Can Adversarially Robust Learning Leverage Computational Hardness?

Saeed Mahloujifar[*] Mohammad Mahmoody[†]

November 5, 2018

Abstract

Making learners robust to adversarial perturbation at test time (i.e., evasion attacks) or training time (i.e., poisoning attacks) has emerged as a challenging task. It is known that for some natural settings, *sublinear* perturbations in the training phase or the testing phase can drastically decrease the quality of the predictions. These negative results, however, are *information theoretic* and only prove the *existence* of such successful adversarial perturbations. A natural question for these settings is whether or not we can make classifiers *computationally* robust to *polynomial-time* attacks.

In this work, we prove strong barriers against achieving such envisioned computational robustness both for evasion and poisoning attacks. In particular, we show that if the test instances come from a product distribution (e.g., uniform over $\{0,1\}^n$ or $[0,1]^n$, or isotropic n-variate Gaussian) and that there is an initial constant error, then there exists a *polynomial-time* attack that finds adversarial examples of Hamming distance $O(\sqrt{n})$. For poisoning attacks, we prove that for any learning algorithm with sample complexity m and any efficiently computable "predicate" defining some "bad" property B for the produced hypothesis (e.g., failing on a particular test) that happens with an initial constant probability, there exist *polynomial-time* online poisoning attacks that tamper with $O(\sqrt{m})$ many examples, replace them with other correctly labeled examples, and increases the probability of the bad event B to ≈ 1.

Both of our poisoning and evasion attacks are *black-box* in how they access their corresponding components of the system (i.e., the hypothesis, the concept, and the learning algorithm) and make no further assumptions about the classifier or the learning algorithm producing the classifier.

[*] saeed@virginia.edu, University of Virginia. Supported by University of Virginia's SEAS Research Innovation Awards.

[†] mohammad@virginia.edu, University of Virginia. Supported by NSF CAREER award CCF-1350939, and University of Virginia's SEAS Research Innovation Awards.

Contents

1 Introduction

Making trained classifiers robust to adversarial attacks of various forms has been an active line of research in machine learning recently. Two major forms of attack are the so called "evasion" and "poisoning" attacks. In an evasion attack, an adversary enters the game during the test phase and tries to perturb the original test instance x into a "close" adversarial instance x' that is misclassified by the produced hypothesis (a.k.a. trained model) h. In a poisoning attack, the adversary manipulates the training data into a "closely related" poisoned version with the goal of increasing the risk (or some other closely related property such as failing on a particular example) of the hypothesis h produced based on the poisoned data. Starting with Szegedy et al. [SZS+14] a race has emerged between evasion attacks that aim to find classified adversarial examples and defences against those attacks [BCM+13, BFR14, GSS15, PMW+16, CW17, XEQ17, ACW18]. In another line of work, many papers studied poisoning attacks and defense mechanisms against them [BNL12, ABL17, XBB+15, PMSW16, RNH+09, SHN+18, KL17, BL17, CSV17, DKS17, MDM18, DKS18a, DKK+18b, PSBR18, DKS18b, DKK+17, DKK+18a]. Although, some specific problems (e.g., that of image classification) naturally has got more attention in this line of work, like other works in the theory literature, we approach the robustness problem from a general and fundamental perspective.

Is adversarially robust classification possible? Recently, started by Gilmer et el. [GMF+18] and followed by [FFF18, DMM18, SHS+18, MDM19], it was shown that for many natural metric probability spaces of instances (e.g., uniform distribution over $\{0,1\}^n$, $[0,1]^n$, unit n-sphere, or isotropic Gaussian in dimension n, all with "normalized" Euclidean or Hamming distance) adversarial examples of sublinear perturbations exist for almost all test instances. Indeed, as shown by Mahloujifar, Diochnos, and Mahmoody [MDM19], if the instances are drawn from any "normal Lévy family" [MS86] of metric probability spaces (that include all the above-mentioned examples), and if there exists an initial non-negligible risk for the generated hypothesis classifier h, an adversary can perturb an initial instance x into an adversarial one x' that is only $\approx \sqrt{n}$-far (which is sublinear in n) from x and that x' is misclassified.

In the context of poisoning attacks, some classic results about malicious noise [Val85, KL93, BEK02] could be interpreted as limitations of learning under poisoning. On the positive side, the recent breakthroughs of Diakonikolas et al. [DKK$^+$16] and Lia et al. [LRV16] showed the surprising power of robust inference over poisoned data in *polynomial-time* with error that does *not* depend on the dimension of the instances. These works led to an active line of work (e.g., see [CSV17, DKS17, DKS18a, DKK$^+$18b, PSBR18, DKS18b]) exploring the possibility of robust statistics over poisoned data with algorithmic guarantees. In particular [CSV17, DKS18a] showed how to perform *list-docodable* learning; outputting a set of hypothesis one of which is of high quality, and [DKK$^+$18b, PSBR18] studied robust stochastic convex optimization.

Studying the power of poisoning *attacks*, the work of Mahloujifar et al. [MDM19] demonstrated the power of poisoning attacks of various forms as long as there is a "small but non-negligible vulnerability" in no attack setting. Namely, assuming that the goal of the adversary is to increase the probability of any "bad event" B over the generated hypothesis, it was proved in [MDM19] that the adversary can always increase the probability of B from any non-negligible (or at least sub-exponentially large) probability to ≈ 1 using sublinear perturbations of the training data. In particular, the adversary can decrease the confidence of the produced hypothesis (to have error at most ε for a fixed ε), or alternatively it can increase the classification error of a particular instance x, using an adversarial poisoning strategy that achieves these goals by changing $\approx \sqrt{m}$ of the training examples, where m is the sample complexity of the learner.

Is *computationally* robust classification possible? All the above-mentioned sublinear-perturbation attacks of [FFF18, DMM18, SHS$^+$18, MDM19], in both evasion and poisoning models, were *information theoretic* (i.e., *existential*). Namely, they only show the existence of such adversarial instances for evasion attacks or that they show the existence of such adversarial poisoned data with sublinear perturbations for poisoning attacks. In this work, we study the next natural question; can we overcome these information theoretic (existential) lower bounds by relying on the fact that the adversary is computationally bounded? Namely, can we design solutions that resist *polynomial-time* attacks on the robustness of the learning algorithms? More specifically, the general question studied in our work is as follows.

> *Can we make classifiers robust to* computationally bounded *adversarial perturbations (of sublinear magnitude) that occur during the training or the test phase?*

In this work we focus on sublinear perturbations as our main results are *negative* (i.e., demonstrating the power of sublinear tampering).[1]

1.1 Our Results

In this work, we prove strong barriers against basing the robustness of classifiers, in both evasion and poisoning settings, on computational intractability. Namely, we show that in many natural settings (i.e., any problem for which the instances are drawn from a product distribution and that their distances are measured by Hamming distance) adversarial examples could be found in *polynomial time*. This result applies to any learning task over these distributions. In the poisoning attacks' setting, we show that for any learning task and any distribution over the labeled instances, if the goal of the adversary is to decrease the confidence of the learner or to increase its error on any particular instance x, it can always do so in polynomial time by only changing $\approx \sqrt{m}$ of the labeled instances and replacing them with yet correctly labeled examples. Below we describe both of these results at a high level.

[1]For any result proved on the positive side, e.g., it would be stronger to resist even a *linear* amount of perturbations.

Theorem 1.1 (Informal: polynomial-time evasion attacks)**.** *Let $\mathcal{P}$ be a classification problem in which the test instances are drawn from a product distribution $\mathbf{x} \equiv \mathbf{u}_1 \times \cdots \times \mathbf{u}_n$. Suppose c is a concept function (i.e., ground truth) and h is a hypothesis that has a constant $\Omega(1)$ error in predicting c. Then, there is a* polynomial-time *(black-box) adversary that perturbs only $\approx O(\sqrt{n})$ of the* blocks *of the instances and make them misclassified with probability ≈ 1.*

(See Theorem 3.3 for the formal version of the following theorem.)

The above theorem covers many natural distributions such as uniform distributions over $\{0,1\}^n$ or $[0,1]^n$ or the isotropic Gaussian of dimension n, so long as the distance measure is Hamming distance. Also, as we will see in Theorem 3.3, the initial error necessary for our polynomial-time evasion attack could be as small as $1/\operatorname{poly}(\log n)$ to keep the perturbations $\widetilde{O}(\sqrt{n})$, and even initial error $\omega(\log n/\sqrt{n})$ is enough to keep the perturbations sublinear $o(n)$. Finally, by "black-box" we mean that our attacker only needs oracle access to the hypothesis h, the ground truth c, and distribution $\mathbf{x}$.[2] This black-box condition is similar to the one defined in previous work of Papernot et al. [PMG+17], however the notion of black box in some other works (e.g., see [IEAL18]) are more relaxed and give some additional data, such as a vector containing probabilities assigned to each label, to the adversary as well.

We also note that, even though *learning* is usually done with respect to a *family* of distributions (e.g., all distributions), working with a particular distribution in our *negative* results make them indeed *stronger*.

We now describe our main result about polynomial-time poisoning attacks. See Theorem 3.6 for the formal version of the following theorem.

Theorem 1.2 (Informal: polynomial-time poisoning attacks)**.** *Let $\mathcal{P}$ be a classification problem with a deterministic learner L that is given m labeled examples of the form $(x, c(x))$ for a concept function c (determining the ground truth).*

- **Decreasing confidence.** *For any risk threshold $\varepsilon \in [0,1]$, let ρ be the probability that L produces a hypothesis of risk at most ε, referred to as the ε-confidence of L. If ρ is at most $1 - \Omega(1)$, then there is a* polynomial-time *adversary that replaces at most $\approx O(\sqrt{m})$ of the training examples with other correctly classified examples and makes the ε-confidence go down to any constant $O(1) \approx 0$.*

- **Increasing chosen-instance[3] error.** *For any fixed test instance x, if the average error of the hypotheses generated by L over instance x is at least $\Omega(1)$, then there is a* polynomial-time *adversary that replaces at most $\approx O(\sqrt{m})$ of the training examples with other correctly classified examples and increases this average error to any constant ≈ 1.*

Moreover, both attacks above are online *and* black-box.

Generalization to arbitrary predicates. More generally, and similarly to the information theoretic attacks of [MDM19], the two parts of Theorem 1.2 follow as special cases of a more general result, in which the adversary has a particular efficiently checkable *predicate* in mind defined over the hypothesis (e.g., mislabelling on a particular x or having more than ε risk). We show that the adversary can significantly increase the probability of this bad event if it originally happens with any (arbitrary small) constant probability.

Our negative results do not contradict the recent successful line of work started by [DKK+16, LRV16], as in our setting, we start with an initial required error in the no-attack scenario and show that any such seemingly benign vulnerability (of say probability $1/1000$) can be significantly amplified.

[2]As mentioned, we need to give our adversary oracle access to a sampler for the instance distribution $\mathbf{x}$ as well, though this distribution is usually polynomial-time samplable.

[3]Poisoning attacks in which the instance is chosen are also called *targeted* [BNS+06].

Other features of our poisoning attacks. Similarly to the previous attacks of Mahloujifar et al. [MM17, MDM19, MMM18], both poisoning attacks of Theorem 1.2 have the following features.

1. Our attacks are online; i.e., during the attack, the adversary is only aware of the training examples sampled *so far* when it decides about the next tampering decision. So, these attacks can be launched against online learners in a way that the tampering happens concurrently with the learning process (see [WC18] for an in-depth study of attacks against online learners). The information theoretic attacks of [MDM19] were "off-line" as the adversary needed the full training sequence before attacking.
2. Our attacks only use *correct labels* for instances that they inject to the training set (see [SHN+18] where attacks of this form are studied in practice).
3. Our attacks are black-box [PMG+17], as they use the learning algorithm L and concept c as oracles.

Further related work. Computational constraints for robust learning were previously considered by the works of Mahloujifar et al. [MM17, MDM18] for poisoning attacks and Bubeck et al. [BPR18] for adversarial examples (i.e., evasion attacks). The works of [MM17, MDM18] studied so called "p-tampering" attacks that are online poisoning attacks in which each incoming training example could become tamperable with independent probability p and even in that case the adversary can substitute them with other *correctly labeled* examples. (The independent probabilities of tampering makes p-tampering attacks a special form of Valiant's malicious noise model [Val85].) The works of [MM17, MDM18] showed that for an initial constant error μ, *polynomial-time* p-tampering attacks can decrease the confidence of the learner or alternatively increase a chosen instance's error by $\Omega(\mu \cdot p)$. Therefore, in order to increase the (chosen instance) error to 50%, their attacks needed to tamper with a *linear* number of training examples. The more recent work of Mahloujifar et al. [MDM19] improved this attack to use only a sublinear $\sqrt{m}$ number of tamperings at the cost of only achieving information theoretic (exponential time) attacks. In this work, we get the best of both worlds, i.e., polynomial-time poisoning attacks of sublinear tampering budget.

The recent work of Bubeck, Price, and Razenshteyn [BPR18] studied whether the difficulty of finding robust classifiers is due to information theoretic barriers or that it is due to computational constraints. Indeed, they showed that (for a broad range of problems with minimal conditions) *if* we assume the existence of robust classifiers then polynomially many samples would contain enough information for guiding the learners towards one of those robust classifiers, even though as shown by Schmidt at al. [SST+18] this could be provably a larger sample complexity than the setting with no attacks. However, [BPR18] showed that *finding* such classifier might not be efficiently feasible, where efficiency here is enforced by Kearns' statistical query (SQ) model [Kea98]. So, even though our work and the work of [BPR18] both study computational constraints, the work of [BPR18] studied barriers against *efficiently finding* robust classifiers, while we study whether or not robust classifiers exist at all in the presence of *computationally efficient* attackers. In fact, in [DKS17] similar computational barriers were proved against achieving robustness in the poisoning attacks in the SQ model (i.e., information-theoretic optimal accuracy cannot be achieved by an efficient learning SQ algorithm). However, as mentioned, in this work we are focusing on the *efficiency of the attacker* and ask whether or not such computational limitation could be leveraged for robust learning.

Other related definitions of adversarial examples. In both of our results (for poisoning and evasion attacks), we use definitions that require *misclassification* of the test instance as the main goal of the adversary. However, other definitions of adversarial examples are proposed in the literature that coincide with this definition under natural conditions for practical problems of study (such as image classification).

Corrupted inputs. Feige, Mansour, and Schapire [FMS15] (and follow-up works of [FMS18, AKM18]) studied learning and inference in the presence of *corrupted inputs*. In this setting, the adversary can corrupt the test instance x to another instance x' chosen from "a few" possible corrupted versions, and then the classifier's job is to predict the label of the *original* uncorrected instance x by getting x' as input.[4] Inspired by robust optimization [BTEGN09], the more recent works of Madry et al. [MMS+18] and Schmidt et al. [SST+18] studied adversarial loss (and risk) for corrupted inputs in metric spaces. (One major difference is that now the number of possible corrupted inputs could be huge.)

Prediction change. Some other works (e.g., [SZS+14, FFF18]) only compare the prediction of the hypothesis over the adversarial example with its own prediction on the honest example (and so their definition is independent of the ground truth c). Even though in many natural settings these definitions become very close, in order to prover our formal theorems we use a definition that is based on the "error region" of the hypothesis in comparison with the ground truth that is implicit in [GMF+18, BPR18] and in [MM17, MDM18] in the context of poisoning attacks. We refer the reader to the work of Diochnos, Mahloujifar, and Mahmoody [DMM18] for a taxonomy of these variants and further discussion.

1.2 Technique: Computational Concentration of Measure in Product Spaces

In order to prove our Theorems 1.1 and 1.2, we make use of ideas developed in a recent beautiful work of Kalai, Komargodski and Raz. [KKR18] in the context of attacking coin tossing protocols. In a nutshell, our proofs proceed by first designing new polynomial-time coin-tossing attacks by first carefully changing the model of [KKR18], and then we show how such coin tossing attacks can be used to obtain evasion and poisoning attacks. Our new coin tossing attacks could be interpreted as polynomial-time algorithmic proofs for concentration of measure in product distributions under Hamming distance. We can then use such algorithmic proofs instead of the information theoretic concentration results used in [MDM19].

To describe our techniques, it is instructive to first recall the big picture of the polynomial-time poisoning attacks of [MM17, MDM19], even though they needed linear perturbations, before describing how those ideas can be extended to obtain stronger attacks with sublinear perturbations in both evasion and poisoning contexts. Indeed, the core idea there is to model the task of the adversary by a Boolean function $f(\overline{u})$ over the training data $\overline{u} = (u_1, \ldots, u_m)$, and roughly speaking define $f(\overline{u}) = 1$ if the training process over $\overline{u}$ leads to a misclassification by the hypothesis (on a chosen instance) or "low confidence" over the produced hypothesis. Then, they showed how to increase the expected value of any such f from an initial constant value μ to $\mu' \approx \mu + p$ by tampering with p fraction of "blocks" of the input sequence $(u_1, \ldots, u_m)$.

The more recent work of [MDM19] improved the bounds achieved by the above poisoning attacks by using an *computationally unbounded* attack who is *more efficient* in its tampering budget and only tampers with a sublinear $\approx \sqrt{m}$ number of the m training examples and yet increase the average of f from $\mu = \Omega(1)$ to $\mu' \approx 1$. The key idea used in [MDM19] was to use the concentration of measure in product probability spaces under the Hamming distance [AM80, MS86, McD89, Tal95]. Namely, it is known that for any product space of dimension m (here, modeling the training sequence that is iid sampled) and any initial set $\mathcal{S}$ of constant probability (here, $\mathcal{S} = \{\overline{u} \mid f(\overline{u}) = 1\}$, "almost all" of the points in the product space are of distance $\leq O(\sqrt{m})$ from $\mathcal{S}$, and so the measure is concentrated around $\mathcal{S}$.

Computational concentration of measure. In a concentrated spaces (e.g., in normal Lévy families) of dimension n, for any sufficiently large set $\mathcal{S}$ (of, say constant measure) the "typical" minimum distance of the space points to $\mathcal{S}$ is sublinear $o(n)$ ($O(\sqrt{n})$ in normal Lévy families). A computational version of this statement shall find such "close" points in $\mathcal{S}$ in polynomial time. The main technical contribution of

[4]The work of [MRT15] also studied robust inference, but with static corruption in which the adversary chooses its corruption before seeing the test instance.

our work is to prove such computational concentration of measure for any product distribution under the Hamming distance. Namely, we prove the following result about biasing Boolean functions defined over product spaces using polynomial time tampering algorithms. (See Theorem 3.1 for a formal variant.)

Theorem 1.3 (Informal: computational concentration of products)**.** *Let $\overline{\mathbf{u}} \equiv \mathbf{u}_1 \times \ldots \mathbf{u}_n$ be any product distribution of dimension n and let $f \colon \operatorname{Supp}(\overline{\mathbf{u}}) \mapsto \{0,1\}$ be any Boolean function with expected value $\mathbb{E}[f(\overline{\mathbf{u}})] = \Omega(1)$. Then, there is a* polynomial-time *tampering adversary who only tampers with $O(\sqrt{n})$ of the blocks of a sample $\overline{u} \leftarrow \overline{\mathbf{u}}$ and increases the average of f over the tampered distribution to ≈ 1.*

Once we prove Theorem 1.3, we can also use it directly to obtain *evasion* attacks that find adversarial examples, so long as the test instances are drawn from a product distribution and that the distances over the instances are measured by Hamming distance. Indeed, using concentration results (or their stronger forms of isoperimetric inequalities) was the key method used in previous works of [GMF$^+$18, FFF18, DMM18, SHS$^+$18, MDM19] to show the existence of adversarial examples. Thus, our Theorem 3.1 is a natural tool to be used in this context as well, as it simply shows that similar (yet not exactly equal) bounds to those proved by the concentration of measure can be achieved algorithmically using polynomial time adversaries.

Relation to approximate nearest neighbor search. We note that computational concentration of measure (e.g., as proved in Theorems 1.3 and 3.1 for product spaces under Hamming distance) bears similarities to the problem of "approximate nearest neighbor" (ANN) search problem [IM98, AI06, AR15, AIR18] in high dimension. Indeed, in the ANN search problem, we are given a set of points $\mathcal{P} \subseteq \mathcal{X}$ where $\mathcal{X}$ is the support set of a metric probability space (of high dimension). We then want to answer approximate near neighbor queries quickly. Namely, for a given $x \in \mathcal{X}$, in case there is a point $y \in \mathcal{P}$ where x and y are "close", the algorithm should return a point y' that is comparably close to x. Despite similarities, (algorithmic proofs of) computational concentration of measure are different in two regards: (1) In our case the set $\mathcal{P}$ could be huge, so it is not even possible to be given as input, but we rather have *implicit* access to $\mathcal{P}$ (e.g., by oracle access). (2) We are not necessarily looking for point by point approximate solutions; we only need the *average* distance of the returned points in $\mathcal{P}$ to be within some (nontrivial) asymptotic bounds.

1.2.1 Ideas behind the Proof of Theorem 1.3

The proof of our Theorem 3.1 is inspired by the recent work of Kalai et al. [KKR18] in the context of attacks against coin tossing protocols. Indeed, [KKR18] proved that in any coin tossing protocol in which n parties send a single message each, there is always an adversary who can corrupt up to $\approx \sqrt{n}$ of the players adaptively and almost fix the output to 0 or 1, making progress towards resolving a conjecture of Ben-Or and Linial [BOL89].

At first sight, it might seem that we should be able to directly use the result of [KKR18] for our purposes of proving Theorem 1.3, as they design adversaries who tamper with $\approx O(\sqrt{n})$ blocks of an incoming input and change the average of a Boolean function defined over them (i.e., the coin toss). However, there are two major obstacles against such approach. (1) The attack of [KKR18] is exponential time (as it is recursively defined over the full tree of the values of the input random process), and (2) their attack can not always *increase* the probability of a function f defined over the input, and it can only guarantee that either we will increase this average or decrease it. In fact (2) is *necessary* for the result of [KKR18], as in their model the adversary has to pick the tampered blocks *before* seeing their contents, and that there are simple functions for which we cannot choose the direction of the bias arbitrarily. Both of these restrictions are acceptable in the context of [KKR18], but not for our setting: here we want to *increase* the average of f as it represents the "error" in the learning process, and we want polynomial time biasing attacks.

Interestingly, the work of [KKR18] also presented an alternative simpler proof for a previously known result of Lichtenstein et al. [LLS89] in the context of adaptive corruption in coin tossing attacks. In that special case, the messages sent by parties only consist of single bits. In the simpler bit-wise setting, it *is* indeed possible to achieve biasing attacks that always increase the average of the output function bit. Thus, there is hope that such attacks could be adapted to our setting, and this is exactly what we do.

To prove Theorem 3.1, we do proceed as follows.

1. We give a new block-wise biasing attack, inspired by the bit-wise attack of [KKR18], that also always increases the average of the final output bit. (This is not possible for block-wise model of [KKR18].)

2. We show that this attack *can* be approximate in polynomial time. (The block-wise attack of [KKR18] seems inherently exponential time).

3. We use ideas from the bit-wise attack of [KKR18] to analyze our block-wise attack. To do this, new subtleties arise that can be handled by using stronger forms of Azuma's inequality (see Lemma 4.1) as opposed to the "basic" version of this inequality used by [KKR18] for their bit-wise attack.

Here, we describe our new attack in a simplified ideal setting in which we ignore computational efficiency. We will then compare it with the attack of [KKR18]. The attack has the form that can be adapted to computationally efficient setting by approximating the partial averages needed for the attack. See Constructions 4.3 and 4.14 for the formal description of the attack in computational settings.

Construction 1.4 (Informal: biasing attack over product distributions)**.** Let $\overline{\mathbf{u}} \equiv \mathbf{u}_1 \times \dots \mathbf{u}_n$ be a product distribution. Our (ideal model) tampering attacker IdTam is parameterized by τ. Given a sequence of blocks $(u_1, \dots, u_n)$, IdTam tampers with them by reading them one by one (starting from u_1) and decides about the tampered values inductively as follows. Suppose $v_1, \dots, v_{i-1}$ are the finalized values for the first $i-1$ blocks (after tampering decisions).

- **Tampering case 1.** If there is *some* value $v_i \in \mathrm{Supp}(\mathbf{u}_i)$ such that by picking it, the average of f goes up by at least τ for the fixed prefix $(v_1, \dots, v_{i-1})$ and for a *random* continuation of the rest of the blocks, then pick v_i as the tampered value for the i^{th} block.

- **Tampering case 2.** Otherwise, if the actual (untampered) content of the i^{th} block, namely u_i, *decreases* the average of f (under a random continuation of the remaining blocks) for the fixed prefix $(v_1, \dots, v_{i-1})$, then ignore the original block u_i, and pick some tampered value $v_i \in \mathrm{Supp}(\mathbf{u}_i)$ that v_i at least does not decrease the average. (Such v_i always exists by an elementary averaging argument.)

- **Not tampering.** If none of the above cases happen, output the original sample $v_i = u_i$.

By picking parameter $\tau \approx 1/\sqrt{n}$, we prove that the attack achieves the desired properties of Theorem 3.1; Namely, the number of tampered blocks is $\approx O(1/\tau)$, while the bias of f under attack is ≈ 1.

The bit-wise attack of [KKR18] can be seen as simpler variant of the attack above in which the adversary (also) has access to an oracle that returns the partial averages for random continuation. Namely, in their attack tampering cases 1 and 2 are combined into one: if the next *bit* can increase (or equivalently, can decrease) the partial averages of the current prefix by τ, then the adversary chooses to corrupt that bit (even without seeing its actual content). The crucial difference between the bit-wise attack of [KKR18] and our block-wise attack of Theorem 1.3 is in tampering case 2. Here we *do* look at the untampered value of the i^{th} block, and doing so is *necessary* for getting an attack in block-wise setting that biases $f(\cdot)$ towards $+1$.

Extension to general product distributions and for coin-tossing protocols. Our proof of Theorem 1.3, and its formalized version Theorem 3.1, with almost no changes extend to any *joint* distributions like $\overline{\mathbf{u}} \equiv (\mathbf{u}_1, \ldots, \mathbf{u}_n)$ under a proper definition of online tampering in which the next "untampered" block is sampled conditioned on the previously tampered blocks chosen by the adversary. This shows that in any n round coin tossing protocol in which each of the n parties sends exactly one message, there is a *polynomial-time strong* adaptive adversary who corrupts $O(\sqrt{n})$ of the parties and biases the output to be 1 with $99/100$ probability. A strong adaptive adversary, introduced by Goldwasser et al. [GKP15], allows the adversary to see the messages before they are delivered and then corrupt a party (and change their message) based on their initial messages that were about to be sent. Our result improves a previous result of [GKP15] that was proved for *one-round* protocols using *exponential time* attackers. Our attack extends to arbitrary (up to) n round protocols and is also polynomial time. Our results are incomparable to those of [KKR18]; while they also corrupt up to $O(\sqrt{n})$ of the messages, attackers do not see the messages of the parties before corrupting them, but our attackers inherently rely on this information. On the other hand, their bias is *either* towards 0 or toward 1 (for the block-wise setting) while our attacks can choose the direction of the biasing.

2 Preliminaries

General notation. We use calligraphic letters (e.g., $\mathcal{X}$) for sets. By $u \leftarrow \mathbf{u}$ we denote sampling u from the probability distribution $\mathbf{u}$. For a randomized algorithm $R(\cdot)$, by $y \leftarrow R(x)$ we denote the randomized execution of R on input x outputting y. By $\mathbf{u} \equiv \mathbf{v}$ we denote that the random variables $\mathbf{u}$ and $\mathbf{v}$ have the same distributions. Unless stated otherwise, by using a bar over a variable $\overline{u}$, we emphasize that it is a vector. By $\overline{\mathbf{u}} \equiv (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n)$ we refer to a joint distribution over vectors with n components. For a joint distribution $\overline{\mathbf{u}} \equiv (\mathbf{u}_1, \ldots, \mathbf{u}_n)$, we use $\mathbf{u}_{\leq i}$ to denote the joint distribution of the first i variables $\overline{\mathbf{u}} \equiv (\mathbf{u}_1, \ldots, \mathbf{u}_i)$. Also, for a vector $\overline{u} = (u_1 \ldots u_n)$ we use $u_{\leq i}$ to denote the prefix $(u_1, \ldots, u_i)$. For a joint distribution $(\mathbf{u}, \mathbf{v})$, by $(\mathbf{u} \mid v)$ we denote the conditional distribution $(\mathbf{u} \mid \mathbf{v} = v)$. By $\mathrm{Supp}(\mathbf{u}) = \{u \mid \Pr[\mathbf{u} = u] > 0\}$ we denote the support set of $\mathbf{u}$. By $T^{\mathbf{u}}(\cdot)$ we denote an algorithm $T(\cdot)$ with oracle access to a sampler for distribution $\mathbf{u}$ that upon every query returns a fresh sample from $\mathbf{u}$. By $\mathbf{u} \times \mathbf{v}$ we refer to the product distribution in which $\mathbf{u}$ and $\mathbf{v}$ are sampled independently. By $\mathbf{u}^n$ we denote the n-fold product $\mathbf{u}$ with itself returning n iid samples. Multiple instances of a random variable $\mathbf{x}$ in the same statement (e.g., $(\mathbf{x}, c(\mathbf{x}))$ refer to the same sample. By PPT we denote "probabilistic polynomial time".

Notation for classification problems. A classification problem $(\mathcal{X}, \mathcal{Y}, \mathbf{x}, \mathcal{C}, \mathcal{H})$ is specified by the following components. The set $\mathcal{X}$ is the set of possible *instances*, $\mathcal{Y}$ is the set of possible *labels*, $\mathbf{x}$ is a distribution over $\mathcal{X}$, $\mathcal{C}$ is a class of *concept* functions where $c \in \mathcal{C}$ is always a mapping from $\mathcal{X}$ to $\mathcal{Y}$. Even though in a learning problem, we usually work with a *family* of distributions (e.g., all distributions over $\mathcal{X}$) here we work with only one distribution $\mathbf{x}$. The reason is that our results are *impossibility* results, and proving limits of learning under a known distribution $\mathbf{x}$ are indeed stronger results. We did not state the loss function explicitly, as we work with classification problems. For $x \in \mathcal{X}, c \in \mathcal{C}$, the *risk* or *error* of a hypothesis $h \in \mathcal{H}$ is equal to $\mathsf{Risk}(h, c) = \Pr_{x \leftarrow \mathbf{x}}[h(x) \neq c(x)]$. We are usually interested in learning problems $(\mathcal{X}, \mathcal{Y}, \mathbf{x}, \mathcal{C}, \mathcal{H})$ with a specific metric d defined over $\mathcal{X}$ for the purpose of defining risk and robustness under instance perturbations controlled by metric d. Then, we simply write $(\mathcal{X}, \mathcal{Y}, \mathbf{x}, \mathcal{C}, \mathcal{H}, \mathsf{d})$ to include d.

The following definition based on the definitions given in [MM17, MDM18, MDM19].

Definition 2.1 (Confidence, chosen-instance error, and their adversarial variants)**.** Let L be a learning algorithm for a classification problem $\mathcal{P} = (\mathcal{X}, \mathcal{Y}, \mathbf{x}, \mathcal{C}, \mathcal{H})$, m be the sample complexity of L, and $c \in \mathcal{C}$ be any concept. We define the (adversarial) confidence function and chosen-instance error as follows.

- **Confidence function.** For any error function $\varepsilon = \varepsilon(m)$, the *adversarial confidence* in the presence of a adversary A is defined as

$$\mathsf{Conf}_{\mathsf{A}}(m, c, \varepsilon) = \Pr_{\substack{\overline{u} \leftarrow (\mathbf{x}, c(\mathbf{x}))^m \\ h \leftarrow L(\mathsf{A}(\overline{u}))}}[\mathsf{Risk}(h, c) < \varepsilon].$$

  By $\mathsf{Conf}(\cdot)$ we denote the confidence without any attack; namely, $\mathsf{Conf}(\cdot) = \mathsf{Conf}_I(\cdot)$ for the trivial (identity function) adversary I that does not change the training data.

- **Chosen-instance error.** For a fixed test instance $x \in \mathcal{X}$, the *chosen-instance* error (over instance x) in presence of a poisoning adversary A is defined as

$$\mathsf{Err}_{\mathsf{A}}(m, c, x) = \Pr_{\substack{\overline{u} \leftarrow (\mathbf{x}, c(\mathbf{x}))^m \\ h \leftarrow L(\mathsf{A}(\overline{u}))}}[h(x) \neq c(x)].$$

  By $\mathsf{Err}(\cdot)$ we denote the chosen-instance error (over x) without any attacks; namely, $\mathsf{Err}(\cdot) = \mathsf{Err}_I(\cdot)$ for the trivial (identity function) adversary I.

2.1 Basic Definitions for Tampering Algorithms

Our tampering adversaries follow a close model to that of p-budget adversaries defined in [MDM18]. Such adversaries, given a sequence of blocks, select at most p fraction of the locations in the sequence and change their value. The p-budget model of [MDM18] works in an online setting in which, the adversary should decide for the ith block, only knowing the first $i-1$ blocks. In this work, we define both online and offline attacks that work in a closely related budget model in which we only bound the *expected* number of tampered blocks. We find this notion more natural for the robustness of learners.

Definition 2.2 (Online and offline tampering)**.** We define the following two tampering attack models.

- **Online attacks.** Let $\overline{\mathbf{u}} \equiv \mathbf{u}_1 \times \cdots \times \mathbf{u}_n$ be an arbitrary product distribution.[5] We call a (potentially randomized and computationally unbounded) algorithm OnTam an *online tampering* algorithm for $\overline{\mathbf{u}}$, if given any $i \in [n]$ and any $u_{\leq i} \in \mathrm{Supp}(\mathbf{u}_1) \times \cdots \times \mathrm{Supp}(\mathbf{u}_i)$, it holds that

$$\Pr_{v_i \leftarrow \mathsf{OnTam}(u_{\leq i})}[v_i \in \mathrm{Supp}(\mathbf{u}_i)] = 1 \ .$$

  Namely, $\mathsf{OnTam}(u_{\leq i})$ outputs (a candidate i^{th} block) v_i in the support set of $\mathbf{u}_i$.[6]

- **Offline attacks.** For an arbitrary joint distribution $\overline{\mathbf{u}} \equiv (\mathbf{u}_1 \ldots, \mathbf{u}_n)$ (that might or might not be a product distribution), we call a (potentially randomized and possibly computationally unbounded) algorithm OffTam an *offline tampering* algorithm for $\overline{\mathbf{u}}$, if given any $\overline{u} \in \mathrm{Supp}(\overline{\mathbf{u}})$, it holds that

$$\Pr_{\overline{v} \leftarrow \mathsf{OnTam}(\overline{u})}[\overline{v} \in \mathrm{Supp}(\overline{\mathbf{u}})] = 1 \ .$$

  Namely, given any $\overline{u} \leftarrow \overline{\mathbf{u}}$, $\mathsf{OnTam}(\overline{u})$ always outputs a vector in $\mathrm{Supp}(\overline{\mathbf{u}})$.

[5]We restrict the case of online attacks to product distribution as they will have simpler notations and that they cover our main applications, however they can be generalized to arbitrary joint distributions as well with a bit more care.

[6]Looking ahead, this restriction makes our attacks stronger in the case of poisoning attacks by always picking correct lables during the attack.

- **Efficiency of attacks.** If $\overline{\mathbf{u}}$ is a joint distribution coming from a *family* of distributions (perhaps based on the index $n \in N$), we call an online or offline tampering algorithm *efficient*, if its running time is $\mathrm{poly}(N)$ where N is the total bit length of any $\overline{u} \in \mathrm{Supp}(\overline{\mathbf{u}})$.

- **Notation for tampered distributions.** For any joint distribution $\overline{\mathbf{u}}$, any $\overline{u} \leftarrow \overline{\mathbf{u}}$, and for any tampering algorithm Tam, by $\langle \overline{u} \parallel \mathsf{Tam} \rangle$ we refer to the distribution obtained by running Tam over $\overline{u}$, and by $\langle \overline{\mathbf{u}} \parallel \mathsf{Tam} \rangle$ we refer to the final distribution by also sampling $\overline{u} \leftarrow \overline{\mathbf{u}}$ at random. More formally,
  - For an offline tampering algorithm OffTam, the distribution $\langle \overline{u} \parallel \mathsf{OffTam} \rangle$ is sampled by simply running OffTam on the whole $\overline{u}$ and obtaining the output $(v_1, \dots, v_n) \leftarrow \mathsf{OffTam}(u_1, \dots, u_n)$.
  - For an online tampering algorithm OnTam and input $\overline{u} = (u_1, \dots, u_n)$ sampled from a *product* distribution $\mathbf{u}_1 \times \dots \mathbf{u}_n$, we obtain the output $(v_1, \dots, v_n) \leftarrow \langle \overline{u} \parallel \mathsf{OnTam} \rangle$ *inductively*: for $i \in [i]$, sample $v_i \leftarrow \mathsf{OnTam}(v_1, \dots, v_{i-1}, u_i)$.[7]

- **Average budget of tampering attacks.** Suppose d is a metric defined over $\mathrm{Supp}(\overline{\mathbf{u}})$. We say an online or offline tampering algorithm Tam has *average budget* (at most) b, if

$$\mathop{\mathbb{E}}_{\substack{\overline{u} \leftarrow \overline{\mathbf{u}}, \\ \overline{v} \leftarrow \langle \overline{u} \parallel \mathsf{Tam} \rangle}} [\mathsf{d}(\overline{u}, \overline{v})] \le b.$$

  If no metric d is specified, we use Hamming distance over vectors of dimension n.

3 Polynomial-time Attacks from Computational Concentration of Products

In this section, we will first formally state our main technical tool, Theorem 3.1, that underlies our polynomial-time evasion and poisoning attacks. Namely, we will prove that product distributions are "computationally concentrated" under the Hamming distance, in the sense that any subset with constant probability, is "computationally close" to most of the points in the probability space. We will then use this tool to obtain our attacks against learners. We will finally prove our main technical tool.

Theorem 3.1 (Computational concentration of product distributions)**.** *Let $\overline{\mathbf{u}} \equiv \mathbf{u}_1 \times \cdots \times \mathbf{u}_n$ be any product distribution and $f \colon \mathrm{Supp}(\overline{\mathbf{u}}) \mapsto \{0,1\}$ be any Boolean function over $\overline{\mathbf{u}}$, and let $\mu = \mathbb{E}[f(\overline{\mathbf{u}})]$ be the expected value of f. Then, for any ρ where $\mu < \rho < 1$, there is an* online *tampering algorithm* OnTam *generating the tampering distribution $\overline{\mathbf{v}} \equiv \langle \overline{\mathbf{u}} \parallel \mathsf{OnTam} \rangle$ with the following properties.*

1. **Achieved bias.** $\mathbb{E}[f(\overline{\mathbf{v}})] \ge \rho$.
2. **Efficiency.** *Having oracle access to f and a sampler for $\overline{\mathbf{u}}$, $\mathsf{OnTam} = \mathsf{OnTam}^{f,\overline{\mathbf{u}}}$ runs in time $\mathrm{poly}\left(\frac{n \cdot \ell}{\mu \cdot (1-\rho)}\right)$ where ℓ is the maximum bit length of any $u_i \in \mathrm{Supp}(\mathbf{u}_i)$ for any $i \in [n]$.*
3. **Average budget.** $\mathsf{OnTam} = \mathsf{OnTam}^{f,\overline{\mathbf{u}}}$ *uses average budget* $(2/\mu) \cdot \sqrt{n \cdot \ln(2/(1-\rho))}$.

In the rest of this section, we will use Theorem 3.1 to prove limitations of robust learning in the presence of polynomial-time poisoning and evasion attackers. We will prove Theorem 3.1 in the next section.

[7]By limiting our online attackers to product distributions, we can sample the whole sequence of "untampered" values $(u_1, \dots, u_n)$ at the beginning; otherwise, for general random processes in which the distribution of blocks are correlated, we would need to sample $(u_1, \dots, u_n)$ and $(v_1, \dots, v_n)$ *jointly* by sampling u_i *conditioned on* $v_1, \dots, v_{i-1}$.

Range of initial and target error covered by Theorem 3.1. For any $(1-\rho) = 1/\operatorname{poly}(n), \mu = O(1/\operatorname{polylog} n)$ Theorem 3.1 uses an average budget of only $\widetilde{O}(\sqrt{n})$. If we start from larger initial error that is still bounded by $\mu = o(1/\sqrt{n})$, the average budget given by the attacker of Theorem 3.1 will still be $o(n)$, which is nontrivial as it is still sublinear in the dimension. However, if we start from $\mu = \Omega(1/\sqrt{n})$, we the attacker of Theorem 3.1 stops to give a nontrivial bound, as the required linear $\Omega(n)$ budget is enough for getting any target error trivially. In contrast, the information theoretic attacks of [MDM19] can handle much smaller initial error all the way to subexponentially small μ. Finding the maximum range of μ for which computationally bounded attackers can increase the error to $1 - 1/\operatorname{poly}(n)$ remains open.

3.1 Polynomial-time Evasion Attacks

The following definition of robustness against adversarial perturbations of the input is based on the previous definitions used in [GMF$^+$18, BPR18, DMM18, MDM19] in which the adversary aims at *misclassification* of the adversarially perturbed instance by trying to push them into the error region.

We define the following definition for a fixed distribution $\mathbf{x}$ (as our negative results are for simplicity stated for such cases) but a direct generalization can be obtained for any *family* of distributions over the instances. Moreover, we only give a definition for the "black-box" type of attacks (again because our attacks are black-box) but a more general definition can be given for non-black-box attacks as well.

Definition 3.2 (Computational (error-region) evasion robustness)**.** Let $\mathcal{P} = (\mathcal{X}, \mathcal{Y}, \mathbf{x}, \mathcal{C}, \mathcal{H}, \mathsf{d})$ be a classification problem. Suppose the components of $\mathcal{P}$ are indexed by $n \in \mathbb{N}$, and let $0 < \mu(n) < \rho(n) \leq 1$ for functions $\mu(n)$ and $\rho(n)$ that for simplicity we denote by μ and ρ. We say that the μ-to-ρ *evasion robustness* of $\mathcal{P}$ is at most $b = b(n)$, if there is a (perhaps computationally unbounded) tampering oracle algorithm $\mathsf{A}^{(\cdot)}$ such that for all $h \in \mathcal{H}, c \in \mathcal{C}$ with error region $\mathcal{E} = \mathcal{E}(h, c), \Pr[\mathbf{x} \in \mathcal{E}] \geq \mu$, we have the following.

1. Having oracle access to h, c and a sampler for $\mathbf{x}$, the oracle adversary $\mathsf{A} = \mathsf{A}^{h,c,\mathbf{x}}(x)$ reaches adversarial risk to at least ρ (for the choice of c, h). Namely, $\Pr_{x \leftarrow \mathbf{x}}[\mathsf{A}^{h,c,\mathbf{x}}(x) \in \mathcal{E}] \geq \rho$.

2. The average budget of the adversary $\mathsf{A} = \mathsf{A}^{h,c,\mathbf{x}}$ (with oracle access to h, c and a sampler for $\mathbf{x}$) is at most b for samples $x \leftarrow \mathbf{x}$ and with respect to metric d.

The μ-to-ρ *computational* evasion robustness of $\mathcal{P}$ is at most $b = b(n)$, if the same statement holds for an *efficient* (i.e., PPT) oracle algorithm A.

Evasion robustness of *problems* vs. that of *learners*. Computational evasion robustness as defined in Definition 3.2 directly deals with learning problems regardless of what learning algorithm is used for them. The reason for such a choice is that in this work, we prove *negative* results demonstrating the *limitations* of computational robustness. Therefore, limiting the robustness of a learning problems *regardless* of their learner is a stronger result. In particular, any negative result (i.e., showing attackers with small tampering budget) about μ-to-ρ (computational) robustness of a learning problem $\mathcal{P}$, immediately implies that any learning algorithm L for $\mathcal{P}$ that produces hypothesis with risk $\approx \mu$ can always be attacked (efficiently) to reach adversarial risk ρ.

Now we state and prove our main theorem about evasion attacks. Note that the proof of this theorem is identical to the reduction shown in [MDM19]. The difference is that, instead of using original concentration inequalities, we use our new results about *computational* concentration of product measures under hamming distance and obtain attacks that work in polynomial time.

Theorem 3.3 (Limits on computational evasion robustness)**.** *Let $\mathcal{P} = (\mathcal{X}, \mathcal{Y}, \mathbf{x}, \mathcal{C}, \mathcal{H}, \mathsf{d})$ be a classification problem in which the instances' distribution $\mathbf{x} \equiv \mathbf{u}_1 \times \cdots \times \mathbf{u}_n$ is a product distribution of dimension n and d is the Hamming distance over vectors of dimension n. Let $0 < \mu = \mu(n) < \rho = \rho(n) \le 1$ be functions of n. Then, the μ-to-ρ* computational *evasion robustness of $\mathcal{P}$ is at most*

$$b = (2/\mu) \cdot \sqrt{n \cdot \ln(2/(1-\rho))}.$$

In particular, if $\mu(n) = \omega(\log n/\sqrt{n})$ and $\rho(n) = 1 - 1/\operatorname{poly}(n)$, then $b = o(n)$ is sublinear in n, and if $\mu(n) = \Omega(1/\operatorname{polylog}(n))$ and $\rho(n) = 1 - 1/\operatorname{poly}(n)$, then $b = \widetilde{O}(\sqrt{n})$.

Proof. We first define a Boolean function $f \colon : \mathcal{X} \to [0, 1]$ as follows:

$$f(x) = \begin{cases} 1 & c(x) \neq h(x), \\ 0 & c(x) = h(x). \end{cases}$$

It is clear that $\mathbb{E}[f(\mathbf{x})] = \Pr[\mathbf{x} \in \mathcal{E}] \ge \mu$. Therefore, by using Theorem 3.1, we know there is an tampering algorithm $A_\mu^{f,\mathbf{x}}$ that runs in time $\operatorname{poly}(n \cdot \ell/\mu \cdot (1-\rho))$ and increases the average of f to ρ while using average budget at most $(2/\mu) \cdot \sqrt{n \cdot \ln(2/(1-\rho))}$. Note that A needs oracle access to $f(\cdot)$ which is computable by oracle access to $h(\cdot)$ and $c(\cdot)$. □

Remark 3.4 (Computationally bounded prediction-change evasion attacks)**.** As we mentioned in the introduction, some works studying adversarial examples (e.g., [SZS+14, FFF18]) study robustness by only comparing the prediction of the hypothesis over the adversarial example with its own prediction on the honest example, and so their definition is independent of the ground truth c. (In the terminology of [DMM18], such attacks are called *prediction-change* attacks.) Here we point out that our biasing attack of Theorem 3.1 can be used to prove limits on the robustness against such evasion attacks as well. In particular, in [MDM19], it was shown that using concentration of measure, one can obtain existential (information theoretic) prediction-change attacks (even of the "targeted" form in which the target label is selected). By combining the arguments of [MDM19] and plugging in our computationally bounded attack of Theorem 3.1 one can obtain impossibility results for basing the robustness of hypotheses on computational hardness.

3.2 Polynomial-time Poisoning Attacks

The following definition formalizes the notion of robustness against computationally bounded poisoning adversaries. Our definition is based on those of [MM17, MDM18] who studied online poisoning attacks and that of [MDM19] who studied offline poisoning attacks.

Definition 3.5 (Computational poisoning robustness)**.** Let $\mathcal{P} = (\mathcal{X}, \mathcal{Y}, \mathbf{x}, \mathcal{C}, \mathcal{H})$ be a classification problem with a learner L of sample complexity m. Let $0 < \mu = \mu(m) < \rho = \rho(m) \le 1$ be functions of m.

- **Computational confidence robustness.** For $\varepsilon = \varepsilon(m)$, we say that the ρ-to-μ *ε-confidence robustness* of the learner L is at most $b = b(m)$, if there is a (computationally unbounded) tampering algorithm A such that for all $c \in \mathcal{C}$ for which $\mathsf{Conf}(m, c, \varepsilon) \le \rho$, the following two conditions hold.
  1. The average budget of $\mathsf{A} = \mathsf{A}^{L,c,\mathbf{x}}$ (who has oracle access to L, c and a sampler for $\mathbf{x}$) tampering with the distribution $(\mathbf{x}, c(\mathbf{x}))^m$ is at most b.

2. The adversarial confidence for $\varepsilon' = 99 \cdot \varepsilon/100$ is at most $\mathsf{Conf}_{\mathsf{A}}(m, c, \varepsilon') \leq \mu$ when attacked by the oracle adversary $\mathsf{A} = \mathsf{A}^{L,c,\mathbf{x}}$.[8]

The ρ-to-μ *computational* ε-confidence robustness of the learner L is at most $b = b(n)$, if the same statement holds for an *efficient* (i.e., PPT) oracle algorithm A .

- **Computational chosen-instance robustness.** For an instance $x \leftarrow \mathbf{x}$, we say that the μ-to-ρ *chosen-instance robustness* of the learner L for x is at most $b = b(m)$, if there is a (computationally unbounded) tampering oracle algorithm A (that could depend on x) such that for all $c \in \mathcal{C}$ for which $\mathcal{E}(m, c, x) \geq \mu$, the following two conditions hold.

  1. The average budget of $\mathsf{A} = \mathsf{A}^{L,c,\mathbf{x}}$ (who has oracle access to L, c and a sampler for $\mathbf{x}$) tampering with the distribution $(\mathbf{x}, c(\mathbf{x}))^m$ is at most b.
  2. Adversary $\mathsf{A} = \mathsf{A}^{L,c,\mathbf{x}}$ increases the chosen-instance error to $\mathsf{Err}_{\mathsf{A}}(m, c, x) \geq \rho$.

  The μ-to-ρ *computational* chosen-instance robustness of the learner L for instance x is at most $b = b(n)$, if the same thing holds for an *efficient* (i.e., PPT) oracle algorithm A.

Now we state and prove our main theorem about poisoning attacks. Again, the proof of this theorem is identical to the reduction from shown in [MDM19]. The difference is that here we use our new results about *computational* concentration of product measures under hamming distance and get attacks that work in polynomial time. Another difference is that our attacks here are online due the online nature of our martingale attacks on product measures.

Theorem 3.6 (Limits on computational poisoning robustness). *Let $\mathcal{P} = (\mathcal{X}, \mathcal{Y}, \mathbf{x}, \mathcal{C}, \mathcal{H})$ be a classification problem with a* deterministic polynomial-time *learner L. Let $0 < \mu = \mu(m) < \rho = \rho(m) \leq 1$ be functions of m, where m is the sample complexity of L.*

- **Confidence robustness.** *Let $\varepsilon = \varepsilon(m) \geq 1/\operatorname{poly}(m)$ be the risk threshold defining the confidence function. Then, the ρ-to-μ* computational *ε-confidence robustness of the learner L is at most $b = (2/(1-\rho)) \cdot \sqrt{m \cdot \ln(2/\mu)}$.*

- **Chosen-instance robustness.** *For any instance $x \leftarrow \mathbf{x}$, the μ-to-ρ* computational *chosen-instance robustness of the learner L for x is at most $b = (2/\mu) \cdot \sqrt{m \cdot \ln(2/(1-\rho))}$.*

In particular, in both cases above if $\mu(m) = \omega(\log m/\sqrt{m})$ and $\rho(m) = 1 - 1/\operatorname{poly}(m)$, then $b = o(m)$ is sublinear in m, and if $\mu(m) = \Omega(1/\operatorname{poly}(\log m))$ and $\rho(m) = 1 - 1/\operatorname{poly}(m)$, then $b = \widetilde{O}(\sqrt{m})$.

Moreover, the polynomial time attacker A bounding the computational poisoning robustness in both cases above has the following features: **(1)** *A is online, and* **(2)** *A is plausible; namely, it never uses any wrong labels in its poisoned training data.*

Proof. We first prove the case of chosen-instance robustness. We define a Boolean function $f\colon \mathcal{X}^m \to [0,1]$ as follows:

$$f_1(x_1, \ldots, x_m) = \begin{cases} 1 & h = L((x_1, c(x_1)), \ldots, (x_n, c(x_m)) \wedge h(x) \neq c(x), \\ 0 & h = L((x_1, c(x_1)), \ldots, (x_n, c(x_m)) \wedge h(x) = c(x). \end{cases}$$

[8]The computationally-unbounded variant of this definition as used in [MDM19] uses $\varepsilon' = \varepsilon$ instead of $\varepsilon' = 99 \cdot \varepsilon/100$, but as observed by [MDM18], due to the computational bounded nature of our attack we need to have a small gap between ε' and ε.

It is clear that $\mathbb{E}[f_1(\mathbf{x}^m)] = \mathcal{E}(m,c,x) \geq \mu$. Therefore, by using Theorem 3.1, we know there is a PPT tampering Algorithm $A_2^{f_1(\cdot),\mu}$ that runs in time $\mathrm{poly}(m \cdot \ell/(\mu \cdot (1-\rho)))$, and increase the average of f_1 to ρ while using average budget at most $(2/\mu) \cdot \sqrt{m \cdot \ln(2/(1-\rho))}$. Note that A_1 needs oracle access to $f_1(\cdot)$ which is computable by oracle access to the learning algorithm $L(\cdot)$ and concept $c(\cdot)$. Now we prove the case of confidence robustness. Again we define a Boolean function $f_2\colon \mathcal{X}^m \to [0,1]$ as follows:

$$f_2(x_1,\dots,x_m) = \begin{cases} 1 & h = L((x_1,c(x_1)),\dots,(x_n,c(x_m)) \wedge \Pr[h(\mathbf{x}) \neq c(\mathbf{x})] \geq \varepsilon, \\ 0 & h = L((x_1,c(x_1)),\dots,(x_n,c(x_m)) \wedge \Pr[h(\mathbf{x}) \neq c(\mathbf{x})] < \varepsilon. \end{cases}$$

We have $\mathbb{E}[f_2(\mathbf{x}^m)] = 1 - \mathsf{Conf}(m,c,\varepsilon) \geq 1-\rho$. Therefore, by using Theorem 3.1, we know there is a PPT tampering Algorithm $A_2^{f_2(\cdot),\mu}$ that runs in time $\mathrm{poly}(m \cdot \ell/(1-\rho)\cdot\mu)$, and increase the average of f_2 to $1-\mu$ while using average budget at most $(2/(1-\rho)) \cdot \sqrt{m \cdot \ln(2/\mu)}$. Note that A_2 needs oracle access to $f_2(\cdot)$, which requires the adversary to know the exact error of a hypothesis. Computing the exact error is not possible in polynomial time but using an emprical estimator, the adversary can find an approximation of the error which is sufficient for the attack (See Corollary 3 of [MDM18]). □

4 Products are Computationally Concentrated under Hamming Distance

In this section, we formally prove Theorem 3.1. For simplicity of presentation, we will prove the following theorem for product distributions $\overline{\mathbf{u}} \equiv \mathbf{u}^n$ over the same $\mathbf{u}$, but the same proof directly holds for more general case of $\overline{\mathbf{u}} \equiv \mathbf{u}_1 \times \dots \mathbf{u}_n$.

We will first present an attack in an idealized model in which the adversary has access to some promised oracles that approximate certain properties of the function f in a carefully defined way. In this first step, we indeed show that our attack (and its proof) are robust to such approximations. We then show that these promised oracles can be obtained with high probability, and by doing so we obtain the final polynomial time biasing attack proving the concentration of product distributions under Hamming distance.

4.1 Biasing Attack Using Promised Approximate Oracles

We first state a usefull lemma that is similar to Azuma inequality but works with approximate martingales.

Lemma 4.1 (Azuma's inequality for approximate conditions). *Let $\overline{\mathbf{t}} \equiv (\mathbf{t}_1,\dots,\mathbf{t}_n)$ be a sequence of n jointly distributed random variables such that for all $i \in [n]$, $\Pr[|\mathbf{t}_i| > \tau] \leq \gamma$ and for all $t_{\leq i-1} \leftarrow \mathbf{t}_{\leq i-1}$, we have $\mathbb{E}[\mathbf{t}_i \mid t_{\leq i-1}] \geq -\gamma$. Then, we have*

$$\Pr\left[\sum_{i=1}^{n} \mathbf{t}_i \leq -s\right] \leq \mathrm{e}^{\frac{-(s-n\cdot\gamma)^2}{2n\cdot(\tau+\gamma)^2}} + n \cdot \gamma$$

Proof. If we let $\gamma = 0$, Lemma 4.1 becomes the standard version of Azuma inequality. Here we sketch why Lemma 4.1 can also be reduced to the case that $\gamma = 0$ (i.e., Azuma inequality). We build a sequence $\mathbf{t}'_i$ from $\mathbf{t}_i$ as follows: Sample $t_i \leftarrow \mathbf{t}_i \mid t'_{\leq i-1}$, if $|t_i + \gamma| \leq \tau + \gamma$, output $t'_i = t_i + \gamma$. Otherwise output 0. We clearly have $\mathbb{E}[\mathbf{t}'_i \mid t'_{\leq i-1}] \geq 0$ and $\Pr[|\mathbf{t}'_i| \geq \tau+\gamma] = 0$. Now we can use Lemma 4.1 for the basic case of $\gamma = 0$ for the sequence $\mathbf{t}'_i$ and use it to get a looser bound for sequence $\mathbf{t}_i$, using the fact that $\exists i \in [n], |t_i| \geq \tau$ happens with probability at most $n \cdot \gamma$. □

Now we define some oracle functions that our tampering attack is based on.

Definition 4.2 (Notation for oracles)**.** Suppose $f\colon \operatorname{Supp}(\overline{\mathbf{u}}) \mapsto \mathbb{R}$ is defined over a product distribution $\overline{\mathbf{u}} \equiv \mathbf{u}_1 \times \cdots \times \mathbf{u}_n$ of dimension n. Then, given a specific parameter $\gamma \in [0,1]$ we define the following *promise* oracles for any $i \in [n]$ and any $u_{\leq i} \in \operatorname{Supp}(\mathbf{u}_{\leq i})$. Namely, our promise oracles could be one out of any oracles that satisfy the following guarantees.

- Oracle $\mathsf{a}(u_{\leq i})$ returns the average gain conditioned on the given prefix:

$$\mathsf{a}(u_{\leq i}) = \mathop{\mathbb{E}}_{(u_{i+1},\ldots,u_n)\leftarrow \mathbf{u}_{i+1}\times\cdots\times\mathbf{u}_n}[f(u_1,\ldots,u_n)].$$

- Oracle $\mathsf{g}(u_{\leq i})$ returns the gain on the average in the last block and is defined as

$$\mathsf{g}(u_{\leq i}) = \mathsf{a}(u_{\leq i}) - \mathsf{a}(u_{\leq i-1}).$$

- Oracle $\tilde{\mathsf{g}}(\cdot)$ approximates the gain of average in the last block, $|\tilde{\mathsf{g}}(u_{\leq i}) - \mathsf{g}(u_{\leq i})| \leq \gamma$.

- Oracle $\tilde{\mathsf{g}}^*(\cdot)$ returns the approximate maximum gain with two promised properties:

$$\begin{aligned} &\text{[Property A:]} \quad \Pr_{u_i \leftarrow \mathbf{u}}\left[\mathsf{g}(u_{\leq i}) > \tilde{\mathsf{g}}^*(u_{\leq i-1}) + 2\gamma\right] < \gamma, \\ &\text{[Property B:]} \qquad\qquad \tilde{\mathsf{g}}^*(u_{\leq i-1}) \geq -2\gamma. \end{aligned}$$

- Oracle $\tilde{\mathsf{h}}(\cdot)$ returns a sample producing the approximate maximum gain $\tilde{\mathsf{g}}^*(\cdot)$. Namely,

$$\tilde{\mathsf{g}}^*(u_{\leq i-1}) = \tilde{\mathsf{g}}(u_{\leq i-1}, \tilde{\mathsf{h}}(u_{\leq i-1})).$$

Following is the construction of our tampering attack based on the oracles defined above.

Construction 4.3 (Attack using promised approximate oracles)**.** For a product distribution $\overline{\mathbf{u}} \equiv \mathbf{u}^n$ and $\tau \in [0,1]$, our (online) efficient tampering attacker $\mathsf{AppTam}_{(\tau,\gamma)}$ is parameterized by $\tau, \gamma \in [0,1]$, but for simplicity it will be denoted as AppTam. The parameter γ determines the approximation promised by the oracles used by AppTam. Given $(v_1,\ldots,v_{i-1},u_i) \in \operatorname{Supp}(\mathbf{u})^i$ as input, AppTam will output some $v_i \in \operatorname{Supp}(\mathbf{u})$. Let $w_i = \tilde{\mathsf{h}}(v_{\leq i-1})$ as defined in Definition 4.2. AppTam chooses its output v_i as follows.

- **Tampering.** If $\tilde{\mathsf{g}}^*(v_{\leq i-1}) \geq \tau$ or if $\tilde{\mathsf{g}}(v_{\leq i-1}, u_i) \leq -\tau$, then output $v_i = w_i$.
- **Not tampering.** Otherwise, output the original sample $v_i = u_i$.

Before proving the bounds, we first define some events based on the conditions/cases that happen during the tampering attack of Construction 4.3.

Definition 4.4. We define the following three Boolean functions over $\cup_{i=1}^n \operatorname{Supp}(\mathbf{u})^i$ based on the actions taken by the tampering algorithm of Construction 4.3. Namely, for any $(v_{\leq i-1}, u_i) \in \operatorname{Supp}(\mathbf{u})^i$, we define

$$\mathrm{C}_1(v_{\leq i-1}) = \begin{cases} 1 & \text{if } \tilde{\mathsf{g}}^*(v_{\leq i-1}) \geq \tau, \\ 0 & \text{otherwise;} \end{cases}$$

$$\mathrm{C}_2(v_{\leq i-1}, u_i) = \begin{cases} 1 & \text{if } \mathrm{C}_1(v_{\leq i-1}) = 0 \wedge \tilde{\mathsf{g}}(v_{\leq i-1}, u_i) \leq -\tau, \\ 0 & \text{otherwise;} \end{cases}$$

$$\mathrm{C}_3(v_{\le i-1}, u_i) = \begin{cases} 1 & \text{if } \mathrm{C}_1(v_{\le i-1}) = 0 \text{ and } \mathrm{C}_2(v_{\le i-1}, u_i) = 0, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, if C_1 or C_2 happens, it means that the adversary has chosen to tamper with block i, and if C_3 happens it means that the adversary has not chosen to tamper with block i. Also, since the above functions are Boolean, we might treat them as events as well. Moreover, for convenience we define the set $\mathrm{C}_1 = \left\{ v_{\le i} \mid i \in [n-1], v_{\le i} \in \mathrm{Supp}(\mathbf{u})^i \wedge \mathrm{C}_1(v_{\le i}) \right\}$.

The following Claim bounds the average of the function when the attack of Construction 4.3 is performed on the distribution.

Claim 4.5. *If* $\overline{\mathbf{v}} \equiv \langle \mathbf{u}^n \parallel \mathsf{AppTam} \rangle$ *is the tampering distribution of the efficient attacker* AppTam *of Construction 4.3, then it holds that*

$$\mathbb{E}[f(\overline{\mathbf{v}})] \ge 1 - \mathrm{e}^{\frac{-(\mu - 2n\cdot\gamma)^2}{2n\cdot(\tau+4\gamma)^2}} - n \cdot \gamma.$$

Proof. Define a function t as follows,

$$t(v_{\le i-1}, u_i) = \begin{cases} 0 & \text{if } \mathrm{C}_1(v_{\le i-1}) \text{ or } \mathrm{C}_2(v_{\le i-1}, u_i), \\ \tilde{\mathrm{g}}(v_{\le i-1}, u_i) & \text{if } \mathrm{C}_3(v_{\le i-1}, u_i). \end{cases}$$

Now consider a sequence of random variables $\overline{\mathbf{t}} = (\mathbf{t}_1, \ldots, \mathbf{t}_n)$ sampled as follows. We first sample $\overline{u} \leftarrow \overline{\mathbf{u}}$, then $\overline{v} \leftarrow \langle \overline{u} \parallel \mathsf{AppTam} \rangle$, and then $t_i = \mathrm{t}(v_{\le i-1}, u_i)$ for $i \in [n]$. Now, for any $t_{\le i-1} \leftarrow \mathbf{t}_{\le i-1}$, we claim that $\mathbb{E}[\mathbf{t}_i \mid t_{\le i-1}] \ge 0$. The reason is as follows.

$$\begin{aligned}
\mathbb{E}[\mathbf{t}_i \mid t_{\le i-1}] &= \mathop{\mathbb{E}}_{v_{\le i-1} \leftarrow \mathbf{v}_{\le i-1} | t_{\le i-1}} \left[\mathop{\mathbb{E}}_{u_i \leftarrow \mathbf{u}} [\tilde{\mathrm{g}}(v_{\le i-1}, u_i) \cdot \mathrm{C}_3(v_{\le i-1}, u_i)] \right] \\
&\ge \mathop{\mathbb{E}}_{v_{\le i-1} \leftarrow \mathbf{v}_{\le i-1} | t_{\le i-1}} \left[\mathop{\mathbb{E}}_{u_i \leftarrow \mathbf{u}} [\tilde{\mathrm{g}}(v_{\le i-1}, u_i) \cdot (\mathrm{C}_3(v_{\le i-1}, u_i) \vee \mathrm{C}_2(v_{\le i-1}, u_i))] \right] \\
&= \mathop{\mathbb{E}}_{v_{\le i-1} \leftarrow \mathbf{v}_{\le i-1} | t_{\le i-1}} \left[\mathop{\mathbb{E}}_{u_i \leftarrow \mathbf{u}} [\tilde{\mathrm{g}}(v_{\le i-1}, u_i) \cdot (1 - \mathrm{C}_1(v_{\le i-1}))] \right] \\
&= \mathop{\mathbb{E}}_{v_{\le i-1} \leftarrow \mathbf{v}_{\le i-1} | t_{\le i-1}} \left[(1 - \mathrm{C}_1(v_{\le i-1})) \cdot \mathop{\mathbb{E}}_{u_i \leftarrow \mathbf{u}} [\tilde{\mathrm{g}}(v_{\le i-1}, u_i)] \right] \\
&\ge \mathop{\mathbb{E}}_{v_{\le i-1} \leftarrow \mathbf{v}_{\le i-1} | t_{\le i-1}} \left[(1 - \mathrm{C}_1(v_{\le i-1})) \cdot \mathop{\mathbb{E}}_{u_i \leftarrow \mathbf{u}} [\mathrm{g}(v_{\le i-1}, u_i) - \gamma] \right] \\
&\ge -\gamma.
\end{aligned}$$

Moreover, for any $t_{\le i-1} \in [\mathrm{Supp}(\mathbf{t}_{\le i-1})]$ we have

$$\begin{aligned}
\Pr_{t_i \leftarrow \mathbf{t}_i | t_{\le i-1}} [|t_i| \ge \tau + 3\gamma] &= \Pr_{v_{\le i} \leftarrow \mathbf{v}_{\le i} | t_{\le i-1}} [|\tilde{\mathrm{g}}(v_{\le i})| \ge \tau + 3\gamma \wedge \mathrm{C}_3(v_{\le i})] \\
&= \Pr_{v_{\le i} \leftarrow \mathbf{v}_{\le i} | t_{\le i-1}} [\tilde{\mathrm{g}}(v_{\le i}) \ge \tau + 3\gamma \wedge \mathrm{C}_3(v_{\le i})] \\
&\le \Pr_{v_{\le i} \leftarrow \mathbf{v}_{\le i} | t_{\le i-1}} [\tilde{\mathrm{g}}(v_{\le i}) \ge \tau + 3\gamma \wedge \overline{\mathrm{C}_1(v_{\le i})}] \\
&= \Pr_{v_{\le i} \leftarrow \mathbf{v}_{\le i} | t_{\le i-1}} [\tilde{\mathrm{g}}(v_{\le i}) \ge \tau + 3\gamma \wedge \tilde{\mathrm{g}}^*(\overline{v}_{\le i-1}) \le \tau] \\
&\le \Pr_{v_{\le i} \leftarrow \mathbf{v}_{\le i} | t_{\le i-1}} [\mathrm{g}(v_{\le i}) \ge \tau + 2\gamma \wedge \tilde{\mathrm{g}}^*(\overline{v}_{\le i-1}) \le \tau] \\
&\le \gamma.
\end{aligned}$$

Therefore, the sequence $\overline{\mathbf{t}} = (\mathbf{t}_1, \ldots, \mathbf{t}_n)$, computed over the same $\overline{v} \leftarrow \overline{\mathbf{v}}$, satisfies the properties required in Lemma 4.1. (Let τ of that Lemma 4.1 to be $\tau + 3\gamma$ here, and and letting s of that lemma to be $-\mu + 2n \cdot \gamma$.) This way, we get

$$\Pr\left[\sum_{i=1}^{n} \mathbf{t}_i \leq -\mu + 2n \cdot \gamma\right] \leq \mathrm{e}^{\frac{-(\mu - 3n\gamma)^2}{2n \cdot (\tau + 4\gamma)^2}} + n \cdot \gamma\ .$$

On the other hand, for every $\overline{v} \in \mathrm{Supp}(\overline{\mathbf{v}})$ we have

$$\begin{aligned} f(\overline{v}) &= \mu + \sum_{i=1}^{n} \mathsf{g}(v_{\leq i}) \\ &\geq \mu + \sum_{i=1}^{n} (\tilde{\mathsf{g}}(v_{\leq i}) - \gamma) \\ &= \mu - n \cdot \gamma + \sum_{i=1}^{n} \left(\mathrm{C}_1\left(v_{\leq i-1}\right) + \mathrm{C}_3\left(v_{\leq i}\right)\right) \cdot \tilde{\mathsf{g}}(v_{\leq i}) \\ &\geq \mu - n \cdot \gamma + \sum_{i=1}^{n} t_i - \sum_{i=1}^{n} \mathrm{C}_1(v_{\leq i-1}) \cdot \gamma \\ &\geq \mu - 2n \cdot \gamma + \sum_{i=1}^{n} t_i. \end{aligned}$$

Therefore, we have

$$\Pr[f(\overline{\mathbf{v}}) = 0] \leq \Pr\left[\sum_{i=1}^{n} \mathbf{t}_i \leq -\mu + 2n \cdot \gamma\right] \leq \mathrm{e}^{\frac{-(\mu - 3n \cdot \gamma)^2}{2n \cdot (\tau + 4\gamma)^2}} + n \cdot \gamma.$$

□

Now we state and prove another Claim which bounds the expected number of tamperings performed by the attack of Construction 4.3

Claim 4.6. *For a tampering sequence $\overline{v} \leftarrow \langle \overline{u} \parallel \mathsf{AppTam} \rangle$, let $T_i = \mathrm{C}_1(v_{\leq i-1}) \vee \mathrm{C}_2(v_{\leq i-1}, u_i)$ be the event (or equivalently the Boolean function) denoting that a tampering choice is made by the adversary of Construction 4.3 over the i'th block. If $T = \sum_{i=1}^{n} T_i$ denotes the total number of tamperings, then*

$$\mathbb{E}[\mathbf{T}] \leq \frac{1 - \mu + n \cdot \gamma}{\tau - 2\gamma}.$$

Proof. For any $v_{\leq i-1} \in \mathrm{C}_1$, we have

$$\mathop{\mathbb{E}}_{v_i \leftarrow \mathbf{v}_i | v_{\leq i-1}}[\mathsf{g}(v_{\leq i})] \geq \tilde{\mathsf{g}}^*(v_{\leq i-1}) - \gamma \geq \tau - \gamma = (\tau - \gamma) \cdot \Pr_{u_i \leftarrow \mathbf{u}}[\mathrm{C}_1(v_{\leq i-1}) \vee \mathrm{C}_2(v_{\leq i-1}, u_i)], \quad (1)$$

and, for any $v_{\leq i-1} \notin \mathrm{C}_1$ we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{v_i \leftarrow \mathbf{v}_i | v_{\leq i-1}}[\mathsf{g}(v_{\leq i})] &\geq \mathop{\mathbb{E}}_{v_i \leftarrow \mathbf{v}_i | v_{\leq i-1}}[\tilde{\mathsf{g}}(v_{\leq i}) - \gamma] \\
&= \mathop{\mathbb{E}}_{u_i \leftarrow \mathbf{u}}[(1 - \mathrm{C}_2(v_{\leq i-1}, u_i)) \cdot \tilde{\mathsf{g}}(v_{\leq i-1}, u_i) + \mathrm{C}_2(v_{\leq i-1}, u_i) \cdot \tilde{\mathsf{g}}^*(v_{\leq i-1})] - \gamma \\
&\geq \mathop{\mathbb{E}}_{u_i \leftarrow \mathbf{u}}[(1 - \mathrm{C}_2(v_{\leq i-1}, u_i)) \cdot \tilde{\mathsf{g}}(v_{\leq i-1}, u_i) + \mathrm{C}_2(v_{\leq i-1}, u_i) \cdot (\tilde{\mathsf{g}}(v_{\leq i-1}, u_i) + \tau - 2\gamma)] \\
&= \mathop{\mathbb{E}}_{u_i \leftarrow \mathbf{u}}[\tilde{\mathsf{g}}(v_{\leq i-1}, u_i) + \mathrm{C}_2(v_{\leq i-1}, u_i) \cdot (\tau - 2\gamma)] \\
&\geq -\gamma + \mathop{\mathbb{E}}_{u_i \leftarrow \mathbf{u}}[\mathrm{C}_2(v_{\leq i-1}, u_i) \cdot (\tau - 2\gamma)] \\
&= (\tau - 2\gamma) \cdot \Pr_{u_i \leftarrow \mathbf{u}}[\mathrm{C}_2(v_{\leq i-1}, u_i)] - \gamma \\
&= (\tau - 2\gamma) \cdot \Pr_{u_i \leftarrow \mathbf{u}}[\mathrm{C}_1(v_{\leq i-1}) \vee \mathrm{C}_2(v_{\leq i-1}, u_i)] - \gamma. \qquad (2)
\end{aligned}$$

By Equations 1 and 2, for any $v_{\leq i-1} \in \mathrm{Supp}(\mathbf{u})^{i-1}$ we have

$$\mathop{\mathbb{E}}_{v_i \leftarrow \mathbf{v}_i | v_{\leq i-1}}[\mathsf{g}(v_{\leq i})] \geq (\tau - 2\gamma) \cdot \Pr_{u_i \leftarrow \mathbf{u}}[\mathrm{C}_1(v_{\leq i-1}) \vee \mathrm{C}_2(v_{\leq i-1}, u_i)] - \gamma. \qquad (3)$$

Also, let $\mathsf{PT}(v_{\leq i-1})$ be the probability of tampering in the ith block conditioned on the prefix $v_{\leq i-1}$. Namely,

$$\mathsf{PT}(v_{\leq i-1}) = \Pr_{u_i \leftarrow \mathbf{u}}[\mathrm{C}_1(v_{\leq i-1}) \vee \mathrm{C}_2(v_{\leq i-1}, u_i)].$$

The definition of $\mathsf{PT}(v_{\leq i-1})$ together with Equation 3 implies that

$$\mathop{\mathbb{E}}_{v_i \leftarrow \mathbf{v}_i | v_{\leq i-1}}[\mathsf{g}(v_{\leq i})] \geq (\tau - 2\gamma) \cdot \mathsf{PT}(v_{\leq i-1}) - \gamma. \qquad (4)$$

We now obtain that

$$\begin{aligned}
\mathbb{E}\left[f(\overline{\mathbf{v}})\right] - \mu &= \mathbb{E}\left[\sum_{i=1}^{n} \mathsf{g}(\mathbf{v}_{\leq i})\right] \\
\text{(by linearity of expectation)} &= \sum_{i=1}^{n} \mathbb{E}\left[\mathsf{g}(\mathbf{v}_{\leq i})\right] \\
&= \sum_{i=1}^{n} \mathop{\mathbb{E}}_{v_{\leq i-1} \leftarrow \mathbf{v}_{\leq i-1}}\left[\mathop{\mathbb{E}}_{v_i \leftarrow \mathbf{v}_i | v_{\leq i-1}}[\mathsf{g}(v_{\leq i})]\right] \\
\text{(by Equation 4)} &\geq (\tau - 2\gamma) \cdot \sum_{i=1}^{n} \mathop{\mathbb{E}}_{v_{\leq i-1} \leftarrow \mathbf{v}_{\leq i-1}}[\mathsf{PT}(v_{\leq i-1})] - n\gamma \\
\text{(by linearity of expectation)} &= (\tau - 2\gamma) \cdot \mathbb{E}\left[\sum_{i=1}^{n} \mathsf{PT}(\mathbf{v}_{\leq i-1})\right] - n\gamma \\
&\geq (\tau - 2\gamma) \cdot \mathbb{E}[\mathbf{T}] - n\gamma.
\end{aligned}$$

Therefore, we have

$$\mathbb{E}[\mathbf{T}] \leq \frac{1 - \mu + n \cdot \gamma}{\tau - 2 \cdot \gamma}.$$

□

4.2 Polynomial-time Biasing Attack Using Probably Approximate Oracles

In this subsection, we finally prove Theorem 3.1 by getting rid of the promised approximate oracles, and having them approximated by the attacker itself, though only with high probability. We will also show how to pick the parameters of the attack to achieve the bounds claimed in Theorem 3.1.

We start by sketching the intuitive fact that the approximate oracles of Definition 4.2 could indeed be provided to our polynomial time attacker of Construction 4.3 with high probability by repeated sampling and applying Chernoff bound.

Construction 4.7 (Oracle $\tilde{\mathsf{g}}(\cdot)$)**.** Given a prefix $v_{\leq i} \in \mathrm{Supp}(\mathbf{u})^i$, let

$$k = -12 \cdot \frac{\ln(\gamma/2) + \ln(\ln(1+\gamma)) - \ln(-\ln(\gamma/2))}{\gamma^2}.$$

Sample k random continuations $\overline{w}_1^1, \ldots, \overline{w}_1^k$ from $\mathbf{u}^{n-i-1}$ and k continuations $\overline{w}_2^1, \ldots, \overline{w}_2^k$ from $\mathbf{u}^{n-i}$. Let

$$\tilde{\mathsf{a}}(v_{\leq i}) = \frac{1}{k} \cdot \Big(f(v_{\leq i}, \overline{w}_1^1) + \cdots + f(v_{\leq i}, \overline{w}_1^k) \Big)$$

$$\tilde{\mathsf{a}}(v_{\leq i-1}) = \frac{1}{k} \cdot \Big(f(v_{\leq i-1}, \overline{w}_2^1) + \cdots + f(v_{\leq i-1}, \overline{w}_2^k) \Big)$$

and output $\tilde{\mathsf{g}}(v_{\leq i}) = \tilde{\mathsf{a}}(v_{\leq i}) - \tilde{\mathsf{a}}(v_{\leq i-1})$.

Claim 4.8. *For the oracle $\tilde{\mathsf{g}}(\cdot)$ of Construction 4.7 We have*

$$\Pr[|\tilde{\mathsf{g}}(y_{\leq i}) - \mathsf{g}(y_{\leq i})| \geq \gamma] \leq \frac{-\gamma \cdot \ln(1+\gamma)}{2 \cdot \ln(\gamma/2)} \leq \frac{\gamma}{2}.$$

Proof. Define independent Boolean random variables $\mathbf{f}_1^1, \ldots, \mathbf{f}_1^k$ and $\mathbf{f}_2^1, \ldots, \mathbf{f}_2^k$ such that for each $j \in [k]$ we have $\mathbf{f}_1^j \equiv f(v_{\leq i}, \mathbf{u}^{n-i})$ and $\mathbf{f}_2^j \equiv f(v_{\leq i-1}, \mathbf{u}^{n-i+1})$. Also let $\mathsf{a}(y_{\leq i}) = \mathbb{E}[f(v_{\leq i}, \mathbf{u}^{n-i})]$ and $\mathsf{a}(y_{\leq i-1}) = \mathbb{E}[f(v_{\leq i-1}, \mathbf{u}^{n-i+1})]$. By Chenroff inequality we have,

$$\Pr\left[\Big|\frac{1}{k} \cdot \sum_{j \in k} \mathbf{f}_1^j - \mathsf{a}(y_{\leq i})\Big| \geq \frac{\gamma}{2}\right] = \Pr\left[|\tilde{\mathsf{a}}(y_{\leq i}) - \mathsf{a}(y_{\leq i})| \geq \frac{\gamma}{2}\right] \leq \mathrm{e}^{\frac{-k \cdot \gamma^2}{12}}. \tag{5}$$

On the other hand, again by Chenroff we have,

$$\Pr\left[\Big|\frac{1}{k} \cdot \sum_{j \in k} \mathbf{f}_2^j - \mathsf{a}(y_{\leq i})\Big| \geq \frac{\gamma}{2}\right] = \Pr\left[|\tilde{\mathsf{a}}(y_{\leq i-1}) - \mathsf{a}(y_{\leq i-1})| \geq \frac{\gamma}{2}\right] \leq \mathrm{e}^{\frac{-k \cdot \gamma^2}{12}}. \tag{6}$$

Now by Inequalities 5 and 6, we have

$$\Pr\left[|\tilde{\mathsf{g}}(y_{\leq i}) - \mathsf{g}(y_{\leq i})| \geq \gamma\right] \leq 2 \cdot \mathrm{e}^{\frac{-k \cdot \gamma^2}{12}} = \frac{-\gamma \cdot \ln(1+\gamma)}{2 \cdot \ln(\gamma/2)}.$$

Note that $\frac{-\ln(1+\gamma)}{\ln(\gamma/2)}$ is less than 1 for $\gamma \in [0,1]$.Therefore, we have,

$$\Pr\left[|\tilde{\mathsf{g}}(y_{\leq i}) - \mathsf{g}(y_{\leq i})| \geq \gamma\right] \leq \frac{\gamma}{2}.$$

□

Claim 4.9. *The implementation of the oracle of Construction 4.7 runs in time* $O(n \cdot \ell/\gamma^3)$.

Proof. The oracle samples

$$k = -12 \cdot \frac{\ln(\gamma/2) + \ln(\ln(1+\gamma)) - \ln(-\ln(\gamma/2))}{\gamma^2} \leq \frac{24}{\gamma^3}$$

continuations. Therefore, the running time is $O(n \cdot \ell/\gamma^3)$. □

Construction 4.10 (Oracles $\tilde{\mathsf{g}}^*(\cdot)$ and $\tilde{\mathsf{h}}(\cdot)$)**.** Given a prefix $v_{\leq i-1} \in \mathrm{Supp}(\mathbf{u})^{i-1}$, sample $k = \frac{-\ln(\gamma/2)}{\ln(1+\gamma)}$ blocks $u^1, \ldots, u^k$ from $\mathbf{u}$. Now let

$$\tilde{\mathsf{h}}(v_{\leq i-1}) = \operatorname*{argmax}_{u^j} \tilde{\mathsf{g}}^*(v_{\leq i-1}, u^j) \quad \text{and} \quad \tilde{\mathsf{g}}^*(v_{\leq i-1}) = \max_{u^j} \tilde{\mathsf{g}}(v_{\leq i-1}, u^j).$$

Claim 4.11. *Let* $\lambda \in [-1, 1]$ *be such that* $\Pr[\mathsf{g}(y_{\leq i-1}, \mathbf{u}) \geq \lambda] \geq \gamma$. *For the* $\tilde{\mathsf{g}}^*(\cdot)$ *oracle of Construction 4.10 We have*

$$\Pr[\tilde{\mathsf{g}}^*(y_{\leq i-1}) \leq \lambda - \gamma] \leq \gamma.$$

Proof. We first bound the probability of all the actual gains being less then λ. We have

$$\Pr[\forall j \in [k], \mathsf{g}(v_{\leq i-1}, u^j) \leq \lambda] = \Pr[\mathsf{g}(v_{\leq i-1}, \mathbf{u}) \leq -\gamma]^k \leq (1-\gamma)^k \leq \frac{\gamma}{2}.$$

Now, consider the following event

$$B = \begin{cases} 0 & \text{if for all } j \in k \text{ we have } |\tilde{\mathsf{g}}(v_{\leq i-1}, u^j) - \mathsf{g}(v_{\leq i-1}, u^j)| \leq \gamma, \\ 1 & \text{otherwise.} \end{cases}$$

By Claim 4.8 and a union bound we have $\Pr[B] \leq k \cdot \frac{-\gamma \cdot \ln(1+\gamma)}{2 \cdot \ln(\gamma)} = \frac{\gamma}{2}$. Therefore, we have

$$\Pr[\forall j \in [k], \tilde{\mathsf{g}}(v_{\leq i-1}, u^j) \leq \lambda - \gamma] \leq \Pr[\forall j \in [k], \mathsf{g}(v_{\leq i-1}, v^j) \leq \lambda] + \frac{\gamma}{2} \leq \gamma.$$

□

Claim 4.12. *For the oracle* $\tilde{\mathsf{g}}^*(\cdot)$ *of Construction 4.10, we have* $\Pr[\tilde{\mathsf{g}}^*(y_{\leq i-1}) \leq -2\gamma] \leq \gamma$.

Proof. We first bound the probability of all the actual gains being less then $-\gamma$. We have

$$\Pr[\forall j \in [k], \mathsf{g}(v_{\leq i-1}, u^j) \leq -\gamma] = \Pr[\mathsf{g}(v_{\leq i-1}, \mathbf{u}) \leq -\gamma]^k \leq \left(\frac{1}{1+\gamma}\right)^k = \frac{\gamma}{2}.$$

Now consider the following event

$$B = \begin{cases} 0 & \text{if for all } j \in k \text{ we have } |\tilde{\mathsf{g}}(v_{\leq i-1}, u^j) - \mathsf{g}(v_{\leq i-1}, u^j)| \leq \gamma, \\ 1 & \text{otherwise.} \end{cases}$$

By Claim 4.8 and a union bound, we have $\Pr[B] \leq k \cdot \frac{-\gamma \cdot \ln(1+\gamma)}{\ln(\gamma/2)} = \frac{\gamma}{2}$. Therefore, we have

$$\Pr[\forall j \in [k], \tilde{\mathsf{g}}(v_{\leq i-1}, u^j) \leq -2\gamma] \leq \Pr[\forall j \in [k], \mathsf{g}(v_{\leq i-1}, u^j) \leq -\gamma] + \frac{\gamma}{2} \leq \gamma.$$

□

Claim 4.13. *The Oracles of Construction 4.10 run in time $O(n \cdot \ell/\gamma^5)$.*

Proof. The oracles generate

$$k = -\frac{\ln(\gamma/2)}{\ln(1+\gamma)} \leq \frac{1}{\gamma^2}$$

samples and for each sample call the oracle $\tilde{\mathsf{g}}(\cdot)$ of Construction 4.7. Therefore, by Claim 4.9 the running time of the Oracles of Construction 4.10 are $O(n \cdot \ell/\gamma^5)$ □

Now we show that using the approximate oracles of Constructions 4.7 and 4.10 we still can achieve the desired bounds.

Construction 4.14 (Attack using probably approximate oracles)**.** Given input vector $(v_1, \ldots, v_{i-1}, u_i) \in \mathrm{Supp}(\mathbf{u})^i$, AppTam will output some $v_i \in \mathrm{Supp}(\mathbf{u})$. Let $w_i = \tilde{\mathsf{h}}(v_{\leq i-1})$ as defined in Definition 4.2. Now, EffTam chooses its output v_i as follows.

- **Tampering.** If $\tilde{\mathsf{g}}^*(v_{\leq i-1}) \geq \tau$ or if $\tilde{\mathsf{g}}(v_{\leq i-1}, u_i) \leq -\tau$, then output $v_i = w_i$.
- **Not tampering.** Otherwise, output the original sample $v_i = u_i$.

where $\tilde{\mathsf{g}}^*(\cdot)$,$\tilde{\mathsf{h}}(\cdot)$ and $\tilde{\mathsf{g}}(\cdot)$ oracles are instantiated using Constructions 4.10 and 4.7.

The following claim bounds the average of function in presence of the attack of Construction 4.14.

Claim 4.15. *If $\overline{\mathbf{v}} \equiv \langle \mathbf{u}^n \parallel \mathsf{EffTam} \rangle$ is the tampering distribution of the efficient attacker EffTam of Construction 4.3, then it holds that*

$$\mathbb{E}[f(\overline{\mathbf{v}})] \geq 1 - \mathrm{e}^{\frac{-(\mu - 2n\cdot\gamma)^2}{2n\cdot(\tau+4\gamma)^2}} - 4n \cdot \gamma.$$

Proof. Let B be the event that for at least one of the queries of Construction 4.14, the promises are not satisfied. Namely,

$$\begin{aligned} B = \exists\, i,\ & \Pr_{u_i \leftarrow \mathbf{u}}\left[\mathsf{g}(v_{\leq i-1}, u_i) > \tilde{\mathsf{g}}^*(v_{\leq i-1}) + 2\gamma\right] \leq \gamma \\ & \vee \tilde{\mathsf{g}}^*(u_{\leq i-1}) < -2\gamma \\ & \vee |\tilde{\mathsf{g}}(v_{\leq i-1}, u_i) - \mathsf{g}(v_{\leq i-1}, u_i)| \geq \gamma. \end{aligned}$$

During the whole course of process of Construction 4.14, there are exactly n, $\tilde{\mathsf{g}}^*(\cdot)$ queries and exactly $\tilde{\mathsf{g}}(\cdot)$ queries. Therefore, using Claims 4.8, 4.11 and 4.12 we have

$$\Pr[B] \leq n \cdot \gamma + n \cdot \gamma + n \cdot \frac{\gamma}{2} \leq 3 \cdot n \cdot \gamma.$$

We know that conditioned on $\overline{B}$, the attack of Construction 4.14 will be identical to the attack of Construction 4.3 and increases the average to $1 - \mathrm{e}^{\frac{-(\mu-2n\cdot\gamma)^2}{2n\cdot(\tau+4\gamma)^2}} - n \cdot \gamma$ by Claim 4.5. Therefore, the attack of Construction 4.14 can unconditionally increases the average to at least

$$1 - \mathrm{e}^{\frac{-(\mu-2n\cdot\gamma)^2}{2n\cdot(\tau+4\gamma)^2}} - n \cdot \gamma - \Pr[B] \geq 1 - \mathrm{e}^{\frac{-(\mu-2n\cdot\gamma)^2}{2n\cdot(\tau+4\gamma)^2}} - 4n \cdot \gamma.$$

□

Now we state and prove another claim that bounds the expected number of tamperings performed by the attack of Construction 4.14

Claim 4.16. *For a tampering sequence $\overline{v} \leftarrow \langle \overline{u} \parallel \mathsf{EffTam} \rangle$, let $T_i = \mathrm{C}_1(v_{\leq i-1}) \vee \mathrm{C}_2(v_{\leq i-1}, u_i)$ be the event (or equivalently the Boolean function) denoting that a tampering choice is made by the adversary of Construction 4.14 over the i'th block. If $T = \sum_{i=1}^{n} T_i$ denotes the total number of tamperings, then*

$$\mathbb{E}[\mathbf{T}] \leq \frac{1-\mu+n\cdot\gamma}{\tau-2\gamma} + 3n^2\cdot\gamma.$$

Proof. Define event B similar to the proof of Claim 4.15. We know that conditioned on $\overline{B}$, the attack of Construction 4.14 will be identical to the attack of Construction 4.3 and uses average budget at most $\frac{1-\mu+n\cdot\gamma}{\tau-2\gamma}$ by Claim 4.6. Therefore, the attack of Construction 4.14 will use average budget atmost

$$\frac{1-\mu+n\cdot\gamma}{\tau-2\gamma} + n\cdot\Pr[B] \leq \frac{1-\mu+n\cdot\gamma}{\tau-2\gamma} + 3n^2\cdot\gamma$$

□

Finally, we prove Theorem 3.1 by relying on the probabilistic guarantees of the estimators of the above constructions for the approximate oracles.

Proof of Theorem 3.1. Let

$$k = \ln\left(\frac{2}{1-\rho}\right) \text{ and } \tau = \frac{\mu}{1.9\sqrt{kn}} \text{ and } \gamma = \min\left\{\frac{\mu}{20n}, \frac{\mu}{80\sqrt{k\cdot n}}, \frac{1-\rho}{8\cdot n}, \frac{\sqrt{\ln(2/(1-\rho))}}{3n\sqrt{n}}\right\}.$$

Also, let $\mathsf{EffTam}_{\tau,\gamma}$ be the attack of Construction 4.14 instantiated with the parameters τ and γ specified above. If $\overline{\mathbf{v}} \equiv \langle \mathbf{u}^n \parallel \mathsf{EffTam} \rangle$, by Claim 4.15 we have

$$\begin{aligned}
\mathbb{E}[f(\overline{\mathbf{v}})] &\geq 1 - \mathrm{e}^{\frac{-(\mu-2n\cdot\gamma)^2}{2n\cdot(\tau+4\gamma)^2}} - 4n\cdot\gamma \\
&\geq 1 - \mathrm{e}^{\frac{-(0.9\mu)^2}{2n\cdot(1.1\tau)^2}} - \frac{1-\rho}{2} \\
&\geq 1 - \frac{1-\rho}{2} - \frac{1-\rho}{2} \\
&= \rho.
\end{aligned}$$

On the other hand, By Claim 4.16 we have

$$\begin{aligned}
\mathbb{E}[\mathbf{T}] &\leq \frac{1-\mu+n\cdot\gamma}{\tau-2\gamma} + 3n^2\cdot\gamma \\
&\leq \frac{1-0.95\mu}{0.95\tau} + \sqrt{n\cdot\ln(2/(1-\rho))} \\
&\leq \frac{2-1.9\mu}{\mu}\cdot\sqrt{n\cdot\ln(2/(1-\rho))} + \sqrt{n\cdot\ln(2/(1-\rho))} \\
&\leq \frac{2}{\mu}\cdot\sqrt{n\cdot\ln(2/(1-\rho))}.
\end{aligned}$$

We also know that $\gamma = \omega(\mu\cdot(1-\rho)/n^2)$. Therefore, by Claims 4.9 and 4.13 we conclude that the running time of $\mathsf{EffTam}_{\tau,\gamma}$ is $O(n^{12}\cdot\ell/(\mu\cdot(1-\rho))^5)$. □

5 Conclusion and Some Questions

In this work, we proved strong barriers against basing the robustness of learning algorithms in both settings of evasion attacks (who aim at fining adversarial examples) and poisoning attacks (who try to increase the misclassification probability by minimally tampering with the training data) on computational hardness assumptions. Namely, we showed that a broad set of learning problems, and in particular classification tasks whose instances are drawn from a product distribution of dimension n, are inherently vulnerable to *polynomial-time* adversaries that find adversarial examples of (sublinear) Hamming distance $O(\sqrt{n})$ and increase the classification error from 1% to 99%.

Our proofs are based on a new coin tossing attack that is inspired by the recent results of [KKR18]. Our coin tossing attacks could be interpreted as a polynomial-time algorithmic proof of the well-known classical result of concentration measure in product distributions [AM80, MS86, McD89, Tal95]. While, concentration of measure guarantees that for any initial large enough set $\mathcal{S}$, "most" points in the probability space are "close" to $\mathcal{S}$, our algorithmic proof shows how to find such close instances in polynomial time.

Our work motivates studying the following natural questions.

Other *computationally* concentrated metric probability spaces (e.g., computational Lévy families)? Normal Lévy families include numerous metric probability spaces in which the measure is concentrated. Our Theorem 3.1 shows that product distributions, as a special form of normal Lévy families are computationally concentrated. How about other spaces (e.g., isotropic Gaussian distribution under Euclidean distance) for which information theoretic concentration of measure is known? Proving any such results about any other metric probability space, directly leads to polynomial-time attacks against evasion robustness of any learning problem whose instances are drawn from those spaces. More generally, for what metric probability spaces can we prove nontrivial computational concentration of measure. Namely, a space would be computationally concentrated, if for any set $\mathcal{S}$ with "large enough" measure, "most" of the points in the probability space can be *efficiently* mapped to "close" points in $\mathcal{S}$. Here, "close" shall be something that beats the trivial bound given by the diameter of the space. The latter question can be also interpreted as proving (approximate) "computational isoperimetric" inequalities. Namely, here the "computational boundary" of the set $\mathcal{S}$ would be defined by a computationally bounded mapping that efficiently maps the points in "computational distance" b to $\mathcal{S}$. As in Theorem 3.1, access to the underlying probability distribution and membership in $\mathcal{S}$ can be provided by sampling and membership oracles.

Handling smaller initial error. Our Theorem 3.1 gives nontrivial attacks with sublinear $o(n)$ perturbation only if the initial average (or equivalently the probability of the target set $\mathcal{S}$) is at least $\omega(\log n/\sqrt{n})$. However, the information theoretic attacks of [MDM19] can handle original error that is only substantially $\exp(-o(n))$ large. What is the smallest probability μ for which we can obtain computational concentration of measure (for product or other distributions) if the target set has probability at least μ? Solving this problem would have immediate impact on polynomial time evasion (and poisoning–in case of product distributions) attacks that attack systems with smaller error (in the no-attack case).

References

[ABL17] Pranjal Awasthi, Maria-Florina Balcan, and Philip M. Long. The Power of Localization for Efficiently Learning Linear Separators with Noise. *Journal of the ACM*, 63(6):50:1–50:27, 2017. 2

[ACW18] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018. 2

[AI06] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 459–468. IEEE, 2006. 7

[AIR18] Alexandr Andoni, Piotr Indyk, and Ilya Razenshteyn. Approximate nearest neighbor search in high dimensions. *arXiv preprint arXiv:1806.09823*, 2018. 7

[AKM18] Idan Attias, Aryeh Kontorovich, and Yishay Mansour. Improved generalization bounds for robust learning. *arXiv preprint arXiv:1810.02180*, 2018. 5

[AM80] D Amir and VD Milman. Unconditional and symmetric sets inn-dimensional normed spaces. *Israel Journal of Mathematics*, 37(1-2):3–20, 1980. 6, 24

[AR15] Alexandr Andoni and Ilya Razenshteyn. Optimal data-dependent hashing for approximate near neighbors. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 793–801. ACM, 2015. 7

[BCM$^+$13] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Srndic, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion Attacks against Machine Learning at Test Time. In *ECML/PKDD*, pages 387–402, 2013. 2

[BEK02] Nader H. Bshouty, Nadav Eiron, and Eyal Kushilevitz. PAC learning with nasty noise. *Theoretical Computer Science*, 288(2):255–275, 2002. 3

[BFR14] Battista Biggio, Giorgio Fumera, and Fabio Roli. Security evaluation of pattern classifiers under attack. *IEEE transactions on knowledge and data engineering*, 26(4):984–996, 2014. 2

[BL17] Cody Burkard and Brent Lagesse. Analysis of causative attacks against svms learning from data streams. In *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics*, pages 31–36. ACM, 2017. 2

[BNL12] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1467–1474. Omnipress, 2012. 2

[BNS$^+$06] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D Joseph, and J Doug Tygar. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, pages 16–25. ACM, 2006. 4

[BOL89] M. Ben-Or and N. Linial. Collective coin flipping. *Randomness and Computation*, 5:91–115, 1989. 7

[BPR18] Sébastien Bubeck, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. *arXiv preprint arXiv:1805.10204*, 2018. 5, 6, 12

[BTEGN09] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. Robust optimization. princeton series in applied mathematics, 2009. 6

[CSV17] Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 47–60. ACM, 2017. 2, 3

[CW17] Nicholas Carlini and David A. Wagner. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 39–57, 2017. 2

[DKK$^+$16] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high dimensions without the computational intractability. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 655–664. IEEE, 2016. 3, 4

[DKK$^+$17] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Being robust (in high dimensions) can be practical. In *International Conference on Machine Learning*, pages 999–1008, 2017. 2

[DKK$^+$18a] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robustly learning a gaussian: Getting optimal error, efficiently. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2683–2702. Society for Industrial and Applied Mathematics, 2018. 2

[DKK$^+$18b] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. Sever: A robust meta-algorithm for stochastic optimization. *arXiv preprint arXiv:1803.02815*, 2018. 2, 3

[DKS17] Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Statistical query lower bounds for robust estimation of high-dimensional Gaussians and Gaussian mixtures. In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, pages 73–84. IEEE, 2017. 2, 3, 5

[DKS18a] Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. List-decodable robust mean estimation and learning mixtures of spherical Gaussians. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1047–1060. ACM, 2018. 2, 3

[DKS18b] Ilias Diakonikolas, Weihao Kong, and Alistair Stewart. Efficient algorithms and lower bounds for robust linear regression. *arXiv preprint arXiv:1806.00040*, 2018. 2, 3

[DMM18] Dimitrios I. Diochnos, Saeed Mahloujifar, and Mohammad Mahmoody. Adversarial Risk and Robustness: General Definitions and Implications for the Uniform Distribution. In *Conference on Neural Information Processing Systems (NIPS)*, 2018. 2, 3, 6, 7, 12, 13

[FFF18] Alhussein Fawzi, Hamza Fawzi, and Omar Fawzi. Adversarial vulnerability for any classifier. *arXiv preprint arXiv:1802.08686*, 2018. 2, 3, 6, 7, 13

[FMS15] Uriel Feige, Yishay Mansour, and Robert Schapire. Learning and inference in the presence of corrupted inputs. In *Conference on Learning Theory*, pages 637–657, 2015. 5

[FMS18] Uriel Feige, Yishay Mansour, and Robert E Schapire. Robust inference for multiclass classification. In *Algorithmic Learning Theory*, pages 368–386, 2018. 5

[GKP15] Shafi Goldwasser, Yael Tauman Kalai, and Sunoo Park. Adaptively secure coin-flipping, revisited. In *International Colloquium on Automata, Languages, and Programming*, pages 663–674. Springer, 2015. 9

[GMF$^+$18] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. Adversarial Spheres. *arXiv preprint arXiv:1801.02774*, 2018. 2, 6, 7, 12

[GSS15] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *ICLR*, 2015. 2

[IEAL18] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. *arXiv preprint arXiv:1804.08598*, 2018. 4

[IM98] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998. 7

[Kea98] Michael Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM)*, 45(6):983–1006, 1998. 5

[KKR18] Yael Tauman Kalai, Ilan Komargodski, and Ran Raz. A lower bound for adaptively-secure collective coin-flipping protocols. In *32nd International Symposium on Distributed Computing*, 2018. 6, 7, 8, 9, 24

[KL93] Michael J. Kearns and Ming Li. Learning in the Presence of Malicious Errors. *SIAM Journal on Computing*, 22(4):807–837, 1993. 3

[KL17] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894, 2017. 2

[LLS89] David Lichtenstein, Nathan Linial, and Michael Saks. Some extremal problems arising from discrete control processes. *Combinatorica*, 9(3):269–287, 1989. 8

[LRV16] Kevin A Lai, Anup B Rao, and Santosh Vempala. Agnostic estimation of mean and covariance. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 665–674. IEEE, 2016. 3, 4

[McD89] Colin McDiarmid. On the method of bounded differences. *Surveys in combinatorics*, 141(1):148–188, 1989. 6, 24

[MDM18] Saeed Mahloujifar, Dimitrios I Diochnos, and Mohammad Mahmoody. Learning under p-Tampering Attacks. In *ALT*, pages 572–596, 2018. 2, 5, 6, 9, 10, 13, 14, 15

[MDM19] Saeed Mahloujifar, Dimitrios I Diochnos, and Mohammad Mahmoody. The curse of concentration in robust learning: Evasion and poisoning attacks from concentration of measure. *AAAI Conference on Artificial Intelligence*, 2019. 2, 3, 4, 5, 6, 7, 9, 12, 13, 14, 24

[MM17] Saeed Mahloujifar and Mohammad Mahmoody. Blockwise p-tampering attacks on cryptographic primitives, extractors, and learners. In *Theory of Cryptography Conference*, pages 245–279. Springer, 2017. 5, 6, 9, 13

[MMM18] Saeed Mahloujifar, Mohammad Mahmoody, and Ameer Mohammed. Multi-party poisoning through generalized p-tampering. *arXiv preprint arXiv:1809.03474*, 2018. 5

[MMS$^+$18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR*, 2018. 6

[MRT15] Yishay Mansour, Aviad Rubinstein, and Moshe Tennenholtz. Robust probabilistic inference. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 449–460. Society for Industrial and Applied Mathematics, 2015. 6

[MS86] Vitali D Milman and Gideon Schechtman. *Asymptotic theory of finite dimensional normed spaces*, volume 1200. Springer Verlag, 1986. 2, 6, 24

[PMG$^+$17] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017. 4, 5

[PMSW16] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814*, 2016. 2

[PMW$^+$16] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks. In *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*, pages 582–597, 2016. 2

[PSBR18] Adarsh Prasad, Arun Sai Suggala, Sivaraman Balakrishnan, and Pradeep Ravikumar. Robust estimation via robust gradient estimation. *arXiv preprint arXiv:1802.06485*, 2018. 2, 3

[RNH$^+$09] Benjamin I.P. Rubinstein, Blaine Nelson, Ling Huang, Anthony D. Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and J.D. Tygar. Antidote: understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 1–14. ACM, 2009. 2

[SHN$^+$18] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *arXiv preprint arXiv:1804.00792*, 2018. 2, 5

[SHS$^+$18] Ali Shafahi, W Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable? *arXiv preprint arXiv:1809.02104*, 2018. 2, 3, 7

[SST$^+$18] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially Robust Generalization Requires More Data. *arXiv preprint arXiv:1804.11285*, 2018. 5, 6

[SZS$^+$14] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. 2, 6, 13

[Tal95] Michel Talagrand. Concentration of measure and isoperimetric inequalities in product spaces. *Publications Mathématiques de l'Institut des Hautes Etudes Scientifiques*, 81(1):73–205, 1995. 6, 24

[Val85] Leslie G. Valiant. Learning disjunctions of conjunctions. In *IJCAI*, pages 560–566, 1985. 3, 5

[WC18] Yizhen Wang and Kamalika Chaudhuri. Data poisoning attacks against online learning. *arXiv preprint arXiv:1808.08994*, 2018. 5

[XBB$^+$15] Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. Is feature selection secure against training data poisoning? In *ICML*, pages 1689–1698, 2015. 2

[XEQ17] Weilin Xu, David Evans, and Yanjun Qi. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. *CoRR*, abs/1704.01155, 2017. 2