Computable PAC Learning of Continuous Features

Computable PAC Learning of Continuous Features

Nathanael Ackerman
Harvard University
Cambridge, MA 02138, USA
nate@aleph0.net

Julian Asilis
Boston College
Chestnut Hill, MA 02467, USA
julian.asilis@bc.edu

Jieqi Di
Boston College
Chestnut Hill, MA 02467, USA
dij@bc.edu

Cameron Freer
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
freer@mit.edu

Jean-Baptiste Tristan
Boston College
Chestnut Hill, MA 02467, USA
tristanj@bc.edu

ABSTRACT

We introduce definitions of *computable PAC learning* for binary classification over computable metric spaces. We provide sufficient conditions on a hypothesis class to ensure than an empirical risk minimizer (ERM) is computable, and bound the strong Weihrauch degree of an ERM under more general conditions. We also give a presentation of a hypothesis class that does not admit any proper computable PAC learner with computable sample function, despite the underlying class being PAC learnable.

KEYWORDS

Computable analysis, PAC learning, VC dimension

ACM Reference Format:
Nathanael Ackerman, Julian Asilis, Jieqi Di, Cameron Freer, and Jean-Baptiste Tristan. 2022. Computable PAC Learning of Continuous Features. In *37th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS) (LICS '22), August 2–5, 2022, Haifa, Israel.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3531130.3533330

1 INTRODUCTION

The modern statistical learning theory framework for the study of uniform learnability is the synthesis of two theories. On the one hand, *Vapnik–Chervonenkis (VC) theory* [19] is a statistical theory that provides a rate of convergence for a uniform law of large numbers for estimates of the form $\frac{1}{n} \cdot \left|\{i < n \ : \ f(X_i) \neq Y_i\}\right|$, where (X_i, Y_i) are i.i.d. samples from an unknown probability measure over $\mathcal{X} \times \mathcal{Y}$ and $f : \mathcal{X} \to \mathcal{Y}$ is a function from a class $\mathcal{H}$ of measurable functions. The rate of convergence is a function of the complexity of the class $\mathcal{H}$, measured using the concept of *VC dimension*. On the other hand, *efficient Probably Approximately Correct (PAC) learnability* [18] is a computational theory that defines the efficient learnability of a function class $\mathcal{H}$ in terms of the existence of a *learner*, given by an algorithm having polynomial runtime, that takes an i.i.d. sample $S = \big((X_i, Y_i)\big)_{i<n}$ from an unknown probability measure μ as input and returns a function $h \in \mathcal{H}$ whose error $\Pr(h(X) \neq Y)$ for $(X, Y) \sim \mu$ can be bounded with high probability over the choice of S. The analogous notion of *PAC learnability* [5], where the learner is merely required to be *measurable* in an appropriate sense, rather than efficiently computable, has also been widely studied.

The synthesis of these two theories culminates with the so-called fundamental theorem of machine learning [5], which establishes, under certain broadly-applicable measurability conditions, that a class of functions is PAC learnable if and only if its VC dimension is finite. This theory provides a justification for the foundational learning paradigm of empirical risk minimization and has become the basis for studying many other learning paradigms and non-uniform theories of learnability. Note, however, that in this framework the learner is only required to be a measurable function, and in particular need not be computable.

Insofar as the goal of studying uniform learning is to determine when a problem admits supervised learning by some program given access to training examples, it is important to investigate the subclass of learners that are in some sense *computable*, a natural object of study intermediate between learners that are efficiently computable and those that are merely measurable. In this direction, Agarwal et al. [3] proposed a notion of computable learner for computably represented hypothesis classes $\mathcal{H}$ on discrete spaces. They principally consider binary classification in the case where $\mathcal{H}$ is a computably enumerable set of computable functions on a countable domain, e.g., $\mathcal{X} = \mathbb{N}$.

However, many natural problems considered in classical PAC learning theory have continuous domains, such as $\mathbb{R}^n$. In the present paper, we consider notions of computable learners and hypothesis classes, without restricting to the discrete setting, e.g., where $\mathcal{X}$ is an arbitrary computable metric space. We do so using the framework of computable analysis [21], and establish upper and lower bounds on the computability of several standard classes of learners in our setting.

We now describe the structure of the paper. Next, in Section 1.1, we describe several other approaches to computability in learning theory, including [3], and their relation to our work. We then in Section 3 provide the relevant preliminaries from computability theory (including computable metric spaces and Weihrauch reducibility) and from classical PAC learning theory. In Section 4 we develop the basic concepts of computable learning theory in our setting, including notions of computability for learners, presentations of hypothesis classes, and sample functions. In Section 5, for any computably presented hypothesis class, we establish an

LICS '22, August 2–5, 2022, Haifa, Israel

ACM ISBN 978-1-4503-9351-5/22/08…$15.00
https://doi.org/10.1145/3531130.3533330

upper bound on the strong Weihrauch degree of an empirical risk minimization (ERM) learner and its *parallelization*, and we provide sufficient conditions for an ERM learner to be computable. Finally, in Section 6 we prove matching lower bounds, via the construction of a (computable presentation of a) hypothesis class that is PAC learnable but which has no computable proper PAC learner that admits a computable sample function.

1.1 Related work

Computability of PAC learners has also been studied by Agarwal et al. [3], who consider the setting of *discrete* features and *countable* hypothesis classes. They provide several positive and negative results on the computability of both proper and improper learners for various notions of computably presented hypothesis classes, in both the realizable and agnostic cases. Our results, when we restrict our setting to discrete spaces, correspond most closely to their results for so-called *recursively enumerably representable* (RER) hypothesis classes. In particular, our Theorem 5.3 can be viewed as a generalization of [3, Theorem 10].

The related notion of *strong* computable learning is introduced and studied by Sterkenburg [17], likewise in the case of discrete features and countable hypothesis classes. Strong computable learners are defined by the existence of a computable sample function, a condition first studied in an earlier version [1] of the present paper. Furthermore, [17] considers the arithmetical complexity of learnability alongside its set-theoretic undecidability.

Computability of *non-uniform* learning, which we do not consider in this paper, has been studied in the discrete setting by Soloveichik [16], as well as in [3].

In the present paper (and [3]) when considering a function with finite codomain (as arises for both learners and presentations of hypothesis classes), the notion of computable function is such that for each input, the output is always eventually given. It is also reasonable to consider settings in which there is a particular value signaling non-halting, which the computable function may never identify. This approach is explored by Crook et al. [10], where non-halting of a learner's output is signaled by the value $\perp$. A related approach is considered by Calvert [9], who studies PAC learning for concepts that are Π^0_1 classes on $2^{\mathbb{N}}$, which can be thought of as equivalent to working with computable functions from $2^{\mathbb{N}}$ to Sierpiński space $\mathbb{S}$ (i.e., the space $\{\perp, \top\}$ with open sets $\{\emptyset, \{\top\}, \{\perp, \top\}\}$), where the inverse image of $\top$ is the Π^0_1 class in question.

The computability of PAC learnability has also been studied by Wehner [20] and Schaefer [14], who characterize the arithmetical complexity of deciding finiteness of the VC dimension for various families of hypothesis classes.

Another interaction between learning theory and computability is in the setting of "learning in the limit" [12], sometimes called *TxtEx learning*. One recent result by Beros [4] in this framework establishes the Σ^0_3-completeness of this learning problem for certain computably enumerable hypothesis classes.

2 SUMMARY OF MAIN RESULTS

The field of *supervised learning* concerns the task of predicting labels from features given a sample of labeled features $S = \big((x_i, y_i)\big)_{i<n}$, and famously underscores successes in such settings as image recognition and spam detection. Given a universe $\mathcal{X}$ of features and $\mathcal{Y}$ of labels, a *learner* can be formalized as a function $A\colon (\mathcal{X} \times \mathcal{Y})^{<\omega} \times \mathcal{X} \to \mathcal{Y}$ sending a sample S and feature x to its predicted label $A(S, x)$.

The success of a learner A is formulated with respect to a particular *hypothesis class* $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ and a family $\mathbb{D}$ of probability measures over $\mathcal{X} \times \mathcal{Y}$, by means of the *PAC learning* paradigm. In particular, A is said to be a PAC learner for $\mathcal{H}$ with respect to $\mathbb{D}$ if there exists a *sample function* $m\colon (0,1)^2 \to \mathbb{N}$ so that, when trained on samples of size at least $m(\epsilon, \delta)$, with probability at least $(1-\delta)$ over the choice of sample, A attains error no more than ϵ worse than any hypotheses in $\mathcal{H}$, when the true underlying distribution lies in $\mathbb{D}$. The case in which $\mathbb{D}$ consists of all measures over $\mathcal{X} \times \mathcal{Y}$ is referred to as learning in the *agnostic case*, and the case in which $\mathbb{D}$ is restricted to those measures for which some $h \in \mathcal{H}$ attains an error of 0 is referred to as learning in the *realizable case* (with respect to $\mathcal{H}$).

In the setting of binary classification, i.e., $\mathcal{Y} = \{0, 1\}$, the existence of a PAC learner for a class $\mathcal{H}$ (with respect to any family $\mathbb{D}$ of measures) is determined by the VC dimension of $\mathcal{H}$, a combinatorial measure of its complexity. The so-called fundamental theorem of machine learning establishes that, under tame measurability conditions, there exists a PAC learner for $\mathcal{H}$ with respect to $\mathbb{D}$ if and only if its VC dimension is finite. Furthermore, it establishes that in this case, every learner A that is an *empirical risk minimizer* (ERM), i.e., for which $A(S, \cdot)$ minimizes empirical error over $\mathcal{H}$ for every sample S, is indeed a PAC learner for $\mathcal{H}$ and any $\mathbb{D}$.

Crucially, the classical fundamental theorem permits learners to be arbitrary Borel measurable functions, which need not be computable. In order to more faithfully capture the setting of machine learning, we consider the computability of learners. In this paper, we investigate to what extent results in classical learning theory have computable analogues; when computable analogues need not exist, we quantify how badly noncomputable they can be.

To this end, we study binary classification over domains $\mathcal{X}$ which are arbitrary *computable metric spaces* and define a *computable learner* to be a learner that is computable as a map of computable metric spaces. In order to study which classes can be learned by computable learners, we consider the case where elements of $\mathcal{H}$ are identified by indices bearing some structure. Namely, given an index space $\mathcal{I}$, we say that a map $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \mathcal{Y}$ is a *presentation* of the underlying hypothesis class

$$\mathfrak{H}^\dagger = \operatorname{range}\big(i \in \mathcal{I} \mapsto \mathfrak{H}(i, \cdot)\big).$$

Such a presentation is said to be computable when $\mathcal{I}$ is a computable metric space and $\mathfrak{H}$ is computable as a map of computable metric spaces. We also study the notion of a *proper learner* for $\mathfrak{H}$, i.e., a map $\mathfrak{A}\colon (\mathcal{X} \times \mathcal{Y})^{<\omega} \to \mathcal{I}$, which is said to *induce* the learner A given by

$$A\big((x_i, y_i)_{i\in[n]}, x\big) = \mathfrak{H}\big(\mathfrak{A}\big((x_i, y_i)_{i\in[n]}\big), x\big).$$

A proper learner is computable when it is computable as a map of computable metric spaces.

We now present our primary results, expressed using the language of represented spaces and strong Weihrauch reducibility (see Section 3.1 for details).

First, we show that hypothesis classes equipped with computable presentations always admit an ERM that is computable in the realizable case.

Corollary 2.1 (of Theorem 5.3). *If $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \mathcal{Y}$ is a computable presentation, then there is an ERM for $\mathfrak{H}^\dagger$ that is computable in the realizable case.*

Observe that this result holds even when the hypothesis class $\mathfrak{H}^\dagger$ has infinite VC dimension (and hence is not PAC learnable). For classes of finite VC dimension, every ERM learner is a PAC learner (see Theorem 3.23). Hence when $\mathfrak{H}^\dagger$ has finite VC dimension, the ERM produced by Corollary 2.1 is a PAC learner.

For learning $\mathfrak{H}$ in the agnostic case, we characterize the worst-case noncomputability of ERM learners by bounding the strong Weihrauch degree of a particular ERM learner as a function of the index set $\mathcal{I}$. We make use of the limit map $\lim_{\mathbb{X}}$, which essentially sends the Cauchy sequences in a computable metric space $\mathbb{X}$ to their limits.

Corollary 2.2 (of Theorem 5.1). *If $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \mathcal{Y}$ is a computable presentation, then there exists an ERM learner for $\mathfrak{H}^\dagger$ in the agnostic case that is strongly Weihrauch reducible to $\lim_{\mathcal{I}}$.*

Likewise, when $\mathfrak{H}^\dagger$ has finite VC dimension, this result shows that we can always find a PAC learner with the stated upper bound.

The *parallelization* of a function can be thought of as simultaneously evaluating countably many instances of the original function. As such, it is often the more appropriate object of study when considering the strong Weihrauch reducibility of functions, such as learners, whose range is finite (as opposed to settings where infinitely many bits are required to describe the output).

When considering the parallelization of a learner as in Corollary 2.2, the upper bound may increase from $\lim_{\mathcal{I}}$ to $\lim_{\mathbb{N}^\mathbb{N}}$ but no further.

Corollary 2.3 (of Corollary 5.2). *If $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \mathcal{Y}$ is a computable presentation, then there is an ERM for $\mathfrak{H}^\dagger$ whose parallelization is strongly Weihrauch reducible to $\lim_{\mathbb{N}^\mathbb{N}}$.*

By a classical result, ERM learners for hypothesis classes of finite VC dimension are furthermore guaranteed to admit computable sample functions (see Theorem 3.24). As such, Corollary 2.3 implies that when $\mathfrak{H}^\dagger$ has finite VC dimension, we can always find a PAC learner A with sample function m for which the pair $(\widehat{A}, m)$ is strongly Weihrauch reducible to $\lim_{\mathbb{N}^\mathbb{N}}$, written $(\widehat{A}, m) \leq_{\mathrm{sW}} \lim_{\mathbb{N}^\mathbb{N}}$, where $\widehat{A}$ denotes the parallelization of A.

We are able to show that the corresponding bound for proper learners is tight.

Corollary 2.4 (of Theorem 6.1). *There is a computable presentation $\mathfrak{H}$ such that for every proper learner $\mathfrak{A}$ that induces a PAC learner for $\mathfrak{H}^\dagger$ and every sample function m for the induced learner, $\lim_{\mathbb{N}^\mathbb{N}} \leq_{\mathrm{sW}} (\widehat{\mathfrak{A}}, m)$.*

When considering the computability of an algorithm that requests samples (of a given size) and outputs hypotheses with a desired error and failure probability, one must consider the computability of not just a learner, but also of an accompanying sample function. Even though ERMs for classes of finite VC dimension admit computable sample functions, this is not the case in general. We exhibit a computable PAC learner A that does not admit any computable sample function.

Corollary 2.5 (of Theorem 4.12). *There exists a computable PAC learner A for a hypothesis class $\mathcal{H}$ and collection of measures $\mathbb{D}$ such that any sample function m for A is noncomputable.*

Corollary 2.5 provides a concrete example of how one must take into account the computability of sample functions as well as learners, when studying the hardness of learning problems.

3 PRELIMINARIES

This section provides a brief treatment of the computability theory and classical learning theory that form the starting point of our study.

We begin by recalling several pieces of notation. For a set I, we write $(s_i)_{i\in I}$ to denote an I-indexed sequence. For $n \in \mathbb{N}$, write $[n]$ to denote the set $\{0, 1, \ldots, n-1\}$. We write $f{\restriction_U}$ to denote the restriction of a function $f\colon X \to Y$ to a subdomain $U \subseteq X$.

For a topological space $\mathcal{X}$, we write $\mathcal{X}^{<\omega}$ for the space $\coprod_{i\in\mathbb{N}} \mathcal{X}^i$ of finite sequences of points in $\mathcal{X}$, endowed with its natural topology as the coproduct of product spaces. An **extended metric space** is a set X equipped with a distance function $d\colon X \times X \to \mathbb{R} \cup \{\infty\}$ satisfying the usual metric axioms (where $\infty + r = \infty$ for any $r \in \mathbb{R} \cup \{\infty\}$). (Note that as a special case, any metric space is also an extended metric space.)

3.1 Computable metric spaces and Weihrauch reducibility

We next describe certain key notions of computability and computable analysis, including the notions of computable metric spaces and computable functions between them. (In this paper, points in computable metric spaces will be allowed to have distance ∞.) For more details and several equivalent formulations of the basic notions, see, e.g., [7, Section 4]. We then describe the notion of Weihrauch reducibility; for more details, see [6].

Recall that a partial function f from $\mathbb{N}$ to $\mathbb{N}$ is said to be **computable** if there is some Turing machine that halts on input n (encoded in binary) precisely when f is defined on n, and in this case produces (a binary encoding of) $f(n)$ as output. We fix a standard encoding of Turing machines and write $\{e\}$ to denote the partial function that the program encoded by $e \in \mathbb{N}$ represents. We write $\{e\}(n)\downarrow$ to mean that the partial function $\{e\}$ is defined on n, i.e., that the program encoded by e halts on input n, and write $\{e\}(n)\uparrow$ otherwise.

In this paper, it will be convenient to take oracles to be elements of $\mathbb{N}^\mathbb{N}$ rather than $2^\mathbb{N}$. For $f \in \mathbb{N}^\mathbb{N}$ we write $\{e\}^f$ to denote the partial function defined by an oracle program encoded by e using f as an oracle. Because we are using oracles in $\mathbb{N}^\mathbb{N}$, we will define the Turing jump to yield a function rather than a set. Given $f \in \mathbb{N}^\mathbb{N}$, the **Turing jump** of f, written f', is defined to be the characteristic function of $\{e \in \mathbb{N} : \{e\}^f(0)\downarrow\}$. By convention, we write $\emptyset'$ for the characteristic function of the halting set $\{e \in \mathbb{N} : \{e\}(0)\downarrow\}$.

A subset of $\mathbb{N}$ is **computable** if its characteristic function is a total computable function, and is **computably enumerable** (c.e.) if it is the domain of a partial computable function (equivalently,

either empty or the range of a total computable function). We will also speak of more elaborate finitary objects (such as sets of finite tuples of rationals) as being computable or c.e. when they are computable or c.e., respectively, under a standard encoding of the objects via natural numbers.

For concreteness, we will use the notion of a *presentation* of a real when defining computable metric spaces, but note that this could also be formulated using represented spaces, as defined later in the section. An **extended real** is an element of $\mathbb{R} \cup \{\infty\}$. A **presentation** of an extended real is a sequence of rationals $(q_i)_{i\in\mathbb{N}}$ such that if there is some $\ell \in \mathbb{N}$ for which $q_{\ell+1} < q_\ell + 1$, then for all $j, k \in \mathbb{N}$ with $j < k$ we have $\left|q_{\ell_0+j} - q_{\ell_0+k}\right| < 2^{-j}$, where ℓ_0 is the least such ℓ. If no such ℓ exists, we say that the sequence is a presentation of ∞. If there is such an ℓ, we say that the sequence is a presentation of the real $\lim_{i\to\infty} q_i$. We say that an extended real is **computable** if it has a computable presentation. A **computable real** is an element of $\mathbb{R}$ admitting a computable presentation as an extended real.

We say that a sequence $(t_i)_{i\in\mathbb{N}}$ in an extended metric space $\mathcal{X} = (X, d)$ is a **rapidly converging Cauchy sequence** when for all $i < j$ we have $d(t_i, t_j) < 2^{-i}$.

Definition 3.1. A **computable metric space** is a triple $\mathbb{X} = (X, d_{\mathbb{X}}, (s_i^{\mathbb{X}})_{i\in\mathbb{N}})$ such that

(1) $(X \cup S, d_{\mathbb{X}})$ is a separable extended metric space, where $S = \{s^{\mathbb{X}} : i \in \mathbb{N}\}$,
(2) $(s_i^{\mathbb{X}})_{i\in\mathbb{N}}$, called the sequence of **ideal points of** $\mathbb{X}$, enumerates a dense subset of $(X \cup S, d_{\mathbb{X}})$,
(3) X, called the **underlying set** of $\mathbb{X}$, is dense in $(X \cup S, d_{\mathbb{X}})$, and
(4) $d_{\mathbb{X}}$, called the **distance function**, is such that $d_{\mathbb{X}}(s_i^{\mathbb{X}}, s_j^{\mathbb{X}})$ is a computable extended real, uniformly in i and j.

An element $x \in X$ is said to be a **computable point** of $\mathbb{X}$ if there is a computable function $f : \mathbb{N} \to \mathbb{N}$ such that $(s_{f(i)}^{\mathbb{X}})_{i\in\mathbb{N}}$ is a rapidly converging Cauchy sequence that converges to x. We will omit the superscripts and subscripts when they are clear from context.

Note that in many papers, computable metric spaces are not allowed to take the value ∞. Further, in some papers (e.g., [8, Definition 2.1] and [7, Definition 7.1]), the notion of computable metric space is defined only in the case where the set S of ideal points is a subset of X, while in others (e.g., [13, Definition 2.4.1]) computable metric spaces are also required to be *complete* metric spaces.

Example 3.2. The set $\mathbb{R}$ of real numbers forms a computable metric space under the Euclidean metric, when equipped with the set $\mathbb{Q}$ of rationals as ideal points under its standard enumeration $(q_i)_{i\in\mathbb{N}}$. Its computable points are precisely the computable reals.

Note that in general, the ideal points of a computable metric space are not required to be in its underlying set, as illustrated by the following example.

Example 3.3. The set of irrational numbers forms a computable metric space under the Euclidean metric, again equipped with $(q_i)_{i\in\mathbb{N}}$ as the sequence of ideal points. The computable points of this computable metric space are the computable irrational numbers.

The two spaces in the next example will be key in many of our constructions.

Example 3.4. *Baire space*, written $\mathbb{N}^{\mathbb{N}}$, is the computable metric space consisting of countably infinite sequences of natural numbers, with ideal points those sequences having only finitely many nonzero values (ordered lexicographically), and where $d_{\mathbb{N}^{\mathbb{N}}}$ is the ultrametric on the countably infinite product of $\{0, 1\}$, i.e.,

$$d_{\mathbb{N}^{\mathbb{N}}}\left((s_i)_{i\in\mathbb{N}}, (t_i)_{i\in\mathbb{N}}\right) = 2^{-\inf_{i\in\mathbb{N}}(s_i \neq t_i)}.$$

Cantor space, written $2^{\mathbb{N}}$, is the computable metric subspace of $\mathbb{N}^{\mathbb{N}}$ consisting of binary sequences.

Let π_0 and π_1 be computable maps from $\mathbb{N}$ to $\mathbb{N}$ such that $i \mapsto (\pi_0(i), \pi_1(i))$ is a computable bijection of $\mathbb{N}$ with $\mathbb{N} \times \mathbb{N}$.

When $\mathbb{X}$ and $\mathbb{Y}$ are computable metric spaces, we write $\mathbb{X} \times \mathbb{Y}$ to denote the computable metric space with underlying set $X \times Y$, with sequence of ideal points $\left((s^{\mathbb{X}}_{\pi_0(i)}, s^{\mathbb{Y}}_{\pi_1(i)})\right)_{i\in\mathbb{N}}$, and where

$$\left((X \cup S^{\mathbb{X}}) \times (Y \cup S^{\mathbb{Y}}), d_{\mathbb{X}\times\mathbb{Y}}\right)$$

is the product extended metric space of $\left(X \cup S^{\mathbb{X}}, d_{\mathbb{X}}\right)$ and $\left(Y \cup S^{\mathbb{Y}}, d_{\mathbb{Y}}\right)$.

We let $\mathbb{X}^{<\omega}$ be the coproduct $\coprod_{n\in\mathbb{N}} \prod_{i\in[n]} \mathbb{X}$, i.e., the space whose underlying set consists of finite sequences of elements of X, whose ideal points are finite sequences of ideal points in X, and where the distance function satisfies

$$d_{\mathbb{X}^{<\omega}}\left((x_i)_{i\in[n]}, (y_i)_{i\in[m]}\right) = \begin{cases} \max\limits_{i\in[n]} \; d_{\mathbb{X}}(x_i, y_i) & \text{if } m = n; \\ \infty & \text{otherwise.} \end{cases}$$

Note that this agrees with the definition as a topological space: the extended metric space $\mathbb{X}^{<\omega}$ indeed has topology that of the coproduct topology for the sequence $(\mathbb{X}^n)_{n\in\mathbb{N}}$ of extended metric spaces considered as a sequence of topological spaces.

Definition 3.5. Suppose $\mathcal{X} = (X, d_{\mathcal{X}})$ and $\mathcal{Y} = (Y, d_{\mathcal{Y}})$ are extended metric spaces and $Z \subseteq X$. We say a map $f : X \to Y$ is **continuous** on Z if for all open sets $U \subseteq Y$, there is an open set $V \subseteq X$ such that $f^{-1}(U) \cap Z = V \cap Z$. In other words, f restricted to Z is continuous as a map from the extended metric space that $\mathcal{X}$ induces on Z to $\mathcal{Y}$.

Definition 3.6. Let $\mathbb{X}$ and $\mathbb{Y}$ be computable metric spaces with ideal points $(s_i)_{i\in\mathbb{N}}$ and $(t_i)_{i\in\mathbb{N}}$ respectively, and suppose $Z \subseteq X$. Suppose $f : W \to Y$ is a map where $Z \subseteq W \subseteq X$. We say that f is **computable on** Z if for all $(j, q) \in \mathbb{N}\times\mathbb{Q}$ there is a set $\Phi_{j,q} \subseteq \mathbb{N}\times\mathbb{Q}$ such that

- $f^{-1}(B(t_j, q)) \cap Z = \left(\bigcup_{(k,p)\in\Phi_{j,q}} B(s_k, p)\right) \cap Z$, and
- the set $\{(j, q, k, p) : (k, p) \in \Phi_{j,q}\}$ is c.e.

This definition captures the notion that the partial map f is continuous on its restriction to Z and has a computable witness to this continuity.

Observe that a computable function from $\mathbb{N}^{\mathbb{N}}$ to a computable metric space $\mathbb{Y}$ can be thought of as a program on an oracle Turing machine that takes the input on its oracle tape, and outputs a "representation" of a point in $\mathbb{Y}$. The notion of a *represented space* is one way of making this notion precise. For more details, see [6].

Definition 3.7. A **represented space** (X, γ) is a set X along with a surjection γ from a subset of $\mathbb{N}^{\mathbb{N}}$ onto X. When the choice of γ is clear from context, we call γ the **representation** of X.

Definition 3.8. Suppose $\mathbb{X} = (X, d_{\mathbb{X}}, (s_i^{\mathbb{X}})_{i\in\mathbb{N}})$ is a computable metric space. Define $\mathrm{CS}_{\mathbb{X}} \subseteq \mathbb{N}^{\mathbb{N}}$ to be the collection of functions $f\colon \mathbb{N} \to \mathbb{N}$ for which $\left(s^{\mathbb{X}}_{f(i)}\right)_{i\in\mathbb{N}}$ is a rapidly converging Cauchy sequence whose limit is in X. The **represented space induced by** $\mathbb{X}$ is defined to be $(X, \gamma_{\mathbb{X}})$, where

$$\gamma_{\mathbb{X}}\colon \mathrm{CS}_{\mathbb{X}} \to X$$

assigns each function f the value $\lim_{i\to\infty} s^{\mathbb{X}}_{f(i)}$.

Intuitively, a *realizer* of a function g takes a description of an input x to a description of the corresponding output $g(x)$, where these descriptions are given in terms of representations.

Definition 3.9. Suppose (X, γ_X) and (Y, γ_Y) are represented spaces, and let $g\colon X \to Y$ be a map. A **realizer** of g is any function $G\colon \mathrm{dom}(\gamma_X) \to \mathrm{dom}(\gamma_Y)$ such that $\gamma_Y \circ G = g \circ \gamma_X$.

A realizer is **computable** if it is computable on $\mathrm{dom}(\gamma_X)$ (considered as a partial map between computable metric spaces $\mathbb{N}^{\mathbb{N}}$ and $\mathbb{N}^{\mathbb{N}}$).

The notion of *strong Weihrauch reducibility* aims to capture the intuitive idea that one function is computable given the other function as an oracle, along with possibly some computable pre-processing and post-processing, where access to the original input is permitted only in pre-processing. (The weaker notion of *Weihrauch reducibility*, in which the input may be used again in post-processing, also arises in computable analysis, but in this paper we are able to show that all of the relevant reductions are strong.)

Definition 3.10. Let (X_i, γ_{X_i}) and (Y_i, γ_{Y_i}) be represented spaces for $i \in \{0, 1\}$, and suppose that $f\colon X_0 \to Y_0$ and $g\colon X_1 \to Y_1$ are functions. Let $\mathcal{F}$ and $\mathcal{G}$ be the sets of realizers of f and g respectively. We say that f is **strongly Weihrauch reducible** to g, and write $f \leq_{\mathrm{sW}} g$, when there are computable functions H and K, each from some subset of $\mathbb{N}^{\mathbb{N}}$ to $\mathbb{N}^{\mathbb{N}}$, such that for every $G \in \mathcal{G}$ there exists an $F \in \mathcal{F}$ satisfying $F = H \circ G \circ K$. We say that f and g are **strongly Weihrauch equivalent**, and write $f \equiv_{\mathrm{sW}} g$, when $f \leq_{\mathrm{sW}} g$ and $g \leq_{\mathrm{sW}} f$.

Note that strong Weihrauch reducibility is usually described in the more general setting of partial multifunctions. Here we will only need single-valued functions with explicitly defined domains, and Definition 3.10 coincides with the standard one in this situation.

The following important map describes the problem of computing limits on a represented space X induced by a computable metric space $\mathbb{X}$. (Note that elsewhere in the literature, $\lim_{\mathbb{X}}$ is typically referred to as $\lim_X$.)

Definition 3.11. Suppose $\mathbb{X}$ is a computable metric space, and let $(X, \gamma_{\mathbb{X}})$ be the represented space it induces. The **limit map** $\lim_{\mathbb{X}}$ is the partial function from $X^{\mathbb{N}}$ to X that assigns every convergent Cauchy sequence in X its limit (and is undefined elsewhere).

One can view $\lim_{\mathbb{N}^{\mathbb{N}}}$ as playing a role in Weihrauch reducibility analogous to the role played by the halting problem $\emptyset'$ with respect to Turing reducibility. For more details, see [6, §11.6].

It will also be useful to introduce the notion of a rich space, which bears a relation to $\lim_{\mathbb{N}^{\mathbb{N}}}$ and is informally a space that computably contains the real numbers.

Definition 3.12. A computable metric space $\mathbb{X}$ is **rich** if there is some computable map $\iota\colon 2^{\mathbb{N}} \to \mathbb{X}$ that is injective and whose partial inverse ι^{-1} is also computable.

Lemma 3.13 ([6, Proposition 11.6.2]). *If $\mathbb{X}$ and $\mathbb{Y}$ are rich spaces, then $\lim_{\mathbb{X}} \equiv_{\mathrm{sW}} \lim_{\mathbb{Y}}$. In particular, $\lim_{\mathbb{X}} \equiv_{\mathrm{sW}} \lim_{\mathbb{N}^{\mathbb{N}}}$.*

This implies that $\lim_{\mathbb{N}^{\mathbb{N}}}$ is maximal (under $\leq_{\mathrm{sW}}$) among limit operators.

Corollary 3.14. *Let $\mathbb{X}$ be a computable metric space. Then $\lim_{\mathbb{X}} \leq_{\mathrm{sW}} \lim_{\mathbb{N}^{\mathbb{N}}}$.*

Proof. Let $\mathbb{V}$ be the space $\mathbb{X} \coprod \mathbb{N}^{\mathbb{N}}$, and note that $\lim_{\mathbb{X}} \leq_{\mathrm{sW}} \lim_{\mathbb{V}}$. Because $\mathbb{V}$ is rich, $\lim_{\mathbb{V}} \equiv_{\mathrm{sW}} \lim_{\mathbb{N}^{\mathbb{N}}}$ by Lemma 3.13. □

We will also work with the *Turing jump* map $\mathrm{J}\colon \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$, given by $z \mapsto z'$, which is strongly Weihrauch equivalent to $\lim_{\mathbb{N}^{\mathbb{N}}}$.

Lemma 3.15 ([6, Theorem 11.6.7]). $\lim_{\mathbb{N}^{\mathbb{N}}} \equiv_{\mathrm{sW}} \mathrm{J}$.

Although $\lim_{\mathbb{N}^{\mathbb{N}}} \equiv_{\mathrm{sW}} \mathrm{J}$, in general $\lim_{\mathcal{I}}$ is weaker. In Section 5 we will establish our upper bounds in terms of $\lim_{\mathcal{I}}$ for appropriate computable metric spaces $\mathcal{I}$, while in Section 6 we will establish a bound using the operator J.

Strong Weihrauch reductions to the *parallelization* of a function allow one to ask for countably many instances of the function to be evaluated. This concept will be important in Sections 5 and 6, as we explain following Theorem 5.1.

Definition 3.16. Let $f\colon X \to Y$ be a map between represented spaces. The **parallelization** of f is the map $\widehat{f}\colon X^{\mathbb{N}} \to Y^{\mathbb{N}}$ defined by $\widehat{f}\big((x_i)_{i\in\mathbb{N}}\big) = \big(f(x_i)\big)_{i\in\mathbb{N}}$.

Observe that for any map f between represented spaces, $f \leq_{\mathrm{sW}} \widehat{f}$. We will need the following standard fact.

Lemma 3.17 ([6, Theorem 11.6.6]). $\widehat{\lim_{\mathbb{N}^{\mathbb{N}}}} \equiv_{\mathrm{sW}} \lim_{\mathbb{N}^{\mathbb{N}}}$.

3.2 Learning theory

We now consider the traditional framework for uniform learnability, formulated for Borel measurable hypotheses. A learning problem is determined by a domain, label set, and hypothesis class, as we now describe.

(i) a **domain** $\mathcal{X}$ of features that is a Borel subset of some complete separable extended metric space $\mathbb{X}$,
(ii) a **label set** $\mathcal{Y}$ that is a complete separable extended metric space, and
(iii) a **hypothesis class** $\mathcal{H}$ consisting of Borel functions from $\mathcal{X}$ to $\mathcal{Y}$.

We will say that any Borel function from $\mathcal{X}$ to $\mathcal{Y}$ is a **hypothesis**; note that such a map is sometimes also called a **predictor**, **classifier**, or **concept**. In this paper, we will only consider problems in binary classification, i.e., where $\mathcal{Y} = \{0, 1\}$, considered as a metric space under the discrete topology.

Let $\mathcal{D}$ be a Borel measure on $\mathcal{X} \times \mathcal{Y}$. The **true error**, or simply **error**, of a hypothesis $h \in \mathcal{H}$ with respect to $\mathcal{D}$ is the probability

that $(x, h(x))$ disagrees with a randomly selected pair drawn from $\mathcal{D}$, i.e.,

$$L_{\mathcal{D}}(h) = \mathcal{D}\big(\{(x,y) \in \mathcal{X} \times \mathcal{Y} \mid y \neq h(x)\}\big).$$

The **empirical error** of a hypothesis h on a tuple $S = \big((x_1, y_1), \ldots, (x_n, y_n)\big) \in (\mathcal{X} \times \mathcal{Y})^n$ of **training examples** is the fraction of pairs in S on which h misclassifies the label of a feature, i.e.,

$$L_S(h) = \frac{\sum_{i=1}^n |h(x_i) - y_i|}{n}.$$

Traditionally, one thinks of a learner as a map which takes finite sequences of $(\mathcal{X} \times \mathcal{Y})^{<\omega}$ and returns a hypothesis, i.e., an element of $\mathcal{Y}^{\mathcal{X}}$. We would then like to define a computable learner as a learner which is computable as a map between computable metric spaces. Unfortunately, here we encounter the obstruction that $\mathcal{Y}^{\mathcal{X}}$ is not, in general, an extended metric space. We overcome it by instead considering a learner as the "curried" version of a map from $(\mathcal{X} \times \mathcal{Y})^{<\omega}$ to $\mathcal{Y}^{\mathcal{X}}$, i.e., as a map $(\mathcal{X} \times \mathcal{Y})^{<\omega} \times \mathcal{X} \to \mathcal{Y}$. In this manner, we will be able to consider learners which are computable as maps between computable metric spaces.

Definition 3.18. A **learner** is a Borel measurable function $A\colon (\mathcal{X} \times \mathcal{Y})^{<\omega} \times \mathcal{X} \to \mathcal{Y}$. For notational convenience, for $S \in (\mathcal{X} \times \mathcal{Y})^{<\omega}$ we let $\widetilde{A}(S)\colon \mathcal{X} \to \mathcal{Y}$ be the function defined by $\widetilde{A}(S)(x) = A(S, x)$.

The goal of a learner A is to return a hypothesis h that minimizes the true error with respect to an unknown Borel distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$. The learner does so by examining a $\mathcal{D}$-i.i.d. sequence $S = \big((x_1, y_1), \ldots, (x_n, y_n)\big)$. Notably, the learner cannot directly evaluate $L_{\mathcal{D}}$; it is guided only by the information contained in the sample S, including evaluations of L_S. However, as it is ignorant of $\mathcal{D}$, the learner does not know how faithfully L_S approximates $L_{\mathcal{D}}$.

The most central framework for assessing learners with respect to hypothesis classes is that of PAC learning (see, e.g., [15, Chapter 3]). In the setting of *efficient* PAC learning [15, Definition 8.1], one further requires that the learning algorithm be polynomial-time in the reciprocal of its inputs ϵ and δ, to be described in the following definition.

Definition 3.19. Let $\mathbb{D}$ be a collection of Borel distributions on $\mathcal{X} \times \mathcal{Y}$ and let $\mathcal{H}$ be a hypothesis class. A learner A is said to **PAC learn $\mathcal{H}$ with respect to $\mathbb{D}$** (or is a *learner for $\mathcal{H}$ with respect to $\mathbb{D}$*) if there exists a function $m\colon (0,1)^2 \to \mathbb{N}$, called a **sample function**, that is non-increasing on each coordinate and satisfies the following property: for every $\epsilon, \delta \in (0,1)$ and every Borel distribution $\mathcal{D} \in \mathbb{D}$, a finite i.i.d. sample S from $\mathcal{D}$ with $|S| \geq m(\epsilon, \delta)$ is such that, with probability at least $(1-\delta)$ over the choice of S, the learner A outputs a hypothesis $\widetilde{A}(S)$ with

$$L_{\mathcal{D}}(\widetilde{A}(S)) \leq \inf_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon. \tag{$\dagger$}$$

(Observe that ($\dagger$) is a Borel measurable condition, as $L_{\mathcal{D}}(\widetilde{A}(S)) = \int \mathbb{1}_{A(S,x) \neq y} \mathcal{D}(dx, dy)$.) The minimal such sample function for A is its **sample complexity**. When there is some learner A that learns $\mathcal{H}$ with respect to $\mathbb{D}$, we say that $\mathcal{H}$ is **PAC learnable with respect to $\mathbb{D}$** (via A).

In the case where $\mathbb{D}$ consists of all Borel distributions on $\mathcal{X} \times \mathcal{Y}$, we say that $\mathcal{H}$ is **agnostically PAC learnable** and that A is an **agnostic PAC learner for $\mathcal{H}$**. In the case where $\mathbb{D}$ consists of the class of Borel distributions $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$ for which $L_{\mathcal{D}}(h) = 0$ for some $h \in \mathcal{H}$, we say that $\mathcal{H}$ is **PAC learnable in the realizable case** and that A **PAC learns $\mathcal{H}$ in the realizable case**.

Remark 3.20. Some sources use "sample complexity" to refer to a property of hypothesis classes $\mathcal{H}$, defined as the pointwise minimum of all of $\mathcal{H}$'s PAC learners' sample complexities (in the sense of Definition 3.19). The learner-dependent definition will be more appropriate for our purposes, in which, for instance, the distinction between computable and noncomputable learners is of central importance.

We will see shortly in Theorem 3.23 that a class that is PAC learnable in the realizable case must also be agnostically PAC learnable (possibly via a different learner with worse sample complexity).

Definition 3.21. A learner E is an **empirical risk minimizer** (or *ERM*) for $\mathcal{H}$, if for all finite sequences $S \in (\mathcal{X} \times \mathcal{Y})^{<\omega}$, we have

$$\widetilde{E}(S) \in \operatorname{argmin}_{h \in \mathcal{H}} L_S(h).$$

Definition 3.22. The **VC dimension** of $\mathcal{H}$ is

$$\sup\big\{\, |C| : C \subseteq \mathcal{X} \quad \text{and} \quad \{h\restriction_C \colon h \in \mathcal{H}\} = \{0,1\}^C \big\}.$$

When $\{h\restriction_C \colon h \in \mathcal{H}\} = \{0,1\}^C$, we say that $\mathcal{H}$ **shatters** the set C.

We now state the relevant portions of the fundamental theorem of learning theory in our setting (binary classification with 0-1 loss), which holds for hypothesis classes satisfying the mild technical assumption of *universal separability* [5, Appendix A]. This condition is satisfied for any hypothesis class having a computable presentation (see Definition 4.2), as is the case for all hypothesis classes considered in this paper.

Theorem 3.23 ([15, Theorem 6.7]). *Let $\mathcal{H}$ be a hypothesis class of functions from a domain $\mathcal{X}$ to $\{0,1\}$. Then the following are equivalent:*

1. *$\mathcal{H}$ has finite VC dimension.*
2. *$\mathcal{H}$ is PAC learnable in the realizable case.*
3. *$\mathcal{H}$ is agnostically PAC learnable.*
4. *Any ERM learner is a PAC learner for $\mathcal{H}$, over any family of measures.*

Because of the equivalence between conditions 2 and 3, we will say that a hypothesis class $\mathcal{H}$ is *PAC learnable* (without reference to a class of distributions $\mathbb{D}$, and without mentioning agnostic learning or realizability) when any of these equivalent conditions hold. Note that while every agnostic PAC learner for $\mathcal{H}$ is in particular a PAC learner for $\mathcal{H}$ in the realizable case, the converse is not true; when we speak of a *PAC learner for $\mathcal{H}$* without mention of $\mathbb{D}$, we will mean the strongest such instance, namely that it is an agnostic PAC learner for $\mathcal{H}$.

Furthermore, there exists a connection between the VC dimension of a PAC learnable class and the sample functions of its ERM learners.

Theorem 3.24 ([15, Chapter 28]). *Let $\mathcal{H}$ be a hypothesis class of functions from a domain $\mathcal{X}$ to $\{0,1\}$ with finite VC dimension d. Then its ERM learners are PAC learners with sample functions*

$$m(\epsilon, \delta) = 4\frac{32d}{\epsilon^2} \cdot \log\left(\frac{64d}{\epsilon^2}\right) + \frac{8}{\epsilon^2} \cdot \big(8d \log(\epsilon/d) + 2\log(4/\delta)\big).$$

4 NOTIONS OF COMPUTABLE LEARNING THEORY

As described before Definition 3.18, the notion of learner we consider in this paper is the *curried* version of the standard one, in order to allow for it to be a computable map between computable metric spaces. We now make use of this, to define when a learner is computable and when a hypothesis class is computably PAC learnable.

Definition 4.1. By a **computable learner** we mean a learner $A\colon (\mathcal{X}\times\mathcal{Y})^{<\omega}\times\mathcal{X}\to\mathcal{Y}$ which is computable as a map of computable metric spaces. We say a hypothesis class $\mathcal{H}$ is **computably PAC learnable** if there is a computable learner that PAC learns it.

It will also be important to have a computable handle on hypothesis classes themselves. As such, we will primarily consider hypothesis classes as collection of hypotheses endowed with (not necessarily unique) indices. This information is collected up into a *presentation* of the class.

Definition 4.2. A **presentation of a hypothesis class** is a Borel measurable function $\mathfrak{H}\colon \mathcal{I}\times\mathcal{X}\to\mathcal{Y}$. We call $\mathcal{I}$ the **index space**. Let $\widetilde{\mathfrak{H}}\colon \mathcal{I}\to\mathcal{Y}^{\mathcal{X}}$ be the function defined by $\widetilde{\mathfrak{H}}(i)(x)=\mathfrak{H}(i,x)$. We write $\mathfrak{H}^{\dagger}$ to denote the underlying hypothesis class, i.e., range$(\widetilde{\mathfrak{H}})$. We say that $\mathfrak{H}$ **presents** the class $\mathfrak{H}^{\dagger}$ and that a hypothesis is an **element of** $\mathfrak{H}$ when it is in $\mathfrak{H}^{\dagger}$.

Definition 4.3. A presentation $\mathfrak{H}\colon \mathcal{I}\times\mathcal{X}\to\mathcal{Y}$ of a hypothesis class is **computable** if $\mathcal{I}$ is a computable metric space and $\mathfrak{H}$ is computable as a map of computable metric spaces.

Classically, a *proper learner* for a hypothesis class $\mathcal{H}$ is usually regarded simply as a learner which happens to always produce hypotheses in the class $\mathcal{H}$. This is a key notion, about which we will want to reason computably.

In our setting, to study the computability of proper learning, it will be valuable to consider the case in which the elements of $\mathcal{H}$ are identified by indices bearing additional structure, and thus to consider learners that identify hypotheses in $\mathcal{H}$ by such indices, using a presentation $\mathfrak{H}$. Consequently, and in contrast to the classical setting, we take proper learners to be slightly different objects than ordinary learners. Our proper learners map samples to indices, rather than mapping samples and features to labels. We then can define a computable proper learner to be simply a proper learner that is computable (similarly to Definition 4.1 of a computable learner).

Definition 4.4. Let $\mathfrak{H}\colon \mathcal{I}\times\mathcal{X}\to\mathcal{Y}$ be a presentation of a hypothesis class. A **proper learner** for $\mathfrak{H}$ is a map $\mathfrak{A}\colon (\mathcal{X}\times\mathcal{Y})^{<\omega}\to\mathcal{I}$. If the map A defined by

$$A((x_i,y_i)_{i\in[n]},x)=\mathfrak{H}(\mathfrak{A}(x_i,y_i)_{i\in[n]},x)$$

is a PAC learner for $\mathfrak{H}^{\dagger}$, then $\mathfrak{A}$ is a **proper PAC learner** for $\mathfrak{H}$, and we call A the **learner induced** by $\mathfrak{A}$ (as a proper learner for $\mathfrak{H}$). If $\mathfrak{H}$ is a computable presentation, we say that a proper learner $\mathfrak{A}$ for $\mathfrak{H}$ is **computable** when it is computable as a map of computable metric spaces.

Note that the learner A induced by a computable proper PAC learner for $\mathfrak{H}$ in Definition 4.4 is a computable learner for $\mathfrak{H}^{\dagger}$, as we have required both $\mathfrak{A}$ and $\mathfrak{H}$ to be computable. Intuitively, $\mathfrak{H}$ is computably properly PAC learnable if there is a computable function which takes in finite sequences of elements of $\mathcal{X}\times\mathcal{Y}$ and outputs the index of an element of $\mathfrak{H}$, and where the corresponding learner PAC learns $\mathfrak{H}^{\dagger}$.

Definition 4.5. Given a hypothesis class $\mathcal{H}$, define $\Phi_{\mathcal{H}}\subseteq(\mathcal{X}\times\mathcal{Y})^{<\omega}$ to be the set of those finite sequences $(x_i,y_i)_{i\in[n]}$ for which $\{(x_1,y_1),\dots,(x_n,y_n)\}$ is a subset of the graph of h for some $h\in\mathcal{H}$, i.e.,

$$\bigcup_{h\in\mathcal{H}}\coprod_{n\in\mathbb{N}}\{(x,h(x))\,:\,x\in\mathcal{X}\}^n.$$

Recall that the realizable case restricts attention to measures $\mathcal{D}$ for which $\mathcal{D}$-i.i.d. sequences are almost surely in the graph of some element of $\mathcal{H}$. In particular, for any such $\mathcal{D}$ and $n\in\mathbb{N}$, the product measure $\mathcal{D}^n$ is concentrated on $\Phi_{\mathcal{H}}\cap(\mathcal{X}\times\mathcal{Y})^n$. Note, however, that $\Phi_{\mathcal{H}}$ itself will not in general be Borel, even though every element of $\mathcal{H}$ is a Borel map. Yet, in the following definition, $\Phi_{\mathcal{H}}$ plays only the role of a subdomain on which the computability of learners in the realizable case is considered, and thus its measure-theoretic properties are of no consequence.

Definition 4.6. Let $\mathcal{H}$ be a hypothesis class. Then a learner A for $\mathcal{H}$ is **computable in the realizable case** for $\mathcal{H}$ if it is computable on $\Phi_{\mathcal{H}}\times\mathcal{X}$ as a function between computable metric spaces $(\mathcal{X}\times\mathcal{Y})^{<\omega}\times\mathcal{X}$ and $\mathcal{Y}$. A proper learner $\mathfrak{A}$ for a computable presentation $\mathfrak{H}$ of $\mathcal{H}$ is **computable in the realizable case** if $\mathfrak{A}$ is computable on $\Phi_{\mathcal{H}}$ as a function between computable metric spaces $(\mathcal{X}\times\mathcal{Y})^{<\omega}$ and $\mathcal{I}$.

Note that it is possible to have a noncomputable learner for $\mathcal{H}$ which is nevertheless computable in the realizable case for $\mathcal{H}$. However, all computable learners for $\mathcal{H}$ are computable in the realizable case for $\mathcal{H}$.

It will be important to impose computability constraints on sample functions as well as learners.

Definition 4.7. A sample function $m\colon (0,1)^2\to\mathbb{N}$ is **computable** if uniformly in $n\in\mathbb{N}$ there are computable sequences of rationals $(\ell_{n,i})_{i\in\mathbb{N}}$, $(r_{n,i})_{i\in\mathbb{N}}$, $(t_{n,i})_{i\in\mathbb{N}}$, and $(b_{n,i})_{i\in\mathbb{N}}$ such that

- $U_n\subseteq m^{-1}(n)$ for every $n\in\mathbb{N}$, and
- the closure of the set $\bigcup_{n\in\mathbb{N}}U_n$ is $(0,1)^2$,

where for each n we define $U_n=\bigcup_{i\in\mathbb{N}}(\ell_{n,i},r_{n,i})\times(t_{n,i},b_{n,i})$.

Given a computable PAC learner and a computable sample function for this learner, one can produce an algorithm that, given an error rate and failure probability, outputs a hypothesis having at most that error rate with at most the stated failure probability. If the computable learner is an ERM, then by Theorem 3.24 it has a computable sample function, and so one obtains such an algorithm. On the other hand, we will see in Theorem 4.12 that not every computable PAC learner (for a given hypothesis class $\mathcal{H}$ and class of distributions $\mathbb{D}$) admits a computable sample function (with respect to $\mathcal{H}$ and $\mathbb{D}$).

4.1 Countable hypothesis classes

Suppose that $\mathcal{X}$ is countable and discrete. Requiring that a learner A be computable is then tantamount to asking that the maps $x\mapsto A(S,x)$ be uniformly computable as S ranges over $(\mathcal{X}\times\mathcal{Y})^{<\omega}$. By collecting up this data, such a computable learner A can be

encoded as a computable map from $\mathbb{N}$ to $\mathbb{N}$. In a similar fashion, a computable presentation of a hypothesis class could be encoded by a single computable map from $\mathbb{N}$ to $\mathbb{N}$.

The paper [3] studies computable PAC learning in the setting where $\mathcal{X} = \mathbb{N}$, a countable discrete metric space. As such, they are able to work with the encodings of these simplified notions of computable learners and presentations of hypothesis classes, as we have just sketched.

4.2 Examples

To illustrate these definitions, we now describe two examples — one a very basic one in this formalism, and the other a standard example from learning theory.

4.2.1 "Apply" function. Let the index space $\mathcal{I}$ be $2^{\mathbb{N}}$ and the sample space $\mathcal{X}$ be $\mathbb{N}$. We define the "apply" presentation of the hypothesis class $2^{\mathbb{N}}$ to be the map $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \{0,1\}$ where $\mathfrak{H}(x,n) = x(n)$. Note that while $\mathfrak{H}$ is computable, there is no single Turing degree which bounds every hypothesis in $\mathfrak{H}^\dagger = 2^{\mathbb{N}}$. In particular, this example demonstrates that the notion of computable hypothesis class that we consider is fundamentally more general than the corresponding notion in [3], which considers only countable collections of hypotheses.

4.2.2 Decision stump. Recall the decision stump problem from classical learning theory defined by $\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \{0,1\}$, and $\mathcal{H} = \{\mathbb{1}_{>c} : c \in \mathbb{R}\}$. In the realizable case, the learning problem amounts to estimating the true cutoff point c from a sample $S = (x_i, y_i)_{i\in[n]}$ for which $y_i = 1$ if and only if $x_i > c$. It is well-known to be PAC learnable in the realizable case via the following algorithm:

1. If S has negatively labeled examples (i.e., (x_i, y_i) with $y_i = 0$), then set m to be the maximal such x_i. Otherwise, set m to be the minimal feature among positively labeled examples.
2. Return $\mathbb{1}_{>m}$.

In particular, this implements an ERM learner for $\mathcal{H}$ in the realizable case. Further, as $\mathcal{H}$ has VC dimension 1, it is a PAC learner for $\mathcal{H}$ in the realizable case by the equivalence of clauses 1 and 4 in Theorem 3.23.

The classical algorithm does not give rise to a computable learner in the sense of Definition 3.18, however, as $\mathbb{1}_{>m}$ cannot be computed from S. In particular, undecidability of equality for real numbers obstructs such a computation from being performed over $\mathbb{R}$. In order to more sensibly cast the problem in a computable setting, we restrict focus to cutoff points located at computable reals and take the noncomputable reals as the domain set $\mathcal{X}$.

Now consider the computable presentation $\mathfrak{H}_{\text{step}}\colon \mathbb{R}_c \times (\mathbb{R}\setminus\mathbb{R}_c) \to \{0,1\}$ of a hypothesis class with index set the computable reals $\mathbb{R}_c$, given by $\mathfrak{H}_{\text{step}}(c,x) = \mathbb{1}_{>c}(x)$. Its underlying hypothesis class $\mathfrak{H}^\dagger_{\text{step}} = \{\mathbb{1}_{>c} : c \in \mathbb{R}_c\}$ consists of computable functions (whose domains are $\mathbb{R}\setminus\mathbb{R}_c$), thus proper learners have a chance of success. Nevertheless, the classical algorithm fails: m will reside in $\mathcal{X}$, and thus $\mathbb{1}_{>m}$ will be noncomputable as a function on $\mathcal{X}$ (even when one has access to m).

We will exhibit a proper learner $\mathfrak{A}_{\text{step}}$ for $\mathfrak{H}_{\text{step}}$ that is computable in the realizable case and whose induced learner is an ERM. Fix a computable enumeration $(q_i)_{i\in\mathbb{N}}$ of $\mathbb{Q}$ and uniformly enumerate a computable presentation of each as a computable real.

Algorithm 4.8 (Algorithm $\mathfrak{A}_{\text{step}}$)**.** Given a sample S, output the first $q_j \in (q_i)_{i\in\mathbb{N}}$ for which the empirical error of $\mathbb{1}_{>q_j}$ is 0.

Proposition 4.9. *$\mathfrak{A}_{\text{step}}$ is a proper learner for $\mathfrak{H}_{\text{step}}$ that is computable in the realizable case and whose induced learner is an ERM.*

Proof. Observe that the sequence of functions $(\mathbb{1}_{>q_i})_{i\in\mathbb{N}}$ is uniformly computable on $\mathcal{X} = \mathbb{R}\setminus\mathbb{R}_c$. The empirical error of each $\mathbb{1}_{>q_i}$ can be computed exactly on any sample (and hence compared with 0). The loop terminates upon reaching a rational q_j that separates the sample S, one of which must exist for any S under consideration in the realizable case. □

Corollary 4.10. *$\mathfrak{A}_{\text{step}}$ is a computable proper PAC learner in the realizable case for $\mathfrak{H}_{\text{step}}$.*

Proof. By Proposition 4.9, $\mathfrak{A}_{\text{step}}$ is a computable proper learner in the realizable case for $\mathfrak{H}_{\text{step}}$, whose induced learner is an ERM. The class $\mathfrak{H}^\dagger_{\text{step}}$ has VC dimension 1, and so by the equivalence of clauses 1 and 4 in Theorem 3.23, the learner induced by $\mathfrak{A}_{\text{step}}$ is a PAC learner in the realizable case. □

In fact, we will see shortly in Theorem 5.3 that Corollary 4.10 is an instance of a more general result, namely that all classes with computable presentations have computable ERM learners in the realizable case.

4.3 Computable learners with noncomputable sample functions

Theorem 4.12 shows that even when a hypothesis class $\mathcal{H}$ and class of distributions $\mathbb{D}$ admit some computable PAC learner with a computable sample function, not all computable learners for $\mathcal{H}$ with respect to $\mathbb{D}$ must have a computable sample function.

Therefore, when investigating the computability of algorithms for outputting a hypothesis (with the desired error rate and failure probability), we must consider the computability of a pair consisting of a PAC learner and sample function, not merely the PAC learner alone.

The intuition behind the proof of Theorem 4.12 is that we can enumerate those programs that halt, and whenever the nth program to halt does so, we then coarsen all samples of size n up to accuracy 2^{-s}, where s is the size of the program. Consequently, for each desired degree of accuracy, we eventually obtain answers that are never coarsened beyond that accuracy. On the other hand, knowing how many samples are needed for a given accuracy allows us to determine a point past which we never again coarsen to a given level. This then lets us deduce when a given initial segment of the halting set has stabilized.

Definition 4.11. For $M \in \mathbb{N}$, let $\mathbb{D}_M$ be the collection of Borel probability distributions $\mathcal{D}$ over $(\mathbb{R}\setminus\mathbb{R}_c)\times\{0,1\}$ such that

(i) $L_{\mathcal{D}}(h) = 0$ for some element h of $\mathfrak{H}_{\text{step}}$, and
(ii) $\mathcal{D}$ is absolutely continuous (with respect to Lebesgue measure) and has a probability density function bounded by M.

Theorem 4.12. *For each $M \in \mathbb{N}$, there is a learner A on $\mathcal{X} = \mathbb{R}\setminus\mathbb{R}_c$ and $\mathcal{Y} = \{0,1\}$ such that*

- *A is computable in the realizable case with respect to $\mathfrak{H}^\dagger_{\text{step}}$,*
- *A is a PAC learner for $\mathfrak{H}^\dagger_{\text{step}}$ over $\mathbb{D}_M$, and*
- *$\emptyset'$ is computable from any sample function for A (as a learner for $\mathfrak{H}^\dagger_{\text{step}}$ over $\mathbb{D}_M$).*

Proof. Define $\alpha\colon \mathbb{Q}\times\mathbb{N}\to\mathbb{Q}$ by $\alpha(q,\ell) = \lfloor 2^\ell q\rfloor/2^\ell$, and let $c\colon (\mathcal{X}\times\mathcal{Y})^{<\omega}\to\mathbb{Q}$ be such that $c(S)$ is the rational q of least index attaining zero empirical error on S if one exists, and 0 otherwise. Hereafter, we will additionally demand that the computable enumeration of $\mathbb{Q}$ employed by c be one which enumerates ½ first. Define $c^*\colon (\mathcal{X}\times\mathcal{Y})^{<\omega}\times\mathbb{N}\to\mathbb{Q}$ by $c^*(S,n) = \alpha(c(S),n)$, i.e., the previous decision stump learner discretized to accuracy 2^{-n}.

Let $(e_k)_{k\in\mathbb{N}}$ be a computable enumeration without repetition of all $e\in\mathbb{N}$ for which $\{e\}(0)\!\downarrow$. For $S\in(\mathcal{X}\times\mathcal{Y})^{<\omega}$, write $\operatorname{len}(S)$ for its length. Define $A\colon (\mathcal{X}\times\mathcal{Y})^{<\omega}\times\mathcal{X}\to\mathcal{Y}$ by $A(S,x) = h_{c^*(S,e_{\operatorname{len}(S)})}(x)$. In other words, we discretize the decision stump algorithm to accuracy $2^{-e_{\operatorname{len}(S)}}$. Note that because $\lim_k e_k = \infty$, we can find arbitrarily good approximations as we increase the sample size, even if (as we will show) we cannot compute how large such samples must be.

Note that A is computable in the realizable case. Further, for every $r\in\mathbb{N}$ there is an $i\in\mathbb{N}$ such that $e_{r^*} > i$ for all $r^*\geq r$. Then for every integer $\ell > 0$, there is an $n\in\mathbb{N}$ such that whenever $\operatorname{len}(S) > n$, the set

$$U = \left\{x \;:\; \mathfrak{H}_{\text{step}}(\mathfrak{U}_{\text{step}}(S),x)\neq A(S,x)\right\}$$

is contained in an interval of length $2^{-\ell}$. A is thus a PAC learner for $\mathfrak{H}^\dagger_{\text{step}}$ over $\mathbb{D}_M$, as the loss incurred by A on U is bounded uniformly over $\mathbb{D}_M$ by $2^{-\ell}\cdot M$.

Let $m(\epsilon,\delta)$ be a sample function for A and consider $n\in\mathbb{N}$. We will compute the function $\emptyset'$ restricted to the set $[n] = \{0,\dots,n-1\}$. Fix any rational $\delta\in(0,1)$, and set $m_n = m(2^{-(n+2)},\delta)$. Suppose there is some $i > m_n$ such that $e_i < n$. Then given a sample S of size i, the function $A(S,\,\cdot\,)$ will discretize $c(S)$ to an accuracy below 2^{-n}. This would cause A to incur a true loss of at least 2^{-n} on the distribution which is uniform on features in $[0,1]$ and takes labels according to $\mathbb{1}_{>1/3}$, as $\alpha(1/3,k)\leq 1/3 - 2^{-(k+2)}$, a contradiction. Hence $i\leq m_n$ whenever $e_i < n$. We can therefore determine membership in $\{e_k \;:\; k\in\mathbb{N}\}\cap[n]$, and hence can compute $\emptyset'$ restricted to $[n]$. □

5 UPPER BOUNDS ON THE COMPUTABILITY OF LEARNERS

We now study upper bounds on the computability of learners, for a hypothesis class with a computable presentation. For the rest of the paper, we remain in the setting of binary classification, i.e., $\mathcal{Y} = \{0,1\}$.

For any computable presentation of a hypothesis class, we establish a concrete upper bound, depending only on the index space, for how computable some ERM must be.

Theorem 5.1. *Suppose $\mathfrak{H}\colon \mathcal{I}\times\mathcal{X}\to\mathcal{Y}$ is a computable presentation of a hypothesis class. Then there is a proper learner for $\mathfrak{H}$ that is strongly Weihrauch reducible to $\lim_{\mathcal{I}}$, and is such that the learner it induces is an ERM for $\mathfrak{H}^\dagger$. In particular, there is an ERM for $\mathfrak{H}^\dagger$ that is strongly Weihrauch reducible to $\lim_{\mathcal{I}}$.*

Proof. Fix a sample $S = (x_i,y_i)_{i\in[n]}$. To invoke $\lim_{\mathcal{I}}$, we introduce a procedure for approximating an input $z = (z_i)_{i\in\mathbb{N}}\in\mathcal{I}^{\mathbb{N}}$. In particular, we approximate z using the sequence $(z_k)_{k\in\mathbb{N}}$, with each $z_k\in\mathcal{I}^{\mathbb{N}}$ taking the form

$$z_k = \left(z_k^1,\dots,z_k^{k-1},z_k^k,z_k^k,\dots\right),$$

i.e., constant after the $(k-1)$th term.

z_k^j is computed as follows, for $j\in[k]$:

1. Take balls around the x_i and around the first j ideal points of $\mathcal{I}$, all of radius 2^{-k}. In addition, calculate which value is taken by $y_i\in\{0,1\}$.
2. For each of the first j ideal points of $\mathcal{I}$, use $\mathfrak{H}$ to determine whether the balls around the x_i and the ideal point suffice to calculate a well-defined empirical error with respect to S.
3. If none of the first j ideal points induce a well-defined empirical error, set z_k^j to be the first ideal point of $\mathcal{I}$. Otherwise, set z_k^j to be the first ideal point which attains minimal empirical error among the first j ideal points.

As $\mathfrak{H}$ is continuous, and as there are only finitely many possible empirical errors, if $w\in\mathcal{I}$ is such that $\widetilde{\mathfrak{H}}(w)$ has minimal empirical error with respect to S, then there must be an open ball around w where all elements of the ball give rise to a function with the same minimal empirical error (with respect to S). In particular, there must be an ideal point c such that $\widetilde{\mathfrak{H}}(c)$ has minimal empirical error with respect to S. Therefore $z = (z_j^j)_{j\in\mathbb{N}}$ converges to the ideal point with minimal index among those that give rise to minimal empirical error with respect to S. Calling $\lim_{\mathcal{I}}$ on z thus is a proper learner whose induced learner for $\mathfrak{H}^\dagger$ is an ERM, as desired. □

When comparing the relative computational strength of two maps f and g, the notion of g being "more complex" than f can be intuitively thought of as the statement that one can compute f when given access to g. This is made precise using the formalism of strong Weihrauch reducibility, in which a single application of f must be computed using a single application of g (possibly along with some uniform pre- and post-processing). In some situations, it is useful to be able to use multiple applications of g when computing an application of f; this capability is formalized using the notion of *parallelization* (see Definition 3.16). For those familiar with classical computability theory, working with strong Weihrauch reductions to the parallelization more closely resembles Turing reductions, as opposed to m-reductions.

When considering learners on continuum-sized metric spaces, it is important to study the parallelization of the learner, and not just the learner itself, for reasons we now describe. Because we have chosen for a learner to be a map from $(\mathcal{X}\times\mathcal{Y})^{<\omega}\times\mathcal{X}$ to $\{0,1\}$ (as opposed to a map from $(\mathcal{X}\times\mathcal{Y})^{<\omega}$ to $\{0,1\}^{\mathcal{X}}$), a single application of a learner can only return a single bit of information about its input. In contrast, $\lim_{\mathbb{N}^{\mathbb{N}}}$ is a map from $\mathbb{N}^{\mathbb{N}}$ to $\mathbb{N}^{\mathbb{N}}$ for which a single application contains countably many bits of information. As such, when comparing a learner to $\lim_{\mathbb{N}^{\mathbb{N}}}$, it is somewhat artificial to allow only a single application of the learner. We can overcome this

obstacle by instead considering the parallelization of the learner, i.e., by allowing ourselves to simultaneously ask countably many questions of the learner, rather than a single one.

In considering the parallelization, the upper bound may increase from $\lim_{\mathcal{I}}$ to $\lim_{\mathbb{N}^{\mathbb{N}}}$, but no further.

Corollary 5.2. *Suppose $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \mathcal{Y}$ is a computable presentation of a hypothesis class. Then there is a proper learner for $\mathfrak{H}$ whose parallelization is strongly Weihrauch reducible to $\lim_{\mathbb{N}^{\mathbb{N}}}$, and is such that the learner it induces is an ERM for $\mathfrak{H}^\dagger$. In particular, there is an ERM for $\mathfrak{H}^\dagger$ whose parallelization is strongly Weihrauch reducible to $\lim_{\mathbb{N}^{\mathbb{N}}}$.*

Proof. Let $\mathfrak{A}$ be the proper learner constructed in the proof of Theorem 5.1, which satisfies $\mathfrak{A} \leq_{\mathrm{sW}} \lim_{\mathcal{I}}$. By Corollary 3.14, we have $\lim_{\mathcal{I}} \leq_{\mathrm{sW}} \lim_{\mathbb{N}^{\mathbb{N}}}$, so that $\mathfrak{A} \leq_{\mathrm{sW}} \lim_{\mathbb{N}^{\mathbb{N}}}$.

Observe that strong Weihrauch reductions are preserved under parallelization, and so $\widehat{\mathfrak{A}} \leq_{\mathrm{sW}} \widehat{\lim_{\mathbb{N}^{\mathbb{N}}}}$. Finally, we have $\widehat{\lim_{\mathbb{N}^{\mathbb{N}}}} \equiv_{\mathrm{sW}} \lim_{\mathbb{N}^{\mathbb{N}}}$ by Lemma 3.17. □

In Section 6 we provide matching lower bounds on the parallelization of proper learners.

We now show in Theorem 5.3 that in the setting of Theorem 5.1, there is always an ERM that is computable in the realizable case. Theorem 5.3 can be viewed as a generalization of [3, Theorem 10].

Theorem 5.3. *Suppose $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \mathcal{Y}$ is a computable presentation of a hypothesis class. Then there is a proper learner for $\mathfrak{H}$ that is computable in the realizable case, and is such that its induced learner is an ERM for $\mathfrak{H}^\dagger$ that is computable in the realizable case. In particular, there is an ERM for $\mathfrak{H}^\dagger$ that is computable in the realizable case.*

Proof. Suppose $S \in \Phi_{\mathfrak{H}^\dagger}$. There is some $w \in \mathcal{I}$ such that $\widetilde{\mathfrak{H}}(w)$ has empirical error 0 with respect to S. Because there are only finitely many possible values of the empirical error with respect to S, there must be some open ball B around w such that for all elements $w^* \in B$, the function $\widetilde{\mathfrak{H}}(w^*)$ has empirical error 0 with respect to S. In particular, there must be some ideal point c in this ball. Therefore the algorithm which searches through all ideal points and returns the first to attain an empirical error 0 with respect to S will eventually halt. This algorithm is a proper learner that is computable in the realizable case, and its induced learner is an ERM for $\mathfrak{H}^\dagger$ that is computable in the realizable case. □

Remark 5.4. The algorithms in Theorems 5.1 and 5.3 would have failed had $\mathcal{Y}$ not been computably discrete, in which case verifying that a hypothesis incurs an empirical error of 0 would not be computable. When $\mathcal{Y} = \{0, 1\}$, as in this paper, the predictions of hypotheses on features can be deduced exactly, allowing for precise computation of empirical errors. If $\mathcal{Y} = \mathbb{R}$, in contrast, then predictions of hypotheses h take the form $(q_k - 2^{-k}, q_k + 2^{-k})$ for $q_k \in \mathbb{Q}$ and chosen $k \in \mathbb{N}$, amounting to the information that $h(x) \in (q_k - 2^{-k}, q_k + 2^{-k})$.

Some such intervals allow one to conclude that $h(x) \neq y$, namely when $y \notin (q_k - 2^{-k}, q_k + 2^{-k})$, and thus that h does not attain an empirical error of 0. Yet no such interval allows one to conclude that $h(x) = y$ for even a single example (x, y) if y may take any real value, much less that h attains an empirical error of 0 across an entire sample.

5.1 Applications

We now apply techniques from the proof of Theorem 5.3 to concrete settings where we have more information about the structure of the hypothesis class.

The restricted setting of computability in the realizable case, as in Theorem 5.3, provides a stopping criterion for detecting a hypothesis in $\mathfrak{H}^\dagger$ attaining minimal empirical risk on S, thereby eliminating the need for $\lim_{\mathcal{I}}$. A similar criterion would arise if the size of the restriction of a (computably presented) class $\mathcal{H}$ to a given sample S could be known in advance. In such a case, one could walk through the ideal points of $\mathcal{I}$ as in the proof of Theorem 5.3 until all such behaviors on S are encountered, subsequently returning one which attains the minimal empirical error.

Theorem 5.5. *Suppose $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \mathcal{Y}$ is a computable presentation of a hypothesis class, and that for all finite $U \subseteq \mathcal{X}$, the size of $\{h\restriction_U : h \in \mathfrak{H}^\dagger\}$ can be computed, uniformly in U. Then an ERM learner for $\mathfrak{H}^\dagger$ is computable.*

Proof. Let $n \in \mathbb{N}$. Define $\mathfrak{H}^n\colon \mathcal{I} \times \mathcal{X}^n \to \mathcal{Y}^n$ to be the map where $\mathfrak{H}^n(w, (x_j)_{j\in[n]}) = (\mathfrak{H}(w, x_j))_{j\in[n]}$. Note that this is a continuous function, and hence for all $w \in \mathcal{I}$ and for every $u \in \mathcal{X}^n$ there is an ideal point c such that $\mathfrak{H}^n(w, u) = \mathfrak{H}^n(c, u)$.

Suppose $U \subseteq \mathcal{X}$ is finite. We then have

$$\left|\{h\restriction_U : h \in \mathfrak{H}^\dagger)\}\right| = \left|\left\{\widetilde{\mathfrak{H}}(c)\restriction_U : c \text{ is an ideal point of } \mathcal{I}\right\}\right|.$$

In particular, by searching through all the ideal points of $\mathcal{I}$ we realize all behavior (restricted to U) that occurs in $\widetilde{\mathfrak{H}}$. So, from $\left|\{h\restriction_U : h \in \mathfrak{H}^\dagger\}\right|$ we can compute ideal points of $\mathcal{I}$ realizing all such behavior. From this it is straightforward to choose an ideal point which minimizes the empirical error on any sample $(x_i, y_i)_{i\in[m]}$ where $\{x_i : i \in [m]\} = U$. □

It has been shown in [11] that the computability condition of Theorem 5.5 is enjoyed by maximum classes, i.e., those which achieve the bound of the Sauer–Shelah lemma. We can thus conclude computable PAC learnability for such maximum classes.

Corollary 5.6. *If $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \mathcal{Y}$ is a computable presentation of a hypothesis class, and $\mathfrak{H}^\dagger$ is a maximum class of finite VC dimension, then it is computably PAC learnable.*

6 LOWER BOUNDS ON THE COMPUTABILITY OF LEARNERS

We now provide a converse to Corollary 5.2 by establishing a lower bound on the computability of a proper PAC learner and a sample function for it.

Let $\mathfrak{B}$ be the proper learner produced by Corollary 5.2. Because $\mathfrak{B}$ induces an ERM, it is a proper PAC learner with computable sample function r (by Theorem 3.24). Hence $(\widehat{\mathfrak{B}}, r) \leq_{\mathrm{sW}} \lim_{\mathbb{N}^{\mathbb{N}}}$. Theorem 6.1 (in the case where sample functions are computable) provides a converse.

Theorem 6.1. *There is a hypothesis class that is PAC learnable but which admits a computable presentation $\mathfrak{H}$ such that $\lim_{\mathbb{N}^{\mathbb{N}}} \leq_{\mathrm{sW}} (\widehat{\mathfrak{A}}, m)$ whenever $\mathfrak{A}$ is a proper PAC learner for $\mathfrak{H}$ and m is a sample function for the PAC learner for $\mathfrak{H}^\dagger$ that $\mathfrak{A}$ induces.*

Proof. Let $\mathcal{X}$ be the product of computable metric spaces $\mathbb{N}$ and $\mathbb{N}^{\mathbb{N}}$, and define the index space $\mathcal{I}$ to be

$$\left\{(e,z) \in \mathbb{N} \times \mathbb{N}^{\mathbb{N}} \;:\; \{e\}^z(0)\!\downarrow\right\}$$

with distance inherited from $\mathcal{X}$ and ideal points of the form (e,z) where z has only finitely many nonzero values, ordered by when the respective programs with oracles halt on input 0.

Given $(e_0,z_0),(e_1,z_1) \in \mathcal{X}$, we write $(e_0,z_0) \sim (e_1,z_1)$ when (a) $e_0 = e_1$ and (b) $\{e_0\}^{z_0}(0)\downarrow$ if and only if $\{e_1\}^{z_1}(0)\downarrow$, with program e_0 with oracle z_0 taking the same number of steps to halt on input 0 as does e_1 with oracle z_1 (when they both halt). Define $\mathfrak{H}\colon \mathcal{I} \times \mathcal{X} \to \{0,1\}$ by

$$\mathfrak{H}\big((e_0,z_0),(e_1,z_1)\big) = \begin{cases} 1 & (e_0,z_0) \sim (e_1,z_1); \\ 0 & \text{otherwise.} \end{cases}$$

Note that $\mathfrak{H}$ is computable because $\{e_0\}^{z_0}(0)\downarrow$ for every $(e_0,z_0) \in \mathcal{I}$.

First we show that $\mathfrak{H}^\dagger$ shatters no set of size 2, so that it has VC dimension 1 and hence is PAC learnable by Theorem 3.23. Let $(e_0,z_0),(e_1,z_1) \in \mathcal{X}$ be distinct. If there exists an $h \in \mathfrak{H}^\dagger$ with $h(e_0,z_0) = 1$ and $h(e_1,z_1) = 0$, then there is some k such that the program e_0 with oracle z_0 halts on input 0 in exactly k steps but either $e_0 \neq e_1$ or the program e_1 with oracle z_1 does not halt on input 0 in exactly k steps. But then there is no $g \in \mathfrak{H}^\dagger$ with $g(e_0,z_0) = 1$ and $g(e_1,z_1) = 1$. Therefore $\mathfrak{H}^\dagger$ does not shatter the set $\{(e_0,z_0),(e_1,z_1)\}$.

Now suppose that $\mathfrak{A}\colon (\mathcal{X} \times \mathcal{Y})^{<\omega} \to \mathcal{I}$ is a proper PAC learner for $\mathfrak{H}$, let A be the induced PAC learner for $\mathfrak{H}^\dagger$, and let m be a sample function for A (as a PAC learner for $\mathfrak{H}^\dagger$). We will show that $\mathrm{J} \leq_{\mathrm{sW}} (\widehat{\mathfrak{A}}, m)$. Then by Lemma 3.15, we will have $\lim_{\mathbb{N}^{\mathbb{N}}} \leq_{\mathrm{sW}} (\widehat{\mathfrak{A}}, m)$.

Let $z \in \mathbb{N}^{\mathbb{N}}$. We aim to uniformly compute z' using $\mathfrak{A}$, m, and z. First we preprocess. Calculate $k = m(\epsilon,\delta)$ for any choice of $\epsilon,\delta \in (0,1)$ and construct the sequence $S_{e,z} = \big(((e,z),1)^k\big)_{e\in\mathbb{N}}$. Then, apply $\widehat{A}$ to obtain a sequence $(\ell_e, s_e)_{e\in\mathbb{N}}$.

Now consider the measure $D_{(e,z)}$ which places a pointmass on $((e,z),1)$. Because A is a PAC learner, we have

$$\Pr_{S\sim D^k_{(e,z)}}\left(\left|L_{D_{(e,z)}}(A(S)) - \min_{w\in\mathcal{I}} L_{D_{(e,z)}}(\widetilde{\mathfrak{H}}(w))\right| < \epsilon\right) > 1-\delta.$$

Therefore, as $D_{(e,z)}$ is a pointmass, we have

$$\left|L_{D_{(e,z)}}(A(S_{e,z})) - \min_{w\in\mathcal{I}} L_{D_{(e,z)}}(\widetilde{\mathfrak{H}}(w))\right| < \epsilon.$$

Again because A is a PAC learner and $D_{(e,z)}$ is atomic, we have an equivalence between the following statements:

(1) $A(S_{e,z})(e,z) = 1$.
(2) $L_{D_{(e,z)}}(A(S_{e,z})) = 0$.
(3) $L_{D_{(e,z)}}(\widetilde{\mathfrak{H}}(w)) = 0$ for some $w \in \mathcal{I}$.
(4) $\mathfrak{H}(w,(e,z)) = 1$ for some $w \in \mathcal{I}$.

In particular, (3) $\Rightarrow$ (2) because $D^k_{(e,z)}$ concentrates mass on $S_{(e,z)}$, so otherwise A would be guaranteed to incur a loss of $1 > \epsilon$ when trained on samples drawn from $D^k_{(e,z)}$, contradicting the PAC condition on $m(\epsilon,\delta)$.

Now note that if $\{e\}^z(0)\downarrow$, then there is a $w = (e,z) \in \mathcal{I}$ such that $\mathfrak{H}(w,(e,z)) = 1$; by the previous equivalence, this implies that $A(S_{e,z})(e,z) = 1$.

We are now equipped to post-process and calculate $z'(n)$. If $n \neq \ell_n$, then $A(S_{n,z})(n,z) = 0$ and, via $\neg(1) \Rightarrow \neg(4)$ in the equivalence, $\{n\}^z(0)\uparrow$, meaning $z'(n) = 0$.

Otherwise, $n = \ell_n$. First compute $\{n\}^{s_n}(0)$. This computation is guaranteed to halt, by definition of $\mathcal{I}$ and the fact that $\mathfrak{A}$ is a proper learner. Let t be the number of steps it took to halt. Next run $\{n\}^z$ on input 0 for t steps. If it halts within t steps, then $\{n\}^z(0)\downarrow$ and so $z'(n) = 1$. If $\{n\}^z$ has not halted on input 0 within t steps, then $A(S_{n,z})(n,z) = 0$ and the equivalence again implies that $\{n\}^z(0)\uparrow$, meaning $z'(n) = 0$. □

Theorem 6.1 establishes that for computably presented hypothesis classes (of finite VC dimension), in general there is no computable procedure for PAC learning the hypothesis class from samples.

ACKNOWLEDGMENTS

The authors would like to thank Caleb Miller for valuable discussion on the topic, particularly in helping refine the notion of computable PAC learning and in describing the computable algorithm for the decision stump.

The authors would also like to thank the anonymous reviewers, whose comments and suggestions improved the presentation.

An extended abstract [2] announcing related results in a different setting was presented at the Eighteenth International Conference on Computability and Complexity in Analysis (July 26–28, 2021), and an earlier version [1] of the present paper was posted on the arXiv in November, 2021.

This material is based upon work supported by the National Science Foundation under grant no. CCF-2106659. Freer's work is funded in part by financial support from the Intel Probabilistic Computing Center.

REFERENCES

[1] Nathanael Ackerman, Julian Asilis, Jieqi Di, Cameron Freer, and Jean-Baptiste Tristan. 2021. On Computable Learning of Continuous Features. *arXiv e-print 2111.14630* (2021). arXiv:2111.14630

[2] Nathanael Ackerman, Julian Asilis, Jieqi Di, Cameron Freer, and Jean-Baptiste Tristan. 2021. On the Computable Learning of Continuous Features. In *Eighteenth International Conference on Computability and Complexity in Analysis.* http://cca-net.de/cca2021/

[3] Sushant Agarwal, Nivasini Ananthakrishnan, Shai Ben-David, Tosca Lechner, and Ruth Urner. 2020. On Learnability with Computable Learners. In *Proceedings of the 31st International Conference on Algorithmic Learning Theory (ALT) (PMLR, Vol. 117).* 48–60. http://proceedings.mlr.press/v117/agarwal20b.html

[4] Achilles A. Beros. 2014. Learning theory in the arithmetic hierarchy. *Journal of Symbolic Logic* 79, 3 (2014), 908–927. https://doi.org/10.1017/jsl.2014.23

[5] Anselm Blumer, A. Ehrenfeucht, David Haussler, and Manfred K. Warmuth. 1989. Learnability and the Vapnik–Chervonenkis Dimension. *Journal of the ACM* 36, 4 (1989), 929–965. https://doi.org/10.1145/76359.76371

[6] Vasco Brattka, Guido Gherardi, and Arno Pauly. 2021. Weihrauch Complexity in Computable Analysis. In *Handbook of Computability and Complexity in Analysis*, Vasco Brattka and Peter Hertling (Eds.). Springer, 367–417. https://doi.org/10.1007/978-3-030-59234-9_11

[7] Vasco Brattka, Peter Hertling, and Klaus Weihrauch. 2008. A tutorial on computable analysis. In *New Computational Paradigms.* Springer, 425–491. https://doi.org/10.1007/978-0-387-68546-5_18

[8] Vasco Brattka and Gero Presser. 2003. Computability on subsets of metric spaces. *Theoretical Computer Science* 305, 1-3 (2003), 43–76. https://doi.org/10.1016/S0304-3975(02)00693-X

[9] Wesley Calvert. 2015. PAC learning, VC dimension, and the arithmetic hierarchy. *Archive for Mathematical Logic* 54, 7-8 (2015), 871–883. https://doi.org/10.1007/s00153-015-0445-8

[10] Tonicha Crook, Jay Morgan, Arno Pauly, and Markus Roggenbach. 2021. A Computability Perspective on (Verified) Machine Learning. *arXiv e-print 2102.06585* (2021). arXiv:2102.06585

[11] Sally Floyd and Manfred Warmuth. 1995. Sample compression, learnability, and the Vapnik–Chervonenkis dimension. *Machine Learning* 21, 3 (1995), 269–304. https://doi.org/10.1023/A:1022660318680

[12] E. Mark Gold. 1967. Language identification in the limit. *Information and Control* 10, 5 (1967), 447–474. https://doi.org/10.1016/S0019-9958(67)91165-5

[13] Mathieu Hoyrup and Cristóbal Rojas. 2009. Computability of probability measures and Martin-Löf randomness over metric spaces. *Information and Computation* 207, 7 (2009), 830–847. https://doi.org/10.1016/j.ic.2008.12.009

[14] Marcus Schaefer. 1999. Deciding the Vapnik–Červonenkis Dimension is Σ_3^p-complete. *J. Comput. System Sci.* 58, 1 (1999), 177–182.

[15] Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press. https://doi.org/10.1017/CBO9781107298019

[16] David Soloveichik. 2008. Statistical Learning of Arbitrary Computable Classifiers. *arXiv e-print 0806.3537* (2008). arXiv:0806.3537

[17] Tom F. Sterkenburg. 2022. On characterizations of learnability with computable learners. *arXiv e-print 2202.05041* (2022). arXiv:2202.05041

[18] L. G. Valiant. 1984. A Theory of the Learnable. *Communications of the ACM* 27, 11 (1984), 1134–1142. https://doi.org/10.1145/1968.1972

[19] V. N. Vapnik and A. Ya. Chervonenkis. 1971. On the Uniform Convergence of Relative Frequencies of Events to their Probabilities. *Theory of Probability and its Applications* 16, 2 (1971), 264–280. https://doi.org/10.1137/1116025

[20] S. Wehner. 1990. *Zur Komplexität des Numerierens*. PhD thesis. Universität Karlsruhe.

[21] Klaus Weihrauch. 2000. *Computable analysis: An introduction*. Springer. https://doi.org/10.1007/978-3-642-56999-9