i.e., it can be accesed *only* through drawing samples from $\mathcal{D}$.

Next, we give an informal definition of a Defense by Detection (DbD). See Definition 2 for a formal version.

Defense by Detection (DbD), informal. Let $\mathbb{L}$ be a learning task. Defense by Detection (DbD) is a pair of algorithms $\mathbf{P} = (\mathbf{P}_{\text{TRAIN}}, \mathbf{P}_{\text{DETECT}})$. The interaction between $\mathbf{P}$ and an attacker $\mathbf{V}$ proceeds as follows

---

[7] Note that $\mathbb{L}$ is *not* a fixed $(\mathcal{D}, h)$ pair.

[8] Note that the samples from $\mathcal{D}$ are "labeled", i.e. $\mathcal{D}$ produces (x,y) pairs, where $x \in \mathcal{X}, y \in \mathcal{Y}$, and not only x.

where $f : \mathcal{X} \to \mathcal{Y}, \mathbf{x} \in \mathcal{X}^q, b \in \{0,1\}$ and priv is the private information that $\mathbf{P}_{\text{TRAIN}}$ might provide to $\mathbf{P}_{\text{DETECT}}$ to aid it's detection. We assume that $\mathbf{P}_{\text{TRAIN}}, \mathbf{P}_{\text{DETECT}}$, and $\mathbf{V}$ are polynomial time algorithms in a security parameter. We denote by $K_{\text{TRAIN}}, K_{\text{DETECT}}, K_{\text{ATTACK}}$ the sample complexity bounds for $\mathbf{P}_{\text{TRAIN}}, \mathbf{P}_{\text{DETECT}}$ and $\mathbf{V}$ respectively. A succesful DbD satisfies:

– **Correctness:** f has a low error, i.e.,

$$\Pr_{f \leftarrow \mathbf{P}_{\text{TRAIN}}} \left[\mathsf{err}(f) \leq \epsilon \right] \geq 1 - \delta.$$

– **Completeness:** Training inputs are not flagged as adversarial, i.e.,

$$\Pr_{\substack{(f,\mathsf{priv}) \leftarrow \mathbf{P}_{\text{TRAIN}}, x \leftarrow \mathbf{V}(f), \\ b \leftarrow \mathbf{P}_{\text{DETECT}}(f,\mathbf{x},\mathsf{priv})}} \left[b = 0 \right] \geq 1 - \delta.$$

– **Soundness:** Adversarial inputs are **detected**, i.e., for every polynomial-size $\mathbf{V}$
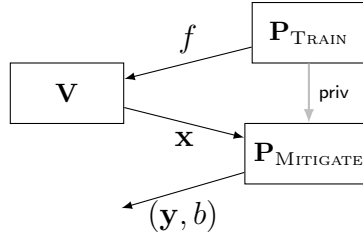
$$\Pr_{\substack{(f,\mathsf{priv}) \leftarrow \mathbf{P}_{\text{TRAIN}}, x \leftarrow \mathbf{V}(f), \\ b \leftarrow \mathbf{P}_{\text{DETECT}}(f,\mathbf{x},\mathsf{priv})}} \left[\mathsf{err}(\mathbf{x}, f(\mathbf{x})) \leq \epsilon \ \text{ or } \ b = 1 \right] \geq 1 - \delta.$$

Now imagine a different version of a defense where $\mathbf{P}$ is required to return $(\mathbf{y}, b)$, where $\mathbf{y} \in \mathcal{Y}^q, b \in \{0,1\}$ resulting in the following soundness condition. See Definition 3 for a formal version.

Defense by Mitigation (DbM), informal. Let $\mathbb{L}$ be a learning task. Defense by Detection (DbD) is a pair of algorithms $\mathbf{P} = (\mathbf{P}_{\text{TRAIN}}, \mathbf{P}_{\text{MITIGATE}})$ that is a DbD, where the interaction is instead as follows:

where $\mathbf{y} \in \mathcal{Y}^q$ and the soundness is replaced by:

– **Soundness:** Adversarial inputs are **detected or mitigated**, i.e., for every polynomial-size $\mathbf{V}$, we have

$$\Pr_{\substack{(f,\mathsf{priv}) \leftarrow \mathbf{P}_{\text{TRAIN}}, x \leftarrow \mathbf{V}(f), \\ (\mathbf{y},b) \leftarrow \mathbf{P}_{\text{MITIGATE}}(f,\mathbf{x},\mathsf{priv})}} \left[\mathsf{err}(\mathbf{x}, \mathbf{y}) \leq \epsilon \ \text{ or } \ b = 1 \right] \geq 1 - \delta.$$

We denote by K_{MITIGATE} the sample complexity bound for $\mathbf{P}_{\text{MITIGATE}}$.

Discussion. An interesting question is that of _resource allocation_ between the training and inference phases. In a DbM defense, the prover $\mathbf{P}$ may choose how to divide its sample budget between $\mathbf{P}_{\mathrm{TRAIN}}$ and $\mathbf{P}_{\mathrm{MITIGATE}}$, corresponding to resources used during training and inference, respectively.

Note that for any pair $(\mathbf{P}_{\mathrm{TRAIN}}, \mathbf{P}_{\mathrm{MITIGATE}})$ using K_{TRAIN} and K_{MITIGATE} samples, we can define a new pair $(\mathbf{P}'_{\mathrm{TRAIN}}, \mathbf{P}'_{\mathrm{MITIGATE}})$ that achieves the same guarantees using $K_{\mathrm{TRAIN}} + K_{\mathrm{MITIGATE}}$ samples in training and none at inference. Specifically, $\mathbf{P}'_{\mathrm{TRAIN}}$ simulates $\mathbf{P}_{\mathrm{TRAIN}}$, draws K_{MITIGATE} fresh samples from $\mathcal{D}$, and stores them in the private string priv; $\mathbf{P}'_{\mathrm{MITIGATE}}$ then executes $\mathbf{P}_{\mathrm{MITIGATE}}$ using these pre-sampled examples. Since $\mathbf{P}'_{\mathrm{MITIGATE}}$ draws no samples directly, we reassign all sample complexity to $\mathbf{P}'_{\mathrm{TRAIN}}$.

While such a transformation is always possible, we argue that charging the K_{MITIGATE} samples to $\mathbf{P}_{\mathrm{MITIGATE}}$ better reflects real-world usage. The key is that these K_{MITIGATE} samples are not used in the training of f, and they remain hidden from the verifier $\mathbf{V}$. In practice, drawing new data at inference time is conceptually and operationally distinct from drawing data at training time.

Another perspective is to interpret $\mathbf{P}_{\mathrm{MITIGATE}}$ as computing a new model f' using fresh data $S \sim \mathcal{D}^{K_{\mathrm{MITIGATE}}}$ and applying it to the batch $\mathbf{x}$: that is, $f' \leftarrow \mathbf{P}_{\mathrm{MITIGATE}}(f, \mathbf{x}, \mathsf{priv}, S)$ and $\mathbf{y} = f'(\mathbf{x})$. This view connects naturally to the notion of _transferable attacks_: a DbM does not exist if and only if there is an attack that _transfers_ from any f trained with K_{TRAIN} samples to any f' trained with $K_{\mathrm{TRAIN}} + K_{\mathrm{MITIGATE}}$ samples. We elaborate on this connection in Section 4.

Remark 1. Our definition of defense by detection (DbD) is essentially equivalent to the definition of robustness to arbitrary test-time examples proposed in Goldwasser et al. [2020], with one key difference: their model permits the learner to abstain from classifying any $x \in \mathcal{X}$.

To show equivalence, one can convert their abstention-based defense into a DbD algorithm by setting $b = 1$ whenever the fraction of abstentions exceeds a fixed threshold. Conversely, given a DbD, we can simulate abstention in their model by choosing to abstain on all inputs whenever $b = 1$. These reductions preserve correctness and soundness up to constant factors in δ and ϵ.

We believe our formulation is more natural for two reasons. First, it shifts the focus from detecting individual adversarial inputs to detecting adversarial _interactions_—a perspective that aligns more closely with the conceptual foundations of interactive proofs and cryptographic indistinguishability. Second, our framework foregrounds computational constraints as a central resource, whereas in Goldwasser et al. [2020] these were treated as secondary.

Remark 2. We note that our definitions are general and allow for $q > 1$. Larger batches have been crucially used in other works. For example in Goldwasser et al. [2020] the larger batches are _necessary_ to fight against distribution shifts.

Increasing q makes attacks harder. For instance, an adversarial input $x \in \mathcal{X}$ cannot simply be repeated to form $x = (x, x, \ldots, x) \in \mathcal{X}^q$, as such repetitions are easily detected. This makes the attack surface more constrained.

A further technical benefit of allowing $q > 1$ is the ability to decouple ϵ from δ. When $q = 1$, the soundness condition $\mathrm{err}(x, f(x)) < \epsilon$ is effectively equivalent to

requiring $h(x, f(x)) = 0$, so the error and soundness bounds align and typically force $\delta > \epsilon$. For larger q, the empirical error enables smaller δ.

Mitigation vs. Detection in Classification. The following result, with a formal version stated in Lemma 1, shows that detection and mitigation are equivalent for classification tasks.

(Lemma) For every classification learning task, the existence of a Defense by Mitigation is equivalent to the existence of a Defense by Detection.

Sketch. Having a DbD we can define a DbM by setting $\mathbf{y} = f(\mathbf{x})$, simulating the DbD to obtain b and returning $(\mathbf{y}, b)$. This algorithm constitutes a DbM because DbD guarantees that whenever f makes errors it is signaled by b, so $\mathbf{y} = f(\mathbf{x})$ "never" makes an error when $b = 1$.[9]

On the other hand, having a DbM we can define a DbD by simulating the DbM and returning $b = 1$ if DbM returned 1 or if $\mathbf{y}$ returned by DbM has a big Hamming distance from $f(\mathbf{x})$. This works because DbM guarantees that, in particular, if $\mathbf{x}$ is such that $\mathsf{err}(\mathbf{x}, f(\mathbf{x})) \gg 0$ then $\mathbf{x}$ is labeled as suspicious by DbM or $\mathsf{err}(\mathbf{x}, \mathbf{y}) \ll 1$. In the second case, it implies that the Hamming distance between $f(\mathbf{x})$ and $\mathbf{y}$ is large, and so $\mathbf{x}$ will be labeled as suspicious.   ☐   ◼

Remark 3. Our reduction crucially uses the fact that we consider classification tasks. Indeed, for generative tasks with many valid outputs for a given input, this DbD would likely label as suspicious almost all $\mathbf{x}$'s. This is because two different generative models likely return different answers for a fixed input.

Remark 4. A related result was shown in the bounded-perturbation model by Tramer [2022]. Their reduction, however, is computationally inefficient, while our equivalence preserves both sample and time complexity. Moreover, our framework does not require an explicit bound on the inputs $\mathbf{x}$.

Mitigation vs. Detection for Generation. The following is the main result of the paper exhibiting a generative learning task for which DbM is *necessary* to achieve "robustness", in contrast to DbD. For a formal version see Theorem 1.

(Theorem) There exists a generative learning task such that for every sample complexity upper-bound K, we have that

- *There exists a DbM with $K_{T_{RAIN}} = K$, $K_{M_{ITIGATE}} = O(\sqrt{K})$ and $K_{A_{TTACK}} = o(K)$.[10]*
- *There is no DbD with $K_{T_{RAIN}} = K$ even if $K_{D_{ETECT}} = \mathsf{poly}(K)$ and $K_{A_{TTACK}} = O(1)$.*

---

[9] We note that the reduction from DbM to DbD works for generative tasks also.

[10] As we noted earlier, the theorem also holds in the setting where $K_{\mathrm{TRAIN}} = K + O(\sqrt{K})$ and $K_{\mathrm{MITIGATE}} = 0$, if the sample acquisition originally attributed to $\mathbf{P}_{\mathrm{MITIGATE}}$ is instead performed by $\mathbf{P}_{\mathrm{TRAIN}}$. While this reallocation strengthens the formal result, we believe it obscures the conceptual message of the theorem.

Remark 5. Note that it is impossible to DbD even when the attacker utilizes a constant number of samples. And yet the DbM is possible by a mitigation algorithm which is much more sample-efficient than the training algorithm.

Sketch.. The informal idea is to define a learning task $\mathbb{L}^{\mathsf{data}}$ with a distribution $\mathcal{D}$ whose inputs $\mathcal{D}_\mathcal{X}$ are partitioned into different levels of difficulty and a valid output for an input on level k will be an input on level at least $k + \lfloor \sqrt{k} \rfloor$. Furthermore, for every $k \in \mathbb{N}$, $O(k)$ samples will be enough to compute outputs for inputs up to level k, while k samples are necessary to produce outputs for level k. This would imply that for any model f trained with k samples, an attacker, in time $O(\sqrt{k})$, could find an adversarial example x, which f cannot answer. The attacker simply calls f on itself, i.e., $x = \underbrace{f(\ldots(f(x'))}_{O(\sqrt{k})}$.

To realize the above proof template, in our concrete construction, we will utilize zk-SNARKs, Fully Homomorphic Encryption (FHE), and Strongly Unforgeable Signatures (SUS) - signature schemes where an adversary cannot produce additional signatures of messages even given older signatures.

A first attempt, containing most of the ideas of the proof, at defining the learning task $\mathbb{L}^{\mathsf{data}}$ is as follows. Let $\mathcal{X} = \mathcal{Y}$ so that inputs and outputs live in the same space. We let distribution $\mathcal{D}$ be decomposed of two equally probable parts

$$\mathcal{D} = \frac{1}{2}\mathcal{D}_{\mathsf{Clear}} + \frac{1}{2}\mathcal{D}_{\mathsf{Enc}}.$$

Every $(x, y) \in \mathsf{supp}(\mathcal{D}_{\mathsf{Clear}})$ are strings such that

$$x = a\|k\|\pi, y = a\|k + \lfloor\sqrt{k}\rfloor\|\pi',$$

where a is a valid signature of $m = 0$, i.e., $\mathsf{Verify}(\mathsf{pk}, m = 0, a) = 1$, $k \in \mathbb{N}$, π is a publicly-verifiable zk-SNARK that there exists k different signatures of $m = 0$, and π' is an zk-SNARK that there exists $\lfloor\sqrt{k}\rfloor$ different signatures of $m = 0$. The probability of (x, y) being sampled is $\mathsf{Geom}(\frac{1}{2})$ in k, uniform in signatures, and uniform over valid proofs. We say x has hardness level k, if it decomposes to $x = a\|k\|\pi$.

We say that y is incorrect on x if x or y contain invalid proofs or signatures or the hardness level of y is smaller than $k + \lfloor\sqrt{k}\rfloor$, where k is the hardest level of x.

$\mathcal{D}_{\mathsf{Enc}}$ is $\mathcal{D}$ under FHE encryption, i.e., it is defined by the process: $(x, y) \sim \mathcal{D}_{\mathsf{Clear}}$, return $(\mathsf{Encrypt}(x), \mathsf{Encrypt}(y))$.

We now show that defense by Mitigation (DbM) for $\mathbb{L}^{\mathsf{data}}$ exists. To construct a DbM $\mathbf{P}_{\mathrm{TRAIN}}$ computes a "weak" classifier f that answers correctly for levels up to K. Next, upon receiving an input x, $\mathbf{P}_{\mathrm{MITIGATE}}$, updates f to a stronger classifier f^{strong} by drawing $O(\sqrt{K})$ fresh samples not used in the training of f, and answers with $f^{\mathrm{strong}}(x)$.

By the properties of the learning task if the attacker $\mathbf{V}$ used $o(K)$ samples then $\mathbf{P}_{\mathrm{MITIGATE}}$ will produce a valid output. Intuitively, this follows since with

$o(K)$ samples and access to f the hardest input $\mathbf{V}$ can produce is on level $K + \lfloor\sqrt{K}\rfloor$. In more detail, there seem to be only two natural attacks $\mathbf{V}$ can mount. In the first attack, she can draw $o(K)$ samples from $\mathcal{D}$ and create an input with hardness level $o(K)$ by producing a zk-SNARK proof using all the samples. This is not a successful attack because f^{strong} answers correctly for levels up to $K + O(\sqrt{K})$. In the second attack, she can try to "extract" hard inputs from f. But f has only inputs with hardness level at most $K + \lfloor\sqrt{K}\rfloor$ "inside its code"!

We now would like to show that Defense by Detection (DbD) for $\mathbb{L}^{\mathsf{data}}$ does not exist. To attack, $\mathbf{V}$ upon receiving f computes a valid input x_1 on level 1 and iterates $x_i = f(x_{i-1})$ until f no longer produces a valid output for x_i. Then she sends as adversarial input $x = \mathsf{Encrypt}(x_i)$ with probability $\frac{1}{2}$ or a fresh sample from the marginal of $\mathcal{D}_{\mathsf{Clear}}$ otherwise. Note that this input is indistinguishable from a sample from $\mathcal{D}_\mathcal{X}$ by the properties of FHE. Nonetheless, we would like to argue that f makes an error on x.

Unfortunately, we cannot quite yet guarantee that f would make an error on x. It is possible that $\mathbf{P}_{\mathrm{TRAIN}}$ could have provided f that is weaker on $\mathcal{D}_{\mathsf{Clear}}$ than it is on $\mathcal{D}_{\mathsf{Enc}}$! After all, this is "how" DbM works, i.e., $\mathbf{P}_{\mathrm{MITIGATE}}$ "hides" from $\mathbf{V}$ a stronger model than f, with which he answers. Maybe, $\mathbf{P}_{\mathrm{TRAIN}}$ did "hide", e.g., by obfuscating, a stronger model on the $\mathcal{D}_{\mathsf{Enc}}$ part.

To circumvent this issue $\mathbf{V}$ needs to generate x in $\mathcal{D}_{\mathsf{Enc}}$ such that $h(x, f(x)) = 1$ (i.e, for which f is incorrect). Unfortunately, it is not feasible to evaluate $h(x, f(x))$ because x is encrypted. To solve this issue we use a special FHE (modifying $\mathbb{L}^{\mathsf{data}}$ slightly) which will allow the parties partial decryption access and the evaluation of $h(x, f(x))$. With this final change, $\mathbf{V}$ can generate a hard and indistinguishable input for $\mathbf{P}_{\mathrm{DETECT}}$. For details please see the full proof in Section 9. $\qquad\square$ $\qquad\qquad\square$

3   Related Work

Robustness and Adversarial Examples. The field of adversarial robustness has a rich and extensive literature [Szegedy et al., 2014, Gilmer et al., 2018, Raghunathan et al., 2018, Wong and Kolter, 2018, Engstrom et al., 2017].

For classification tasks, techniques such as adversarial training [Madry et al., 2018], which involves training the model on adversarial examples, have shown empirical promise in improving robustness. Another empirically successful inference-time method for improving robustness is *randomized smoothing* [Cohen et al., 2019]. Certified defenses [Raghunathan et al., 2018] provide provable guarantees, ensuring robustness within a specified perturbation bound. A setup that goes beyond the standard threat model of bounded perturbations was considered in [Goldwasser et al., 2020]. The authors show a defense against *all inference-time examples* for bounded VC-dimension classes provided that the learner can abstain. The setup of [Goldwasser et al., 2020] is, see Remark 1, equivalent to our notion od DbD.

Alignment and Jailbreaking. In the context of LLMs, *alignment* aims at making sure that the models act consistently with human goals, e.g., by not answering dangerous prompts or not using offensive language, etc. There is a fast-growing literature about trying to achieve alignment [Amodei et al., 2016, Greenblatt et al., 2024, Ji et al., 2023]. In consequence, there is a lot of research on trying to avoid alignment by designing *jailbreaks* [Andriushchenko et al., 2024, Chao et al., 2023, Mehrotra et al., 2024, Wei et al., 2023], which involves crafting prompts that bypass filters and protections embedded in the models. There is also some interest in adversarial examples for LLMs [Zou et al., 2023, Carlini et al., 2023, Wen et al., 2023].

Transferability of Adversarial Examples. One of the most intriguing facts about adversarial examples is their *transferability*, i.e., adversarial examples crafted for one model often transfer to other architectures or training sets. This phenomenon was discovered early and studied extensively [Goodfellow et al., 2014, Papernot et al., 2016] and more recently was observed in the context of LLMs [Zou et al., 2023]. In Ilyas et al. [2019] it is hypothesized that adversarial examples exist because of highly predictive but non-robust features existing in the data, thus explaining transferability.

4   Connections to Transferable Attacks

A *non-transferable* adversarial attack is an attack that is effective for a specific f but not for other models from the same class, which are trained with a similar amount of resources but on a potentially different training set. Intuitively, one might expect most adversarial attacks to be non-transferable. However, as discussed in Section 3, adversarial attacks are often found to be transferable in practice. Surprisingly, to the best of our knowledge, no theoretical construction has demonstrated a learning task where only non-transferable attacks exist.[11]

This observation leads to a somewhat unintuitive conjecture:

(Conjecture) If a learning task has an adversarial attack, then it also has a transferable attack.

The conjecture has significant implications for the problem of robustness. If true, it suggests that every learning task is either defensible against adversarial examples or fundamentally vulnerable. In particular, if humans can be modeled as resource-bounded learners, they too must be susceptible to adversarial attacks. Notably, adversarial examples capable of fooling time-limited humans have been demonstrated [Elsayed et al., 2018].

To formalize this conjecture, one must specify the resource constraints for each party and define the class of models for which transferability should hold. Our main result refutes the conjecture under a specific formalization, but it leaves open the possibility that the conjecture holds under a different, perhaps more natural, formulation.

---

[11] For a definition of transferability where the resources of parties are properly bounded.

Reinterpreting our theorem, we show that there exists a learning task where:

1. There exists an efficient attack that is effective against every model f trained with K samples.
2. No such attack transfers to models trained with slightly more data—specifically, models f' trained with $K + O(\sqrt{K})$ samples.

Consequently, our result disproves the conjecture if transferability is required for models that are only slightly stronger than f. However, the attack from our construction *does* transfer to models trained with K samples. Thus, if the conjecture is interpreted as requiring transferability only among models with the same resource constraints—a seemingly more natural requirement—then our result does not disprove it. Recently, it was shown [Głuch et al., 2024] that every learning task has at least one of: a transferable attack, a defense, and a watermarking scheme (a process of embedding hidden, detectable signals within the model's outputs to verify the source of the generated text). It would be interesting to see if the methods from Głuch et al. [2024] can be generalized to show the conjecture.

5   Preliminaries

For $n \in \mathbb{N}$ we define $[n] := \{0, 1, \ldots, n-1\}$. For $q \in \mathbb{N}$, space $\mathcal{X}$ and vectors $\mathbf{x}, \mathbf{x}' \in \mathcal{X}^q$ we denote the normalized Hamming distance between two vectors by $d(\mathbf{x}, \mathbf{x}') := |\{i \in [q] \mid x_i \neq x_i'\}|/q$. We say a function $\eta : \mathbb{N} \to \mathbb{N}$ is negligible if for every polynomial $p : \mathbb{N} \to \mathbb{N}$ it holds that $\lim_{n\to\infty} \eta(n) \cdot p(n) = 0$.

Learning. For a set Ω, we write $\Delta(\Omega)$ to denote the set of all probability measures defined on the measurable space $(\Omega, \mathcal{F})$, where $\mathcal{F}$ is some fixed σ-algebra that is implicitly understood.

We denote the sample space by $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the input space and $\mathcal{Y}$ is the output space. An *error oracle* is a function $h : \mathcal{Z} \to \{0, 1\}$.[12] A *model* is a function (potentially randomized) $f : \mathcal{X} \to \mathcal{Y}$. For a distribution $\mathcal{D} \in \Delta(\mathcal{Z})$ we denote it's marginal distribution over $\mathcal{X}$ as $\mathcal{D}_\mathcal{X} \in \Delta(\mathcal{X})$. For a distribution $\mathcal{D} \in \Delta(\mathcal{Z})$ and an error oracle $h : \mathcal{Z} \to \{0, 1\}$ we define the error of a model $f : \mathcal{X} \to \mathcal{Y}$ as

$$\mathsf{err}(f) := \mathbb{E}_{x \sim \mathcal{D}_\mathcal{X}}\Big[h(x, f(x))\Big],$$

where the randomness of expectation includes the potential randomness of f.

Definition 1 (*Learning Task*). *A* learning task $\mathbb{L}$ *is a family* $\{\mathbb{L}_n\}_{n\in\mathbb{N}}$, *where for every* n, $\mathbb{L}_n$ *is an element of* $\Delta\big(\Delta(\mathcal{Z}_n) \times \{0,1\}^{\mathcal{Z}_n}\big)$, *i.e., it is a distribution over* $(\mathcal{D}_n, h_n)$ *pairs, where* $\mathcal{D}_n \in \Delta(\mathcal{Z}_n)$ *and* $h_n : \mathcal{Z}_n \to \{0, 1\}$.

We say that $\mathbb{L}$ *is* efficiently representable *if there exist polynomials* p, q, *such that for every* n, $\mathcal{X}_n = \{0,1\}^{p(n)}, \mathcal{Y}_n = \{0,1\}^{q(n)}$. *All learning tasks in this work will be efficiently representable and so we will often drop it.*

---

[12] We assumed for simplicity that the range of h is $\{0, 1\}$ but we believe that most of our results extend to, e.g., regression, where h maps for instance to $[0, 1]$.

Note 1. We note that Definition 1, which is similar to the one used in Gluch et al. [2024], is nonstandard. One difference from more common definitions is that we consider a sequence of learning tasks indexed by the size parameter n. This is needed to measure the computational resources of learning algorithms properly. The outer distribution, that is, the distribution over $(\mathcal{D}_n, h_n)$ pairs, represents the prior knowledge of the learner. The reason we had to restrict the support of allowed $(\mathcal{D}_n, h_n)$ pairs is that the hypothesis classes we will consider do not have a bounded VC-dimension but are still learnable for some class of distributions.

Definition 1 rather than requiring learnability for every domain distribution, it allows adaptation to a fixed distribution over $(\mathcal{D}_n, h_n)$ pairs, effectively incorporating a prior. This idea parallels the PAC-Bayes framework McAllester [1999], which uses priors over hypotheses to obtain strong generalization guarantees where standard PAC learning may fail. Related extensions, incorporating priors over distributions and sample sizes, have been explored in meta- and transfer learning Rothfuss et al. [2020], Amit and Meir [2018].

Unlike distribution-specific or restricted family models Kalai et al. [2008], Feldman et al. [2006], our definition does not constrain the support. While standard PAC learning demands generalization over all domain distributions, it often fails to explain the behavior of complex models like deep neural networks Zhang et al. [2021], Nagarajan and Kolter [2019].

Interaction. For a *message space* $\mathcal{M} = \{\mathcal{M}_n\}_n = \{\{0,1\}^{m(n)}\}_n$, where m is some polynomial, a *representation class* is a collection of mappings $\{\mathcal{R}_n\}_n$, where for every n, $\mathcal{R}_n : \mathcal{M}_n \to \Delta(\mathcal{Y}_n)^{\mathcal{X}_n}$, i.e., it is a mapping from the space of messages to probabilistic functions $\mathcal{X}_n \to \mathcal{Y}_n$ Thus, there is a function class corresponding to a representation, i.e., for every n there is a function class $\mathcal{F}_n$, which is an image of $\mathcal{R}_n$. All function classes considered in this work have an implicit representation class and an underlying message space.

6   Formal Definitions

Definition 2 (*Defense by Detection*). *Let $\mathbb{L}$ be a learning task. A* Defense by Detection *(DbD) is a pair of algorithms* $\mathbf{P} = (\mathbf{P}_{T_{RAIN}}, \mathbf{P}_{D_{ETECT}})$. *We assume that* $\mathbf{P}_{T_{RAIN}}, \mathbf{P}_{D_{ETECT}}$, *and* $\mathbf{V}$ *run in time polynomial in* $n, \log(\frac{1}{\epsilon}), \frac{1}{\delta}$. *We denote by* $K_{T_{RAIN}}, K_{D_{ETECT}}, K_{A_{TTACK}} : \mathbb{N} \times (0,1)^2 \to \mathbb{N}$ *the sample complexity bounds for* $\mathbf{P}_{T_{RAIN}}, \mathbf{P}_{D_{ETECT}}$ *and* $\mathbf{V}$ *respectively. A succesful DbD is such that for every* $\epsilon, \delta : \mathbb{N} \to (0, \frac{1}{2})$, *where* $\log(\frac{1}{\epsilon}), \frac{1}{\delta}$ *are upper-bounded by a polynomial in* n, *for every* n:

– **Correctness** *(f has low error).*

$$\Pr\left[\mathsf{err}_{\mathcal{D},h}(f) \leq \epsilon \,\middle|\, \begin{array}{l} (\mathcal{D}, h) \leftarrow \mathbb{L}_n \\ f \leftarrow \mathbf{P}^{\mathcal{D}}_{T_{RAIN}}(\epsilon, \delta) \end{array}\right] \geq 1 - \delta,$$

where $f \in \mathcal{F}_n$.[13]

---

[13] Formally $f_n \in \mathcal{M}_n$ and the model it represents is $\mathcal{R}_n(f_n)$.

– **Completeness** (*natural inputs are not flagged as adversarial*).

$$\Pr\left[b = 0 \,\middle|\, \begin{array}{l} (\mathcal{D}, h) \leftarrow \mathbb{L}_n \\ (f, \mathsf{priv}) \leftarrow \mathbf{P}^{\mathcal{D}}_{T_{RAIN}}(\epsilon, \delta) \\ \mathbf{x} \leftarrow (\mathcal{D}_{\mathcal{X}})^q \\ b \leftarrow \mathbf{P}^{\mathcal{D}}_{D_{ETECT}}(\mathsf{priv}, \mathbf{x}, \epsilon, \delta) \end{array}\right] \geq 1 - \delta - \mathsf{negl}(n),$$

where $\mathbf{x} \in \mathcal{X}^q, b \in \{0, 1\}$ and q is some polynomial in $n, \log(\frac{1}{\epsilon}), \frac{1}{\delta}$.

– **Soundness** (*adversarial inputs are* **detected**). *For every polynomial-size* $\mathbf{V}$

$$\Pr\left[\mathsf{err}_{\mathcal{D}, h}(\mathbf{x}, f(\mathbf{x})) > \epsilon \text{ and } b = 0 \,\middle|\, \begin{array}{l} (\mathcal{D}, h) \leftarrow \mathbb{L}_n \\ (f, \mathsf{priv}) \leftarrow \mathbf{P}^{\mathcal{D}}_{T_{RAIN}}(\epsilon, \delta) \\ \mathbf{x} \leftarrow \mathbf{V}^{\mathcal{D}}(f, \epsilon, \delta) \\ b \leftarrow \mathbf{P}^{\mathcal{D}}_{D_{ETECT}}(\mathsf{priv}, \mathbf{x}, \epsilon, \delta) \end{array}\right] \leq \delta + \mathsf{negl}(n).$$

For simplicity of notation, we omitted the dependence of ϵ, δ *on* n.

Definition 3 (*Defense by Mitigation*). *Let* $\mathbb{L}$ *be a learning task. A pair of algorithms* $\mathbf{P} = (\mathbf{P}_{T_{RAIN}}, \mathbf{P}_{M_{ITIGATE}})$ *is a* Defense by Mitigation *(DbM) if it is a Defense by Detection, where Soundness is replaced by: for every* $\epsilon, \delta : \mathbb{N} \to (0, \frac{1}{2})$, *where* $\log(\frac{1}{\epsilon}), \frac{1}{\delta}$ *are upper-bounded by a polynomial in* n, *for every* n:

– **Soundness** (*adversarial inputs are* **detected or mitigated**). *For every polynomial-size* $\mathbf{V}$

$$\Pr\left[\mathsf{err}_{\mathcal{D}, h}(\mathbf{x}, \mathbf{y}) > \epsilon \text{ and } b = 0 \,\middle|\, \begin{array}{l} (\mathcal{D}, h) \leftarrow \mathbb{L}_n \\ (f, \mathsf{priv}) \leftarrow \mathbf{P}^{\mathcal{D}}_{T_{RAIN}}(\epsilon, \delta) \\ \mathbf{x} \leftarrow \mathbf{V}^{\mathcal{D}}(f, \epsilon, \delta) \\ (\mathbf{y}, b) \leftarrow \mathbf{P}^{\mathcal{D}}_{M_{ITIGATE}}(\mathsf{priv}, \mathbf{x}, \epsilon, \delta) \end{array}\right] \leq \delta + \mathsf{negl}(n),$$

where $\mathbf{y} \in \mathcal{Y}^q$.

Remark 6. Note that we require the running time of $\mathbf{P}_{T_{RAIN}}, \mathbf{P}_{D_{ETECT}}, \mathbf{P}_{M_{ITIGATE}}$ to be polynomial in $\log(\frac{1}{\epsilon})$ and not in $\frac{1}{\epsilon}$ as often defined in computational learning theory. For instance the sample complexity of learning a class of VC-dimension d is $d \cdot (\frac{1}{\epsilon} + \log(\frac{1}{\delta}))$. This implies that defenses, and thus also learning algorithms, are required to run faster (use fewer samples) than what VC theory guarantees. In other words, we focus on a regime of small ϵ. Interestingly, in the so-called Equivalence Query (EQ) model, where samples are acquired through an interaction between a teacher and a learner, where the teacher provides counterexamples to hypotheses given by the learner $d \cdot \mathsf{polylog}(\frac{1}{\epsilon})$ samples suffices as shown in Gluch and Urbanke [2021]. In the same work, the authors provide some connections of this result to adversarial robustness.

Remark 7. Note that in both definitions we allow $\mathbf{P}_{T_{RAIN}}$ to produce, during learning, a secret piece of data priv that is not sent to $\mathbf{V}$. Later it will turn out that this is key in obtaining an efficient mitigation procedure.

7 Mitigation vs. Detection for Classification

Lemma 1 (Mitigation vs. Detection for Classification). *For every classification learning task, the existence of a Defense by Mitigation is equivalent to the existence of a Defense by Detection.*

Proof. Let $\mathbb{L}$ be a learning task, K_{ATTACK} a sample complexity bound, $\epsilon, \delta \in (0, 1)$.

Assume $(\mathbf{P}_{\text{TRAIN}}, \mathbf{P}_{\text{DETECT}})$ is a Defense by Detection for $\mathbb{L}$. Let $\mathbf{P}_{\text{MITIGATE}}$ be an algorithm that simulates $b = \mathbf{P}_{\text{DETECT}}$ and returns $(f(\mathbf{x}), b)$. Then $(\mathbf{P}_{\text{TRAIN}}, \mathbf{P}_{\text{MITIGATE}})$ is a Defense by Mitigation. Indeed, all three probabilities in correctness, completeness, and soundness are identical by construction.

Assume $(\mathbf{P}_{\text{TRAIN}}, \mathbf{P}_{\text{MITIGATE}})$ is a Defense by Mitigation for $\mathbb{L}$. Let $\mathbf{P}_{\text{DETECT}}$ be an algorithm that simulates $(\mathbf{y}, b) = \mathbf{P}_{\text{MITIGATE}}$ and returns $b' = 1$ if $b = 1$ or the normalized Hamming distance of $\mathbf{y}$ and $f(\mathbf{x})$ is large, i.e., $d(\mathbf{y}, f(\mathbf{x})) > 4\epsilon$, where $d(\mathbf{y}, \mathbf{y}') := |\{i \in [q] \mid y_i \neq y_i'\}|/q$.

Then $(\mathbf{P}_{\text{TRAIN}}, \mathbf{P}_{\text{DETECT}})$ is a Defense by Detection. Indeed, correctness transfers directly. Combining the soundness with completeness and the union bound we get

$$\Pr_{\mathbf{x} \leftarrow \mathcal{D}^q}[\text{err}(\mathbf{x}, \mathbf{y}) \leq 2\epsilon] \geq 1 - 4\delta. \tag{1}$$

By correctness and the Chernoff bound we get

$$\Pr_{\mathbf{x} \leftarrow \mathcal{D}^q}[\text{err}(\mathbf{x}, f(\mathbf{x})) \leq 2\epsilon] \geq 1 - 2\delta. \tag{2}$$

To show completeness we bound

$$\Pr_{\mathbf{x} \leftarrow \mathcal{D}^q}[b' = 1]$$
$$= \Pr_{\mathbf{x} \leftarrow \mathcal{D}^q}[b = 1 \text{ or } d(\mathbf{y}, f(\mathbf{x})) > 4\epsilon]$$
$$= \Pr_{\mathbf{x} \leftarrow \mathcal{D}^q}[b = 1] + \Pr_{\mathbf{x} \leftarrow \mathcal{D}^q}[d(\mathbf{y}, f(\mathbf{x})) > 4\epsilon]$$
$$\leq 2\delta + 2\delta + 4\delta, \qquad\qquad \text{Correctness, (1) and (2).} \tag{3}$$

where in the last step we crucially used that if $\text{err}(\mathbf{x}, \mathbf{y}) \leq 2\epsilon$ and $\text{err}(\mathbf{x}, f(\mathbf{x})) \leq 2\epsilon$ then $d(\mathbf{y}, f(\mathbf{x})) \leq 4\epsilon$. Equation (3) guarantees completeness with confidence $1 - 8\delta$.

Remark 8. Notice that to obtain equation (3) we used the fact that $\mathbb{L}$ is a classification task. Indeed, if it was a generative task then having two sets of outputs $\mathbf{y}_1, \mathbf{y}_2 \in \mathcal{Y}^q$ and knowing both have a low error we could not deduce anything about $d(\mathbf{y}_1, \mathbf{y}_2)$!

To show soundness we bound

$$\Pr[\text{err}(\mathbf{x}, f(\mathbf{x})) > 7\epsilon \text{ and } b' = 0]$$
$$= \Pr[\text{err}(\mathbf{x}, f(\mathbf{x})) > 7\epsilon \text{ and } b = 0 \text{ and } d(\mathbf{y}, f(\mathbf{x})) \leq 4\epsilon]$$
$$\leq \Pr[\text{err}(\mathbf{x}, \mathbf{y}) > 3\epsilon \text{ and } b = 0]$$
$$\leq 2\delta \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{By soundness.}$$

Remark 9. Note that when reducing a Defense by Detection to a Defense by Mitigation we lost multiplicative factors in probabilities and error bounds, e.g., for soundness we considered an event $\{\mathsf{err}(\mathbf{x}, f(\mathbf{x})) > 7\epsilon$ and $b' = 0\}$ and not $\{\mathsf{err}(\mathbf{x}, f(\mathbf{x})) > 2\epsilon$ and $b' = 0\}$.

$\square$

8   Learning Task $\mathbb{L}^{\mathsf{data}}$

In this section, we define the learning task that will be used to prove Theorem 1.

Defining learning task $\mathbb{L}^{\mathsf{data}}$. Let $\mathsf{IB} - \mathsf{FHE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$ be an Identity-Based Fully Homomorphic Encryption (Definition 11) for the class of polynomially-sized circuits, $(\mathcal{P}, \mathcal{G}, \mathcal{V})$ be a publicly verifiable zk-SNARK (Definition 9), and $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ be a Strongly Unforgeable Singature scheme (Definition 7). See Gentry et al. [2013], Bitansky et al. [2022], Boneh et al. [2006] for constructions.

For $n \in \mathbb{N}$ we define the learning task $\mathbb{L}^{\mathsf{data}}_n$ as the result of the following procedure: $\mathbb{L}^{\mathsf{data}}_n =$

$$
\left\{ (\mathcal{D}_{\mathsf{sk},\mathsf{pk},y,\mathsf{PP}_{\mathsf{fhe}},\mathsf{MSK},\mathsf{PP}_{\mathsf{snark}}}, h_{\mathsf{sk},\mathsf{pk},y,\mathsf{PP}_{\mathsf{fhe}},\mathsf{MSK},\mathsf{PP}_{\mathsf{snark}}}) \; \middle| \; \begin{array}{l} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^n) \\ (\mathsf{PP}_{\mathsf{fhe}}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^n) \\ \mathsf{PP}_{\mathsf{snark}} \leftarrow \mathcal{G}(1^n, T = \mathrm{bin}(2^n)) \end{array} \right\},
$$

where $\mathrm{bin}(2^n)$ specifies that 2^n is presented in binary.[14]

Next, we define $\mathcal{D}_{\mathsf{sk},\mathsf{pk},y,\mathsf{PP},\mathsf{MSK},\mathsf{PP}'}$. For the rest of this proof we omit the subscripts of $\mathcal{D}$. $\mathcal{D}$ will have to equally probable parts $\mathcal{D}_{\mathsf{Clear}}$ and $\mathcal{D}_{\mathsf{Enc}}$, i.e.,

$$
\mathcal{D} = \frac{1}{2}\mathcal{D}_{\mathsf{Clear}} + \frac{1}{2}\mathcal{D}_{\mathsf{Enc}}.
$$

Let $\mathcal{X} = \mathcal{Y} = \{0,1\}^{\mathsf{poly}(n)}$, i.e. the input and output space of $\mathcal{D}$ are equal.

Defining $\mathcal{D}_{\mathsf{Clear}}$. For every $(x,y) \in \mathsf{supp}(\mathcal{D}_{\mathsf{Clear}})$ both x and y are of the form

$$
a\|k\|\pi\|0^i,
$$

i.e. they are a concatenation of a, k, π and a padding 0^i, separated by a special character $\|$. Moreover, a is in the space of potential signatures, $k \in \mathbb{N}$, and π is a bitstring of some length.

Additionally, for every $(x,y) \in \mathsf{supp}(\mathcal{D}_{\mathsf{Clear}})$ we have

$$
x = a\|k\|\pi\|0^i, y = a\|(k + \lfloor\sqrt{k}\rfloor)\|\pi'\|0^{i'},
$$

and $\mathsf{Verify}(\mathsf{pk}, m = 0, a) = 1$, π is a zk-SNARK proof that $m = 0$ has k signatures. Additionally, π' is a zk-SNARK proof that $m = 0$ has $k + \lfloor\sqrt{k}\rfloor$ signatures.

---

[14] We chose $T = 2^n$ to allow for all (for a large enough n) polynomial-time computations to be provable by the SNARK.

Let $\mathsf{Geom}(\frac{1}{2})$ be a geometric distribution, i.e. a distribution that samples $k \in \mathbb{N}$ with probability 2^{-k}. For $k \in \mathbb{N}$ let M_k be an NP machine that accepts as a potential witness k elements from the domain of potential signatures, i.e. $(a_i)_{[k]}$, and accepts iff all a_i's are pairwise different and for every $i \in [k]$ we have that $\mathsf{Verify}(\mathsf{pk}, m = 0, a) = 1$.

Let

$$
\mathcal{D}_{\mathsf{Clear}} = \left\{ (x, y) \middle|
\begin{array}{l}
a \leftarrow \mathsf{Sign}(\mathsf{sk}, m = 0) \\
k \leftarrow \mathsf{Geom}(\frac{1}{2}) \\
(a_i)_{[k]} \leftarrow \mathsf{Sign}(\mathsf{sk}, m = 0) \\
\pi \leftarrow \mathcal{P}\left((M_k, \bot, t), (a_i)_{[k]}, \mathsf{PP}_{\mathsf{snark}}\right) \\
(a'_i)_{[k+\lfloor \sqrt{k} \rfloor]} \leftarrow \mathsf{Sign}(\mathsf{sk}, m = 0) \\
\pi' \leftarrow \mathcal{P}\left((M_{k+\lfloor \sqrt{k} \rfloor}, \bot, t'), (a_i)_{[k+\lfloor \sqrt{k} \rfloor]}, \mathsf{PP}_{\mathsf{snark}}\right) \\
x = a \| k \| \pi \| 0^i \\
y = a \| (k + \lfloor \sqrt{k} \rfloor) \| \pi' \| 0^{i'}
\end{array}
\right\}
$$

In words, $\mathcal{D}_{\mathsf{Clear}}$ is "uniform" over signatures and proofs and decreases exponentially with k.

Defining $\mathcal{D}_{\mathsf{Enc}}$. Intuitively, $\mathcal{D}_{\mathsf{Enc}}$ is $\mathcal{D}_{\mathsf{Clear}}$ under an IB-FHE encryption. More formally, let $\mathcal{I}$ be a space of identities of size $2^{\Omega(n)}$. Let

$$
\mathcal{D}_{\mathsf{Enc}} = \left\{ (x^{\mathsf{enc}} \| \mathsf{id}_1 \| \mathsf{id}_2 \| \mathsf{sk}_{\mathsf{id}_2}, y^{\mathsf{enc}}) \middle|
\begin{array}{l}
\mathsf{id}_1, \mathsf{id}_2 \leftarrow \mathcal{I} \\
(x, y) \leftarrow \mathcal{D}_{\mathsf{Clear}} \\
x^{\mathsf{enc}} \leftarrow \mathsf{Encrypt}(x, \mathsf{id}_1) \\
y^{\mathsf{enc}} \leftarrow \mathsf{Encrypt}(y, \mathsf{id}_1) \\
\mathsf{sk}_{\mathsf{id}_2} \leftarrow \mathrm{KeyGen}(\mathsf{MSK}, \mathsf{id}_2)
\end{array}
\right\}
$$

In words, it is a sample from $\mathcal{D}_{\mathsf{Clear}}$ encrypted under a random identity id_1 supplied with a secret key for a random identity id_2. The reason for supplying this additional secret key will become clear later.

Defining the quality oracle h. Oracle h is defined via the following algorithm

1. Determine if (x, y) has the right form and if not return 0.[15]
2. If (x, y) is encrypted then run:
   - Let x^{enc} be the part of x up to the first separator $\|$.
   - Let id_1 be the the part of x between the first and the second separator $\|$.
   - $\mathsf{sk}_{\mathsf{id}_1} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id}_1)$.
   - $x = \mathsf{Decrypt}(\mathsf{sk}_{\mathsf{id}_1}, x^{\mathsf{enc}})$.
   - $y = \mathsf{Decrypt}(\mathsf{sk}_{\mathsf{id}_1}, y)$.
3. Decompose $x = a \| k \| \pi \| 0^i, y = a' \| k' \| \pi' \| 0^{i'}$. If at least one of them is of the wrong format return 0.

---

[15] Note that it will be important what happens outside of the support because $\mathbf{V}$, during an attack, can send samples that lie outside the support of the distribution.

4. Verify that $a = a', \mathsf{Verify}(\mathsf{pk}, m = 0, a) = 1, k' \geq k + \lfloor\sqrt{k}\rfloor,$[16] and that $\mathcal{V}(\mathsf{PP_{snark}}, (M_k, \bot, t_k), \pi) = 1$ and that $\mathcal{V}(\mathsf{PP_{snark}}, (M_{k+\lfloor\sqrt{k}\rfloor}, \bot, t_{k+\lfloor\sqrt{k}\rfloor}), \pi') = 1$. If all the checks are successful return 0 and otherwise return 1.

In words, h first decrypts if (x, y) was encrypted and then checks if both x and y contain valid zk-SNARKS for the right values of k and valid signatures.

Definition 4 (Hardness Levels of $\mathbb{L}^{\mathsf{data}}$). *For $n \in \mathbb{N}$ and $(\mathcal{D}, h) \in \mathsf{supp}(\mathbb{L}_n^{\mathsf{data}})$ we say that $(x, y) \in \mathsf{supp}(\mathcal{D})$ is of hardness level k if: when x is unencrypted it decomposes to $x = a\|k\|\pi\|0^i$, and otherwise it satisfies that $x = x^{\mathsf{enc}}\|\mathsf{id}_1\|\mathsf{id}_2\|\mathsf{sk}_{\mathsf{id}_2}$ and $\mathsf{Decrypt}(x^{\mathsf{enc}}, \mathsf{id}_1) = a\|k\|\pi\|0^i$.*

Lemma 2 ($\mathbb{L}^{\mathsf{data}}$ learnability.). *There exists a universal constant $C \in \mathbb{N}$ such that for every $n \in \mathbb{N}, \epsilon, \delta \in (0, \frac{1}{2}), \mathbb{L}_n^{\mathsf{data}}$ is learnable to error ϵ with probability $1 - \delta$ with sample complexity $C(\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta}))$ and time $O((\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta}))\mathsf{poly}(n))$.*
Moreover, for every $K : \mathbb{N} \to \mathbb{N}, C(K + \log(\frac{1}{\delta}))$ samples suffices to, with probability $1 - \delta$, learn a classifier that answers correctly on all hardness levels of $\mathbb{L}^{\mathsf{data}}$ up to level K. Additionally, f can be represented in size $O(\sqrt{K} \cdot \mathsf{poly}(n))$, for a universal polynomial $\mathsf{poly}(n)$.

Proof. We define a procedure that learns $\mathbb{L}^{\mathsf{data}}$ "up to level K".

1. Draw $m = O(K)$ samples from $\mathcal{D}$, i.e., $S = ((x_i, y_i))_{[m]} \sim \mathcal{D}^m$.
2. Choose a subset of $S' = ((x_i, y_i))_{[K]} \subseteq S$ to be K samples from S that are unencrypted. If there is fewer than K samples return a dummy f.
3. Let $(a_i)_{[K]}$ be elements of the domain of g, where for every $i \in [K]$, a_i is the part of x up to the first separator $\|$.
4. For $j = 1, \ldots, \lfloor\sqrt{K}\rfloor$:
   - $\pi_j \leftarrow \mathcal{P}\left((M_{j\cdot\lfloor\sqrt{K}\rfloor}, \bot, t_{j\cdot\lfloor\sqrt{K}\rfloor}), (a_{j'})_{[j\cdot\lfloor\sqrt{K}\rfloor]}, \mathsf{PP_{snark}}\right)$
5. Let f' be defined as an algorithm that on input x decomposes it to $x = a\|k\|\pi\|0^i$ and if $k + \lfloor\sqrt{k}\rfloor \leq K$ it returns $y = a\|j\cdot\lfloor\sqrt{K}\rfloor\|\pi_{j\cdot\lfloor\sqrt{K}\rfloor}\|0^{i'}$, for the smallest j such that $\cdot\lfloor\sqrt{K}\rfloor \geq k + \sqrt{k}$. This will make sure that y is a valid output for x. If $k + \lfloor\sqrt{k}\rfloor > K$ it returns $\bot$.
6. Let f be defined as an algorithm that on input x first determines if it is encrypted and if it is not it returns $f'(x)$. If x has an encrypted part it first decomposes $x = x^{\mathsf{enc}}\|\mathsf{id}_1\|\mathsf{id}_2\|\mathsf{sk}_{\mathsf{id}_2}$ and then returns $y \leftarrow \mathsf{Eval}(\mathsf{PP_{fhe}}, f', x^{\mathsf{enc}})$.
7. Return f.

We claim that if invoked with $K = O(\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta}))$ it satisfies the conditions of the lemma. Indeed, because $\mathcal{D}$ is defined such that $k \leftarrow \mathsf{Geom}(\frac{1}{2})$, it is enough to learn up to level $\lceil\log(\frac{1}{\epsilon})\rceil$ to achieve a classifier of error ϵ. This in turn implies that $O(\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta}))$ samples are enough because the Chernoff bound guarantees that the learner sees at least $\lceil\log(\frac{1}{\epsilon})\rceil$ unencrypted samples with probability $1 - \delta$.

Note that f can be represented in size $O(\sqrt{K} \cdot \mathsf{poly}(n))$ because it contains only $O(\sqrt{K})$ different proofs. $\qquad\square$

---

[16] Note that h checks if $k' \geq k + \lfloor\sqrt{k}\rfloor$ and not $k' = k + \lfloor\sqrt{k}\rfloor$. This will be important when designing a DbM, because it will allow to make the size of representation of f small.

9   Main Result

Theorem 1. *There exists a learning task and a universal constant $C \in \mathbb{N}$ such that for every $K : \mathbb{N} \times (0,1)^2 \to \mathbb{N}$, where for every $\epsilon, \delta \in (0,1)$, $K(n,\epsilon,\delta) \geq C((\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta})) \cdot n)$,[17] and is upper bounded by a polynomial in n, where n is the security parameter, it holds that:*

- *There exists a DbM with $K_{\text{TRAIN}} = K$, $K_{\text{MITIGATE}} = O(\sqrt{K})$ and $K_{\text{ATTACK}} = \frac{K}{10C}$.*
- *There is no DbD with $K_{\text{TRAIN}} = K$ even if $K_{\text{DETECT}} = \mathsf{poly}(n)$ and $K_{\text{ATTACK}} = O(1)$.*

Remark 10. Note that the sample complexities of the attack on DbD and the mitigation procedure of DbM are much smaller than the sample complexity of the training phase.

Proof. We claim that $\mathbb{L}^{\mathsf{data}}$ satisfies the conditions of the theorem. It follows from Lemma 3 and 4 the we prove in what follows. $\square$

Lemma 3 (DbD does not exist for $\mathbb{L}^{\mathsf{data}}$.). *For every $K : \mathbb{N} \times (0,1)^2 \to \mathbb{N}$ such that for every $\epsilon, \delta \in (0,1)$, $K(n,\epsilon,\delta) = \Omega(n)$ we have that there is no DbD for $\mathbb{L}^{\mathsf{data}}$ with $K_{\text{TRAIN}} = K, K_{\text{ATTACK}} = O(1)$. Moreover, the attack runs in time $O(\sqrt{K} \cdot |f| \cdot \mathsf{poly}(n))$.*

Proof. We start with defining the attack.

Definition of an attack on DbDs. $\mathbf{V}$ upon receiving f can perform the following attack.

1. Draw $K_{\text{ATTACK}} = O(1)$ samples from $\mathcal{D}$, i.e., $S \sim \mathcal{D}^{K_{\text{ATTACK}}}$.
2. Let $(x,y) \in S$ be any unencrypted sample. If it does not exist abort. Let a be the part of x up to the first separator $\|$.
3. $\pi_1 \leftarrow \mathcal{P}\left((M_1, \perp, t_1), (a), \mathsf{PP_{snark}}\right)$
4. Let $x_1 = a\|1\|\pi_1\|0^i$.
5. Let $(\mathsf{id}^*, \mathsf{sk}_{\mathsf{id}^*}), (\mathsf{id}^\dagger, \mathsf{sk}_{\mathsf{id}^\dagger})$ be two different identity, secret-key pairs from two encrypted samples from S. If they do not exist abort.
6. For $j = 2, \ldots, \infty$
    - $x_j := \mathsf{Decrypt}(\mathsf{Eval}(f, \mathsf{Encrypt}(x_{j-1}, \mathsf{id}^*)), \mathsf{sk}^*)$
    - Decompose $x_j = a_j\|k_j\|\pi_j\|0^{i_j}$
    - If at least one of the following checks: $a_j = a$, $k_j \geq k_{j-1} + \lfloor\sqrt{k_{j-1}}\rfloor$, $\mathcal{V}(\mathsf{PP_{snark}}, (M_{k_j}, \perp, t_{k_j}), \pi_j) = 1$ fails then:
        - Draw $(x_{\mathsf{Clear}}, y_{\mathsf{Clear}}) \sim \mathcal{D}_{\mathsf{Clear}}$.
        - With probability $\frac{1}{2}$ return $\mathsf{Encrypt}(x_i, \mathsf{id}^*)\|\mathsf{id}^*\|\mathsf{id}^\dagger\|\mathsf{sk}_{\mathsf{id}^\dagger}$ and return x_{Clear} otherwise.

In words, $\mathbf{V}$ starts with a valid question on level 1 and iterates f on itself, i.e., computes $x_i := \mathsf{Decrypt}(\mathsf{Eval}(f, \mathsf{Encrypt}(x_{i-1}, \mathsf{id}^*)), \mathsf{sk}^*)$, until f produces valid answers. Finally, with probability $\frac{1}{2}$ she sends a question from the marginal distribution of $\mathcal{D}_{\mathsf{Clear}}$, and $\mathsf{Encrypt}(x_i, \mathsf{id}^*)\|\mathsf{id}^*\|\mathsf{id}^\dagger\|\mathsf{sk}_{\mathsf{id}^\dagger}$ otherwise.

---

[17] This lower-bound makes sure that we are in the regime, where K_{TRAIN} is enough samples to learn up to error ϵ with confidence $1 - \delta$.

Analysis of the attack. Recall that $\mathbf{P}_{\text{TRAIN}}$ uses K samples. Let S_{Train} be the sample from $\mathcal{D}^K$ that $\mathbf{P}_{\text{TRAIN}}$ draws. By the union bound and the fact that in $\mathcal{D}$, k is distributed according to $\text{Geom}(\frac{1}{2})$ the probability that S_{Train} contains a sample from a level at least $K + \lfloor\sqrt{K}\rfloor + 1$ is at most

$$K \cdot 2^{-K+1} \leq 2^{-K/2} = \text{negl}(n),$$

from the assumption that $K = \Omega(n)$. By strong unforgeability the signature scheme (Definition 7) and the knowledge soundness of zk-SNARK this implies that $\mathbf{P}_{\text{TRAIN}}$ computes answers incorrectly for all levels $k \geq K + \lfloor\sqrt{K}\rfloor + 1$ with all but negligible in n probability.

   This implies that $\mathbf{V}$ needs only to repeat the iteration procedure, i.e., $x_j := \text{Decrypt}(\text{Eval}(f, \text{Encrypt}(x_{j-1}, \text{id}^*)), \text{sk}^*)$, $O(\sqrt{K})$ times until it finds an incorrect output. The attack is indistinguishable from questions from $\mathcal{D}$ because of the security of IB-FHE, and the fact that there is a negligible probability that $\mathbf{P}$ has a secret key corresponding to id^*, because the space of identities $\mathcal{I}$ is $2^{\Omega(n)}$. f produces a wrong output with high probability by construction.

Efficiency of the attack. Recall that $\mathbf{V}$ needs only $K_{\text{ATTACK}} = O(1)$ samples for a succesful attack. Moreover, its running time is $O(\sqrt{K} \cdot |f| \cdot \text{poly}(n))$. $\qquad\square$

Lemma 4 (DbM exists for $\mathbb{L}^{\text{data}}$.). *There exists a universal constant $C \in \mathbb{N}$ such that for every $K : \mathbb{N} \times (0,1)^2 \to \mathbb{N}$, where for every $\epsilon, \delta \in (0,1)$, $K(n, \epsilon, \delta) \geq C((\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta})) \cdot n)$,[18] and is upper bounded by a polynomial in n, there exists a DbM with $K_{\text{TRAIN}} = K$, $K_{\text{MITIGATE}} = O(\sqrt{K})$ and $K_{\text{ATTACK}} = \frac{K}{10C}$. Moreover, f can be represented in size $O(\sqrt{K} \cdot \text{poly}(n))$.*

Proof. Let C be the same universal constant as in Lemma 2.
   We start by defining a DbM.

DbM definition. $\mathbf{P}_{\text{TRAIN}}$ runs the following algorithm:

1. Learn f by running a procedure from Lemma 2. Let S be the sample drawn by this procedure.
2. Choose a subset of $S' = ((x_i, y_i))_{[m]} \subseteq S$ to be samples from S that are unencrypted.
3. Set $(a_i)_{[m]}$ to be such that for every $i \in [m]$, a_i is the part of x_i up to the first separator $\|$.
4. Set $\text{priv} = (f, (a_i)_{[m]})$.

In words, $\mathbf{P}_{\text{TRAIN}}$ runs the learning algorithm from Lemma 2 to compute f. Next, it stores privately all the signatures, i.e., $\text{priv} = (f, (a_i)_{[m]})$.

Fact 1. *$\mathbf{P}_{TRAIN}$ with probability $1 - \delta$ computes f that (i) satisfies $\text{err}(f) \leq \epsilon$, and (ii) is correct up to level $\frac{K}{C}$.*

---

[18] This lower-bound makes sure that we are in the regime, where K is enough samples to learn up to error ϵ with confidence $1 - \delta$.

Proof. By the assumption that $K \geq C(\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta}))$ and Lemma 2.   □

$\mathbf{P}_{\text{MITIGATE}}$ receives as input $\mathbf{x} \in \mathcal{X}^q$. It runs the following algorithm

1. Draw $S_{\text{Mitigate}} \sim \mathcal{D}^{O(\lfloor\sqrt{K}\rfloor)}$.
2. Choose a subset of $S'_{\text{Mitigate}} = ((x_i, y_i))_{[\lfloor 2\sqrt{K}\rfloor]} \subseteq S_{\text{Mitigate}}$ to be $\lfloor 2\sqrt{K}\rfloor$ samples from S_{Mitigate} that are unencrypted. If they don't exist abort.
3. Let $(a_i^{\text{Mitigate}})_{[\lfloor 2\sqrt{K}\rfloor]}$ to be such that for every $i \in [\lfloor 2\sqrt{K}\rfloor]$, a_i^{Mitigate} is the part of x_i up to the first separator $\|$.
4. For $j = K + 1, \ldots, K + \lfloor 2\sqrt{K}\rfloor$:
   - $\pi_j^{\text{Mitigate}} \leftarrow \mathcal{P}\left((M_j, \bot, t_j), (a_{j'})_{[K]} \cup (a_{j'}^{\text{Mitigate}})_{[\lfloor 2\sqrt{K}\rfloor]}, \mathsf{PP}_{\text{snark}}\right)$
5. Let f' be the following augmentation of f. On input x it decomposes it to $x = a\|k\|\pi\|0^i$ and if $k + \lfloor\sqrt{k}\rfloor \leq K$ it returns $y = f(x)$. If $k + \lfloor\sqrt{k}\rfloor \leq K + \lfloor 2\sqrt{K}\rfloor$ it returns $\pi_{k+\lfloor\sqrt{k}\rfloor}$. Otherwise, it returns $\bot$.
6. Let f^{strong} be defined as an algorithm that on input x first determines if it is encrypted and if it is not it returns $f'(x)$. If x has an encrypted part it first decomposes $x = x^{\text{enc}}\|\mathsf{id}_1\|\mathsf{id}_2\|\mathsf{sk}_{\mathsf{id}_2}$ and then returns $y \leftarrow \mathsf{Eval}(\mathsf{PP}_{\text{fhe}}, f', x^{\text{enc}})$.
7. Return $(f^{\text{strong}}(x), b = 0)$.[19] Note that it always returns $b = 0$, i.e., it never suspects that a sample is an adversarial example.

In words, $\mathbf{P}_{\text{MITIGATE}}$ first collects $\lfloor 2\sqrt{K}\rfloor$ additional signatures by drawing $O(\sqrt{K})$ fresh samples. Next, it creates zk-SNARKS for hardness levels $K + 1, \ldots, K + \lfloor 2\sqrt{K}\rfloor$ using the privately stored signatures that $\mathbf{P}_{\text{TRAIN}}$ used to compute f. Finally, it creates a strong classifier f^{strong} by augmenting f with the newly created zk-SNARKs.

Security of the DbM. We will show that properties of Definition 3 are satisfied.
     Firstly, Fact 1 guarantees *correctness*.
     Secondly, *completeness* is trivially satisfied because $\mathbf{P}_{\text{MITIGATE}}$ always returns $b = 0$.
     Thirdly, we argue that *soundness* is satisfied. Let $\mathbf{V}$ be a polynomial-size attacker using $K_{\text{ATTACK}} \leq \frac{K}{10C}$ samples. Let $K_{\text{LEVEL}} := \frac{K}{C}$.
     Assume towards contradiction that there exists a polynomial-size $\mathbf{V}$ that, when given as additional input the MSK of the IB-FHE, can produce a valid zk-SNARK proof for a level at least $K_{\text{LEVEL}} + \lfloor\sqrt{K_{\text{LEVEL}}}\rfloor + 1$.
     We proceed with a series of Hybrids for the distribution of $\mathbf{V}$ to show that this leads to a contradiction.

1. Hybrid 0: the original distribution of the output of $\mathbf{V}$.
2. Hybrid 1: Let S_{Attack} be the sample from $\mathcal{D}^{K_{\text{ATTACK}}}$.[20] By the union bound and the fact that in $\mathcal{D}$, k is distributed according to $\mathsf{Geom}(\frac{1}{2})$ the probability

---

[19] We slightly abused the notation here. What we mean is that f^{strong} is applied to every element of $\mathbf{x}$ separately.

[20] Note that by the assumption that $\mathbf{V}$ has access to MSK we can assume that all samples in S_{Attack} are unencrypted.

that S_{Attack} contains a sample from a level at least $K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor + 1$ is at most

$$K_{\mathrm{Level}} \cdot 2^{-K_{\mathrm{Level}}+1} \leq 2^{-K_{\mathrm{Level}}/2} = \mathsf{negl}(n),$$

from the assumption that $K = \Omega(n)$. We replace the oracle access of $\mathbf{V}$ to $\mathcal{D}$ by an oracle access to $\mathcal{D}'$ that is identical to $\mathcal{D}$ apart from the fact that it samples k from a modified distribution $\mathsf{Geom}(\frac{1}{2})$, where $\Pr[k = K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor] = 2^{-(K_{\mathrm{Level}}+\lfloor\sqrt{K_{\mathrm{Level}}}\rfloor+1)}$, and for all $k' > K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor$ we have $\Pr[k = k'] = 0$. We have that $\mathcal{D}$ and $\mathcal{D}'$ are statistically negligibly close.

3. Hybrid 2: By construction f, in it's code, contains $O(\sqrt{K})$ hardcoded proofs for levels k up to $K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor$. Additionally $\mathbf{V}$ received K_{Attack} samples from $\mathcal{D}'$, which by construction produces proofs for level up to $K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor$. We replace all these proofs using the simulator, which is guaranteed to exist by the zero-knowledge property of the zk-SNARKs. By zero-knowledge $\mathbf{V}$ still produces a valid proof for level at least $K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor + 1$ (because otherwise, it could distinguish the real from the simulated proofs because zk-SNARK is publicly verifiable and because $\mathbf{V}$ has access to MSK).

If we now apply the efficient extractor of the zk-SNARK on $\mathbf{V}$ when executed as in Hybrid 2 it would produce a valid witness for a proof at level $K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor + 1$, which by construction would constitute of $K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor + 1$ different signatures of $m = 0$. However, $\mathbf{V}$ received at most $\frac{K}{10C} < K_{\mathrm{Level}}$ signatures in S_{attack} and 1 additional signature encoded in f. Indeed, observe that the code of f includes only 1 signature, zk-SNARK proofs, and an instruction on how to run f on $\mathcal{D}_{\mathsf{Enc}}$. The fact that there is an efficient procedure (running the extractor) that produces $K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor + 1$ signatures having access to only $K_{\mathrm{Level}} + \lfloor\sqrt{K_{\mathrm{Level}}}\rfloor$ signatures is a contradiction with strong unforgeability of the signature scheme.

Consider now the original $\mathbf{V}$ without access to MSK. If it was able to produce $x \in \mathrm{supp}(\mathcal{D}_{\mathsf{Enc}})$ that contains a proof for level at least $K_{\mathrm{Level}} + \sqrt{K_{\mathrm{Level}}} + 1$ then $\mathbf{V}$ with access to MSK would be able to produce a proof for level at least $K_{\mathrm{Level}} + \sqrt{K_{\mathrm{Level}}} + 1$. But we showed that it is impossible.

Summarizing $\mathbf{V}$ is able to produce $x \in \mathrm{supp}(\mathcal{D})$ with a proof for a level at most $K_{\mathrm{Level}} + \sqrt{K_{\mathrm{Level}}}$. This implies soundness because $\mathbf{P}_{\mathrm{Mitigate}}$ answers correctly for all these levels.

Efficiency of the DbM. Note that the size of f is much smaller than K. Indeed, Lemma 2 guarantees that f can be represented in size $O(\sqrt{K} \cdot \mathsf{poly}(n))$.[21]

$\square$

---

[21] Note that $O(\sqrt{K} \cdot \mathsf{poly}(n)) \ll K$, for big enough K.

10 Cryptographic tools

Definition 5 (NP Relation). *A binary relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ is an NP relation if there exists a polynomial-time algorithm $\mathcal{V}$ such that:*

$$\forall(x,w) \in \{0,1\}^* \times \{0,1\}^*, \quad (x,w) \in \mathcal{R} \iff \mathcal{V}(x,w) = 1.$$

The associated language $\mathcal{L}_{\mathcal{R}}$ is the set of statements x for which there exists a witness w such that $(x,w) \in \mathcal{R}$:

$$\mathcal{L}_{\mathcal{R}} = \{x \in \{0,1\}^* \mid \exists w \text{ such that } (x,w) \in \mathcal{R}\}.$$

10.1 Interactive Proofs

We use the standard definitions of interactive proofs (and interactive Turing machines) and arguments (also known as computationally sound proofs). Given a pair of interactive Turing machines, $(\mathcal{P}, \mathcal{V})$, we denote by $\langle \mathcal{P}(w), \mathcal{V} \rangle(x)$ the random variable representing the final (local) output of $\mathcal{V}$, on common input x, when interacting with machine $\mathcal{P}$ with private input w, when the random tape to each machine is uniformly and independently chosen.

Definition 6 (Interactive Proof System). *A pair $(\mathcal{P}, \mathcal{V})$ of interactive machines is called an* interactive proof system *for a language $\mathcal{L}$ with respect to a witness relation $\mathcal{R}_{\mathcal{L}}$ if the following two conditions hold:*

– **Completeness:** *For every $x \in \mathcal{L}$ and $w \in \mathcal{R}_{\mathcal{L}}(x)$,*

$$\Pr\left[\langle \mathcal{P}(w), \mathcal{V} \rangle(x) = 1\right] = 1.$$

– **Soundness:** *For every interactive machine $\mathcal{B}$, there is a negligible function $\nu(\cdot)$, such that, for every $x \notin \mathcal{L}$,*

$$\Pr\left[\langle \mathcal{B}, \mathcal{V} \rangle(x) = 1\right] \leq \nu(|x|).$$

In the case that the soundness condition is required to hold only with respect to a polynomial-size prover, the pair $\langle \mathcal{P}, \mathcal{V} \rangle$ is called an *interactive argument system*.

10.2 Strongly Unforgeable Signatures

We formally define *Strongly Unforgeable Digital Signatures*.

Definition 7 (Strongly Unforgeable Signature Scheme). *A* Strongly Unforgeable Signature Scheme *consists of a tuple of probabilistic polynomial-time (PPT) algorithms* (Gen, Sign, Verify) *with the following properties:*

– **Key Generation:**
$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^n)$$

takes as input the security parameter n and outputs a secret key sk and a public key pk.

– **_Signing:_**

$$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$$

takes as input the secret key sk _and a message_ m, _and outputs a signature_ σ.
– **_Verification:_**

$$\mathsf{Verify}(\mathsf{pk}, m, \sigma) \in \{0, 1\}$$

takes as input the public key pk, _a message_ m, _and a signature_ σ, _and outputs 1 if_ σ _is a valid signature on_ m _under_ pk, _and 0 otherwise._

The scheme must satisfy the following properties:

– **_Correctness:_** _For all_ $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^n)$ _and all messages_ m,

$$\Pr\left[\mathsf{Verify}(\mathsf{pk}, m, \sigma) = 1 \mid \sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)\right] = 1.$$

– **_Strong Unforgeability:_** _For any PPT adversary_ $\mathcal{A}$, _there exists a negligible function_ $\mathsf{negl}(n)$ _such that:_

$$\Pr\left[\mathsf{Verify}(\mathsf{pk}, m^*, \sigma^*) = 1 \ and \ (m^*, \sigma^*) \notin \mathcal{Q}\right] \leq \mathsf{negl}(n),$$

where $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^n)$, _the adversary_ $\mathcal{A}^{\mathsf{Sign}(\mathsf{sk}, \cdot)}(\mathsf{pk})$ _interacts with a signing oracle, obtaining signatures on at most polynomially many messages forming the query set_ $\mathcal{Q}$, _and then produces a new message-signature pair_ (m^*, σ^*) _that was not in_ $\mathcal{Q}$.

10.3 SNARKs

Definition 8 (Universal Relation). _The_ universal relation $\mathcal{R}_U$ _consists of all instance-witness pairs_ (y, w) _where:_ $y = (M, x, t)$, $|y|, |w| \leq t$, _such that_ M _is a random-access machine that accepts_ (x, w) _within at most_ t _steps. The corresponding language is denoted by_ $\mathcal{L}_U$.

We now proceed to define SNARKs. A _fully-succinct non-interactive argument of knowledge_ (SNARK) [Bitansky et al., 2022] is a triple of algorithms $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ that work as follows. The (probabilistic) generator $\mathcal{G}$, on input the security parameter n (presented in unary) and a time bound T, outputs public parameters PP. The honest prover $\mathcal{P}(y, w, \mathsf{PP})$ produces a proof π for the instance $y = (M, x, t)$ given a witness w, provided that $t \leq T$; then, $\mathcal{V}(\mathsf{PP}, y, \pi)$ verifies the validity of π.

Definition 9 (SNARK). _A_ Succinct Non-Interactive Argument of Knowledge _(SNARK) for the relation_ $\mathcal{R} \subseteq \mathcal{R}_U$ _consists of a triple of algorithms_ $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ _with the following properties:_

– **_Completeness:_** _For every large enough security parameter_ $n \in \mathbb{N}$, _every time bound_ $T \in N$, _and every instancewitness pair_ $(y, w) = (M, x, t), w \in \mathcal{R}$ _with_ $t \leq T$,

$$\Pr\left[\mathcal{V}(\mathsf{PP}, y, \pi) = 1 \mid (\sigma, \tau) \leftarrow \mathcal{G}(1^n, T), \ \pi \leftarrow \mathcal{P}(y, w, \mathsf{PP})\right] = 1.$$

- **Adaptive Proof of Knowledge:** *For any polynomial-size prover $\mathcal{P}^*$, there exists a polynomial-size extractor $\mathcal{E}^{\mathcal{P}^*}$ such that for every large enough $n \in \mathbb{N}$, all auxiliary inputs $z \in \{0,1\}^{poly(n)}$, and all time bounds $T \in \mathbb{N}$,*

$$\Pr\left[\mathcal{V}(\mathsf{PP}, y, \pi) = 1 \text{ and } (y, w) \notin \mathcal{R} \;\middle|\; \begin{array}{l} \mathsf{PP} \leftarrow \mathcal{G}(1^\lambda, T), \\ (y, \pi) \leftarrow \mathcal{P}^*(z, \mathsf{PP}), \\ w \leftarrow \mathcal{E}^{\mathcal{P}^*}(z, \mathsf{PP}) \end{array}\right] \leq \mathsf{negl}(n).$$

- **Efficiency:** *There exists a universal polynomial p (independent of $\mathcal{R}$) such that, for every large enough security parameter $n \in \mathbb{N}$, every time bound $T \in \mathbb{N}$, and every instance $y = (M, x, t)$ with $t \leq T$:*
  - *Generator Runtime: $p(n + \log T)$*
  - *Prover Runtime: $p(n + |M| + |x| + t + \log T)$*
  - *Verifier Runtime: $p(n + |M| + |x| + \log T)$*
  - *Proof Size: $p(n + \log T)$*

A *zero-knowledge SNARK* (or "succinct NIZK of knowledge") satisfies a zero-knowledge property. That is, an honest prover can generate valid proofs without leaking any information beyond the truth of the statement, particularly without revealing any knowledge of the witness used to generate the proof. For zero-knowledge SNARKs, the reference string σ must be a common reference string that is trusted by both the verifier and the prover.

Definition 10 (Zero-Knowledge SNARK). *A triple of algorithms $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ is a zero-knowledge SNARK for the relation $\mathcal{R} \subseteq \mathcal{R}_U$ if it is a SNARK for $\mathcal{R}$ and satisfies the following property:*

- **Zero Knowledge:** *There exists a stateful interactive polynomial-size simulator S such that for all stateful interactive polynomial-size distinguishers D, every large enough security parameter $\lambda \in \mathbb{N}$, every auxiliary input $z \in \{0,1\}^{poly(\lambda)}$, and every time bound $T \in \mathbb{N}$:*

$$\Pr\left[t \leq T, (y, w) \in \mathcal{R}_U, D(\pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{PP} \leftarrow \mathcal{G}(1^n, T), \\ (y, w) \leftarrow D(z, \mathsf{PP}), \\ \pi \leftarrow \mathcal{P}(y, w, \mathsf{PP}) \end{array}\right]$$
$$\approx \Pr\left[t \leq T, (y, w) \in \mathcal{R}_U, D(\pi) = 1 \;\middle|\; \begin{array}{l} (\mathsf{PP}, \mathsf{trap}) \leftarrow S(1^n, T), \\ (y, w) \leftarrow D(z, \mathsf{PP}), \\ \pi \leftarrow S(z, y, \mathsf{PP}, \mathsf{trap}) \end{array}\right].$$

10.4   IB-FHE

Definition 11 (Identity-Based Fully Homomorphic Encryption (IB-FHE)). *An* Identity-Based Fully Homomorphic Encryption *(IB-FHE) scheme with message space $\mathcal{M}$, identity space $\mathcal{I}$, and a class of circuits $\mathcal{C} \subseteq \mathcal{M}^* \to \mathcal{M}$ consists of a tuple of probabilistic polynomial-time (PPT) algorithms* $\mathsf{IB-FHE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$ *with the following properties:*

– **Setup:**
$$(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^n)$$

  takes as input the security parameter n and outputs public parameters PP and a master secret key MSK.

– **Key Generation:**
$$\mathsf{sk_{id}} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id})$$

  takes as input the master secret key MSK and an identity $\mathsf{id} \in \mathcal{I}$, outputting a secret key $\mathsf{sk_{id}}$.

– **Encryption:**
$$c \leftarrow \mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, m)$$

  takes as input the public parameters PP, an identity id, and a message $m \in \mathcal{M}$, and outputs a ciphertext c.

– **Decryption:**
$$m \leftarrow \mathsf{Decrypt}(\mathsf{sk_{id}}, c)$$

  takes as input a secret key $\mathsf{sk_{id}}$ and a ciphertext c, outputting the plaintext m if c is a valid encryption under id; otherwise, it outputs $\perp$.

– **Evaluation:**
$$c' \leftarrow \mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_k)$$

  takes as input the public parameters PP, a circuit $C \in \mathcal{C}$, and a collection of ciphertexts $(c_1, \ldots, c_k)$, outputting a new ciphertext c'.

  The scheme must satisfy the following properties:

– **Correctness:** For all $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^n)$, identities $\mathsf{id} \in \mathcal{I}$, circuits $C : \mathcal{M}^k \to \mathcal{M}$, and messages $m_1, \ldots, m_k \in \mathcal{M}$:

$$\Pr\left[\mathsf{Decrypt}(\mathsf{sk_{id}}, \mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_k)) = C(m_1, \ldots, m_k)\right] = 1 - \mathsf{negl}(n),$$

  where $c_i \leftarrow \mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}, m_i)$ and $\mathsf{sk_{id}} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, \mathsf{id})$.

– **Compactness:** There exists a polynomial $p(\cdot)$ such that for all n, C, and messages:
$$\Pr\left[|\mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_k)| \leq p(n)\right] = 1.$$

– **IND-sID-CPA Security:** For any PPT adversary $\mathcal{A}$,

$$\left|\Pr\left[\mathsf{IB} - \mathsf{FHEGame}_0^{\mathcal{A}}(n) = 1\right] - \Pr\left[\mathsf{IB} - \mathsf{FHEGame}_1^{\mathcal{A}}(n) = 1\right]\right| \leq \mathsf{negl}(n),$$

  where $\mathsf{IB} - \mathsf{FHEGame}_b^{\mathcal{A}}(n)$ is defined as follows:
  1. $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^n)$ is generated, and PP is given to $\mathcal{A}$.
  2. $\mathcal{A}$ chooses id^* and submits $(m_0, m_1) \in \mathcal{M}$.
  3. The challenger samples $b \leftarrow \{0, 1\}$ and returns $c^* \leftarrow \mathsf{Encrypt}(\mathsf{PP}, \mathsf{id}^*, m_b)$.
  4. $\mathcal{A}$ outputs a bit b'.

Bibliography

Ron Amit and Ron Meir. Meta-learning by adjusting priors based on extended pac-bayes theory. In *International Conference on Machine Learning*, pages 205–214. PMLR, 2018.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Jailbreaking leading safety-aligned llms with simple adaptive attacks, 2024.

Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. Succinct non-interactive arguments via linear interactive proofs. *J. Cryptol.*, 35(3), July 2022. ISSN 0933-2790. https://doi.org/10.1007/s00145-022-09424-4. URL https://doi.org/10.1007/s00145-022-09424-4.

Dan Boneh, Emily Shen, and Brent Waters. Strongly unforgeable signatures based on computational diffie-hellman. In *Proceedings of the 9th International Conference on Theory and Practice of Public-Key Cryptography*, PKC'06, page 229–240, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3540338519. https://doi.org/10.1007/11745853_15. URL https://doi.org/10.1007/11745853_15.

Nicholas Carlini, Milad Nasr, Christopher A. Choquette-Choo, Matthew Jagielski, Irena Gao, Anas Awadalla, Pang Wei Koh, Daphne Ippolito, Katherine Lee, Florian Tramèr, and Ludwig Schmidt. Are aligned neural networks adversarially aligned? *ArXiv*, abs/2306.15447, 2023. URL https://api.semanticscholar.org/CorpusID:259262181.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries, 2023.

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.

Gamaleldin Elsayed, Shreya Shankar, Brian Cheung, Nicolas Papernot, Alexey Kurakin, Ian Goodfellow, and Jascha Sohl-Dickstein. Adversarial examples that fool both computer vision and time-limited humans. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/8562ae5e286544710b2e7ebe9858833

Logan Engstrom, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with

simple   transformations.   *ArXiv*,   abs/1712.02779,   2017.   URL
https://api.semanticscholar.org/CorpusID:21929206.

Jon Feldman, Rocco A Servedio, and Ryan O'Donnell. Pac learning axis-aligned
mixtures of gaussians with no separation assumption. In *Learning Theory:
19th Annual Conference on Learning Theory, COLT 2006, Pittsburgh, PA,
USA, June 22-25, 2006. Proceedings 19*, pages 20–34. Springer, 2006.

Craig Gentry, Amit Sahai, and Brent Waters.   Homomorphic encryp-
tion from learning with errors: Conceptually-simpler, asymptotically-faster,
attribute-based.   Cryptology ePrint Archive, Paper 2013/340, 2013.   URL
https://eprint.iacr.org/2013/340.

Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S. Schoenholz, Maithra
Raghu, Martin Wattenberg, and Ian J. Goodfellow. Adversarial spheres. In
*6th International Conference on Learning Representations, ICLR 2018, Van-
couver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*,
2018. URL https://openreview.net/forum?id=SkthlLkPf.

Grzegorz Gluch and Ruediger Urbanke.   Exponential separation be-
tween two learning models and adversarial robustness.   In M. Ran-
zato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman
Vaughan, editors, *Advances in Neural Information Processing Systems*,
volume 34, pages 20785–20797. Curran Associates, Inc., 2021.   URL
https://proceedings.neurips.cc/paper_files/paper/2021/file/ae06fbdc519bddaa88aa1b24bace45

Shafi Goldwasser, Adam Tauman Kalai, Yael Tauman Kalai, and Omar Mon-
tasser. Beyond perturbations: Learning guarantees with arbitrary adversarial
test examples. In *Proceedings of the 34th International Conference on Neural
Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran
Associates Inc. ISBN 9781713829546.

Shafi   Goldwasser,   Michael   P.   Kim,   Vinod   Vaikuntanathan,
and   Or   Zamir.   Planting   undetectable   backdoors   in   ma-
chine   learning   models.   *ArXiv*,   abs/2204.06974,   2022.   URL
https://api.semanticscholar.org/CorpusID:248177888.

Shafi Goldwasser, Jonathan Shafer, Neekon Vafa, and Vinod Vaikuntanathan.
Oblivious defense in ml models: Backdoor removal without detection, 11 2024.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and
harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Ryan Greenblatt, Carson Denison, Benjamin Wright, Fabien Roger, Monte Mac-
Diarmid, Sam Marks, Johannes Treutlein, Tim Belonax, Jack Chen, David Du-
venaud, Akbir Khan, Julian Michael, Sören Mindermann, Ethan Perez, Linda
Petrini, Jonathan Uesato, Jared Kaplan, Buck Shlegeris, Samuel R. Bowman,
and Evan Hubinger. Alignment faking in large language models. *arXiv preprint
arXiv:2412.14093*, 2024.   https://doi.org/10.48550/arXiv.2412.14093.
URL https://arxiv.org/abs/2412.14093.

Grzegorz Głuch, Berkant Turan, Sai Ganesh Nagarajan, and Sebastian Pokutta.
The good, the bad and the ugly: Watermarks, transferable attacks and adver-
sarial defenses, 2024. URL https://arxiv.org/abs/2410.08864.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon
Tran, and Aleksander Madry.   Adversarial examples are not bugs, they

are features. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/e2c420d928d4bf8ce0ff2ec19b3715

Jiaming Ji, Tianyi Qiu, Boyuan Chen, Borong Zhang, Hantao Lou, Kaile Wang, Yawen Duan, Zhonghao He, Jiayi Zhou, Zhaowei Zhang, Fanzhi Zeng, Kwan Yee Ng, Juntao Dai, Xuehai Pan, Aidan O'Gara, Yingshan Lei, Hua Xu, Brian Tse, Jie Fu, Stephen Marcus McAleer, Yaodong Yang, Yizhou Wang, Song-Chun Zhu, Yike Guo, and Wen Gao. Ai alignment: A comprehensive survey. *ArXiv*, abs/2310.19852, 2023. URL https://api.semanticscholar.org/CorpusID:264743032.

Adam Tauman Kalai, Adam R Klivans, Yishay Mansour, and Rocco A Servedio. Agnostically learning halfspaces. *SIAM Journal on Computing*, 37(6):1777–1805, 2008.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

David A McAllester. Pac-bayesian model averaging. In *Proceedings of the twelfth annual conference on Computational learning theory*, pages 164–170, 1999.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box llms automatically, 2024.

Vaishnavh Nagarajan and J Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Nicolas Papernot, Patrick Mcdaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *ArXiv*, abs/1605.07277, 2016. URL https://api.semanticscholar.org/CorpusID:17362994.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=Bys4ob-Rb.

Jonas Rothfuss, Martin Josifoski, and Andreas Krause. Meta-learning bayesian neural network priors based on pac-bayesian theory. 2020. URL https://api.semanticscholar.org/CorpusID:232085156.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6199.

Florian Tramer. Detecting adversarial examples is (Nearly) as hard as classifying them. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 21692–21702. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/tramer22a.html.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? *ArXiv*, abs/2307.02483, 2023. URL https://api.semanticscholar.org/CorpusID:259342528.

Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 51008–51025. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/a00548031e4647b13042c97c922fad

Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5283–5292. PMLR, 2018. URL http://proceedings.mlr.press/v80/wong18a.html.

Wojciech Zaremba, Evgenia Nitishinskaya, Boaz Barak, Stephanie Lin, Sam Toyer, Yaodong Yu, Rachel Dias, Eric Wallace, Kai Xiao, Johannes Heidecke, and Amelia Glaese. Trading inference-time compute for adversarial robustness, 2025.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *ArXiv*, abs/2307.15043, 2023. URL https://api.semanticscholar.org/CorpusID:260202961.