TEACHERS, LEARNERS AND ORACLES

ACHILLES A. BEROS AND COLIN DE LA HIGUERA

ABSTRACT. We exhibit a family of computably enumerable sets which can be learned within polynomial resource bounds given access only to a teacher, but which requires exponential resources to be learned given access only to a membership oracle. In general, we compare the families that can be learned with and without teachers and oracles for four measures of efficient learning.

1. Introduction

In this paper, we address the question of whether or not the presence of a teacher as a computational aide improves learning. A teacher is a computable machine that receives data and selects a subset of the data. In the models we consider, a teacher receives an enumeration for a target and passes its data selection to the learner – the learner does not have access to the original data. The first natural question is if there are families that are learnable with a teacher, but not learnable without. As will be obvious from the definitions presented in the next section, the answer is no: the learner can always perform an internal simulation of the learner-teacher interaction and output the result. The second question is whether a teacher can improve efficiency. For teacher models of learning, only the computational activity of the learner counts against the efficiency bound; the computational activity of the teacher is not counted. Heuristically, the question is whether there is benefit to pre-processing data. We will prove there can be an exponential improvement in efficiency. In fact, there are situations where access to a teacher is better than access to a membership oracle about the target.

Various forms of and questions related to teaching have arisen in learning theory over the last few decades. Work on the complexity of teaching families has given rise to the classical teaching dimension [6] and more recently the recursive teaching dimension ([12] and [3]). In [3], Zilles et al. establish deep and interesting connections between recursive teaching dimension, Vapnik-Chervonenkis dimension and sample compression schemes (see [4] for more about sample compression). Query learning has been a central topic in learning theory for even longer than teaching. Numerous papers have been written both on the abilities of machines equipped with oracles to learn ([9], [1] and [2]) and on the properties of oracles that allow learning of certain target families ([11], [7], [5] and [8]).

We add to the body of research on teaching and query learning by comparing the efficiency of the two learning modes and by highlighting a recursion theory technique for constructing examples. In particular, we use the recursion theorem and the s-m-n theorem as a way to build sets that have computational “backdoors” in the sense that they encode a great deal of complexity, but that complexity can also be described easily given a small piece of non-uniform knowledge about the target. We recommend Soare’s classic [10] as a reference for the recursion theoretic concepts used in this paper.

Key words and phrases. Learning, Efficiency, Oracle, Teacher, Recusion Theory.
2. Background

We will examine variants of Gold-style text learning of effectively describable sets of natural numbers. In particular, the target objects will be computably enumerable sets.

**Definition 2.1.** A set, $S$, is **computably enumerable (c.e.)** if there is a partial computable function, $f$, such that $S = \text{dom}(f)$. A sequence of sets, $\{A_n\}_{n \in \mathbb{N}}$ is called uniformly computably enumerable (u.c.e.) if the set $\{(a, i) : a \in A_i\}$ is c.e.

We now remind the reader of some standard notation and concepts as well as introducing some notation specific to this paper.

1. $\phi$ denotes an acceptable universal Turing machine and hence, a partial computable function. $\phi_e(x)$ is the state or value of the function described by the program coded by $e \in \mathbb{N}$ after $s$ computation stages on input $x$. If the program execution has terminated, we write $\phi_e(x) \downarrow$, otherwise we write $\phi_e(x) \uparrow$.

2. $W_e$ is the c.e. set coded by the program $e$ as the domain of $\phi_e$. $\{W_e\}_{e \in \mathbb{N}}$ is a u.c.e. sequence of sets and enumerates all the c.e. sets. We write $E$ for the set of all c.e. sets.

3. For $n \in \mathbb{N}$, $\langle x_0, x_1, \ldots, x_n \rangle : \mathbb{N}^{n+1} \rightarrow \mathbb{N}$ is a computable encoding function such that $x_i < \langle x_0, x_1, \ldots, x_n \rangle$ for all $i \leq n$. We define $A \otimes B = \{\langle a, b \rangle : (a \in A) \land (b \in B)\}$. We use $\otimes$ to partition $\mathbb{N}$ into an infinite number of infinite computable sets, $\mathbb{N} \otimes \{0\}, \mathbb{N} \otimes \{1\}, \ldots$. Sets of this form are known as columns, whereby $\mathbb{N} \otimes \{i\}$ is the $i^{th}$ column of $\mathbb{N}$. As a shorthand, we will represent the $i^{th}$-column of $\mathbb{N}$ with the symbol $C_i$ and the $i^{th}$-column of $A \subseteq \mathbb{N}$ by $C_i(A)$. Associated with $C_i$, we define $c_i$ to be a computable function such that $W_{c_i(x)} = W_x \cap C_i$.

4. We write $(x_0, x_1, \ldots, x_n)$ to denote the ordered tuple of elements (as opposed to the encoding of the ordered tuple, $(x_0, x_1, \ldots, x_n))$.

5. $\text{signedInt} : \mathbb{N} \rightarrow \mathbb{Z}$ is the computable bijection such that $\text{signedInt}(2n) = n$ and $\text{signedInt}(2n + 1) = -(n + 1)$.

6. If $a$ is a string or natural number, then $a^i$ denotes the string which consists of $a$ repeated $i$ times.

7. For function composition we use the notation $f \circ g$ where $(f \circ g)(x) = f(g(x))$.

8. If $\sigma = a_0 \cdots a_n$ is a string, then $|\sigma| = n + 1$ is the length of the string, $\sigma(k) = a_k$ and content($\sigma$) = $|\sigma| \downarrow (k < |\sigma|)$.

9. An enumeration of a set $A$ is an infinite sequence of elements of $A \cup \{\ast\}$ such that every element of $A$ appears in the sequence at least once. We regard an enumeration as a stream of bits with markers between individual elements.

10. A learning machine (or learner) is a partial computable function that receives a string as input, may have access to oracle queries and outputs a natural number that is interpreted as a code for a set. The outputs are called hypotheses and the sequence of hypotheses produced by a learner on initial segments of an enumeration is called the hypothesis stream. When measuring efficiency, we allow a learner to skip an element of an enumeration for some fixed computational cost.

11. Given an interval $[0, n]$, where $n$ is unknown, but bounded by $a^n$, $n$ can be determined with $(m + 1)^{2n+1}$ or fewer oracle queries using the following algorithm. First, determine the least $k_0$ such that $a^{n+1} \notin [0, n]$. We will obtain $k_0$ after at most $m + 1$ queries. Next, we repeat the process to determine the least $k_1$ such that $a^{k_0 + a^{k_1+1}} \notin [a^{k_0}, n]$. By iterating this process at most $a + 1$ times we find $n$. We call this an exponential query search algorithm.

We will make extensive use of two very well-known theorems: the s-m-n theorem and the recursion theorem.
Theorem 2.2 (s-m-n Theorem). There exists a computable function, \( s \), such that for every partial computable function \( f = \phi_e \) we have that \( f(x, y) = \phi_{s(p,x)}(y) \), where \( x = (x_0, \ldots, x_n) \) and \( y = (y_0, \ldots, y_m) \) are tuples of natural numbers.

Theorem 2.3 (Recursion Theorem). If \( f \) is total computable function, then there are infinitely many \( e \) such that \( \phi_e = \phi_{f(e)} \).

Both theorems have powerful generalizations, but we only need the following for our purposes.

Theorem 2.4 (Smullyan’s Double Recursion Theorem). If \( f \) and \( g \) are total computable functions, then there are \( a \) and \( b \) such that \( \phi_a = \phi_{f(a,b)} \) and \( \phi_b = \phi_{g(a,b)} \).

We will consider learning models using combinations of three different data sources: enumeration, oracle and teacher. All of the models we consider are forms of TxtEx-learning, or learning in the limit. We begin with the definition of this fundamental learning model.

Definition 2.5. Let \( M \) be a computable learning machine.

1. \( M \) TxtEx-identifies an enumeration (text) \( \{a_n\}_{n \in \mathbb{N}} \) if

   \[
   (\exists i)(\forall j)(M(a_0 \ldots a_{j+1}) = M(a_0 \ldots a_i) \land \text{content}(M(a_0 \ldots a_i) = \{a_k : k \in \mathbb{N}\})
   \]

   If only the first condition above is met, i.e., \( (\exists i)(\forall j)(M(a_0 \ldots a_{j+1}) = M(a_0 \ldots a_i)) \), then we say that \( M \) has converged on the enumeration \( \{a_n\}_{n \in \mathbb{N}} \).

2. \( M \) TxtEx-learns \( A \subseteq \mathbb{N} \) if \( M \)TxtEx-identifies every enumeration of \( A \).

3. \( M \) TxtEx-learns \( \mathcal{F} \subseteq \mathcal{P}(\mathbb{N}) \) if \( M \) TXTEx-learns every member of \( \mathcal{F} \).

All of the models we examine in this paper are variants of TxtEx-learning. The parameters we will vary are linked to sources of information and the measurement of efficiency. We state definitions of these variants starting from an arbitrary learning model.

Definition 2.6. Let L-learning be an arbitrary learning model.

1. We say that \( \mathcal{F} \) is \( L \)-learnable with a membership oracle (denoted L[O]-learnable) if there is a learning machine, \( M \), that L-learns \( \mathcal{F} \) and has access to a membership oracle for the target it is learning. As membership oracles are the only oracles we will consider, we often simply refer to a membership oracle as an oracle.

2. A function \( T : 2^{\mathbb{N}} \to 2^{\mathbb{N}} \) is a teacher if it is a computable function, \( T(\sigma) \) is a prefix of \( T(\tau) \) whenever \( \sigma \) is a prefix of \( \tau \), and \( T(\sigma) \subseteq \text{content}(\sigma) \). We say that \( \mathcal{F} \) is \( L \)-learnable with a teacher (denoted L[T]-learnable) if there is a learner-teacher pair \( (M, T) \) such that \( M \) L-identifies every enumeration of the form \( T \circ f \), where \( f \) enumerates a member of \( \mathcal{F} \).

3. We say that \( \mathcal{F} \) is \( L \)-learnable with a teacher and a membership oracle (denoted L[T,O]-learnable) if there is a learner-teacher pair, \( (M, T) \), such that \( M \) has access to a membership oracle, \( T \) has access to the query responses \( M \) receives, and \( M \) L-identifies every enumeration of the form \( T \circ f \), where \( f \) enumerates a member of \( \mathcal{F} \).

As is clear from the definition, the teacher serves to pre-process the text input before passing the elements deemed important to the learner. In the subsequent sections, we will consider the different combinations of teacher and oracle with certain variants of TxtEx-learning.

When defining efficiency notions for learning, the first natural notion is that of polynomial run-time: the learner must converge within \( p(\epsilon) \) computation steps, where \( p \) is a
polynomial and e is a code for the target. There are two problems with this definition. First, apart from trivial cases, any learning process can be delayed arbitrarily by using an enumeration that repeats a single element of the target set. Second, if a learning machine has produced an encoding of the target, but has failed to do so in polynomial run-time, a suitably larger and equivalent encoding can be chosen instead so that the run-time is appropriately bounded. Thus, we choose to restrict our attention to a limited class of target families. We associate with each member of the target family an encoding which we regard as the “reasonable” encoding and we measure efficiency with respect to this chosen encoding.

Definition 2.7. An indexed target family is a pair \((\text{ind}_F, \mathcal{F})\) where \(\mathcal{F}\) is a family of c.e. sets and \(\text{ind}_F\) is a function whose domain includes \(\mathcal{F}\), whose co-domain is \(\mathbb{N}\) and such that for \(F \in \mathcal{F}\), \(\text{ind}_F(F) \in F\).

By restricting our attention to indexed target families, we have a well-defined concept of polynomial bounds in the size of the target, thereby addressing the second problem. In the absence of an oracle or teacher the first problem remains. Nevertheless, we include polynomial run-time among the notions of e.

We will address four measures of learning efficiency: Polynomial run-time, polynomial size dataset, polynomial size characteristic sample, and polynomial mind-changes.

3. Polynomial Run-Time

Definition 3.1. An indexed target family, \((\text{ind}_F, \mathcal{F})\), is polynomial run-time learnable (PRT-learnable) if there is a machine \(M\) and a polynomial \(p\) such that for every enumeration \(f\) of \(F \in \mathcal{F}\), the learner \(M\) converges to a correct hypothesis on \(f\) in fewer than \(p(\text{ind}_F(F))\) computation steps. If an oracle is accessed, oracle use must also be bounded by \(p(\text{ind}_F(F))\). We use PRT to denote the set of all PRT-learnable indexed target families.

We will apply Definition 2.6 to Definition 3.1 to obtain, for example, PRT\([T]\)-learning and PRT\([T]\), the PRT\([T]\)-learnable indexed target families.

Proposition 3.2 demonstrates that PRT-learnability is much too restrictive in the absence of an oracle or teacher. In particular, for all but a very limited collection of families, there is no index function that renders the family learnable.

Proposition 3.2. Let \((\text{ind}_F, \mathcal{F})\) be an indexed target family. If there are \(A, B \in \mathcal{F}\) such that \(A \not= B\) and \(A \cap B \not= \emptyset\), then \((\text{ind}_F, \mathcal{F})\) is not PRT-learnable.

Proof. Let \(A, B\) and \((\text{ind}_F, \mathcal{F})\) be as in the statement and let \(M\) be an arbitrary learning machine. Also, let \(a = \text{ind}_F(A), b = \text{ind}_F(B)\) and \(x \in A \cap B\). Define \(f_A\) to be an enumeration of \(A\) that begins with \(x^{i+1}\) and \(f_B\) be an enumeration of \(B\) that begins with \(x^{i+1}\). If \(M\) PRT-identifies \(f_A\), then \(M(\sigma)\) must be a code for \(A\) for any \(\sigma = x^{i+1}\) for \(i \geq 0\). Similarly, if \(M\) PRT-identifies \(f_B\), then \(M(\sigma)\) must be a code for \(B\) for any \(\sigma = x^{i+1}\) for \(i \geq 0\). Thus, no machine can PRT identify both \(f_A\) and \(f_B\) and \((\text{ind}_F, \mathcal{F})\) is not PRT-learnable.

On the other hand, there are many non-trivial indexed target families which are PRT\([O]\)-, PRT\([T]\)- or PRT\([T,O]\)-learnable.

Example 3.3. Define \(\text{ind}_{\min} : \mathcal{P}(\mathbb{N}) \to \mathbb{N}\) and \(\text{ind}_{\max} : \mathcal{P}(\mathbb{N}) \to \mathbb{N}\) by \(\text{ind}_{\min}(F) = \min(F)\) and \(\text{ind}_{\max}(F) = \max(F)\), if \(F\) is finite and \(\text{ind}_{\max}(F) = 0\) otherwise. The following indexed
target families are \(\text{PRT}[O]\)-learnable: \(\text{(ind}_{\text{min}}, \{[n, \infty) : n \in \mathbb{N}\})\), \(\text{(ind}_{\text{max}}, \{[m, n] : m, n \in \mathbb{N}\})\) and \(\text{(ind}_{\text{max}}, \{F : F \subseteq \mathbb{N} \land |F| = k\})\) for any \(k \in \mathbb{N}\).

**Example 3.4.** The indexed target families in Example 3.3 are also \(\text{PRT}[T]\)-learnable. For example, consider \(\mathcal{F} = \{F : F \subseteq \mathbb{N} \land |F| = n\}\), for some fixed \(n\), and \(\text{ind}_{\mathcal{F}}(F) = \max(F)\). Define a teacher \(T\) such that \(T(a_0 \cdots a_k) = a_k\) if \(a_k \notin \{a_0, \ldots, a_{k-1}\}\) and outputs nothing otherwise. Define a learner \(M\) that waits until it has received \(n\) distinct numbers from \(T\) and then outputs a code for the finite set that consists of those distinct numbers. \((M, T)\) \(\text{PRT}[T]\)-learns \((\text{ind}_{\mathcal{F}}, \mathcal{F})\).

**Proposition 3.5.** \(\text{PRT} \subset \text{PRT}[O] \subseteq \text{PRT}[T, O]\) and \(\text{PRT} \subset \text{PRT}[T] \subseteq \text{PRT}[T, O]\).

**Proof.** Let \(\text{ind}(F) = \max(F)\) and \(\mathcal{F} = \{F : F \subseteq \mathbb{N} \land |F| = 2\}\). As observed in Examples 3.3 and 3.4, \((\text{ind}, \mathcal{F})\) is \(\text{PRT}[O]\)-learnable and \(\text{PRT}[T]\)-learnable, but by Proposition 3.2, not only is \((\text{ind}, \mathcal{F})\) not \(\text{PRT}\)-learnable, there is no index function that renders \(\mathcal{F}\) \(\text{PRT}\)-learnable. Thus, \(\text{PRT} \subset \text{PRT}[O] \cap \text{PRT}[T]\). The other containments follow from the definitions.

We now produce indexed target families that distinguish \(\text{PRT}[T]\)-learning from \(\text{PRT}[O]\)-learning and \(\text{PRT}[O,T]\)-learning from both \(\text{PRT}[O]\) and \(\text{PRT}[T]\)-learning. In order to prove that all of these distinctions are non-trivial, we introduce the concept of marked self-description.

**3.1. Marked Self-Describing Sets.** Including self-description in an object is an encoding technique on which many important learning theory examples are based. Examples of self-description include the self-describing sets \(\text{SD} = \{A \in \mathcal{E} : W_{\text{min}}(A) = A\}\), and the almost self-describing functions \(\mathcal{ASD} = \{f : \phi_f(z) =^* f\}\). Many variants on the self-description theme have been explored in learning theory and inductive inference.

Our interest is in families that use carefully engineered self-description to calibrate the difficulty in identifying their members. We will construct families whose members are not only self-describing, but also have their self-describing elements marked for ease of identification. We say that such families exhibit **marked self-description**. In particular, we will use encapsulating objects that we call descriptors.

**Definition 3.6.** For finite \(X \subseteq \mathbb{N}\), a **descriptor on the \(i\)-th column** is a finite set \(D = \{(x, c_i, 1, i) : x \in X\} \subseteq C_i(C_1)\) such that

1. \(\sum_{x \in X} \text{SignedInt}(c_x) = 0\)
2. \((\forall X \subseteq X)(\sum_{x \in X} \text{SignedInt}(c_x) \neq 0)\)
3. \(\sum_{x \in X} \text{SignedInt}(x) \geq 0\).

Such a descriptor is said to **describe** the natural number \(n = \sum_{x \in X} \text{SignedInt}(x)\). For \((x, c_i, 1, i) \in D\), we call \(c_i\) the completion index of the element. For \(n \in \mathbb{N}\), we define \(\text{Descriptors}_i(n)\) to be the set of all descriptors on the \(i\)-th column that describe \(n\).

A descriptor can be thought of as a stream of data that includes parity bits to check the integrity of the data stream and where the intended message is the number described by the descriptor. Thus, a machine can decide not only which elements are pieces of the descriptor (packets in the stream), but also decide when the entire descriptor has appeared in the enumeration (all the packets have been received). By using a descriptor to encode the self-description for a set, we make the self-description instantly recognizable upon appearance in the enumeration. For this reason, learning such a self-describing set can be achieved with no mind-changes. In contrast to the degree to which we have made
learning easier, we have potentially made efficient learning harder. By distributing the self-description into a large descriptor, we will create a scenario in which a very large amount of data is required to reach a correct decision. We now proceed to our first result using these tools.

**Lemma 3.7.** Let $\mathcal{M} = \{W_e : W_e$ describes $e\}$ and define $\text{IND}_M$ by

$$\text{IND}_M(D) = \begin{cases} e & \text{if } D \text{ is a descriptor and describes } e, \\ 0 & \text{otherwise.} \end{cases}$$

$\mathcal{M}$ is infinite and the indexed target family $(\text{IND}_M, \mathcal{M})$ is PRT[T]-learnable, but not PRT[O]-learnable. We call this indexed target family the marked self-describing sets and designate it by $\text{MSD}$.

**Proof.** To see that $\mathcal{M}$ is infinite, observe that, by the s-m-n theorem, there is a computable function, $f$, such that $f(x)$ is a code for a descriptor that describes $x$. Thus, by the recursion theorem, there are an infinite number of distinct values, $e$, such that $W_e = W_{f(e)}$. In other words, there are an infinite number of $e$ such that $W_e$ describes $e$.

Next, we show that $(\text{IND}_M, \mathcal{M})$ is PRT[T]-learnable. We define a teacher $T$ as follows. If $\text{content}(\sigma)$ is not a descriptor, $T(\sigma)$ is the empty string. If $D = \text{content}(\sigma)$ describes $n$, then $T(\sigma) = \min(D)$ if $|\sigma| = |\sigma_0| + i$ where $\sigma_0$ is the shortest initial segment of $\sigma$ whose content contains $D$ and $i < n$; if $i \geq n$ then $T(\sigma) = 0$. Let $M$ be a machine that reads the output of $T$ and returns the number of elements in the output of $T$. The teacher-learner pair learns $(\text{IND}_M, \mathcal{M})$ and the run-time of the learner is linear in the index of the target and the number of mind-changes is linear in the index given by $\text{IND}_M$.

We now show that $(\text{IND}_M, \mathcal{M})$ is not PRT[O]-learnable. To prove that $(\text{IND}_M, \mathcal{M})$ is not PRT[O]-learnable, fix a learner $M$ and a polynomial $p$. Consider the situation where $M$ has access to the membership oracle for the singleton $\langle 0, 1, 1, 0 \rangle$ and define a computable function $q$ such that, for $\ell \in \mathbb{N}$, $q(\ell)$ is the greatest number about which $M$ queries the oracle when it receives inputs which are substrings of $\langle 0, 1, 1, 0 \rangle^\ell$. Note that $q$ is an increasing function. Since $q$ is a computable function, we can define two computable functions, $h_0$ and $h_1$, such that for $i \in \{0, 1\}$, $W_{h_i(x)}$ is a descriptor on the $0^i$ column that describes $x$ and whose set of completion indices contains $1$, but no other numbers less than $q(p(x))$, and $W_{h_i(x)} \neq W_{h_j(x)}$. By the recursion theorem, there are $e_0$ and $e_1$ such that $W_{e_0} = W_{h_0(e_0)}$ and $W_{e_1} = W_{h_1(e_1)}$. Clearly $W_{e_0}, W_{e_1} \in \mathcal{M}$. If $M$ PRT[O]-learns $(\text{IND}_M, \mathcal{M})$ with polynomial bound $p$, then it must succeed at identifying $W_{e_0}$ and $W_{e_1}$ within $p(\text{IND}(W_{e_0})) = p(e_0)$ and $p(\text{IND}(W_{e_1})) = p(e_1)$ computation stages, respectively. Let $m = \max(p(\text{IND}(W_{e_0})), p(\text{IND}(W_{e_1})))$. Choose $T_0$ and $T_1$ to be any enumerations of $W_{e_1}$ and $W_{h_1(e_1)}$, respectively, which have $\langle 0, 1, 1, 0 \rangle^{m+1}$ as an initial segment. When trying to identify $T_0$ and $T_1$, the learner must reach its final hypothesis before finding any elements of the target sets, $W_{e_0}$ and $W_{e_1}$, other than $\langle 0, 1, 1, 0 \rangle$. Whatever hypothesis $M$ converges to before completing the $m + 1$ length initial segment of either enumeration cannot code both sets. Thus, $M$ fails to learn at least one of the two sets. Since $M$ and $p$ were chosen arbitrarily we conclude that $(\text{IND}_M, \mathcal{M})$ is not PRT[O]-learnable.

**Lemma 3.8.** There is a PRT[O]-learnable indexed target family that is not PRT[T]-learnable.

**Proof.** Define

$$\mathcal{L} = \{W_e : (\exists n \leq 2^e)(\forall i > n)(W_{c(e)} = [0, e] \land 0 \in W_{c(e)})\}.$$
Define the family of finite column interval sets, $FCI$, to be $\mathcal{L}$ together with the index function that returns the greatest element of the highest index non-empty column of a set, or 0 if no greatest element exists or if infinitely many columns are non-empty. To PRT[O]-learn $FCI$, define $M$ to be a learning machine that uses the exponential query search algorithm to find the highest index non-empty column, queries about the members of the column, in increasing order, until the greatest element is found, and returns the value of this element. Since the number of queries involved is polynomially bounded in $e$, $M$ witnesses the desired learnability.

For $i \in \mathbb{N}$, define $k_i$ as above. Let $f$ be a computable function such that

$$W_{f(x)} = \bigcup_{i \leq 2^i} [0, k_i(x)].$$

By the recursion theorem, we can pick an $e$ such that $W_{f(e)} = W_e$. Clearly, $W_e \in \mathcal{L}$. Furthermore, $W_{k_i(e)} \in \mathcal{L}$ for all $i \leq 2^e$. Fix a machine $M$ and suppose that $M$ PRT[T]-learns $FCI$. There must be a string $\sigma_0$, such that content($\sigma_0$) $\subseteq$ $W_{k_i(e)}$, on which $M$ has converged to a correct hypothesis for $W_{k_i(e)}$. Proceeding inductively for $n < 2^e$, given $\sigma_n$, we can define $\sigma_{n+1} > \sigma_n$ such that $M$ converges to a correct hypothesis for $W_{k_{n+1}}$ on $\sigma_{n+1}$ and content($\sigma_{n+1}$) $\subseteq$ $W_{k_{n+1}}$. Because $\sigma_i$ is defined for $i \leq 2^e$, $M$ fails to PRT[T]-learn $FCI$. Note that if we make the additional assumption that content($\sigma_{n+1}$) $\not\subseteq$ $W_{k_{n+1}}$, then $M$ receives an exponential number of distinct elements before converging to a correct hypothesis.

Finally, we wish to distinguish PRT[T,O]-learning from both PRT[T]-learning and PRT[O]-learning.

**Lemma 3.9.** There is an indexed target family which is PRT[T,O]-learnable, but neither PRT[T]-learnable nor PRT[O]-learnable.

**Proof.** To prove the claim, we must combine the strategy used in the proof of Lemma 3.7 with a second strategy – in essence adding a second dimension to the difficulty in finding the self-description. Define

$$\mathcal{H} = \{ W_e : (\exists n \leq 2^e)(W_e = \bigcup_{i \in [0,n]} D_i)$$

$$\wedge (\forall i \leq n)(\exists a_i)(D_i \in \text{descriptors}_i(a_i) \wedge \langle 0, 1, 1, i \rangle \in D_i \wedge a_n = e) \}.$$ 

Note that $\mathcal{H}$ is non-empty as $M \in \mathcal{H}$, where $(\text{ind}_M, M) = \text{MSD}$. Naturally, we define the associated index function $\text{ind}_\mathcal{H}$ to return $e$ on sets as in the definition of $\mathcal{H}$ and 0 on all other sets. Define $E_{\text{MSD}} = (\text{ind}_\mathcal{H}, \mathcal{H})$ to be the set of exponential marked self-describing sets.

To see that $E_{\text{MSD}} \in \text{PRT}[T,O]$, consider a learner, $M$, and teacher, $T$, defined as follows. Let $m_i = \langle 0, 1, 1, i \rangle$. $M$ determines the greatest non-empty column by applying a modified exponential query search algorithm. Query about $m_i$ until the answer is false. If $k_0$ is the last value of $i$ for which the answer to the query is true, then $M$ queries about the membership of $m_{2^i-1}$, for $0 \leq j < k_0$ until an answer of false is returned. Proceeding in this manner, the greatest non-empty column can be ascertained with at most $e^2$ queries. Let $n$ be the index of the greatest non-empty column. Having determined the value of $n$, $M$ waits for input from $T$. If it receives a string of elements from $T$ of the form $a^e$ followed by no further elements, then $M$ outputs $e$. Otherwise, $M$ outputs no hypothesis. $T$, which has access to the responses to $M$’s queries, outputs nothing until it receives a
complete descriptor on the $n^{th}$-column. Suppose the descriptor describes $e$. $T$ then outputs $m_n$ $e$-times and produces no further output. The learner-teacher pair $(M, T)$ succeeds in PRT[T,O]-learning.

The fact that $E_{MSD}$ is not PRT[O]-learnable follows from Lemma 3.7 and the observation that $\mathcal{M} \subseteq \mathcal{H}$ and $\text{ind}_M(M) = \text{ind}_H(M)$, where $(\text{ind}_M, M) = \text{MSD}$.

We now prove that $E_{MSD} \not\subseteq \text{PRT}[T]$. Define uniformly computable functions $f_i$, such that $W_{f_i(x)} = \bigcup_{x \leq 2^i} c_i(x)$ and define a computable function $f$ such that $W_{f(x)} = \bigcup_{x \leq 2^i} D_i$, where $D_i \in \text{descriptors}(k_i(x))$ and $\langle 0, 1, 1 \rangle \in D_i$. By the recursion theorem, there exists an $e$ such that $W_e = W_{f(x)} \in \mathcal{H}$. Fix such an $e$, let $N$ be any learning machine and $V$ be any teacher. Suppose that $(N, V)$ succeeds in PRT[T]-learning $E_{MSD}$. Since $W_{f_i(x)} \in \mathcal{H}$ for all $i \leq 2^e$, $(N, V)$ must learn $W_{f_i(x)}$ for all $i \leq 2^e$. Let $\sigma_0$ be the shortest initial segment of an enumeration of $W_{f_i(x)}$ on which the learner-teacher pair produces a correct code for $W_{f_i(x)}$. Using the fact that $W_{f_i(x)} \in \text{MSD}$ for $n + 1 \leq 2^e$, and proceeding inductively, define $\sigma_{n+1}$ to be shortest initial segment of an enumeration extending $\sigma_n$ on which the learner-teacher pair outputs a correct code for $W_{f_i(x)}$. The learner-teacher pair will change its hypothesis $2^e$ many times on an enumeration of $W_e$ which extends $\sigma_{2^e}$. Trivially, the run-time of the learner is also exponential in $e$. Thus, $E_{MSD} \not\subseteq \text{PRT}[T]$.

For clarity, we summarize the results of Section 3 in the following theorem.

**Theorem 3.10.** (1) $\text{PRT} \subset \text{PRT}[O] \subset \text{PRT}[T, O]$. (2) $\text{PRT} \subset \text{PRT}[T] \subset \text{PRT}[T, O]$. (3) $\text{PRT}[O] \setminus \text{PRT}[T] \neq \emptyset$. (4) $\text{PRT}[T] \setminus \text{PRT}[O] \neq \emptyset$.

**Proof.** All of the claims in the statement follow from Lemmas 3.7, 3.8 and 3.9 and Proposition 3.5.

4. POLYNOMIAL SIZE DATASET

**Definition 4.1.** An indexed target family, $(\text{ind}_F, \mathcal{F})$, is polynomial size dataset learnable (PSD-learnable) if there is a machine $M$ and a polynomial $p$ such that for any enumeration $f$ of $F \in \mathcal{F}$, $M$ converges to a correct hypothesis on an initial segment $f|n$ such that $||f(x) : x < n|| < p(\text{ind}_F(F))$. If an oracle is accessed, oracle use must also be bounded by $p(\text{ind}_F(F))$.

We shall apply Definition 2.6 to Definition 4.1 much as we did in the case of Definition 3.1.

**Proposition 4.2.** $\text{PSD} \subseteq \text{PSD}[O] \subseteq \text{PSD}[T, O]$ and $\text{PSD} \subseteq \text{PSD}[T] \subseteq \text{PSD}[T, O]$.

**Proof.** The claim follows from the definitions of PSD, PSD[T], PSD[O] and PSD[T,O].

Unlike PRT-learning, there are non-trivial PSD-learnable indexed target families.

**Example 4.3.** Let $\mathcal{F} = \{F : |F| < \infty \}$ and $\text{ind}_F(F) = |F| + e$, where $e$ is the canonical code for $F$. $(\text{ind}_F, \mathcal{F})$ is PSD-learnable by the learning machine $M$ where $M(a_0 \cdots a_n)$ is a code for the finite set $\{a_0, \ldots, a_n\}$. 
Example 4.4. Let $\mathcal{F} = \{[0, 2^e] : e \in \mathbb{N}\}$ and $\text{indi}_F([0, 2^e]) = e$. (indi$_F$, $\mathcal{F}$) is PSD(O)-learnable by a learning machine that uses the exponential query search algorithm to find the greatest element. (indi$_F$, $\mathcal{F}$) is PSD(T)-learnable by the pair $(M, T)$ where $M(a_0, \ldots, a_{n-1}) = n$ and $T(a_0 \cdots a_{k+1}) = \min\{a_0, \ldots, a_{k+1}\}$ if $2^n \leq \max\{a_0, \ldots, a_{k+1}\} < 2^{n+1}$ and $\max\{a_0, \ldots, a_k\} < 2^n$. (indi$_F$, $\mathcal{F}$) is not PSD-learnable as the learner may be forced to receive $2^{e-1}$ distinct elements before converging to a correct hypothesis.

Lemma 4.5. There is an indexed target family which is PSD(T)-learnable, but not PSD(O)-learnable.

Proof. Since every PRT(T)-learnable family is also PSD(T)-learnable, we need only verify that $\text{MSD} = (\text{indi}_M, M)$ is not PSD(O)-learnable. Fix a learning machine $M$ and polynomial $p$. We define computable functions $f$, $g$ and $q$ as follows.

1. Let $\sigma_i = (0, 1, 1, 0) \cdots (0, i, 1, 0)$ and $S = \bigcup_{i \in \mathbb{N}} \text{content}((\sigma_i))$. Consider the situation where $M$ has access to the membership oracle for $S$ and define a computable function $q$ such that, for $\ell \in \mathbb{N}$, $q(\ell)$ is the greatest number about which $M$ queries the oracle when it receives an input $\sigma_i$ for some $i \leq \ell$.

2. $W_{f(x,y)}$ is a descriptor and describes $x$.

3. $W_{g(x,y)}$ is a descriptor and describes $y$.

4. $W_{f(x,y)} \neq W_{g(x,y)}$.

5. $(W_{f(x,y)} \cap W_{g(x,y)}) \cap [0, q(x+y) + 1] = S \cap [0, q(x+y) + 1]$.

By Smullyan’s Double Recursion Theorem, we may fix $e_0$ and $e_1$ such that $W_{e_0} = W_{f(e_0, e_1)}$ and $W_{e_1} = W_{g(e_0, e_1)}$. Both sets are clearly members of $M$. Let $T_0$ and $T_1$ be enumerations of $W_{e_0}$ and $W_{e_1}$, respectively, both of which begin with the string $\sigma_i$, where $\text{content}(\sigma_i) = S \cap [0, q(e_0 + e_1) + 1]$. As above, $M$ cannot identify both enumerations successfully. Since $M$ and $p$ are arbitrary, $\text{MSD}$ is not PSD(O)-learnable.

Theorem 4.6. (1) $\text{PSD} \subset \text{PSD(O)} \subset \text{PSD(T,O)}$.

(2) $\text{PSD} \subset \text{PSD(T)} \subset \text{PSD(T,O)}$.

(3) $\text{PSD(O)} \setminus \text{PSD(T)} \neq \emptyset$.

(4) $\text{PSD(T)} \setminus \text{PSD(O)} \neq \emptyset$.

Proof. By Lemma 4.5 and Example 4.4, we need only prove that $\text{PSD(O)} \setminus \text{PSD(T)} \neq \emptyset$ and $\text{PSD(T)} \cup \text{PSD(O)} \subset \text{PSD(T)}$.

By the comments at the end of the proof of Lemma 3.8, the set of finite column intervals, $\mathcal{FCI}$, is not PSD(T)-learnable. Since PRT(O) $\subseteq$ PSD(O), $\mathcal{FCI} \in \text{PSD(O)} \setminus \text{PSD(T)}$.

$\mathcal{EMS}_D \subset \text{PRT(O)} \subset \text{PSD(T)}$ and so we need only prove that $\mathcal{EMS}_D \notin \text{PSD(T)} \cup \text{PSD(O)}$. Since $\text{MSD} \notin \text{PSD(O)}$ and $\text{MSD}$ is contained in $\mathcal{EMS}_D$, we conclude that $\mathcal{EMS}_D$ is not PSD(O)-learnable. Furthermore, in the proof of Lemma 3.3, if we choose $\sigma_{n+1}$ to contain an element of $W_{k_i(e)} \setminus W_{k_i(e)}$, we guarantee that the arbitrarily chosen learner does not arrive at a correct hypothesis before it has received an exponential size dataset. Hence, $\mathcal{EMS}_D$ is not PSD(T)-learnable.

5. Polynomial Mind Changes

Definition 5.1. An indexed target family, $(\text{indi}_F, \mathcal{F})$, is polynomial mind-changes learnable (PMC-learnable) if there is a machine $M$ and a polynomial $p$ such that for every enumeration $f$ of $F \in \mathcal{F}$, the hypothesis stream generated by $M$ on $f$ contains fewer than
\(p(\text{ind}_F(F))\) distinct hypotheses and the only one that appears infinitely many times is an encoding of \(F\). If an oracle is accessed, oracle use must also be bounded by \(p(\text{ind}_F(F))\).

We begin with an example exhibiting three PMC-learnable indexed target families.

**Example 5.2.** Let \(\mathcal{F} = \{F : |F| < \infty\}\) and \(\text{ind}_F(F) = |F|\). \((\text{ind}_F, \mathcal{F})\) is PMC-learnable as witnessed by the learning machine \(M\) such that \(M(a_0 \cdots a_k)\) is a code for the finite set of distinct elements in \(a_0, \ldots, a_k\). On any enumeration of a finite set, \(F, M\) will change its hypothesis at most \(|F|\) times.

Let \(\mathcal{F} = \{[0, 2^e] : e \in \mathbb{N}\}\) and \(\text{ind}_F([0, 2^e]) = e\). \((\text{ind}_F, \mathcal{F})\) is PMC-learnable. Define \(M\) such that \(M(\sigma)\) is a code for \([0, 2^s]\), where \(s\) is the least integer greater than or equal to \(\log_2(\max(\sigma))\).

\(\mathcal{MSD}\) is PMC-learned by a learning machine that waits until a descriptor has appeared in the enumeration and then outputs the number the descriptor describes.

**Theorem 5.3.** \(\text{PMC}[T] = \text{PMC} = \text{PSD}[T]\).

**Proof.** Fix an arbitrary \((\text{ind}_F, \mathcal{F})\). If \((M, T)\) \(\text{PMC}[T]\)-learns \((\text{ind}_F, \mathcal{F})\), then \(M \circ T\) PMC-learns \((\text{ind}_F, \mathcal{F})\). Since every PMC-learnable indexed target family is also \(\text{PMC}[T]\)-learnable, \(\text{PMC} = \text{PMC}[T]\).

Suppose \((M, T)\) \(\text{PSD}[T]\)-learns \((\text{ind}_F, \mathcal{F})\) and define \(M^*\) such that \(M^*(a_0 \cdots a_{k+1}) = M \circ T(a_0 \cdots a_k)\) when \(T(a_0 \cdots a_i) \neq \{T(a_0 \cdots a_i) : i < k + 1\}\) and \(M^*(a_0 \cdots a_{k+1}) = M^*(a_0 \cdots a_k)\), otherwise. Since \((M, T)\) \(\text{PSD}[T]\)-learns \((\text{ind}_F, \mathcal{F})\), the number of distinct elements that \(T\) outputs before \(M \circ T\) converges to a correct hypothesis is polynomially bounded, hence \(M^*\) changes hypothesis a polynomially bounded number of times. Thus, \((\text{ind}_F, \mathcal{F})\) \(\in\) \(\text{PMC}\).

Define functions \(f\) and \(g\) such that \(f(\sigma) = |\sigma|\) and \(g(n, x) = x^n\), the string \(x\) repeated \(n\) times. Suppose \(M\) PMC-learns \((\text{ind}_F, \mathcal{F})\). Define \(T\) such that \(T(\sigma) = g(M(\sigma), \min(\sigma))\) if \(M(\sigma)\) is different from \(M(\tau)\) for all \(\tau < \sigma\). \(T(\sigma)\) is undefined otherwise. \((f, T)\) \(\text{PSD}[T]\)-learns \((\text{ind}_F, \mathcal{F})\) because it converges to a correct hypothesis after reading a polynomially bounded number of outputs from \(T\).

\(\square\)

**Theorem 5.4.** \(\text{PMC} = \text{PMC}[T] \subset \text{PMC}[O] \subset \text{PMC}[T,O]\)

**Proof.** The proof of Lemma 3.3 shows that \(\mathcal{FCI}\) is also not PMC-learnable. Similarly, the proof of Lemma 3.3 shows that \(\mathcal{EMSD}\) is not \(\text{PMC}[O]\)-learnable. Thus, \(\text{PMC}[O] \subset \text{PMC}[T,O]\) and \(\text{PMC} \subset \text{PMC}[O]\). By Theorem 5.3 \(\text{PMC} = \text{PMC}[T]\), and so the desired claims are true.

\(\square\)

6. Polynomial Size Characteristic Sample and Further Questions

**Definition 6.1.** An indexed target family, \((\text{ind}_F, \mathcal{F})\), is polynomial size characteristic sample learnable (PCS-learnable) if there is a machine \(M\), a polynomial \(p\) and a family \(\mathcal{H}\) such that for each \(F \in \mathcal{F}\), there is a corresponding \(H \in \mathcal{H}\) such that \(|H| < p(\text{ind}_F(F))\) and if \(f\) is an enumeration of \(F\), then \(M\) outputs the same encoding of \(F\) on every initial segment of \(f\) whose content includes \(H\). If an oracle is accessed, oracle use must also be bounded by \(p(\text{ind}_F(F))\).

**Proposition 6.2.** There is a \(\text{PCS}[O]\)-learnable indexed target family that is not PCS-learnable.
Proof. Define \( \mathcal{F} = \{[0, n] : n \in \mathbb{N}\} \cup \{\mathbb{N}\} \) and let \( \text{ind}_F \) be an index function such that \( \text{ind}_F([0, n]) = n \) and \( \text{ind}_F(\mathbb{N}) = a \), where \( a \) is a fixed code for \( \mathbb{N} \). \((\text{ind}_F, \mathcal{F})\) is PCS\([O]\)-learned by \( M \), where \( M(a_0 \cdot \ldots \cdot a_n) = a \) if the answer to a query about \( \max\{a_0, \ldots, a_n\} + 1 \) is true and is a code for \([0, \max\{a_0, \ldots, a_n\}]\) otherwise.

To obtain a contradiction, suppose that \((\text{ind}_F, \mathcal{F})\) is PCS-learned by \( M \). Let \( A \) be a finite set which is the characteristic sample by which \( M \) identifies \( \mathbb{N} \) and let \( B = [0, \max(A)] \). By the definition of \( A \), \( M \) cannot PMC-learn \( B \). Thus, \((\text{ind}_F, \mathcal{F})\) is not PCS-learnable.

\( \square \)

The results so far are summarized in the following diagram.

\[
\begin{align*}
\text{PRT} & \subset \text{PRT}[T] \neq \text{PRT}[O] \subset \text{PRT}[T,O] \\
\subseteq & \\
\text{PSD} & \subset \text{PSD}[T] \neq \text{PSD}[O] \subset \text{PSD}[T,O] \\
= & \\
\text{PMC} & = \text{PMC}[T] \subset \text{PMC}[O] \subset \text{PMC}[T,O]
\end{align*}
\]

In light of this, it is natural to ask the following question.

**Question 6.3.** What is the relationship between PCS, PCS\([T]\) and PCS\([T,O]\) and how do they fit in the framework illustrated above?

**References**

[1] Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.
[2] Dana Angluin. Queries revisited. *Theor. Comput. Sci.*, 313(2):175–194, February 2004.
[3] Thorsten Doliwa, Hans Ulrich Simon, and Sandra Zilles. Recursive teaching dimension, learning complexity, and maximum classes. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory (ALT’10)*, pages 209–223. Springer, 2010.
[4] Sally Floyd and Manfred Warmuth. Sample compression, learnability, and the vapnik-chervonenkis dimension. *Machine Learning*, 21:1–36, 1995.
[5] Lance Fortnow, William Gasarch, Sanjay Jain, Efim Kinber, Martin Kummer, Stuart Kurtz, Mark Pleszkovich, Theodore Slaman, Robert Solovay, and Frank Stephan. Extremes in the degrees of inferability. *Annals of Pure and Applied Logic*, 66(3):231 – 276, 1994.
[6] Sally A. Goldman and Michael J. Kearns. On the complexity of teaching. *J. Comput. Syst. Sci.*, 50(1):20–31, February 1995.
[7] Sanjay Jain and Arun Sharma. On the non-existence of maximal inference degrees for language identification. *Information Processing Letters*, 47(2):81 – 88, 1993.
[8] Martin Kummer and Frank Stephan. On the structure of degrees of inferability. *Journal of Computer and System Sciences*, 52(2):214 – 238, 1996.
[9] Claude Sammut and Ranan Banerji. Learning concepts by asking questions. In R.S. Michalski, J. Carbonell, and T.M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach, II*. Morgan Kaufmann, Los Altos, CA, 1986.
[10] Robert I. Soare. *Recursively enumerable sets and degrees: a study of computable functions and computably generated sets*. Springer-Verlag, 1987.
[11] Frank Stephan. Noisy inference and oracles. *Theoretical Computer Science*, 185:129–157, 1997.
[12] Sandra Zilles, Steffen Lange, Robert C. Holte, and Martin Zinkevich. Teaching dimensions based on cooperative learning. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT’08)*, pages 135–146, 2008.
(A. Beros) Laboratoire d’Informatique de Nantes Atlantique, Université de Nantes, 2 rue de la Houssinière
BP 92208, 44322 Nantes Cedex 03, FRANCE,
E-mail address: achilles.beros@univ-nantes.fr

(C. de la Higuera) Laboratoire d’Informatique de Nantes Atlantique, Université de Nantes, 2 rue de la Houssinière
BP 92208, 44322 Nantes Cedex 03, FRANCE,
E-mail address: cdlh@univ-nantes.fr