On the Computability of Robust PAC Learning

Pascale Gourdeau PASCALE.GOURDEAU@VECTORINSTITUTE.AI
Vector Institute & University of Toronto, Toronto, ON, Canada

Tosca Lechner TLECHNER@UWATERLOO.CA
Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada

Ruth Urner RUTH@EECS.YORKU.CA
Lassonde School of Engineering, EECS Department, York University, Toronto, ON, Canada

Editors: Shipra Agrawal and Aaron Roth

Abstract

We initiate the study of computability requirements for adversarially robust learning. Adversarially robust PAC-type learnability is by now an established field of research. However, the effects of computability requirements in PAC-type frameworks are only just starting to emerge. We introduce the problem of robust computable PAC (robust CPAC) learning and provide some simple sufficient conditions for this. We then show that learnability in this setup is not implied by the combination of its components: classes that are both CPAC and robustly PAC learnable are not necessarily robustly CPAC learnable. Furthermore, we show that the novel framework exhibits some surprising effects: for robust CPAC learnability it is not required that the robust loss is computably evaluable! Towards understanding characterizing properties, we introduce a novel dimension, the computable robust shattering dimension. We prove that its finiteness is necessary, but not sufficient for robust CPAC learnability. This might yield novel insights for the corresponding phenomenon in the context of robust PAC learnability, where insufficiency of the robust shattering dimension for learnability has been conjectured, but so far a resolution has remained elusive.

Keywords: Adversarial robustness, Computability, PAC learning, CPAC learning

1. Introduction

Formal studies of learnability mostly fall into one of two extremes: the focus is either on purely statistical (or information-theoretic) aspects, where predictors and learners are treated as functions; or requirements of computational efficiency, namely runtime that is polynomial in various parameters, are imposed. Recent work has introduced the study of learnability under a more basic, yet arguably essential requirement, namely that both learners and predictors are computable functions (Agarwal et al., 2020), a framework termed Computable Probably Approximately Correct (CPAC) learnability. For binary classification, a setting where various versions of standard learnability (realizable, agnostic, proper, improper, learnable by any ERM) are well known to be characterized (in the information-theoretic sense) by finiteness of the VC dimension, the addition of computability requirements has revealed a somewhat more fine-grained landscape: CPAC learnability has been shown to be equivalent to a computable version of the VC dimension, the so called *effective VC dimension*, while proper CPAC learnability was proven to be equivalent to the finiteness of the VC dimension combined with the existence of a computable *approximate* ERM learner (Sterkenburg, 2022; Delle Rose et al., 2023).

In this work, we initiate the study of computable PAC learning in the (adversarially) robust setting. Adversarial robust PAC learning has by now been extensively studied, e.g., by Montasser et al.

(2019, 2021); Gourdeau et al. (2021); Awasthi et al. (2023); Lechner et al. (2023) and recent work has provided a characterization of learnability (again, in the purely information-theoretic sense) by a parameter of the one inclusion graph of the learning problem (Montasser et al., 2022). We here explore how adding a requirement of computability adds more subtle aspects to the questions of robust PAC learnability, and expose some perhaps surprising aspects of this setup.

We start by setting up a formal framework for robust learning under computability constraints on the learners and hypotheses, and also the perturbation sets employed. As a warm-up, we provide some simple sufficient conditions for learnability in our setting, but also show that the question of robust CPAC learnability is more subtle than the combination of its components: we prove that there exist classes that are CPAC learnable and robustly PAC learnable with respect to perturbation types that are decidable, yet are not robustly CPAC learnable.

We then explore the role of computability of the robust loss for robust CPAC learnability. Perhaps surprisingly, the robust loss being computably evaluable is neither a necessary nor a sufficient requirement for robust learnability. The insufficiency result relies on showing that there are CPAC learnable classes with computably evaluable robust loss that do not admit a robust empirical risk minimization (RERM) oracle (Montasser et al., 2019) which would return a hypothesis with minimal empirical robust risk on any sample) in the robust agnostic setting. This result is significant as many works assume access to an RERM oracle. In particular, the standard reduction of agnostic to realizable learning in the robust learning setting requires such an oracle (Montasser et al., 2019), and we here demonstrate an example where learnability in the robust realizable case does not extend to the agnostic case through this reduction, since RERM is not computable.

Finally, we explore the role of dimensions for computable robust learnability. We introduce a novel, computable version of the robust shattering dimension. Finiteness of the robust shattering dimension has been shown to to be necessary for robust learnability (Montasser et al., 2019), however it has remained an open question whether it is also a sufficient condition (Montasser et al., 2022). For the computable version, we prove that the finiteness of our *computable robust shattering dimension* is necessary, yet not sufficient, to ensure robust CPAC learnability. This result might provide novel insights for the case without computability constraints. Our results involve deriving a new no free lunch theorem for robust learning, which might also be of independent interest.

1.1. Related Work

Computable Learning. Incorporating aspects of computability into formal learning frameworks is a very recent field of study, originally motivated by the establishement of problems whose learnability is independent of set-theory (Ben-David et al., 2019). The introduction of the notion of computable PAC (CPAC) learning (Agarwal et al., 2020) provided a framework for standard binary classification. Follow up works have by now resulted in a full characterization of CPAC learnability in terms of the so-called effective VC dimension (Sterkenburg, 2022; Delle Rose et al., 2023). Very recent works have extended the study of computability in learning to other learning settings such as continuous domains (Ackerman et al., 2022) and online learning (Hasrati and Ben-David, 2023).

Robust Learning. There is a rich learning theory literature on adversarial robustness. Earlier work focused on the existence of adversarial examples for the stability notion of robustness (Fawzi et al., 2016, 2018a,b; Gilmer et al., 2018; Shafahi et al., 2019; Tsipras et al., 2019), and for its true-label counterpart (Diochnos et al., 2018; Mahloujifar et al., 2019). Many of the works in the former category highlighted the incompatibility between robustness and accuracy, while Bhattacharjee and

Chaudhuri (2021) and Chowdhury and Urner (2022) later argued that these two objectives should not be in conflict by proposing notions of adaptive robusteness. Some studies focus on the sample complexity of robust learning in a PAC setting through a notion called the adversarial VC dimension (Cullina et al., 2018). Some upper bounds depend on the number of perturbations allowed for each instance (Attias et al., 2022b), others on the VC and dual VC dimension of a hypothesis class obtained through an improper learner (Montasser et al., 2019), thus also covering the case of infinite perturbation sets. Ashtiani et al. (2020) later gave an upper bound for *proper* robust learning in terms of the VC dimensions of the class and induced margin class. Montasser et al. (2022) exhibited a characterization of robust learning based on the one-inclusion graph of Haussler et al. (1994) adapted to robust learning. The sample complexity of robust learning has also been studied in the semi-supervised setting (Attias et al., 2022a), through the lens of transformation invariances (Shao et al., 2022), through Rademacher complexity bounds (Khim et al., 2019; Yin et al., 2019; Awasthi et al., 2020) and by the use of online learning algorithms (Diakonikolas et al., 2020; Bhattacharjee et al., 2021). Robust risk relaxations have been studied by Viallard et al. (2021); Ashtiani et al. (2023); Bhattacharjee et al. (2023); Raman et al. (2023), while Balcan et al. (2022, 2023) offered reliability guarantees for robust learning under various notions of robustness. In terms of the true-label robust risk, sample complexity upper and lower bounds with access to random examples only have been derived with respect to distributional assumptions (Diochnos et al., 2020; Gourdeau et al., 2021, 2022b), as this set-up does not allow distribution-free robust learning (Gourdeau et al., 2021). To circumvent computational or information-theoretic obstacles of robust learning with random examples only, recent work has studied robust learning with the help of oracles (Montasser et al., 2021; Gourdeau et al., 2022a; Lechner et al., 2023), while other work has instead looked at curtailing the (computational) power of the adversary (Mahloujifar and Mahmoody, 2019; Garg et al., 2020). Prior works have also analyzed the role of computational efficiency in robust learnability (Bubeck et al., 2019; Degwekar et al., 2019; Gourdeau et al., 2021), however we are the first to exhibit effects of issues arising from the basic, essential requirement of computability.

2. Problem Set-up

Notation. Denote by Σ a finite alphabet, with Σ^* denoting the set of all finite words, or strings, over Σ. We use standard notation for sets and functions, for example, we let $[n] = \{1, 2, \ldots, n\} \subset \mathbb{N}$ denote the set of the first n natural numbers.

Computability. Throughout this paper, we will assume we fixed a programming language. The existence of a program or algorithm is then equivalent to the existence of a Turing machine. We use $(T_i)_{i\in\mathbb{N}}$ to denote a fixed enumeration of all Turing machines/programs. We use the notation $T \equiv S$ to indicate that two Turing Machines have the same behavior as functions (not necessarily the same encoding, thus not necessarily the same index in the enumeration). Further, for a Turing machine T and input x, we write $T(x)\downarrow$ to indicate that T halts on input x and we write $T(x)\uparrow$ to indicate that T loops (does not halt) on input x.

We say that a function $f : \Sigma^* \to \Sigma^*$ is *total computable* if there exists a program P that halts on every string $\sigma \in \Sigma^*$ with $P(\sigma) = f(\sigma)$. A set $S \subseteq \Sigma^*$ is called *decidable* (or recursive) if there exists a program P such that for every $\sigma \in \Sigma^*$, P halts on σ and outputs whether $\sigma \in S$. Finally, a set $S \subseteq \Sigma^*$ is called *recursively enumerable* (or semidecidable) if there exists a program P that enumerates all strings in S, or, equivalently, if there exists a program P that halts on every input $\sigma \in S$ and, if it halts, correctly indicates whether $\sigma \in S$.

Moreover, we will fix a language and proof system for first-order logic that is both sound and complete: a first-order formula has a proof if and only if it is a tautology. We additionally require that this language has a rich enough vocabulary (with respect to function and relation symbols) to guarantee that the set of all its tautologies is undecidable (e.g. Mendelson (2009), Proposition 3.54 (Church's Theorem)).

Learnability. Let $\mathcal{X}$ be the input space and $\mathcal{Y}$ the label space. We will focus on the case $\mathcal{Y} = \{0,1\}$, i.e., binary classification, and countable domains, thus $\mathcal{X} = \mathbb{N}$. We let $\mathcal{H} \subseteq \{0,1\}^{\mathbb{N}}$ denote a *hypothesis class* on $\mathcal{X}$. Given a joint distribution D on $\mathcal{X} \times \mathcal{Y}$ and a hypothesis $h \in \mathcal{H}$, the *risk* (or error) of h with respect to D is defined as

$$\mathsf{R}(h; D) = \Pr_{(x,y)\sim D}(h(x) \neq y) \quad .$$

We will also denote by $\ell : \mathcal{H} \times \mathcal{X} \times \{0,1\} \to \{0,1\}$ the 0-1 loss function $\ell(h, x, y) = \mathbf{1}[h(x) \neq y]$, and we use the notation $\mathsf{R}(h; S) = \frac{1}{m}\sum_{i=1}^{m} \ell(h, x_i, y_i)$ to denote the *empirical risk* of h on a sample $S = \{(x_i, y_i)\}_{i=1}^{m} \in (\mathcal{X} \times \mathcal{Y})^m$. We operate in the PAC-learning framework of (Valiant, 1984).

Definition 1 (Agnostic PAC Learnability) *A hypothesis class $\mathcal{H}$ is* PAC learnable in the agnostic setting *if there exists a learner $\mathcal{A}$ and function $m(\cdot,\cdot)$ such that for all $\epsilon, \delta \in (0,1)$ and for any distribution D, if the input to $\mathcal{A}$ is an i.i.d. sample S from D of size at least $m(\epsilon, \delta)$, then, with probability at least $(1-\delta)$ over the samples, the learner outputs a hypothesis $\mathcal{A}(S)$ with $\mathsf{R}(\mathcal{A}(S); D) \leq \inf_{h\in\mathcal{H}} \mathsf{R}(h; D) + \epsilon$. The class is said to be* PAC learnable in the realizable setting *if the above holds under the condition that $\inf_{h\in\mathcal{H}} \mathsf{R}(h; D) = 0$.*

It is well known that a (binary) hypothesis class is PAC learnable if and only if it has finite *VC dimension* (Vapnik and Chervonenkis, 1971), which is defined below.

Definition 2 (Shattering and VC dimension (Vapnik and Chervonenkis, 1971)) *Given a class of functions $\mathcal{H}$ from $\mathcal{X}$ to $\{0,1\}$, we say that a set $S \subseteq \mathcal{X}$ is* shattered by $\mathcal{H}$ *if the restriction of $\mathcal{H}$ to S is the set of all function from S to $\{0,1\}$. The VC dimension of a hypothesis class $\mathcal{H}$, denoted $\mathsf{VC}(\mathcal{H})$, is the size d of the largest set that can be shattered by $\mathcal{H}$. If no such d exists then $\mathsf{VC}(\mathcal{H}) = \infty$.*

Computable learnability. The above notion of learnability only considers the size of a sample needed to ensure generalization; there are no computational limitations. *Efficient* PAC learnability (Valiant, 1984) requires that the algorithm $\mathcal{A}$ run in time that is polynomial in the learning parameters, and that its output be polynomially evaluable. Sitting in the middle of these two viewpoints is *computable* PAC (CPAC) learnability (Agarwal et al., 2020), where we do not require computational efficiency, but rather that the learning algorithm and its output be computable. We additionally require that the hypothesis class $\mathcal{H}$ be computably representable (Agarwal et al. (2020), Remark 4).

Definition 3 (Computable Representation of a Hypothesis class (Agarwal et al., 2020)) *A class of functions $\mathcal{H}$ is* decidably representable (DR) *if there exists a decidable set of programs $\mathcal{P}$ such that the set of all functions computed by a program in $\mathcal{P}$ equals $\mathcal{H}$. We call it* recursively enumerably *representable (RER) if there exists such a set of programs that is recursively enumerable.*

The following definition incorporates these requirements for the class that a learner outputs.

Definition 4 (CPAC Learnability, (Agarwal et al., 2020)) *We say that a class $\mathcal{H}$ is (agnostic) CPAC learnable, if there is a computable (agnostic) PAC learner for $\mathcal{H}$ that outputs total computable functions as predictors and uses a decidable (recursively enumerable) representation for these.*

Adversarial robustness. Let $\mathcal{U} : \mathcal{X} \to 2^{\mathcal{X}}$ be a *perturbation type*, assigning each point $x \in \mathcal{X}$ a region $\mathcal{U}(x)$ accessible to an adversary, with the convention $x \in \mathcal{U}(x)$. We define the robust risk as:

$$\mathsf{R}_{\mathcal{U}}(h; D) = \Pr_{(x,y)\sim D}(\exists z \in \mathcal{U}(x) \,.\, h(z) \neq y) \quad .$$

Similarly as in the standard setting, we will consider the robust loss function $\ell^{\mathcal{U}} : \mathcal{H} \times \mathcal{X} \times \{0,1\} \to \{0,1\}$ defined as $\ell^{\mathcal{U}}(h, x, y) = \mathbf{1}[\exists z \in \mathcal{U}(x) \,.\, h(x) \neq y]$, and denote the empirical robust risk of predictor h over a sample S by $\mathsf{R}_{\mathcal{U}}(h; S)$.

We will here work with this *stability* notion of robustness, while prior work has also explored other notions of robustness with emphasis on label correctness (Mahloujifar et al., 2019; Gourdeau et al., 2021; Bhattacharjee and Chaudhuri, 2021; Chowdhury and Urner, 2022).

Definition 5 (Agnostic Robust PAC Learnability (Montasser et al., 2019)) *A hypothesis class $\mathcal{H}$ is $\mathcal{U}$-robustly PAC learnable in the agnostic setting if there exists a learner $\mathcal{A}$ and a function $m(\cdot,\cdot)$ such that for all $\epsilon, \delta \in (0,1)$ and for every distribution D, if the input to $\mathcal{A}$ is an i.i.d. sample S from D of size at least $m(\epsilon, \delta)$, then, with probability at least $(1-\delta)$ over the samples, the learner outputs a hypothesis $\mathcal{A}(S)$ with $\mathsf{R}_{\mathcal{U}}(\mathcal{A}(S); D) \leq \inf_{h\in\mathcal{H}} \mathsf{R}_{\mathcal{U}}(h; D) + \epsilon$. The class is said to be $\mathcal{U}$-robustly PAC learnable in the realizable setting if the above holds under the condition that $\inf_{h\in\mathcal{H}} \mathsf{R}_{\mathcal{U}}(h; D) = 0$.*

Computable robust learnability. We can straightforwardly adapt the definition of CPAC learnability to its robust counterpart:

Definition 6 (Robust CPAC Learnability) *We say that a class $\mathcal{H}$ is $\mathcal{U}$-robustly (agnostic) CPAC learnable, if there exists a $\mathcal{U}$-robust PAC learner for $\mathcal{H}$ that also satisfies the computability requirements of a CPAC learner.*

Similarly to the requirements for $\mathcal{H}$, we can require the perturbation type $\mathcal{U}$ also be DR or RER:

Definition 7 (Representations of Perturbation Types) *Let $\mathcal{U} : \mathcal{X} \to 2^{\mathcal{X}}$ be a perturbation type. Then $\mathcal{U}$ is said to be* decidably representable (or recursively enumerable) *if the set $\{(x, z) \,.\, x \in \mathcal{X}, z \in \mathcal{U}(x)\}$ is decidable (or recursicely enumerable, repectively).*

Asking for perturbation types to be decidably representable is quite a natural requirement. Indeed, we show in Appendix A.2 an example that uses a perturbation type $\mathcal{U}$ that is not DR and thus makes the impossibility of $\mathcal{U}$-robust CPAC learning trivial.

As opposed to the binary loss, which depends only on a simple comparison between two labels, and can thus always be evaluated for computable predictors, the robust loss is not always computably evaluable. In Section 4.2, we explore the role of the following notion of loss computability.

Definition 8 (Robust loss computably evaluable) *Given a hypothesis class $\mathcal{H}$, and perturbation type $\mathcal{U}$, we say that the robust loss for $\mathcal{U}$ is* computably evaluable on $\mathcal{H}$ *if $\ell^{\mathcal{U}} : \mathcal{H} \times \mathcal{X} \times \{0,1\} \to \{0,1\}$ is a total computable function.*

Proper learnability and empirical risk minimization. For all the above notions of learnability, we call the class $\mathcal{H}$ *properly PAC/ CPAC/$\mathcal{U}$-robustly PAC/$\mathcal{U}$-robustly CPAC learnable* if there exists a learner that satisfies the corresponding definition of learnability and always outputs functions from $\mathcal{H}$. A learner $\mathcal{A}$ is said to *perform empirical risk minimization (ERM)*, or *implement an* $\mathsf{ERM}_{\mathcal{H}}$ *oracle*, if it always outputs a predictor from the hypothesis class of minimal empirical risk, that is for all $m \in \mathbb{N}$ and samples $S \in (\mathcal{X} \times \mathcal{Y})^m$,

$$\mathcal{A}(S) \in \arg\min_{h \in \mathcal{H}} \mathsf{R}(h, S) \ .$$

Robust empirical risk minimization (RERM) is defined analogously for the robust risk $\mathsf{R}_{\mathcal{U}}$. We now outline different types of RERM oracles that we will study throughout this work. A labelled sample S is said to be *robustly realizable* if $\min_{h \in \mathcal{H}} \mathsf{R}_{\mathcal{U}}(h; S) = 0$, and *robustly non-realizable* otherwise. We will distinguish between three cases:

• **weak-realizable** $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ **oracle:** for any robustly realizable sample S, the oracle outputs h with $\mathsf{R}_{\mathcal{U}}(h; S) = 0$. On robustly non-realizable samples such an oracle is allowed not to halt.

• **strong-realizable** $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ **oracle:** for any robustly realizable sample S, the oracle outputs h with $\mathsf{R}_{\mathcal{U}}(h; S) = 0$. For any robustly non-realizable sample, it halts and outputs "not-robustly-realizable".

• **agnostic** $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ **oracle:** the oracle performs RERM on both robustly realizable and robustly non-realizable samples.

Note that, if the robust loss is computably evaluable, then the existence of a computable strong realizable $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ oracle is equivalent to the existence of an agnostic $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ oracle. Moreover, a strong-realizable oracle always implies the existence of an agnostic oracle.

Useful facts. Recall the following sufficient condition for CPAC learnability, which we later use:

Fact 9 (Agarwal et al. (2020); Sterkenburg (2022)) *If $\mathsf{VC}(\mathcal{H})$ is finite and there exists a total computable function that implements an $\mathsf{ERM}_{\mathcal{H}}$ oracle, then $\mathcal{H}$ is CPAC learnable.*

We now recall the following result by Ashtiani et al. (2020), which outlines sufficient conditions for the proper robust learnability of a class. Throughout this text, we will use the notation of Ashtiani et al. (2020), where the *margin class* $\mathcal{H}^{\mathcal{U}}_{\text{mar}} := \left\{\text{mar}^{\mathcal{U}}_h\right\}_{h \in \mathcal{H}}$ consists of the margin sets $\text{mar}^{\mathcal{U}}_h := \{x \in \mathcal{X} \mid \exists z \in \mathcal{U}(x) \ . \ h(x) \neq h(z)\}$, i.e., given $\mathcal{H}$ and $\mathcal{U}$, the instances in $\mathcal{X}$ that incur a robust loss of 1 due to a perturbation that is labeled differently by h (rather than due to mislabeling).

Fact 10 (Theorem 7 in (Ashtiani et al., 2020)) *If both the VC dimension of a class $\mathcal{H}$ and the VC dimension of $\mathcal{H}^{\mathcal{U}}_{mar}$ are finite, then $\mathcal{H}$ is properly (agnostically) $\mathcal{U}$-robustly PAC learnable.*

3. Warm-up: Simple Sufficient Conditions for Robust CPAC Learnability

We start by exhibiting conditions ensuring (proper) robust CPAC learnability in the agnostic setting.

Fact 11 *Let hypothesis class $\mathcal{H}$ and perturbation type $\mathcal{U}$ be such that $\mathsf{VC}(\mathcal{H}) + \mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{mar}) < \infty$ and there exists a total computable function that implements an agnostic $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ oracle. Then $\mathcal{H}$ is properly $\mathcal{U}$-robustly CPAC learnable. If there exists a computable function that implements a weak-realizable $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ oracle, then $\mathcal{H}$ is $\mathcal{U}$-robustly CPAC learnable in the realizable case.*

Proof By Fact 10, $D := \mathsf{VC}(\mathcal{H}) + \mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}}) < \infty$ implies that a finite sample of size $O\left(\frac{D \log D + \log 1/\delta}{\epsilon^2}\right)$ is sufficient to guarantee robust generalization w.r.t. $\mathcal{U}$, whenever a robust risk minimizer for a given sample is returned. Since robust ERM is computably implementable, we are done. ■

Fact 12 *Let hypothesis class $\mathcal{H}$ be RER and perturbation type $\mathcal{U}$ be such that $\ell^{\mathcal{U}}$ is computably evaluable on $\mathcal{H}$. Then $\mathcal{H}$ admits a weak realizable RERM oracle.*

Proof Let S be a robustly realizable input sample. The following procedure computes $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$. Since the class is RER, we can iterate through the class $\mathcal{H} = (h_i)_{i \in \mathbb{N}}$ and for each h_i:
• Compute $\mathsf{R}^{\mathcal{U}}(h_i, S)$, which is possible since the robust loss is computably evaluable;
• Check whether $\mathsf{R}^{\mathcal{U}}(h_i, S) = 0$. If so, halt and output h_i, otherwise continue.
As we know that there exists $h \in \mathcal{H}$ with $R^{\mathcal{U}}(h, S) = 0$, we know that the algorithm will halt. ■

Remark 13 *Combining Facts 11 and 12 shows that for a RER hypothesis class $\mathcal{H}$ and a perturbation type $\mathcal{U}$, the conditions $\mathsf{VC}(\mathcal{H}) + \mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{mar}) < \infty$ and $\ell^{\mathcal{U}}$ on $\mathcal{H}$ being computably evaluable are sufficient to guarantee proper robust CPAC learnability in the robustly realizable case.*

Now, we show that having access to a computable online learner (see Definition 15 in (Hasrati and Ben-David, 2023)) and a counterexample oracle, together with the margin class having finite VC dimension, is sufficient to guarantee robust learnability. The counterexample oracle we will use is the Perfect Attack Oracle (PAO) of (Montasser et al., 2021), which takes as input $h, x, y \in \mathcal{H} \times \mathcal{X} \times \{0, 1\}$ and returns $z \in \mathcal{U}(x)$ such that $h(z) \neq y$, if such a counterexample exists. The proof is included in Appendix A.1.

Proposition 14 *Let $\mathcal{H}$ be computably online learnable, and let $\mathcal{U}$ be such that $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{mar}) < \infty$. Then $\mathcal{H}$ is $\mathcal{U}$-robustly CPAC learnable with access to the PAO in the realizable setting.*

4. Relating CPAC and Robust CPAC Learning

In this section, we study the relationship between CPAC learnability, robust CPAC learnability and the computable evaluability of the robust loss. We start in Section 4.1 with examples where we have CPAC learnability, but not robust CPAC learnability. The constructions in that section also show that the robust loss being computably evaluable is not sufficient for robust CPAC learnability. In Section 4.2, we show that the robust loss being computably evaluable is in general also not necessary to ensure robust CPAC learning, which might be unexpected.

4.1. Robust CPAC learnability is not implied by Robust PAC + CPAC learnability

In this section, we first look at two examples of hypothesis classes and perturbation types where CPAC learnability and robust learnability do not imply robust CPAC learnability. In both cases, the perturbation types are DR, in contrast to Example 1. Our first result holds in the robust agnostic case for general, improper learners, and the second, in the robust realizable case for proper learners.

Theorem 15 *There exists a hypothesis class $\mathcal{H}$ and perturbation type $\mathcal{U}$ such that (i) $\mathcal{H}$ is properly CPAC learnable, (ii) $\mathcal{U}$ is decidably representable and (iii) $\mathcal{H}$ is properly $\mathcal{U}$-robustly PAC learnable, but $\mathcal{H}$ is not (improperly) $\mathcal{U}$-robustly CPAC learnable in the agnostic setting.*

Proof [Sketch] Fix a proof system for first-order logic over a rich enough vocabulary that is sound and complete. Let $\{\varphi_i\}_{i\in\mathbb{N}}$ and $\{\pi_j\}_{j\in\mathbb{N}}$ be enumerations of all theorems and proofs, respectively. We define the following hypothesis class on $\mathbb{N}$:

$$\mathcal{H} = \{h_a \ : \ \forall i \in \mathbb{N},\ h_a(2i) = \mathbf{1}[i \leq a] \wedge h_a(2i+1) = 1\}_{a\in\mathbb{N}\cup\{\infty\}} \ .$$

Thus, $\mathcal{H}$ is the the concept class where each function defines a threshold on even integers, and is the constant function 1 on odd integers.

The perturbation sets are defined as follows. For any $i \in \mathbb{N}$ we set:

$$\begin{aligned}
&\mathcal{U}(6i) = \{6i\} \cup \{2j+1\}_{j\in\mathbb{N}} \ , \\
&\mathcal{U}(6i+2) = \{6i+2\} \ , \\
&\mathcal{U}(6i+4) = \{6i+2, 6i+4\} \cup \{2j+1 \mid \pi_j \text{ is a proof of theorem } \varphi_i\}_{j\in\mathbb{N}} \ , \\
&\mathcal{U}(2i+1) = \{2i+1\} \ .
\end{aligned}$$

The properties (i)-(iii) are fulfilled by this construction, as we will show in detail in the appendix. Furthermore, we note that the question of whether $\mathcal{U}(6i)\cap\mathcal{U}(6i+4)$ is empty is equivalent to whether there exists a proof for theorem φ_i, and thus is undecidable. We now show that this undecidable problem can be reduced to agnostically robustly CPAC learning $\mathcal{H}$ with respect to $\mathcal{U}$. To see this define the distributions D_i on $\mathbb{N} \times \{0,1\}$ as follows:

$$D_i((6i,1)) = 1/2, \qquad D_i((6i+2,1)) = 1/6, \qquad D_i((6i+4,0)) = 1/3 \ .$$

For a learner to succeed on all the D_i, it needs to decide whether $\mathcal{U}(6i) \cap \mathcal{U}(6i+4) = \emptyset$ for all i. ■

For more detail and the full proof we refer the reader to Appendix B.1.

The result above holds in the improper agnostic setting. Next, we look at the robustly realizable setting, and show an impossibility result in case we require a *proper* robust learning algorithm.

Theorem 16 *There exists a hypothesis class $\mathcal{H}$ and perturbation type $\mathcal{U}$ such that (i) $\mathcal{H}$ is properly CPAC learnable, (ii) $\mathcal{U}$ is decidably representable, and (iii) $\mathcal{H}$ is properly $\mathcal{U}$-robustly PAC learnable, but not properly $\mathcal{U}$-robustly CPAC learnable in the realizable setting.*

Proof Let $(T_i)_{i\in\mathbb{N}}$ be an enumeration of Turing Machines. We define the hypothesis class on $\mathbb{N}$:

$$\mathcal{H} = \{h_{a,b,c} \ : \ \forall i \ (h_{a,b,c}(2i) = 1 \text{ iff } i \in \{a,b\}) \wedge (h_{a,b,c}(2i+1) = c)\}_{a,b\in\mathbb{N},\ c\in\{0,1\}} \ ,$$

i.e., $\mathcal{H}$ is the class of functions that only maps at most two even numbers to 1 and are constant on the odd numbers. First note that $\mathsf{VC}(\mathcal{H}) = 3$, as for any shattered set $X \subseteq \mathbb{N}$, we can have at most one odd integer in X, and at most two even ones. It is straightforward to show that any set of the form $\{2i, 2j, 2k+1\}$ for $i, j, k \in \mathbb{N}$ can be shattered. This implies, by Theorem 6 in (Montasser et al., 2019), that $\mathcal{H}$ is (improperly) robustly learnable for any perturbation type $\mathcal{U}$. We will argue below that, for the perturbation type we employ, $\mathcal{H}$ is in fact properly $\mathcal{U}$-robustly CPAC learnable. Second, as ERM is easily implementable for $\mathcal{H}$, we have that $\mathcal{H}$ is properly CPAC learnable.

We define the perturbation sets as follows:

$$\begin{aligned}
&\mathcal{U}(2i) = \{2i\} \cup \{2j+1 : T_i(i) \text{ halts within } j \text{ steps } \} \ , \\
&\mathcal{U}(2i+1) = \{2i+1\} \ .
\end{aligned}$$

Note that $\mathcal{U}$ is decidably representable, as for any two points x_1, x_2, the program outlined in Appendix B.2 decides whether $x_2 \in \mathcal{U}(x_1)$.

We now show that for this $\mathcal{U}$, $\mathcal{H}$ is *properly* $\mathcal{U}$-robustly learnable. To see this, we again use the terminology of Ashtiani et al. (2020), and consider the margin class $\mathcal{H}^{\mathcal{U}}_{\text{mar}}$, where each $h_{a,b,c} \in \mathcal{H}$ fixes a set $\text{mar}^{\mathcal{U}}_{h_{a,b,c}}$, which we will denote by $\text{mar}_{a,b,c}$ for brevity. Note that, from the property $k \subseteq \mathcal{U}(k)$ that holds for all $k \in \mathbb{N}$, only even integers can belong in some set $\text{mar}_{a,b,c}$. Recall that $h_{a,b,c}$ maps at most two even numbers a and b to 1 and all odd numbers to $c \in \{0, 1\}$.

We distinguish two cases:

- $c = 0$: then $\text{mar}_{a,b,0} = \{2a \mid T_a(a) \text{ halts}\} \cup \{2b \mid T_b(b) \text{ halts}\}$,
- $c = 1$: then $\text{mar}_{a,b,1} = \{2i \mid T_i(i) \text{ halts}\} \setminus \{2a, 2b\}$.

From this, we can see that shattered sets must only contain even integers k such that $T_{k/2}(k/2)$ halts. We now argue that $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}}) = 5$. Consider a set $X = \{k_1, \ldots, k_6\}$ of size 6 such that k_i is even and $T_{k_i/2}(k_i/2)$ halts for all $i = 1, \ldots, 6$. Then the subset $\{k_1, k_2, k_3\}$ cannot be part of the projection of $\mathcal{H}^{\mathcal{U}}_{\text{mar}}$ onto X, and thus $\{k_1, \ldots, k_6\}$ cannot be shattered. Indeed, note that for $c = 0$, the margin set $\text{mar}_{a,b,0}$ has size 2 only for any $a, b \in \mathbb{N}$. Thus $\{k_1, k_2, k_3\} \in \text{mar}_{a,b,c}$ would imply that there exists $a, b \in \mathbb{N}$ such that $k_1, k_2, k_3 \in \text{mar}_{a,b,1}$. However then, by definition, there are at most two of k_4, k_5, k_6 that are not in $\text{mar}_{a,b,1}$, a contradiction. Finally, consider a set $X = \{k_1, \ldots, k_5\}$ of size 5 such that k_i is even and $T_{k_i/2}(k_i/2)$ halts for all $i = 1, \ldots, 5$. Any subset $X' \subseteq X$ is contained in the projection of $\mathcal{H}^{\mathcal{U}}_{\text{mar}}$ onto X: if $|X'| \geq 3$ there exists $a, b \in \mathbb{N}$ such that $X' \in \text{mar}_{a,b,1}$, and otherwise if $|X'| \leq 2$ there exists $a, b \in \mathbb{N}$ such that $X' \in \text{mar}_{a,b,0}$, and thus X is shattered.

Now, for any $i, k \in \mathbb{N}$, with $i \neq k$, we define a distribution $D_{i,k}$ on $\mathbb{N} \times \{0, 1\}$ as follows:

$$D_{i,k}(2i, 1) = D_{i,k}(2k, 0) = 1/2 \ .$$

We will now show that $\mathcal{H}$ is not properly $\mathcal{U}$-robustly CPAC learnable with respect to the set of distributions $\{D_{i,k}\}_{i,k \in \mathbb{N}}$ in the realizable case. Note that robustly realizability here implies that at most one of the sets $\mathcal{U}(2i)$ or $\mathcal{U}(2k)$ is not a singleton. Indeed, if $\mathcal{U}(2i) = \{2i\}$ then $T_i(i)$ does not halt, whereas if $\{2i\} \subset \mathcal{U}(2i)$ (strict inequality) then $T_i(i)$ halts, and similarly for $2k$. Hence, since we have $D_{i,k}(2i, 1) = D_{i,k}(2k, 0) = 1/2$, to choose between $h_{i,i',0}$ and $h_{i,i',1}$ for some fixed $i' \in \mathbb{N}$ with $i' \neq k$ and $i' \neq i$ we need to know whether either of $\mathcal{U}(2i)$ or $\mathcal{U}(2k)$ is a singleton.

Now consider the following decision problem, which we term TwoHalt. Given two Turing machines T_i and T_j, the task is to correctly predict which run out of $T_i(i)$ and $T_j(j)$ halts if only one of them does. If both runs loop, the solver should still halt and produce an output; in case both halt the decider may run indefinitely or halt and produce some output. That is, we only require a correct decision if exactly one of $T_i(i)$ and $T_j(j)$ halts, and require the program to halt if both loop. We show below (see Lemma 29) that no algorithm can satisfy these requirements.

We now argue that a $\mathcal{U}$-robust proper CPAC learner can be used to solve TwoHalt, and thus, by invoking Lemma 29, which is stated and proved in Appendix B.3, conclude that no such learner exists. If a $\mathcal{U}$-robust proper CPAC learner for $\mathcal{H}$ existed, there is a sample size M that suffices for robust loss at most $\epsilon = 1/3$ with probability $1 - \delta = 2/3$ over all M size samples from $D_{i,k}$. We then solve TwoHalt by running the learner on all M-length sequences over the points $(2i, 1)$ and $(2k, 0)$, and taking the majority for the label that the resulting predictor gives on the odd numbers. ■

4.2. Robust CPAC learnability and its relationship to the computability of $\ell^{\mathcal{U}}$ and RERM

One of the obstacles with robust CPAC learning is that, even if a class consists of computable functions, the robust loss might not be pointwise computable. This is in contrast to the CPAC setting for which the binary loss is always computable for computable functions. It is then natural to ask what relationship exists between the computability of the robust loss and the robust CPAC learnability of a class. The following result demonstrates that the computability of the pointwise robust loss for all hypotheses in the class is not a necessary criterion for robust CPAC learnability.

Theorem 17 *There exist $\mathcal{H}$ and $\mathcal{U}$ such that (i) $\mathcal{H}$ is properly CPAC learnable, (ii) $\mathcal{U}$ is decidably representable, and (iii) $\mathcal{H}$ is properly $\mathcal{U}$-robustly CPAC learnable in the agnostic setting and admits a computable agnostic $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ oracle, but the function $\ell^{\mathcal{U}}$ is not computably evaluable on $\mathcal{H}$.*

Proof [Sketch] This theorem is proven with the following construction. Let $\mathcal{X}$, $(\varphi_i)_{i\in\mathbb{N}}$ and $(\pi_i)_{i\in\mathbb{N}}$ be defined as in the construction for Theorem 15. Let $(i,0)$, represent the i-th formula and for any i,j let (i,j) represent the j-th proof (independently of i). Let $\mathcal{H} = \{h_{a,b}\}_{a,b\in\mathbb{N}\cup\{0,\infty\}}$, where

$$h_{a,b}(i,j) = \begin{cases} 1 & \text{if } i < a \text{ or } (i = a \text{ and } j \leq b) \\ 0 & \text{otherwise} \end{cases} .$$

It is clear that $\mathsf{VC}(\mathcal{H}) = 1$ and that ERM is computably implementable, so $\mathcal{H}$ is properly CPAC learnable. We define the perturbation regions as follows:

$$\mathcal{U}(i,0) = \{(i,0)\} \cup \{(i,j) : \pi_j \text{ proves } \varphi_i\} \ ,$$
$$\mathcal{U}(i,j) = \{(i,0),(i,j)\} \ .$$

It is also easy to verify that this perturbation type is DR. In the full proof, we show that in general there is no algorithm that evaluates the loss of each hypotheses $h_{a,b}$ on every point. However, there is an RER subclass $\mathcal{H}' = \{h_{a,\infty} : a \in \mathbb{N}\} \subset \mathcal{H}$ such that every hypothesis in $\mathcal{H}'$ is computably evaluable and such that on any distribution, $\mathcal{H}'$ contains an optimal (with respect to $\mathcal{H}$) hypothesis. It is thus possible to robustly CPAC learn $\mathcal{H}$ by performing RERM over $\mathcal{H}'$. ■

The full proof is included in Appendix C.1. We next show that, even if the robust loss is computably evaluable on $\mathcal{H}$, and the class $\mathcal{H}$ is properly CPAC learnable and properly robustly PAC learnable, neither the existence of a robust ERM oracle nor robust CPAC learnability are implied.

Theorem 18 *There exist a DR hypothesis class $\mathcal{H}$ and a DR perturbation type $\mathcal{U}$ such that (i) $\mathcal{H}$ is properly CPAC learnable, (ii) $\mathcal{H}$ is properly $\mathcal{U}$-robustly PAC learnable, and (iii) the robust loss is computably evaluable, but there is no strong realizable $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$-oracle and $\mathcal{H}$ is not properly agnostically $\mathcal{U}$-robustly CPAC learnable.*

Proof [Sketch] We prove this theorem with the following construction. Let the instance space be $\mathcal{X} = \mathbb{N} \times \mathbb{N}$. Let the hypothesis class be $\mathcal{H} = \{h_{i,j} : i,j \in \mathbb{N}\}$, where

$$h_{i,j}(x) = \begin{cases} 1 & \text{if } x = (i,k) \text{ with } k \leq j \\ 0 & \text{otherwise} \end{cases} ,$$

i.e., $\mathcal{H}$ is the class of initial segments on $\{(i,j)\}_{j\in\mathbb{N}}$ for all $i \in \mathbb{N}$. We set the perturbation types to:

$$\mathcal{U}((i,0)) = \{(i,0)\} \cup \{(i,k) : T_i \text{ does not halt after } k \text{ steps on the empty input } \} \ ,$$
$$\mathcal{U}((i,j)) = \begin{cases} \{(i,j),(i,0)\} & T_i \text{ does not halt after } j \text{ steps on the empty input} \\ \{(i,j)\} & \text{otherwise.} \end{cases}$$

The proof of the impossibility of agnostic robust CPAC learning and of the non-existence of a strong realizable RERM oracle relies on reductions from the Halting problem. ■

The detailed proof shows that the construction in the sketch above fulfills all the requirements for proving the theorem and is included in Appendix C.2. Recall from Fact 12 that the class being RER (a weaker assumption than being DR) and the robust loss being computably evaluable imply the existence of a weak realizable $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$-oracle. We therefore demonstrated a separation between the existence of RERM oracles in the robust realizable versus the agnostic case. Our proof also shows that the existence of weak realizable RERM oracles does not imply the existence of agnostic RERM orcalce. Furthermore, note that the existence of a strong realizable RERM oracle implies computable evaluablility of robust loss. Thus, the result of Theorem 17 shows that the existence of a agnostic RERM oracle does not imply the existence of a strong realizable RERM oracle.

In the non-robust setting, proper *strong* CPAC (SCPAC) learnability (which is CPAC learnability with the additional requirement that the sample complexity function $m(\cdot,\cdot,\cdot)$ be computable) implies the existence of an $\mathsf{ERM}_{\mathcal{H}}$ oracle (Sterkenburg (2022), Theorem 8). We finish this section by showing that a similar result holds in robust learning if the robust loss is computably evaluable. The proof of Proposition 19 below is included in Appendix C.3.

Proposition 19 *Let $\mathcal{H}$ be a hypothesis class and $\mathcal{U}$ a perturbation type such that $\ell^{\mathcal{U}}$ is computably evaluable. Then, if $\mathcal{H}$ is properly $\mathcal{U}$-robustly SCPAC learnable in the agnostic setting, $\mathcal{H}$ admits a computable agnostic $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ oracle; and if $\mathcal{H}$ is properly $\mathcal{U}$-robustly SCPAC learnable in the realizable setting, then $\mathcal{H}$ admits a computable weak realizable $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ oracle.*

5. Necessary Conditions for Robust CPAC Learning

In this section, we define a complexity measure, denoted by $c\text{-dim}_{\mathcal{U}}$, and prove its finiteness is implied by robust CPAC learnability for RER perturbation types. We later study its relationship with the VC dimension, and show that it cannot characterize robust CPAC learnability.

5.1. No-Free-Lunch Theorems

Prior work has shown that standard CPAC learnability is equivalent to the hypothesis class having finite *effective VC dimension*, which can be thought of as a computable version of the VC dimension (Sterkenburg, 2022; Delle Rose et al., 2023). The effective VC dimension of $\mathcal{H}$ is the smallest k, such that there exists a computable function $w : \mathbb{N}^{k+1} \to \{0,1\}^{k+1}$ that, when given a set of domain points of size $k+1$, outputs a labelling that cannot be achieved by any $h \in \mathcal{H}$. The function w is called a *witness function*. The complexity measure $c\text{-dim}_{\mathcal{U}}(\mathcal{H})$ that we introduce in this section is a computable version of the robust shattering dimension $\dim_{\mathcal{U}}(\mathcal{H})$ which has been shown to lowerbound the robust sample complexity (Montasser et al., 2019).

Definition 20 ($\mathcal{U}$-robust shattering dimension Montasser et al. (2019)) *A set $X = \{x_i\}_{i=1}^m$ is $\mathcal{U}$-robustly shattered by $\mathcal{H}$ if there exists a set $Z = \left\{(z_i^0, z_i^1)\right\}_{i=1}^m$ with $x_i \in \mathcal{U}(z_i^0) \cap \mathcal{U}(z_i^1)$ for all $i = 1, \ldots, m$ such that for all $y_1 \ldots, y_m \in \{0,1\}$ there exists $h \in \mathcal{H}$ such that for all $i = 1, \ldots, m$ and $z' \in \mathcal{U}(z_i^{y_i})$ we have that $h(z') = y_i$. The* robust shattering dimension of $\mathcal{H}$, *denoted $\dim_{\mathcal{U}}(\mathcal{H})$, is the largest integer k such that there exists a set of size k that is $\mathcal{U}$-robustly shattered. If arbitrarily large $\mathcal{U}$-robustly shattered sets exist, then $\dim_{\mathcal{U}}(\mathcal{H}) = \infty$.*

Note that, for X to be robustly shattered, for all $i \neq j$ the regions $\mathcal{U}(z_i^1)$ and $\mathcal{U}(z_j^0)$ must be disjoint. Now, contrary to the standard notion of shattering, there are two factors at play in a set being robustly shattered. The first one is whether a set is robustly *"shatterable"*, namely whether, by definition of $\mathcal{U}$–and without considering $\mathcal{H}$–the set X *could* be robustly shattered by an arbitrary set of functions (which holds trivially in the standard case). The second factor, if robust shatterability holds, is as in the standard case: whether there does in fact exists a set of hypotheses from $\mathcal{H}$ that can realize the robust shattering. To highlight this distinction, we introduce the following property:

Definition 21 (Robust shatterability) *A set $X = \{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ is called* $\mathcal{U}$-robustly shatterable *if there exists a set $Z = \left\{(z_i^0, z_i^1)\right\}_{i=1}^m \subseteq \mathcal{X}$ with $x_i \in \mathcal{U}(z_i^0) \cap \mathcal{U}(z_i^1)$ and for all $i \neq j$ the regions $\mathcal{U}(z_i^1)$ and $\mathcal{U}(z_j^0)$ are disjoint. In this case, Z is said to* admit a $\mathcal{U}$-robustly shatterable set.

We now define the computable version of Definition 20, which also uses the notion of a witness function. If $\dim_{\mathcal{U}}(\mathcal{H}) < k$ then for all $m \geq k$, for all $X = \{x_i\}_{i=1}^m$ and for all $Z = \left\{(z_i^0, z_i^1)\right\}_{i=1}^m$ with $x_i \in \mathcal{U}(z_i^0) \cap \mathcal{U}(z_i^1)$ there exists a labelling that is *not* $\mathcal{U}$-robustly realizable by $\mathcal{H}$. Thus:

Definition 22 (Computable robust shattering dimension) *A* k-witness of $\mathcal{U}$-robust shattering dimension *for $\mathcal{H}$ is a function $w : \mathbb{N}^{2(k+1)} \to \{0,1\}^{k+1}$ such that when given a set $Z = \{(z_i^0, z_i^1)\}_{i=1}^{k+1}$ that admits a robustly shatterable set, it outputs $y_1 \ldots, y_{k+1} \in \{0,1\}$ such that for all $h \in \mathcal{H}$ there exist $i = 1, \ldots, m$ and $z' \in \mathcal{U}(z_i^{y_i})$ with $h(z') \neq y_i$. The* computable robust shattering dimension of $\mathcal{H}$, *denoted c-$\dim_{\mathcal{U}}(\mathcal{H})$, is the smallest integer k such that there exists a* computable k*-witness of robust shattering dimension. If no such k exists, c-$\dim_{\mathcal{U}}(\mathcal{H}) = \infty$.*

We note that, by definition, if $\mathcal{U}$ does not admit a shatterable set of size k, then any function is a k-witness. Towards showing that finiteness of the computable robust shattering dimension is a necessary condition for $\mathcal{U}$-robust CPAC learnability, we first formulate a No-Free-Lunch theorem for robust learnability, leaving aside computability considerations for the time being.

Lemma 23 (Robust No-Free-Lunch Theorem) *Let $\mathcal{X}$ be the domain and $\mathcal{U} : \mathcal{X} \to 2^{\mathcal{X}}$ a perturbation type. Let $\mathcal{A}$ be any computable learner. Let m be a number smaller than $|\mathcal{X}|/4$ representing the training set size, with the property that there exist $4m$ points $Z = \left\{(z_i^0, z_i^1)\right\}_{i=1}^{2m}$ such that Z admits a $\mathcal{U}$-robustly shatterable set. Then there exists a distribution D over $\mathcal{X} \times \{0,1\}$ such that (i) there exists a function $f : \mathcal{X} \to \{0,1\}$ such that $\mathsf{R}_{\mathcal{U}}(f; D) = 0$; and (ii) with probability at least $1/7$ over $S \sim D^m$, $\mathsf{R}_{\mathcal{U}}(\mathcal{A}(S); D) \geq 1/8$.*

The proof of Lemma 23 is included in Appendix D.1. We comment on the robust shatterability in the impossibility results from Section 4 in Appendix E.

Remark 24 *First note that we work with $\mathcal{X} = \mathbb{N}$, so the condition $m \leq |\mathcal{X}|/4$ is immediately satisfied for any m. Second, note that the robust shatterability condition in the theorem, which depends on the perturbation type $\mathcal{U}$, is satisfied if the regions $\mathcal{U}(x)$ are, e.g., balls of finite radii.*

We are now ready to state the computable version of Lemma 23, which involves not only the existence of a distribution and function that are robustly realizable on which the learner performs poorly, but also the ability to computably find such a pair:

Lemma 25 (Computable Robust No-Free-Lunch Theorem) *Fix instance space $\mathcal{X}$ and perturbation type $\mathcal{U} : \mathcal{X} \to 2^{\mathcal{X}}$. Let $\mathcal{A}$ be any computable learner and furthermore suppose that the perturbation type $\mathcal{U}$ is recursively enumerably representable (RER). Then for any $m \in \mathbb{N}$, any domain $\mathcal{X}$ of size at least $4m$ and any subset $Z = \left\{(z_i^0, z_i^1)\right\}_{i=1}^{2m}$ that admits a $\mathcal{U}$-robustly shatterable set, we can computably find a function $f : \mathcal{X} \to \{0,1\}$ that is computable on Z such that*

$$\mathop{\mathbb{E}}_{S\sim D^m}\left[R_{\mathcal{U}}(\mathcal{A}(S); D)\right] \geq 1/4 \qquad \textit{and} \qquad \Pr_{S\sim D^m}\left(R_{\mathcal{U}}(\mathcal{A}(S); D) \geq 1/8\right) \geq 1/7 \ , \tag{1}$$

where D is the uniform distribution on $\{(z_i^{y_i}, y_i)\}_{i=1}^{2m}$ for some $y \in \{0,1\}^{2m}$ such that for all $i = 1, \ldots, 2m$ and for all $z' \in \mathcal{U}(z_i^{y_i})$ we have that $f(z') = y_i$.

The proof of Lemma 25 follows a similar argument as the proof of the computable No-Free-Lunch Theorem shown in Agarwal et al. (2020), but the computation of the robust loss differs:

Proof We first computably find a shatterable set $X = \{x_1, x_2, \ldots, x_{2m}\}$ corresponding to the set of pairs $Z = \left\{(z_i^0, z_i^1)\right\}_{i=1}^{2m}$. That is $x_i \in \mathcal{U}(z_i^0) \cap \mathcal{U}(z_i^1)$ for all $i \in [2m]$. Since the sets $\mathcal{U}(z_i^0)$ and $\mathcal{U}(z_i^1)$ are recursively enumerable, we can computably find such an x_i in their intersection for each i. We simply enumerate the instances from $\mathcal{U}(z_i^0)$ and $\mathcal{U}(z_i^1)$ by alternating between the two sets until we find an instance that belongs to both enumerations. Because we are guaranteed that Z admits a $\mathcal{U}$-robustly shatterable set, this process is guaranteed to halt.

The existence of a function f and distribution D satisfying Equation 1 follows directly from Lemma 23, so it remains to show that, using $\mathcal{A}$ we can computably find such an f and D. Note that the construction of the pairs (f_j, D_j) in Lemma 23 corresponds to pairs of $\mathcal{U}$-robustly realizable distributions. There are $T = 2^{2m}$ such pairs, as putting mass on $z_i^{y_i}$ implies not putting mass on $z_i^{\neg y_i}$. Second, for a given (f_j, D_j) and a fixed sample size m, there are $k = (2m)^m$ sequences $\{S_l^j\}_{l=1}^k$ of m instances drawn from D_j, each equally likely as D_j is uniform on its support. We now run $\mathcal{A}$ on all these sequences, and compute a lower bound on the expected robust loss of $\mathcal{A}$ on (f_j, D_j) by employing the labels of the learner's output on the shatterable set X:

$$\begin{aligned}\mathop{\mathbb{E}}_{S\sim D_j^m}\left[\mathsf{R}_{\mathcal{U}}(\mathcal{A}(S); D_j)\right] = \frac{1}{k}\sum_{l=1}^{k}\mathsf{R}_{\mathcal{U}}(\mathcal{A}(S_l^j); D_j) &= \frac{1}{k}\sum_{l=1}^{k}\frac{1}{2m}\sum_{i=1}^{2m}\ell^{\mathcal{U}}\left(\mathcal{A}(S_l^j), z_i^{(j)}, y_i^{(j)}\right) \ , \\ &\geq \frac{1}{k}\sum_{l=1}^{k}\frac{1}{2m}\sum_{i=1}^{2m}\mathbb{1}\left[\mathcal{A}(S_l^j)(x_i) \neq y_i^{(j)}\right]\end{aligned}$$

As argued in the proof of Lemma 23, there exist f_j, D_j such that $\frac{1}{k}\sum_{l=1}^{k}\frac{1}{2m}\sum_{i=1}^{2m}\mathbb{1}\left[\mathcal{A}(S_l^j)(x_i) \neq y_i^{(j)}\right] \geq 1/8$. Furthermore, we note that the for every j, the set of sequences $\{S_l^j\}_{j=1}^k$ can be computably generated and that the expression $\frac{1}{k}\sum_{l=1}^{k}\frac{1}{2m}\sum_{i=1}^{2m}\mathbb{1}\left[\mathcal{A}(S_l^j)(x_i) \neq y_i^{(j)}\right]$ can be computably evaluated. Thus by iterating through $j = 1, \ldots, T$ until finding j with $\frac{1}{k}\sum_{l=1}^{k}\frac{1}{2m}\sum_{i=1}^{2m}\mathbb{1}\left[\mathcal{A}(S_l^j)(x_i) \neq y_i^{(j)}\right] \geq 1/8$, we can computably find (f_j, D_j) with $\mathop{\mathbb{E}}_{S\sim D_j^m}\left[\mathsf{R}_{\mathcal{U}}(\mathcal{A}(S); (D_j)\right] \geq 1/8$. We further note that the corresponding labelling f_j can be computably evaluated on all elements of Z as required. ■

Note that if the perturbation sets are DR, then the function f above is total computable on $\mathcal{X}$. We now state the main result of this section:

Theorem 26 *Let $\mathcal{H}$ be (improperly) $\mathcal{U}$-robustly CPAC learnable and suppose the perturbation type $\mathcal{U}$ is RER. Then the computable $\mathcal{U}$-robust shattering dimension $c\text{-}\dim_{\mathcal{U}}(\mathcal{H})$ of $\mathcal{H}$ is finite, i.e., $\mathcal{H}$ admits a computable k-witness of $\mathcal{U}$-robust shattering dimension for some $k \in \mathbb{N}$.*

The proof, included in Appendix D.2, follows the reasoning of Lemma 9 in Sterkenburg (2022).

5.2. Relationship between the effective VC dimension and $c\text{-}\dim_{\mathcal{U}}$

It has been shown that the dimension $\dim_{\mathcal{U}}(\mathcal{H})$ yields a lower bound on the sample complexity for $\mathcal{U}$-robust learning (Montasser et al., 2019). However, it has remained an open question whether this dimension also provides an upper bound on the sample complexity, thus whether it actually characterizes $\mathcal{U}$-robust learnability. Here we show that a corresponding conjecture in the CPAC setting is not true. We first relate the effective VC dimension, which we will identify as c-VC to denote computability, to the computable robust shattering dimension.

Lemma 27 *For any class $\mathcal{H}$ and perturbation type $\mathcal{U}$ admitting a recursively enumerable representation, we have $c\text{-}\dim_{\mathcal{U}}(\mathcal{H}) \leq c\text{-}\mathsf{VC}(\mathcal{H})$.*

Proof Let $\mathcal{H}$ have finite effective VC dimension k. Then there exist a computable function $w : \mathbb{N}^{k+1} \to \{0,1\}^{k+1}$ such that w is a k-witness of VC dimension for $\mathcal{H}$. Now, let $Z = \left\{(z_i^0, z_i^1)\right\}_{i=1}^{k+1}$ admit a robustly shatterable set. Because $\mathcal{U}$ is RER, we can find a robustly shatterable set X of size $k+1$ for Z. It suffices to output the labelling $w(X)$, which is not achievable by any $h \in \mathcal{H}$, and thus not robustly achievable by any $h \in \mathcal{H}$ as well, and so $c\text{-}\dim_{\mathcal{U}}(\mathcal{H}) \leq k$, as required. ■

We know that CPAC learnability of a class $\mathcal{H}$ implies $c\text{-}\mathsf{VC}(\mathcal{H}) < \infty$ (Sterkenburg, 2022), and thus, by the above lemma, CPAC learnability implies $c\text{-}\dim_{\mathcal{U}}(\mathcal{H}) \leq c\text{-}\mathsf{VC}(\mathcal{H}) < \infty$. On the other hand, we have shown in this work that there exist $\mathcal{H}, \mathcal{U}$ such that CPAC learnability does not imply $\mathcal{U}$-robust CPAC learnability (see Theorem 15). Thus there are classes that are not $\mathcal{U}$-robust CPAC learnable while having finite effective robust shattering dimension, meaning that this dimension does not characterize robust CPAC learnability:

Corollary 28 *The dimension $c\text{-}\dim_{\mathcal{U}}(\mathcal{H})$ does not characterize $\mathcal{U}$-robust CPAC learnability.*

6. Conclusion

We have initiated the study of robust computable PAC learning, and provided a formal framework for its analysis. We showed sufficient conditions that enable robust CPAC learnability, as well as showed that CPAC learnability and robust learnability are not in themselves sufficient to guarantee robust CPAC lernability. We also exhibited a counterintuitive relationship between the computability of the robust loss and robust CPAC learnability. This is of particular interest, as the evaluability of the robust loss has often implicitly been used in the literature on the theory of robust learning, or potential issues on the evaluability of the robust loss have been circumvented by the use of oracles. We finished by studying the role of the computable robust shattering dimension in robust CPAC learnability, showing that its finiteness is a necessary but not sufficient condition.

Acknowledgments

Pascale Gourdeau has been supported by a Vector Postdoctoral Fellowship and an NSERC Postdoctoral Fellowship. Tosca Lechner has been supported by a Vector Research Grant and a Waterloo Apple PhD Fellowship. Ruth Urner is also an Affiliate Faculty Member at Toronto's Vector Institute, and acknowldeges funding through an NSERC Discovery grant.

References

Nathanael Ackerman, Julian Asilis, Jieqi Di, Cameron Freer, and Jean-Baptiste Tristan. Computable pac learning of continuous features. In *Proceedings of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 1–12, 2022.

Sushant Agarwal, Nivasini Ananthakrishnan, Shai Ben-David, Tosca Lechner, and Ruth Urner. On learnability wih computable learners. In *Algorithmic Learning Theory*, pages 48–60. PMLR, 2020.

Hassan Ashtiani, Vinayak Pathak, and Ruth Urner. Black-box certification and learning under adversarial perturbations. In *International Conference on Machine Learning*, pages 388–398. PMLR, 2020.

Hassan Ashtiani, Vinayak Pathak, and Ruth Urner. Adversarially robust learning with tolerance. In *International Conference on Algorithmic Learning Theory*, pages 115–135. PMLR, 2023.

Idan Attias, Steve Hanneke, and Yishay Mansour. A characterization of semi-supervised adversarially robust pac learnability. *Advances in Neural Information Processing Systems*, 35:23646–23659, 2022a.

Idan Attias, Aryeh Kontorovich, and Yishay Mansour. Improved generalization bounds for adversarially robust learning. *Journal of Machine Learning Research*, 23(175):1–31, 2022b.

Pranjal Awasthi, Natalie Frank, and Mehryar Mohri. Adversarial learning guarantees for linear hypotheses and neural networks. In *International Conference on Machine Learning*, pages 431–441. PMLR, 2020.

Pranjal Awasthi, Anqi Mao, Mehryar Mohri, and Yutao Zhong. Theoretically grounded loss functions and algorithms for adversarial robustness. In *International Conference on Artificial Intelligence and Statistics*, pages 10077–10094. PMLR, 2023.

Maria-Florina Balcan, Avrim Blum, Steve Hanneke, and Dravyansh Sharma. Robustly-reliable learners under poisoning attacks. In *Conference on Learning Theory*, pages 4498–4534. PMLR, 2022.

Maria-Florina F Balcan, Steve Hanneke, Rattana Pukdee, and Dravyansh Sharma. Reliable learning in challenging environments. *Advances in Neural Information Processing Systems*, 36, 2023.

Shai Ben-David, Pavel Hrubes, Shay Moran, Amir Shpilka, and Amir Yehudayoff. Learnability can be undecidable. *Nat. Mach. Intell.*, 1(1):44–48, 2019.

Robi Bhattacharjee and Kamalika Chaudhuri. Consistent non-parametric methods for maximizing robustness. In *Advances in Neural Information Processing Systems*, volume 34, pages 9036–9048, 2021.

Robi Bhattacharjee, Somesh Jha, and Kamalika Chaudhuri. Sample complexity of robust linear classification on separated data. In *International Conference on Machine Learning*, pages 884–893. PMLR, 2021.

Robi Bhattacharjee, Max Hopkins, Akash Kumar, Hantao Yu, and Kamalika Chaudhuri. Robust empirical risk minimization with tolerance. In *International Conference on Algorithmic Learning Theory*, pages 182–203. PMLR, 2023.

Sebastien Bubeck, Yin Tat Lee, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 831–840. PMLR, 2019.

Sadia Chowdhury and Ruth Urner. Robustness should not be at odds with accuracy. In *3rd Symposium on Foundations of Responsible Computing (FORC 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

Daniel Cullina, Arjun Nitin Bhagoji, and Prateek Mittal. PAC-learning in the presence of evasion adversaries. *Advances in Neural Information Processing Systems*, 2018.

Akshay Degwekar, Preetum Nakkiran, and Vinod Vaikuntanathan. Computational limitations in robust classification and win-win results. In *Conference on Learning Theory, COLT*, volume 99, pages 994–1028. PMLR, 2019.

Valentino Delle Rose, Alexander Kozachinskiy, Cristóbal Rojas, and Tomasz Steifer. Find a witness or shatter: the landscape of computable pac learning. In *The Thirty Sixth Annual Conference on Learning Theory*, pages 511–524. PMLR, 2023.

Ilias Diakonikolas, Daniel M Kane, and Pasin Manurangsi. The complexity of adversarially robust proper learning of halfspaces with agnostic noise. *Advances in Neural Information Processing Systems*, 33:20449–20461, 2020.

Dimitrios Diochnos, Saeed Mahloujifar, and Mohammad Mahmoody. Adversarial risk and robustness: General definitions and implications for the uniform distribution. In *Advances in Neural Information Processing Systems*, 2018.

Dimitrios I. Diochnos, Saeed Mahloujifar, and Mohammad Mahmoody. Lower bounds for adversarially robust PAC learning under evasion and hybrid attacks. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 717–722, 2020.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In *Advances in Neural Information Processing Systems*, pages 1632–1640, 2016.

Alhussein Fawzi, Hamza Fawzi, and Omar Fawzi. Adversarial vulnerability for any classifier. *Advances in neural information processing systems*, 31, 2018a.

Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Analysis of classifiers robustness to adversarial perturbations. *Machine Learning*, 107(3):481–508, 2018b.

Sanjam Garg, Somesh Jha, Saeed Mahloujifar, and Mahmoody Mohammad. Adversarially robust learning could leverage computational hardness. In *Algorithmic Learning Theory*, pages 364–385. PMLR, 2020.

Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S. Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian J. Goodfellow. Adversarial spheres. In *6th International Conference on Learning Representations, ICLR Workshop Track Proceedings*. OpenReview.net, 2018.

Pascale Gourdeau, Varun Kanade, Marta Kwiatkowska, and James Worrell. On the hardness of robust classification. *The Journal of Machine Learning Research*, 22(1):12521–12549, 2021.

Pascale Gourdeau, Varun Kanade, Marta Kwiatkowska, and James Worrell. When are local queries useful for robust learning? *Advances in Neural Information Processing Systems*, 35:33920–33933, 2022a.

Pascale Gourdeau, Varun Kanade, Marta Kwiatkowska, and James Worrell. Sample complexity bounds for robustly learning decision lists against evasion attacks. In *International Joint Conference in Artificial Intelligence*, 2022b.

Niki Hasrati and Shai Ben-David. On computable online learning. In *International Conference on Algorithmic Learning Theory*, pages 707–725. PMLR, 2023.

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

Justin Khim, Varun Jog, and Po-Ling Loh. Adversarial influence maximization. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 1–5. IEEE, 2019.

Tosca Lechner, Vinayak Pathak, and Ruth Urner. Adversarially robust learning with uncertain perturbation sets. In *Advances in Neural Information Processing Systems*, 2023.

Saeed Mahloujifar and Mohammad Mahmoody. Can adversarially robust learning leverage computational hardness? In *Algorithmic Learning Theory*, pages 581–609. PMLR, 2019.

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.

Elliott Mendelson. *Introduction to mathematical logic (5. ed.)*. Chapman and Hall, 2009.

Omar Montasser, Steve Hanneke, and Nathan Srebro. VC classes are adversarially robustly learnable, but only improperly. In *Conference on Learning Theory*, pages 2512–2530. PMLR, 2019.

Omar Montasser, Steve Hanneke, and Nathan Srebro. Adversarially robust learning with unknown perturbation sets. In *Conference on Learning Theory*, pages 3452–3482. PMLR, 2021.

Omar Montasser, Steve Hanneke, and Nati Srebro. Adversarially robust learning: A generic minimax optimal learner and characterization. *Advances in Neural Information Processing Systems*, 35:37458–37470, 2022.

Vinod Raman, Unique Subedi, and Ambuj Tewari. On proper learnability between average-and worst-case robustness. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Ali Shafahi, W Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable? In *7th International Conference on Learning Representations (ICLR 2019)*, 2019.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014.

Han Shao, Omar Montasser, and Avrim Blum. A theory of pac learnability under transformation invariances. *Advances in Neural Information Processing Systems*, 35:13989–14001, 2022.

Tom F Sterkenburg. On characterizations of learnability with computable learners. In *Conference on Learning Theory*, pages 3365–3379. PMLR, 2022.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

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

Vladimir Vapnik and Alexey Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Theory of Probability and Its Applications*. 1971.

Paul Viallard, Eric Guillaume VIDOT, Amaury Habrard, and Emilie Morvant. A pac-bayes analysis of adversarial robustness. *Advances in Neural Information Processing Systems*, 34, 2021.

Dong Yin, Ramchandran Kannan, and Peter Bartlett. Rademacher complexity for adversarially robust generalization. In *International conference on machine learning*, pages 7085–7094. PMLR, 2019.

Appendix A. Proofs and Remarks from Section 3

A.1. Proof of Theorem 14

Proof We will use the computable online algorithm $\mathcal{A}$ as a black-box. Let $D := \mathsf{VC}(\mathcal{H}) + \mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}})$ and note that online learnability implies that $\mathsf{VC}(\mathcal{H}) \leq \mathsf{Lit}(\mathcal{H}) < \infty$, and so $D < \infty$.

Let $S = \{(x_i, y_i)\}_{i=1}^m$ be drawn i.i.d. from an arbitrary robustly-realizable distribution D on $\mathcal{X} \times \{0,1\}$. We outline below a robust CPAC algorithm, which is essentially the CycleRobust algorithm of (Montasser et al., 2021) with the online algorithm $\mathcal{A}$ being used, implements robust ERM on S:

Algorithm 1 $\mathcal{U}$-robust CPAC algorithm

Input: $S = \{(x_i, y_i)\}_{i=1}^m$, computable online algorithm $\mathcal{A}$ for $\mathcal{H}$
Output: $h \in \mathcal{H}$ such that h achieves zero robust loss on S

```
K ← 0, h ← A(∅)
while K ≤ Lit(H) do
    for i = 1, ..., m do
        if (z, y) ← PAO(h, x_i, y_i) then        ▷ robust loss is 1, get counterexample
            K++                                   ▷ Update mistake count
            h ← A((z, y))                         ▷ Update hypothesis
            Break                                 ▷ Exit for-loop
        else if i = m then
            return h                              ▷ h is robustly consistent on S
                                                  ▷ No counterexample ⇒ go to next index
    end
end
return h
```

Note that, because of robust realizability, we are guaranteed that $\mathcal{A}$ makes at most $\mathsf{Lit}(\mathcal{H})$ mistakes, after which h will be $\mathcal{U}$-robustly consistent on S. The program above makes a finite number of calls to $\mathcal{A}$, and thus is computable by the fact that $\mathcal{A}$ is itself a computatble online learner. By Fact 11, we are done. ■

A.2. Decidable Representation of Perturbation Types

Example 1 *We exhibit $\mathcal{H}$, $\mathcal{U}$ such that $\mathcal{H}$ is CPAC learnable and $\mathcal{U}$-robustly PAC learnable but $\mathcal{H}$ is not (improperly) $\mathcal{U}$-robustly CPAC learnable. In this example, $\mathcal{U}$ is not DR. The hypothesis class is as follows:*

$$\mathcal{H} := \{h_{a,b}(i) = \mathbf{1}[i \in \{a,b\}]\} \quad .$$

Clearly $\mathsf{VC}(\mathcal{H}) = 2$ and ERM is computably implementable, so by Fact 9, $\mathcal{H}$ is CPAC learnable. Moreover, $\mathcal{H}$ having finite VC dimension implies that it is $\mathcal{U}$-robustly PAC learnable (Montasser et al., 2019).

We now define the perturbation type:

$$\mathcal{U}(2i) = \{2i\} \cup \{2i+1 \mid T_i \text{ halts on the empty word}\} \quad ,$$

$$\mathcal{U}(2i+1) = \{2i+1\} \cup \{2i \mid T_i \text{ halts on the empty word}\} \quad .$$

We now define distributions D_i:

$$D_i(2i,1) = D_i(2i+1,0) = 1/2 \ .$$

Clearly, the labelling $l(2_i) = 1$ and $l(2i+1) = 0$ is optimal (and gives zero robust risk) if and only if T_i does not halt on the empty word. Otherwise, if T_i halts, l incurs a robust risk of 1, and any function satisfying $h(2i) = h(2i+1)$ is a robust risk minimizer incurring a robust risk of $1/2$.

If $\mathcal{H}$ were (improperly) $\mathcal{U}$-robustly CPAC learnable, there would be a robust learning algorithm $\mathcal{A}$ for $\mathcal{H}$ with sample complexity $m(\epsilon,\delta)$. Letting $M \in \mathbb{N}$ being an upper bound on $m(1/3,1/2)$, then with probability at least $2/3$, $\mathcal{A}$ returns a hypothesis with robust risk within $1/3 < 1/2$ of the robust risk minimizer in $\mathcal{H}$ when run on a sample of size M. Then, running $\mathcal{A}$ on all 2^M sequences of length M with elements in $\{2i, 2i+1\}$ (which are equally probable), we end up with optimal hypotheses on two thirds of the runs. Checking, for each hypothesis h, whether $h(2i) = h(2i+1)$ and taking the majority vote over all sequences, we get a decider for whether T_i halts on the empty word. And so $\mathcal{H}$ is not (improperly) $\mathcal{U}$-robustly CPAC learnable. Note here that if we had a way to know whether $k \in \mathcal{U}(2i)$, i.e., if $\mathcal{U}$ was DR, we would immediately be able to implement robust ERM for samples from the distributions defined above. Not requiring perturbation regions to be DR thus makes deriving impossibility results for robust CPAC learning relatively trivial.

Appendix B. Proofs and Algorithms from Section 4.1

B.1. Proof of Theorem 15

Proof Fix a proof system for first-order logic over a rich enough vocabulary that is sound and complete. Let $\{\varphi_i\}_{i\in\mathbb{N}}$ and $\{\pi_j\}_{j\in\mathbb{N}}$ be enumerations of all theorems and proofs, respectively. We define the following hypothesis class on $\mathbb{N}$:

$$\mathcal{H} = \{h_a \ : \ \forall i \in \mathbb{N},\ h_a(2i) = \mathbf{1}[i \leq a] \wedge h_a(2i+1) = 1\}_{a\in\mathbb{N}\cup\{\infty\}} \ .$$

Thus, $\mathcal{H}$ is the the concept class where each function defines a threshold on even integers, and is the constant function 1 on odd integers.

First, to show (i), note that $\mathsf{VC}(\mathcal{H}) = 1$, as for any set X of two even integers, the labelling $h(x) = 0$, $h(z) = 1$ cannot be realized on two even integers $x < z$, and odd integers can only have labelling 1, and thus cannot be in a shattered set. As ERM is easily implementable for $\mathcal{H}$, we have that $\mathcal{H}$ is properly CPAC learnable by Fact 9.

Now, we define the perturbation sets as follows. For any $i \in \mathbb{N}$ we set:

$$\begin{aligned}
&\mathcal{U}(6i) = \{6i\} \cup \{2j+1\}_{j\in\mathbb{N}} \ , \\
&\mathcal{U}(6i+2) = \{6i+2\} \ , \\
&\mathcal{U}(6i+4) = \{6i+2, 6i+4\} \cup \{2j+1 \mid \pi_j \text{ is a proof of theorem } \varphi_i\}_{j\in\mathbb{N}} \ , \\
&\mathcal{U}(2i+1) = \{2i+1\} \ .
\end{aligned}$$

Note that the question of whether $\mathcal{U}(6i) \cap \mathcal{U}(6i+4)$ is empty is equivalent to whether there exists a proof for theorem φ_i, and thus is undecidable.

It is straightforward to show (ii), as for every integer $i \in \mathbb{N}$, we can represent $\mathcal{U}(6i+4)$ with the program P_{6i+4}, which we have included below.

Algorithm 2 Program P_{6i+4} to decide whether $x \in \mathcal{U}(6i+4)$

Input: $x, i \in \mathbb{N}$
Output: Whether $x \in \mathcal{U}(6i+1)$
if $x \in \{6i+2, 6i+4\}$ **then**
  output yes
else if x *is even* **then**
  output no
else
  $j \leftarrow \frac{x-1}{2}$
  if π_j *proves* φ_i **then**
    output yes
  else
    output no
  end
end

Now, for all other k, deciding whether $x \in \mathcal{U}(k)$ is also easily representable as a program P_k (we can just assume $\mathcal{U}(x) = \{x\}$ if there does not exists $i \in \mathbb{N}$ such that $k \in \{6i, 6i+2, 6i+4\}$, in order for $\mathcal{U}$ to be well-defined).

To show (iii), we follow the notation of (Ashtiani et al., 2020) and show that $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}}) = 1$, which implies proper robust learnability by Fact 11 (also Fact 10). Recall that the class $\mathcal{H}^{\mathcal{U}}_{\text{mar}}$ consists of the sets $\text{mar}^{\mathcal{U}}_h := \{x \in \mathcal{X} \mid \exists z \in \mathcal{U}(x)\ .\ h(x) \neq h(z)\}$, defined for each $h \in \mathcal{H}$. Now, fix $a \in \mathbb{N}$, which induces the function h_a and the set $\text{mar}^{\mathcal{U}}_{h_a}$, which we will denote by mar_a for simplicity. Because of the condition $h(x) \neq h(z)$ in the definition of mar_a and the property that $\{k\} \subseteq \mathcal{U}(k)$, the only integers $k \in \mathbb{N}$ that can belong to mar_a are those for which $\{k\}$ is a proper subset of $\mathcal{U}(k)$, i.e., those for which there exists $i \in \mathbb{N}$ such that $k = 6i$ or $k = 6i+4$. We can see that

$$\begin{aligned}\text{mar}_a = &\{6i \mid 6i > 2a\}_{i\in\mathbb{N}} \cup \{6i+4 \mid \varphi_i \text{ is a tautology} \wedge 6i+4 > 2a\}_{i\in\mathbb{N}} \\ &\cup \{6i+4 \mid \varphi_i \text{ not a tautology} \wedge 6i+2 = 2a\}_{i\in\mathbb{N}} \quad ,\end{aligned}$$

as these are precisely the instances which incur a robust loss of 1 with respect to h_a:

- If $k = 6i$ then since $\mathcal{U}(6i) = \{6i\} \cup \{2j+1\}_{j\in\mathbb{N}}$, we have that $k \in \text{mar}_a$ iff $h_a(k) = 0$,
- If $k = 6i+4$ and φ_i is a tautology, then since

  $$\mathcal{U}(6i+4) = \{6i+2, 6i+4\} \cup \{2j+1 \mid \pi_j \text{ is a proof of theorem } \varphi_i\}_{j\in\mathbb{N}} \quad ,$$

  we have that $k \in \text{mar}_a$ iff $h_a(k) = 0$ (we cannot have $h_a(6i+4) = 1$ while $h_a(6i+2) = 0$),
- Else, if $k = 6i+4$ and φ_i is not a tautology, $\mathcal{U}(6i+4) = \{6i+2, 6i+4\}$, and $k \in \text{mar}_a$ iff $h(6i+2) \neq h(6i+4)$ iff $6i+2 = 2a$.

Now, it suffices to consider the subsets of $X = \{6i, 6i+4\}_{i\in\mathbb{N}}$ to find the largest shattered set, as these are the only instances that can belong to some set mar_a. We partition X into two parts P_1 and P_2, where $P_1 = \{6i\}_{i\in\mathbb{N}} \cup \{6i+4 \mid \varphi_i \text{ is a tautology}\}_{i\in\mathbb{N}}$ and $P_2 = \{6i+4 \mid \varphi_i \text{ is not a tautology}\}_{i\in\mathbb{N}}$. On the one hand, $\{\text{mar}_a\}_{a\in\mathbb{N}}$ restricted to P_1 is simply thresholds on P_1, and so at most one element

from P_1 can be in a shattered set. On the other hand, $\{\text{mar}_a\}_{a\in\mathbb{N}}$ restricted to P_2 is the class of singletons, so again at most one element from P_2 can be in a shattered set. Thus, we have established that $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}}) \leq 2$. To see that $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}}) = 1$ consider arbitrary $k_1 \in P_1$ and $k_2 = 6i + 4 \in P_2$ for some $i, i' \in \mathbb{N}$. It is easy to see that $\{k_1, k_2\}$ or $\{k_2\}$ won't be in the projection of $\mathcal{H}^{\mathcal{U}}_{\text{mar}}$ onto $\{k_1, k_2\}$. Indeed, k_2 being in a set in the projection implies that there is a unique mar_a such that $k_2 \in \text{mar}_a$, and k_1 is either in mar_a or not.

Finally, to show that $\mathcal{H}$ not (improperly) $\mathcal{U}$-robustly CPAC learnable in the agnostic setting we define a set of distributions for which no computable learner for $\mathcal{H}$ succeeds. For each $i \in \mathbb{N}$ we define a distribution D_i on $\mathbb{N} \times \{0, 1\}$ as follows:

$$\begin{aligned} D_i((6i, 1)) &= 1/2 \ , \\ D_i((6i + 2, 1)) &= 1/6 \ , \\ D_i((6i + 4, 0)) &= 1/3 \ . \end{aligned}$$

To determine optimal predictors on these distributions, we distinguish two cases:

1. $\mathcal{U}(6i) \cap \mathcal{U}(6i + 4) = \emptyset$: this implies that $\mathcal{U}(6i + 4) = \{6i + 2, 6i + 4\}$. Since $\mathcal{U}(6i + 2) \cap \mathcal{U}(6i + 4) \neq \emptyset$ and these points have different labels under distribution D_i, no predictor can achieve robust loss 0 on both points simultaneously. Thus no predictor can achieve robust risk less than $1/6$. Note that $h_{3i,\infty}$ has robust risk $1/6$ on D_i, and is therefore an optimal predictor on D_i. Furthermore, note that any optimal predictor h on D_i must satisfy $h(6i + 2) = h(6i + 4) = 0$.

2. $\mathcal{U}(6i) \cap \mathcal{U}(6i+4) \neq \emptyset$: in this case, since $6i$ and $6i+4$ have different labels under distribution D_i and intersecting perturbation sets, no predictor can achieve robust loss 0 on both points and thus no predictor can achieve robust loss less than $1/3$ on D_i. Note that $h_{\infty,\infty}$ (the constant function 1) has robust risk $1/3$ on D_i and is thus a robust risk minimizer in this case. Furthermore, any optimal predictor h on D_i must satisfy $h(6i) = h(6i + 2) = 1$, to not incur additional robust loss.

We thus established that we have $h(6i+2) = 1$ for an optimal predictor on D_i (and any predictor that has robust loss less than $1/6$ more than the optimal) if and only if $\mathcal{U}(6i) \cap \mathcal{U}(6i + 4) \neq \emptyset$ and thus if and only if φ_i has a proof.

We now argue that for any $i \in \mathbb{N}$, a $\mathcal{U}$-robust CPAC learner $\mathcal{A}$ for $\mathcal{H}$, proper or not, can be used to determine whether $\mathcal{U}(6i) \cap \mathcal{U}(6i + 4) = \emptyset$, and thus can be used to decide if φ_i is a tautology or not: we let $\mathcal{A}$ be a robust computable learner for $\mathcal{H}$ and let $M \in \mathbb{N}$ be an upper bound for learning $\mathcal{H}$ with $\epsilon = 1/7 < 1/6$ and $\delta = 1/3$. Running $\mathcal{A}$ on all sequences of length M over the set $\{(6i, 1), (6i + 2, 1), (6i + 4, 0)\}$ and evaluating the resulting predictors has to yield a $1/7$-close to optimal hypothesis on two thirds of the sample. Thus taking the majority vote over the predictions on $(6i + 2)$ is a decider for whether φ_i is a tautology or not. ■

B.2. Algorithm for the Decidable Representation in Theorem 16

Algorithm 3 Program to decide whether $x_2 \in \mathcal{U}(x_1)$

Input: $x_1, x_2 \in \mathbb{N}$
Output: Whether $x_2 \in \mathcal{U}(x_1)$
if x_1 *is odd* **then**
  if $x_2 = x_1$ **then**
    output $x_2 \in \mathcal{U}(x_1)$
  else
    output $x_2 \notin \mathcal{U}(x_1)$
  end
else ▷ x_1 is even
  if x_2 *is odd* **then**
    run $T_{x_1}(x_1)$ for at most $\frac{x_2-1}{2}$ steps
     if $T_{x_1}(x_1)$ *halted* **then**
      output $x_2 \in \mathcal{U}(x_1)$
    else ▷ $T_{x_1}(x_1)$ is still running after $\frac{x_2-1}{2}$ steps
      output $x_2 \notin \mathcal{U}(x_1)$
    end
  else ▷ x_2 is even
    if $x_2 = x_1$ **then**
      output $x_2 \in \mathcal{U}(x_1)$
    else
      output $x_2 \notin \mathcal{U}(x_1)$
    end
  end
end

B.3. Useful Lemmas for Theorem 16

Lemma 29 *There is no program that solves the problem* TwoHalt*, that is, no program with the following behavior: given a pair of indices of Turing Machines* (i, j) *as input:*

- *if* $T_i(i)$ *halts and* $T_j(j)$ *loops, output* 1
- *if* $T_i(i)$ *loops and* $T_j(j)$ *halts, output* 2
- *if both* $T_i(i)$ *and* $T_j(j)$ *loop, halt and output* 1 *or output* 2
- *if both* $T_i(i)$ *and* $T_j(j)$ *halt, output* 1 *or output* 2 *or loop*

Before proving Lemma 29, let us state and prove the following result, which will be used in the proof of Lemma 29.

Lemma 30 *For any two 2-place total computable functions* f_1*,* f_2 *there are indices* c_1, c_2*, such that* $T_{c_1} \equiv T_{f_1(c_1,c_2)}$ *and* $T_{c_2} \equiv T_{f_2(c_1,c_2)}$

Proof We start by defining the following Turing Machines:

$$T_a(x_1, x_2, y_1, y_2) = \begin{cases} T_{T_{x_1}(x_1,x_2)}(y_1, y_2) & \text{if } T_{x_1}(x_1, x_2) \text{ halts} \\ \uparrow & \text{otherwise.} \end{cases}$$

and

$$T_b(x_1, x_2, y_1, y_2) = \begin{cases} T_{T_{x_2}(x_1,x_2)}(y_1, y_2) & \text{if } T_{x_2}(x_1, x_2) \text{ halts} \\ \uparrow & \text{otherwise.} \end{cases}$$

By the S_m^n-Theorem, there exist total computable functions h_1 and h_2, such that $T_a(x_1, x_2, y_1, y_2) = T_{h_1(x_1,x_2)}(y_1, y_2)$ and $T_b(x_1, x_2, y_1, y_2) = T_{h_2(x_1,x_2)}(y_1, y_2)$ for all x_1, x_2, y_1, y_2. Now let e_1 be the index of the Turing machine that computes the function

$$g_1(x_1, x_2) = f_1(h_1(x_1, x_2), h_2(x_1, x_2))$$

and e_2 be the index of the Turing machine that computes the function

$$g_2(x_1, x_2) = f_2(h_1(x_1, x_2), h_2(x_1, x_2)).$$

Since g_1 is a total computable function, we have $T_{h_1(e_1,e_2)} \equiv T_{T_{e_1}(e_1,e_2)}$. By definition, $T_{e_1}(e_1, e_2)$ computes $f_1(h_1(e_1, e_2), h_2(e_1, e_2))$. Thus $T_{h_1(e_1,e_2)} \equiv T_{f_1(h_1(e_1,e_2),h_2(e_1,e_2))}$. Furthermore, by the same argument, we have that $T_{h_2(e_1,e_2)} \equiv T_{T_{e_2}(e_1,e_2)} \equiv T_{f_2(h_1(e_1,e_2),h_2(e_1,e_2))}$. Thus if we choose $c_1 = h_1(e_1, e_2)$ and $c_2 = h_2(e_1, e_2)$, we get $T_{c_1} \equiv T_{f_1(c_1,c_2)}$ and $T_{c_2} \equiv T_{f_2(c_1,c_2)}$, as required. ■

We are now ready to prove Lemma 29, which completes the result of this section.

Proof [of Lemma 29] By way of contradiction, let's assume that there existed a program/Turing Machine T^{TwoHalt} that solved the TwoHalt problem. Recall that we had fixed an enumeration $(T_i)_{i\in\mathbb{N}}$ of all Turing Machines (or programs), and that we use the notaion $T(i)\downarrow$ to indicate that a Turing Machine T halts on input i and $T(i)\uparrow$ to indicate that T loops on input i. We can now define two natural numbers l_1 and l_2 as being the indices of Turing Machines T_{l_1} and T_{l_2} which have the following behavior:

$$T_{l_1}(i, j, z) = \begin{cases} 1 & \text{if } z = 0 \\ \uparrow & \text{if } z = i > 0 \text{ and } T^{\mathsf{TwoHalt}}(i, j) = 1 \\ 1 & \text{if } z = j > 0 \text{ and } T^{\mathsf{TwoHalt}}(i, j) = 1 \\ \uparrow & \text{if } z = j > 0 \text{ and } T^{\mathsf{TwoHalt}}(i, j) = 2 \\ 2 & \text{if } z = i > 0 \text{ and } T^{\mathsf{TwoHalt}}(i, j) = 2 \\ \uparrow & \text{otherwise} \end{cases}$$

$$T_{l_2}(i, j, z) = \begin{cases} 2 & \text{if } z = 0 \\ \uparrow & \text{if } z = i > 0 \text{ and } T^{\mathsf{TwoHalt}}(i, j) = 1 \\ 1 & \text{if } z = j > 0 \text{ and } T^{\mathsf{TwoHalt}}(i, j) = 1 \\ \uparrow & \text{if } z = j > 0 \text{ and } T^{\mathsf{TwoHalt}}(i, j) = 2 \\ 2 & \text{if } z = i > 0 \text{ and } T^{\mathsf{TwoHalt}}(i, j) = 2 \\ \uparrow & \text{otherwise} \end{cases}$$

Now by S^n_m-Theorem, there are total computable functions f_1 and f_2 such that

$$T_{f_1(i,j)}(z) = \begin{cases} 1 & \text{if } z = 0 \\ \uparrow & \text{if } z = i > 0 \text{ and } T^{\mathsf{TwoHalt}}(i,j) = 1 \\ 1 & \text{if } z = j > 0 \text{ and } T^{\mathsf{TwoHalt}}(i,j) = 1 \\ \uparrow & \text{if } z = j > 0 \text{ and } T^{\mathsf{TwoHalt}}(i,j) = 2 \\ 2 & \text{if } z = i > 0 \text{ and } T^{\mathsf{TwoHalt}}(i,j) = 2 \\ \uparrow & \text{otherwise} \end{cases}$$

and

$$T_{f_2(i,j)}(z) = \begin{cases} 2 & \text{if } z = 0 \\ \uparrow & \text{if } z = i \text{ and } T^{\mathsf{TwoHalt}}(i,j) = 1 \\ 1 & \text{if } z = j \text{ and } T^{\mathsf{TwoHalt}}(i,j) = 1 \\ \uparrow & \text{if } z = j \text{ and } T^{\mathsf{TwoHalt}}(i,j) = 2 \\ 2 & \text{if } z = i \text{ and } T^{\mathsf{TwoHalt}}(i,j) = 2 \\ \uparrow & \text{otherwise} \end{cases}$$

Now, by Lemma 30, there are $c_1, c_2 > 0$, such that $T_{f_1(c_1,c_2)} \equiv T_{c_1}$ and $T_{f_2(c_1,c_2)} \equiv T_{c_2}$. Recall that we use $T \equiv S$ to indicate that two Turing Machines have the same behavior as functions.

Now let us look at $T^{\mathsf{TwoHalt}}(c_1, c_2)$. There are three options:

- $T^{\mathsf{TwoHalt}}(c_1, c_2) = 1$. Then by definition $T_{c_1}(c_1) = T_{f_1(c_1,c_2)}(c_1) \uparrow$ and $T_{c_2}(c_2) = T_{f_2(c_1,c_2)}(c_2) \downarrow$. Thus T^{TwoHalt} would output 1, the wrong answer in a case where 2, the correct answer, was required.
- $T^{\mathsf{TwoHalt}}(c_1, c_2) = 2$. Then by definition $T_{c_1}(c_1) = T_{f_1(c_1,c_2)}(c_1) \downarrow$ and $T_{c_2}(c_2) = T_{f_2(c_1,c_2)}(c_2) \uparrow$. Thus T^{TwoHalt} would output 2, the wrong answer in a case where 1, the correct answer, was required.
- $T^{\mathsf{TwoHalt}}(c_1, c_2) \uparrow$. Note that, by definition, the Turing Machines $T_{f_1(c_1,c_2)}$ and $T_{f_2(c_1,c_2)}$ only halt and produce an output if either their input was 0 or the call to T^{TwoHalt} inside their definition halts and produces an output. Thus, for this case, we get $T_{c_1}(c_1) = T_{f_1(c_1,c_2)}(c_1) \uparrow$ and $T_{c_2}(c_2) = T_{f_2(c_1,c_2)}(c_2) \uparrow$. However, if both $T_{c_1}(c_1)$ and $T_{c_2}(c_2)$ don't halt, T^{TwoHalt} was supposed to halt. Thus T^{TwoHalt} also has the wrong behaviour in this case.

In summary, the assumption that T^{TwoHalt} existed lead to a contradiction in all cases and thus the problem TwoHalt does not admit a solution. ■

Appendix C. Proofs from Section 4.2

C.1. Proof of Theorem 17

Proof Let the instance space be $\mathcal{X} = (\mathbb{N} \times (\{0\} \cup \mathbb{N})$. Let $(\varphi_i)_{i\in\mathbb{N}}$ be an enumeration of formulas and $(\pi_j)_{j\in\mathbb{N}}$ be an enumeration of proofs. Let $(i, 0)$, represent the i-th formula and for any i, j let

(i, j) represent the j-th proof (independently of i). Let $\mathcal{H} = \{h_{a,b}\}_{a,b \in \mathbb{N} \cup \{0\}}$, where

$$h_{a,b}(i,j) = \begin{cases} 1 & \text{if } i < a \text{ or } (i = a \text{ and } j \leq b) \\ 0 & \text{otherwise} \end{cases} .$$

It is clear that $\mathsf{VC}(\mathcal{H}) = 1$ and that ERM is computably implementable, so $\mathcal{H}$ is properly CPAC learnable.

We define the perturbation region as follows:

$$\mathcal{U}(i,0) = \{(i,0)\} \cup \{(i,j) : \pi_j \text{ proves } \varphi_i\} \ ,$$
$$\mathcal{U}(i,j) = \{(i,0),(i,j)\} \ ,$$

which is clearly DR, as to check whether $(i,j) \in \mathcal{U}((i,0))$ we can check whether π_j proves φ_i.

It is immediate to see that $\ell^{\mathcal{U}}$ is not computable for all hypotheses in $\mathcal{H}$. Indeed, for every $i \in \mathbb{N}$, deciding whether for the formula φ_i is a tautology reduces to deciding whether

$$\ell^{\mathcal{U}}(h_{i,0},(i,0),1) = 0 \ .$$

Now, it remains to show that $\mathcal{H}$ is properly $\mathcal{U}$-robustly CPAC learnable. To this end, we will first show that $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}})$ is finite, and second, show that robust ERM is computable in the agnostic setting. By Fact 11, we will have robust CPAC learnability.

To show $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}}) < \infty$, we consider the sets $\text{mar}^{\mathcal{U}}_{h_{a,b}}$, which we will denote by $\text{mar}_{a,b}$ for brevity. First note that either $a = \infty$ or $b = \infty$ implies that $\text{mar}_{a,b} = \emptyset$. Let $a, b \in \mathbb{N}$ and observe that only instances of the form (a,j) can belong to $\text{mar}_{a,b}$, and thus a shattered set cannot contain both (a,j) and (a',j') where $a \neq a'$. We distinguish two cases:

- φ_a is not a tautology: then $\text{mar}_{a,b} = \{(a,0)\}$,
- φ_a is a tautology: then $\text{mar}_{a,b} = \{(a,0)\} \cup \{(a,j) \mid j > b \wedge \pi_j \text{ proves } \varphi_a\}$.

From this, it is clear that $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}}) = 1$.

Now, to show that robust ERM is computable, let $S \subseteq \mathcal{X} \times \{0,1\}$ be an arbitrary labelled sample of size m. For a fixed $i \in \mathbb{N}$ and $y \in \{0,1\}$, consider the subset S^y_i of S, consisting of "i-instances" with label y, i.e., $S^y_i := \{((i,j),y) \mid ((i,j),y) \in S\}$, and let $S_i = S^0_i \cup S^1_i$. We can see that $h_{i,\infty}$ incurs robust loss 1 on all the instances in S^0_i and robust loss 0 on all the instances in S^1_i. Furthermore, while every $h_{i,b}$ incurs robust loss 1 on all S^0_i, it might also incur loss 1 on some instances S^1_i. Finally, for $b \in \mathbb{N}$, on every instance in $S \setminus S_i$, the hypotheses $h_{i,b}$ and $h_{i,\infty}$ incur the same robust loss. Thus $h_{i,\infty}$ always "dominates" $h_{i,b}$ in the sense that $\mathsf{R}_{\mathcal{U}}(h_{i,\infty};S) \leq \mathsf{R}_{\mathcal{U}}(h_{i,b};S)$. We furthermore note that for every i, the loss of the function $h_{i,\infty}$ can be evaluated on every point. Thus, performing RERM boils down to comparing all candidate hypotheses of the form $h_{i,\infty}$, which can be done computably. ■

Remark 31 *In the above proof, let $\mathcal{H}' \subset \mathcal{H}$ be the set of all hypotheses in $\mathcal{H}$ with $b = \infty$, i.e., $\mathcal{H}' = \{h_{a,b} \in \mathcal{H} \mid b = \infty\}$. Then it is clear that, in the robust case, because of the definition of $\mathcal{U}$, we can get away with only outputting hypotheses from $\mathcal{H}'$, no matter what the underlying distribution is, and that for all hypotheses in $\mathcal{H}'$ the robust loss is computable. In general, for some hypothesis class $\mathcal{H}$, let $\mathcal{H}'$ be a class, such that (i) for every $h \in \mathcal{H}$, there is $h' \in \mathcal{H}'$, such that for all $(x,y) \in \mathcal{X} \times \mathcal{Y}$: $\ell^{\mathcal{U}}(h',x,y) \leq \ell^{\mathcal{U}}(h,x,y)$, (ii) the robust loss of every $h' \in \mathcal{H}'$ is computably evaluable. Then if $\mathcal{H}'$ is realizable/agnostic robustly CPAC learnable, so is $\mathcal{H}$. Furthermore, if $\mathcal{H}' \subset \mathcal{H}$ and $\mathcal{H}'$ is* proper *realizable/agnostic robustly learnable, so is $\mathcal{H}$.*

C.2. Proof of Theorem 18

Proof Let the instance space be $\mathcal{X} = \mathbb{N} \times \mathbb{N}$. Let the hypothesis class be $\mathcal{H} = \{h_{i,j} : i, j \in \mathbb{N}\}$, where

$$h_{i,j}(x) = \begin{cases} 1 & \text{if } x = (i,k) \text{ with } k \leq j \\ 0 & \text{otherwise} \end{cases} ,$$

i.e., $\mathcal{H}$ is the class which defines initial segments on the sets $\{(i,j)\}_{j\in\mathbb{N}}$ for all $i \in \mathbb{N}$.

We define the perturbation types as follows.

$$\mathcal{U}((i,0)) = \{(i,0)\} \cup \{(i,k) : T_i \text{ does not halt after } k \text{ steps on the empty input }\} ,$$

$$\mathcal{U}((i,j)) = \begin{cases} \{(i,j),(i,0)\} & T_i \text{ does not halt after } j \text{ steps on the empty input} \\ \{(i,j)\} & \text{otherwise.} \end{cases} ,$$

We now prove properties (i)-(iii) from the theorem statement.

To prove (i), i.e., that $\mathcal{H}$ is properly CPAC learnable, it suffices to observe that $\mathsf{VC}(\mathcal{H}) = 1$ and that ERM is easily implementable (in both the agnostic and the realizable cases). In order to verify $\mathsf{VC}(\mathcal{H}) = 1$, let us take to arbitrary distinct points (i_1, j_1) and (i_2, j_2). If $i_1 \neq i_2$, then the labelling $((i_1, j_1), 1), ((i_2, j_2), 1)$ cannot be achieved by $\mathcal{H}$. If $i_1 = i_2$, then $j_1 \neq j_2$. Let us assume without loss of generality that $j_1 < j_2$. Then the labelling $((i_1, j_1), 0), ((i_2, j_2), 1)$ cannot be achieved. For computably realizing ERM, we note that for any sample S it is sufficient to do search over the finite class $\mathcal{H}_S = \{h_{i,j} : ((i,j),1) \in S\} \cup \{h_0^S\}$, where $h_0^S = h_{i,0}$, with i being the smallest index for which there is no $((i,j),1) \in S$. Searching over these finitely many candidate hypotheses can be done computably.

To prove (ii), i.e., that $\mathcal{H}$ is properly $\mathcal{U}$-robustly learnable, we show that the VC dimension $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}})$ of the margin class is finite. To this end, we identify the sets $\text{mar}^{\mathcal{U}}_{\mathcal{H}}$ and distinguish two cases:

- T_i does not halt after j steps on the empty input: then

$$\text{mar}^{\mathcal{U}}_{h_{i,j}} = \{(i,0)\} \cup \{(i,k) : k > j \text{ and } T_i \text{ does not halt after } k \text{ steps}\} ,$$

- T_i halts on the empty input after j steps: then $\text{mar}^{\mathcal{U}}_{h_{i,j}} = \emptyset$.

We now argue that $\mathsf{VC}(\mathcal{H}^{\mathcal{U}}_{\text{mar}}) = 1$. To this end, let $x_1 = (i_1, j_1)$ and $x_2 = (i_2, j_2)$ be two distinct domain points, and note that, to be shattered, we need $i_1 = i_2$, as otherwise there is no $\text{mar}^{\mathcal{U}}_h \in \mathcal{H}^{\mathcal{U}}_{\text{mar}}$ with $x_1, x_2 \in \text{mar}^{\mathcal{U}}_h$. Now, without loss of generality, let $j_1 < j_2$. There are three cases:

- If T_{i_1} halts after j_2 steps or less, then there is no $\text{mar}^{\mathcal{U}}_h \in \mathcal{H}^{\mathcal{U}}_{\text{mar}}$ with $x_2 \in \text{mar}^{\mathcal{U}}_h$.
- If T_{i_1} does not halt after j_2 and $j_1 \neq 0$, then there is no $\text{mar}^{\mathcal{U}}_h \in \mathcal{H}^{\mathcal{U}}_{\text{mar}}$ with $x_1 \in \text{mar}^{\mathcal{U}}_h$ and $x_2 \notin \text{mar}^{\mathcal{U}}_h$.
- If T_{i_1} does not halt after j_2 steps and $j_1 = 0$, then there is no $\text{mar}^{\mathcal{U}}_h \in \mathcal{H}^{\mathcal{U}}_{\text{mar}}$ with $x_2 \in \text{mar}^{\mathcal{U}}_h$ and $x_1 \notin \text{mar}^{\mathcal{U}}_h$.

To prove (iii), i.e., that the pointwise robust loss is computably evaluable, let $h_{i,j} \in \mathcal{H}$, $x \in \mathcal{X}$, $y \in \mathcal{Y}$ be arbitrary, and consider the following procedure:

- If $x = (i, 0)$ and $y = 1$, we have that $\ell^{\mathcal{U}}(h_{i,j}, x, y) = 1$ if and only if $(i, j + 1) \in \mathcal{U}((i, 0))$, which can be determined by running T_i for $j + 1$ steps and checking whether it halts.
- If $x = (i, 0)$ and $y = 0$, then $\ell^{\mathcal{U}}(h_{i,j}, x, y) = 1$.
- If $x = (i, k)$ with $k > 0$ and $y = 1$, check whether $k \leq j$. If so, $\ell^{\mathcal{U}}(h_{i,j}, x, y) = 0$, otherwise $\ell^{\mathcal{U}}(h_{i,j}, x, y) = 1$.
- If $x = (i, k)$ with $k > 0$ and $y = 0$, check whether $k \leq j$. If so, $\ell^{\mathcal{U}}(h_{i,j}, x, y) = 1$. Otherwise $\ell^{\mathcal{U}}(h_{i,j}, x, y) = 0$ if and only if $(i, 0) \notin \mathcal{U}((i, j))$, which can be checked by running T_i for k many steps and checking whether it halts.
- If $x = (i', k)$ with $i' \neq i$, then $\ell^{\mathcal{U}}(h_{i,j}, x, y) = y$.

Finally, we show that there is no strong realizable $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$-oracle and that $\mathcal{H}$ is not properly agnostically $\mathcal{U}$-robustly CPAC learnable. We first show the non-existence of a computable strong realizable $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$-oracle or agnostic $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$-oracle We prove this with a reduction from the Halting problem: for all $i \in \mathbb{N}$, a strongly realizable $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$ oracle (or agnostic $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$-oracle) ran on the single labelled instance $((i, 0), 1)$ outputs a hypothesis h with robust loss $\ell^{\mathcal{U}}(h, (i, 0), 1) = 0$ if and only if the Turing Machine T_i halts on the empty input. As a side note, we observe that a weak realizable $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$-oracle does exist for this construction.

We show that $\mathcal{H}$ is not agnostically properly $\mathcal{U}$-robustly CPAC learnable in a similar way. Assume there was an agnostically properly $\mathcal{U}$-robustly CPAC learner $\mathcal{A}$ with sample complexity function m. For all $i \in \mathbb{N}$, define D_i as $D_i((i, 0), 1) = 1$. Let $m = m(1/8, 1/8)$. By the learning guarantee of $\mathcal{A}$, we have that $\mathcal{A}(S) = h \in \mathcal{H}$ with $\mathcal{L}_{D_i}(h) \leq \inf_{h \in \mathcal{H}} \mathsf{R}(h; D_i) + \frac{1}{8}$ with probablility $7/8$ over $S \sim D_i^m$. Observe that $\inf_{h \in \mathcal{H}} \mathsf{R}(h; D_i) = 0$ if and only if T_i halts on the empty input, and otherwise $\inf_{h \in \mathcal{H}} \mathsf{R}(h; D_i) = 1$. We note that $S' = (((i, 0), 1), \ldots, ((i, 0), 1))$, where $|S'| = m$ is the only possible sample drawn from D_i. Thus we have $\mathcal{A}(S') = h' \in \mathcal{H}$ with $\mathcal{L}_{D_i}(h') \leq 1/8$ if and only if T_i halts on the empty input. Furthermore, $\mathcal{L}_{D_i}(h') = \ell^{\mathcal{U}}(h', (i, 0), 1) \in \{0, 1\}$ for all $h' \in \mathcal{H}$. Thus we have $\mathcal{L}_{D_i}(\mathcal{A}(S'), (i, 0), 1) = \ell^{\mathcal{U}}(\mathcal{A}(S'), (i, 0), 1) = 0$ if and only if T_i halts on the empty input. As established above, the robust loss $\ell^{\mathcal{U}}$ can be computably evaluated on any $h \in \mathcal{H}$. Thus a proper agnostic learner $\mathcal{A}$ yields a computable procedure for solving the Halting problem, which is a contradiction to the computable undedicability of the Halting problem. Thus $\mathcal{H}$ is not agnostically properly $\mathcal{U}$-robustly CPAC learnable. ■

C.3. Proof of Proposition 19

Proof Let $S = ((x_1, y_1), \ldots, (x_k, y_k))$ be an input sample for $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$, and let $\mathcal{A}$ be the robust SCPAC learning algorithm with computable sample complexity function $m_{\mathcal{H},\mathcal{U}}$. Consider the uniform distribution D_S over S. Let $m \geq m_{\mathcal{H},\mathcal{U}}(\frac{1}{k+1}, 1/7)$, which is computable by the SCPAC guarantee. There are exactly k^m different possible samples $S_1, \ldots S_{k^m}$ of size m that can be drawn from D_S, all of which being equally likely. Using the proper learner $\mathcal{A}$ on all samples S_i, we can generate the class $\hat{\mathcal{H}} = \{\mathcal{A}(S_i) : i \in [k^m]\}$. By the robust learning guarantee, we know that at least 7/8 of the hypotheses in $\hat{\mathcal{H}}$ are successful hypotheses, i.e., hypotheses within robust risk $1/(k + 1)$ of the optimal one. Furthermore, since $\epsilon = \frac{1}{k+1}$, any successful hypothesis must have optimal loss on S

(as any error on a sample point would incur loss $\frac{1}{k}$). It now suffices to use $\ell^{\mathcal{U}}$ on $\hat{\mathcal{H}}$ and return any one of the hypotheses with minimal empirical robust risk on S. Note that, for the robust realizable setting (the second case in the theorem statement), we only get a weak realizable oracle $\mathsf{RERM}^{\mathcal{U}}_{\mathcal{H}}$, as if the sample is not robustly realizable, we don't have any guarantees on $\mathcal{A}$'s behaviour. ■

Appendix D. Proofs from Section 5

D.1. Proof of Lemma 23

Proof The idea is similar to the standard No-Free-Lunch Theorem. We let m and $Z = \left\{(z_i^0, z_i^1)\right\}_{i=1}^{2m}$ as in the theorem statement, meaning that for all $i, \mathcal{U}(z_i^0) \cap \mathcal{U}(z_i^1) \neq \emptyset$ and for all $i \neq j$ the regions $\mathcal{U}(z_i^1)$ and $\mathcal{U}(z_j^0)$ are disjoint. We now look at a robust labelling $y \in \{0,1\}^{2m}$. There are $T = 2^{2m}$ such labellings, and for each $j \in [T]$ associated with labelling $y^{(j)} \in \{0,1\}^{2m}$, there exists a function f_j such that for all $i \in [2m]$, for all $z' \in \mathcal{U}(z_i^{y_i^{(j)}})$ we have that $f_j(z') = y_i^{(j)}$. Namely, for $i \in [2m]$, the label $y_i^{(j)}$ dictates which of z_i^0 or z_i^1 will have its perturbation region constantly (and thus robustly) labelled. For each function f_j and its robust labelling $y^{(j)}$, we define the distribution D_j as follows:

$$D_j((z,y)) = \begin{cases} 1/2m & \text{if } y = y_i^{(j)} \text{ and } z = z_i^y \text{ for some } i \in [2m] \\ 0 & \text{otherwise} \end{cases} .$$

It is easy to check that $\mathsf{R}_{\mathcal{U}}(f_j; D_j) = 0$.

Let $X = \{x_i\}_{i=1}^m$ be some $\mathcal{U}$-robustly shatterable set corresponding to the set of pairs $Z = \left\{(z_i^0, z_i^1)\right\}_{i=1}^{2m}$, that is $x_i \in \mathcal{U}(z_i^0) \cap \mathcal{U}(z_i^1)$ for each $i \in [2m]$. Now note any hypothesis $h : \mathcal{X} \to \{0,1\}$ (that the learner may output) has to assign labels to the the points in the shatterable set X and $h(x_i) = 1$ implies $\ell^{\mathcal{U}}(h, z_i^0) = 1$ while $h(x_i) = 0$ implies $\ell^{\mathcal{U}}(h, z_i^1) = 1$. Thus each hypothesis incurs robust loss on at least one of the members $\{z_i^0, z_i^1\}$ of each pair, and such a member can be deduced from observing $h(x_i)$. Now the remaining argument to show that there exist f_j, D_j such that the conditions of the theorem statement are met is exactly as in the proof of the standard No-Free-Lunch theorem (see, e.g., Theorem 5.1 in Shalev-Shwartz and Ben-David (2014) for details). We note that the argument there implies that the high expected loss of the learner's output is witnessed by its behaviours on the shatterable set X. Namely, there exists f_j, D_j such that

$$\begin{aligned} \mathop{\mathbb{E}}_{S \sim D_j^m} \left[\mathsf{R}_{\mathcal{U}}(\mathcal{A}(S); D_j)\right] = \frac{1}{k}\sum_{l=1}^{k} \mathsf{R}_{\mathcal{U}}(\mathcal{A}(S_l^j); D_j) &= \frac{1}{k}\sum_{l=1}^{k}\frac{1}{2m}\sum_{i=1}^{2m} \ell^{\mathcal{U}}\left(\mathcal{A}(S_l^j), z_i^{(j)}, y_i^{(j)}\right) , \\ &\geq \frac{1}{k}\sum_{l=1}^{k}\frac{1}{2m}\sum_{i=1}^{2m} \mathbb{1}\left[\mathcal{A}(S_l^j)(x_i) \neq y_i^{(j)}\right] \geq 1/8 \end{aligned}$$

where $k = (2m)^m$ is the number of possible sequences of length m drawn from $\mathrm{supp}(D_j) := \left\{(z_i^{(j)}, y_i^{(j)})\right\}_{i=1}^{2m}$, and where we have denoted by S_l^j the l-th such sequence. ■

D.2. Proof of Theorem 26

Proof First note that, if the perturbation type $\mathcal{U}$ is such that there is an upper bound M on the maximal number of pairs $Z = \{(z_i^0, z_i^1)\}_{i=1}^{2m}$ admitting a robustly shatterable set, then the computable robust shattering dimension is vacuously finite for any hypothesis class $\mathcal{H}$. Otherwise, let $\mathcal{A}$ be a $\mathcal{U}$-robust CPAC learner for $\mathcal{H}$ with sample complexity function $m(\epsilon, \delta)$. Then, fixing $\epsilon = 1/8$ and $\delta = 1/7$, we have that for all distributions on $\mathcal{X} \times \{0,1\}$, for any sample of size $m := m(1/8, 1/7)$

$$\Pr_{S\sim D^m}\left(\mathsf{R}_{\mathcal{U}}(\mathcal{A}(S); D) \geq \min_{h\in\mathcal{H}} \mathsf{R}_{\mathcal{U}}(h; D) + 1/8\right) < 1/7 \ . \tag{2}$$

Now, let $Z = \{(z_i^0, z_i^1)\}_{i=1}^{2m}$ admit a robustly shatterable set X, and note that, as explained in the proof of Lemma 25, we can computably find such a set X. Note that $\mathcal{H}$ and $\mathcal{U}$ satisfying the conditions of the theorem statement imply, by Lemma 25, that we can find a distribution D on $Z \times \{0,1\}$ such that

$$\Pr_{S\sim D^m}\left(\mathsf{R}_{\mathcal{U}}(\mathcal{A}(S); D) \geq 1/8\right) \geq 1/7 \ , \tag{3}$$

and whose support is on some $\{z_i^{y_i}\}_{i=1}^{2m}$ for $y \in \{0,1\}^{2m}$. This support induces a labelling y on X that is not robustly achievable with respect to Z by any $h \in \mathcal{H}$, as we would otherwise have $\min_{h\in\mathcal{H}} L_D^{\mathcal{U}}(h) = 0$, implying that Equation 3 and $\Pr_{S\sim D^m}\left(\mathsf{R}_{\mathcal{U}}(\mathcal{A}(S); D) \geq 1/8\right) < 1/7$ must hold simultaneously, a contradiction. ■

Appendix E. Remarks on the robust shatterability in the impossibility results from Section 4

We now comment on the robust shatterability in the impossibility results from Section 4. On the one hand, the perturbation type in Theorem 16 does not allow for sets of arbitrary size to admit a robustly shatterable set. Indeed, any sequence of halting Turing machines (which are associated with the only instances in $\mathbb{N}$ that have a strict inequality $k \subset \mathcal{U}(k)$) will have an ordering such that their perturbation regions form an ascending chain, which makes the condition $i \neq j \implies \mathcal{U}(z_i^1) \cap \mathcal{U}(z_j^0) = \emptyset$ unsatisfiable for any candidate set $Z = \{(z_i^0, z_i^1)\}_{i=1}^{2m}$. On the other hand, the perturbation type in Theorem 15 satisfies the robust shatterability requirement. Indeed, we will add the mild assumption that each proof proves at most one formula. This implies that for each $i \neq i'$, $\mathcal{U}(6i+4)$ and $\mathcal{U}(6i'+4)$ are disjoint, thus, letting $T = (\varphi_{i_k})_{k\in\mathbb{N}}$ be an enumeration of the tautologies, the set $Z = \{(6i_k + 2, 6i_k + 4)\}_{k\in\mathbb{N}}$ admits the robustly shatterable set $X = \{6i_k + 2\}_{k\in\mathbb{N}}$, while the sets $\mathcal{U}(6i+4)$ may be infinite.