Being Taught can be Faster than Asking Questions

Ronald L. Rivest[*] **Yiqun Lisa Yin**[†]

Abstract

We explore the power of teaching by studying two on-line learning models: teacher-directed learning and self-directed learning. In both models, the learner tries to identify an unknown concept based on examples of the concept presented one at a time. The learner predicts whether each example is positive or negative with immediate feedback, and the objective is to minimize the number of prediction mistakes. The examples are selected by the teacher in teacher-directed learning and by the learner itself in self-directed learning. Roughly, teacher-directed learning represents the scenario in which a teacher teaches a class of learners, and self-directed learning represents the scenario in which a smart learner asks questions and learns by itself. For all previously studied concept classes, the minimum number of mistakes in teacher-directed learning is always larger than that in self-directed learning. This raises an interesting question of whether teaching is helpful for all learners including the smart learner. Assuming the existence of one-way functions, we construct concept classes for which the minimum number of mistakes is linear in teacher-directed learning but superpolynomial in self-directed learning, demonstrating the power of a helpful teacher in a learning process.

[*] Laboratory for Computer Science, MIT, Cambridge, MA 02139. Supported in part by NSF grant CCR-9310888 and a grant from Siemens. Email: rivest@theory.lcs.mit.edu.

[†] RSA Laboratories, 100 Marine Parkway, Redwood City, CA 94065. The work was done when the author was at MIT and supported in part by DARPA Contracts N00014-91-J-1698 and N00014-92-J-1799. Email: yiqun@rsa.com.

1 Introduction

In this paper, we study the power of a teacher in helping students to learn concept classes. In the literature of learning theory, the teacher has been modeled differently in various learning frameworks [3, 8, 9, 10, 11, 15, 20, 22, 23], and the impact of teaching depends on how much the teacher is involved in the learning process. We study the importance of teaching by investigating two learning models:

- teacher-directed learning in which the learner highly relies on the information provided by the teacher to accomplish learning, and
- self-directed learning in which the learner actively queries the information needed and accomplishes learning solely by itself.

Teacher-directed learning and self-directed learning were first introduced by Goldman, Rivest, and Schapire [11]. In both models, the learner tries to identify an unknown concept based on examples of the concept presented one at a time. The learner predicts whether each example is positive or negative with immediate feedback, and the objective is to minimize the number of prediction mistakes. The examples are selected by the teacher in teacher-directed learning and by the learner itself in self-directed learning. The picture behind the formulation of the two models is roughly as follows. Self-directed learning reflects the situation in which a smart learner asks questions and learns by itself; teacher-directed learning reflects the situation in which a teacher teaches a class of learners, some of which may be stupid. Throughout the paper, we use *smart learner* to denote an optimal self-directed learner for a given concept class. To study the power of teaching, we compare the number of mistakes made by the smart learner with the number of mistakes made by the stupidest learner with the help of a powerful teacher.

Goldman and Kearns [8, 9] studied the teacher-directed learning model and gave tight bounds on the number of mistakes for several concept classes. Goldman and Sloan [8, 12] studied the self-directed learning model and derived optimal bounds on the number of mistakes for several concept classes. For all previously studied concept classes [8, 9, 11, 12, 27], the minimum number of mistakes made by the stupidest learner in teacher-directed learning is always larger than the minimum number of mistakes made by the smart learner in self-directed learning. This raises an interesting question of whether teaching is helpful for all learners including the smart learner. In other words, can a smart learner learn faster when being taught instead of asking questions and working on its own?

In this paper, we answer this question in the affirmative. We construct concept classes for which the minimum number of mistakes in self-directed learning is strictly larger than that in teacher-directed learning, assuming that cryptographically strong pseudorandom bit generators exist. In fact, our results are much stronger: the concept classes that we create have the property that the minimum number of mistakes is superpolynomial in self-directed learning but only linear in teacher-directed learning. In particular, without the help from a teacher, the concept classes are not learnable even for the smart learner. This demonstrates the power of teaching in a learning process. It has been shown that the existence of cryptographically strong pseudorandom bit generators is equivalent to the existence of one-way functions [14, 18]. So our results hold if any one-way function exists.

In the past, cryptography has had considerable impact on learning theory, and virtually every non-learnability result has at its heart a cryptographic construction [1, 2, 4, 16, 17, 21]. Although the construction of our concept classes is also based on a cryptographic assumption, our non-learnability result for self-directed learning is stronger than previous non-learnability results in the following sense: Most of the previous results of this type rely on the fact that the examples are chosen according to a distribution or by an adversary which might be "malicious" to the learner. Since the examples are selected by the learner itself in self-directed learning, the non-learnability of our concept classes is solely inherent in the structure of the concept classes and does not depend on having the learner see examples in a way that is less desirable than could have been chosen by itself.

As a by-product, our results also imply that the minimum number of mistakes for learning a concept class in self-directed learning can be substantially larger than the Vapnik-Chervonenkis dimension [25] of the concept class. This answers an open question posed by Goldman and Sloan [12].

The remainder of the paper is organized as follows. In §2, we formally define teacher-directed learning and self-directed learning. In §3, we review some useful definitions in cryptography. In §4, we present the construction of our concept classes and show that they have the desired property. In §5, we further discuss some other properties of our concept classes. We conclude in §6 with some open problems.

2 The learning models

In this section, we first introduce some basic definitions in learning theory and review Littlestone's on-line learning model. Then, we formally define teacher-directed learning and self-directed learning, which are two variants of Littlestone's model.

A *concept* c is a Boolean function on some domain of instances X. A *concept class* $\mathcal{C}$ is a family of concepts. An *example* is an instance $x \in X$, and, for a given concept c, a *labeled example* of c is a pair $\langle x, c(x) \rangle$. An example x is called a *positive example* if $c(x) = 1$, and it is called a *negative example* otherwise. An instance domain X is often decomposed into subsets $\{X_n\}$ according to some natural dimension measure n. Accordingly, a concept class $\mathcal{C}$ is decomposed into subclasses $\{\mathcal{C}_n\}$. In all models for concept learning, the objective of the learner is to learn an unknown *target concept* in a known concept class using labeled examples of the target concept. Since we are interested in designing efficient algorithms, we will focus our discussion on polynomial-time algorithms throughout the paper unless otherwise specified.

One of the commonly used models in learning theory is Littlestone's mistake-bound model [19] in which learning is done on-line in a series of stages. In each stage, an adversary first presents an unlabeled example x to the learner. The learner predicts if x is positive or negative and is then told the correct answer. The goal of the learner is to minimize the number of prediction mistakes. We say that the learner *learns* a concept class $\mathcal{C} = \{\mathcal{C}_n\}$ if there exists a polynomial P such that for all target concepts in $\mathcal{C}_n$, the learner makes at most $P(n)$ mistakes using polynomial time in each stage.

We say that a learner is *consistent* if, in every stage, there is a concept in $\mathcal{C}_n$ that agrees with the learner's current prediction and all previously seen labeled examples. A consistent learner is a reasonable learner in the sense that it pays attention to what has been presented. We define a *polynomial-time consistent learner* as a learner that makes consistent predictions using polynomial time in each stage.

The self-directed learning model

Self-directed learning is a variant of Littlestone's model in which the adversary is replaced by the learner itself. Let A be a self-directed learning algorithm for selecting examples and making predictions. We use $M_S(\mathcal{C}_n, A)$ to denote the maximum number of mistakes made by A for any target concept $c \in \mathcal{C}_n$, and we define $optM_S(\mathcal{C}_n) = \min_A M_S(\mathcal{C}_n, A)$. In other words, $optM_S(\mathcal{C}_n)$ is the number of mistakes made by an optimal self-directed learner (i.e., a smart learner) in the worst case.

Note that a self-directed learner selects examples by itself, and the selection in each stage is based on the learner's current knowledge of the target concept obtained from previously seen labeled examples. This reflects the situation in which a smart learner actively asks questions and learns by itself.

The teacher-directed learning model

Teacher-directed learning is a variant of Littlestone's model in which the adversary is replaced by a helpful teacher who *knows* the target concept. Let A be a teacher's algorithm for selecting examples. We define $M_T(\mathcal{C}_n, A)$ as the maximum number of mistakes made by any polynomial-time consistent learner for any target concept $c \in \mathcal{C}_n$. (We make the convention that $M_T(\mathcal{C}_n, A) = |X_n|$ if $\mathcal{C}_n$ has no polynomial-time consistent learner.) We define $optM_T(\mathcal{C}_n) = \min_A M_T(\mathcal{C}_n, A)$. In other words, $optM_T(\mathcal{C}_n)$ is the number of mistakes made by the stupidest learner in the worst case when the teacher uses an optimal algorithm.

Note that the teacher is required to teach any polynomial-time consistent learner. Equivalently, the teacher is required to present a sequence of labeled examples that uniquely specifies the target concept. This requirement represents the situation in which a teacher teaches a class of learners who may be stupid but pay attention to (i.e., is consistent with) what the teacher has presented.

The requirement is essential since it prevents possible collusions between the teacher and the learner, which would make both teaching and learning trivial. An easy collusion strategy is the following: The teacher and the learner agree beforehand on an "encoding" of the concepts in a concept class by certain sequences of examples. When teaching a concept, the teacher just presents the sequence of examples that encodes the concept, even though there may be several concepts consistent with the sequence of examples.

$\mathbf{optM_S}(\mathcal{C})$ versus $\mathbf{optM_T}(\mathcal{C})$

For all the natural concept classes that have been previously studied [9, 12, 27], $optM_S(\mathcal{C})$ is always smaller than $optM_T(\mathcal{C})$. As we analyze these algorithms, it is always the case that a smart self-directed learner can obtain some information about the target concept "for free".

We illustrate this phenomenon by a simple concept class. Let $\mathcal{C}$ consist of all the concepts with exactly one positive example and the unique concept whose examples are all negative. To learn $\mathcal{C}$, a smart learner can simply follow any order of the examples and always predict negative. In such a way, the smart learner makes at most one mistake, and hence $optM_S(\mathcal{C}) = 1$. On the other hand, to teach the concept that only has negative examples, the teacher must present all the examples in domain X; otherwise, some consistent learner may still make prediction mistakes for unseen examples. This implies $optM_T(\mathcal{C}) = |X|$.

3 Some background in cryptography

In this section, we first introduce some notation and definitions in cryptography. We then review the Goldreich, Goldwasser, and Micali pseudorandom function construction [13], which will be useful for constructing our concept classes.

Let $R = \bigcup_n R_n$, where each R_n is the set of all possible 0-1 strings of length n. Let $S = \bigcup_n S_n$ be a set of strings, where each S_n consists of n-bit-long strings. We use notation $s \in_{rand} S_n$ to denote that s is chosen uniformly at random from S_n. Let T be a probabilistic polynomial-time algorithm that takes as input strings from S_n and outputs either 0 or 1. We use $P_n(T, S)$ to denote the probability that T outputs 1 on $s \in_{rand} S_n$.

For a polynomial P, a *Cryptographically Strong pseudorandom Bit generator (CSB generator)* [6] with stretch P is defined as a deterministic polynomial-time algorithm G with the following properties: (1) On an input string $s \in \{0,1\}^n$, G generates a $P(n)$-bit-long output string. (2) The set of strings $S = \bigcup_n S_n$ that G generates cannot be efficiently distinguished from set R. More precisely, for any probabilistic polynomial-time algorithm T and any polynomial Q, $|P_n(T,S) - P_n(T,R)| < \frac{1}{Q(n)}$ for sufficiently large n.

Let $F = \{F_n\}$ be a collection of functions, where each F_n consists of functions from $\{0,1\}^n$ to $\{0,1\}^n$. Let A be a probabilistic polynomial-time algorithm capable of oracle calls. On an input function f: $\{0,1\}^n \rightarrow \{0,1\}^n$, A outputs either 0 or 1 by querying an oracle for f about some instances. We use $P_n(A,F)$ to denote the probability that A outputs 1 on a function $f \in_{rand} F_n$.

Let $F = \{F_n\}$ and $F' = \{F'_n\}$ be two collections of functions, where both F_n and F'_n consist of functions from $\{0,1\}^n$ to $\{0,1\}^n$. We say that F and F' are *polynomially indistinguishable* if, for any probabilistic polynomial-time algorithm A and any polynomial Q, $|P_n(A,F) - P_n(A,F')| < \frac{1}{Q(n)}$ for sufficiently large n. It is easy to prove that polynomial indistinguishability is transitive.

Let $F = \{F_n\}$ be a collection of functions, where each F_n consists of functions from $\{0,1\}^n$ to $\{0,1\}^n$. Let A be a probabilistic polynomial-time algorithm capable of oracle calls. On an input function $f \in_{rand} F_n$, A queries an oracle O_f for f about some instances and then chooses a different instance y. At this point, A is disconnected from O_f and is presented with values $f(y)$ and $r \in_{rand} \{0,1\}^n$ in a random order. For any polynomial Q, we say that algorithm A *Q-infers* F if, for infinitely many n, A correctly guesses which of the two values is $f(y)$ with probability at least $\frac{1}{2} + \frac{1}{Q(n)}$ for $f \in_{rand} F_n$. We say that F can be *polynomially inferred* if there exist a polynomial Q and a probabilistic polynomial-time algorithm A that Q-infers F.

The pseudorandom function collection constructed by Goldreich, Goldwasser, and Micali is a set of functions $F^* = \{F^*_n\}$, where each $F^*_n = \{f_s\}_{s\in\{0,1\}^n}$ is defined as follows. Let G be a CSB generator that stretches a seed $s \in \{0,1\}^n$ into a $2n$-bit-long sequence $G(s) = b^s_1 \cdots b^s_{2n}$. Let $G_0(s)$ be the leftmost n bits $b^s_1 \cdots b^s_n$ and $G_1(s)$ be the rightmost n bits $b^s_{n+1} \cdots b^s_{2n}$. For $t \geq 1$, let

$$G_{x_1\cdots x_t}(s) = G_{x_t}(G_{x_{t-1}}(\cdots G_{x_1}(s)\cdots)).$$

Then function f_s:$\{0,1\}^n \rightarrow \{0,1\}^n$ is defined as

$$f_s(x_1\cdots x_n) = G_{x_1\cdots x_n}(s) = G_{x_n}(G_{x_{n-1}}(\cdots G_{x_1}(s)\cdots)).$$

Goldreich *et al.* showed that if CSB generators exist, then the collection of functions F^* is polynomially indistinguishable from $H = \{H_n\}$ where H_n is the set of all functions from $\{0,1\}^n$ to $\{0,1\}^n$. It is easy to see that F^* also has the following two properties: (1) *Indexing*: each function $f_s \in F_n$ has a unique n-bit index s associated with it. (2) *Polynomial-time evaluation*: there exists a polynomial-time algorithm that on inputs $s, x \in \{0,1\}^n$ computes $f_s(x)$.

Goldreich *et al.* further studied how to infer a function in F^*_n given its input-output values. They obtained the following general result which immediately implies that $F^* = \{F^*_n\}$ cannot be polynomially inferred if CSB generators exist.

Theorem 1 [13] *Let $F = \{F_n\}$ be a collection of functions, where each F_n consists of functions from $\{0,1\}^n$ to $\{0,1\}^n$. If F has the properties of indexing and polynomial-time evaluation, then F cannot be polynomially inferred if and only if F is polynomially indistinguishable from H.*

We remark that the above theorem also holds for collections of functions with domain $\{0,1\}^n$ and range $\{0,1\}$ as opposed to $\{0,1\}^n$. This fact will be used in the next section.

4 The power of teaching

In this section, we construct concept classes for which the learner makes substantially fewer mistakes in teacher-directed learning than in self-directed learning.

We first prove a useful lemma. Let Z_n denote the set of all functions from $\{0,1\}^n$ to $\{0,1\}$ and let $Z = \{Z_n\}$.

Lemma 2 *If a concept class $\mathcal{C} = \{\mathcal{C}_n\}$ is polynomially indistinguishable from Z, then for any polynomial P and for infinitely many n, $optM_S(\mathcal{C}_n) > P(n)$.*

Proof. We assume for contradiction that there exist a polynomial-time self-directed learning algorithm A^* and a polynomial P such that $M_S(\mathcal{C}_n, A^*) \leq P(n)$ for sufficiently large n. Let $\pi = \langle x_1, x_2, \ldots, x_t\rangle$ be the query sequence that A^* chooses. (Note that for different target concepts, π may be different. So each query x_i depends on the target concept.)

By the assumption $M_S(\mathcal{C}_n, A^*) \leq P(n)$, we obtain that, for any fixed target concept $c \in \mathcal{C}_n$, the number of prediction mistakes that A^* makes over the first $6P(n)$ queries $\langle x_1, \ldots, x_{6P(n)}\rangle$ is at most $P(n)$. Therefore, for sufficiently large n, with probability one, the number of prediction mistakes that algorithm A^* makes over the first $6P(n)$ queries is at most $P(n)$ if $c \in_{rand} \mathcal{C}_n$.

On the other hand, by Theorem 1, we know that $\mathcal{C}$ cannot be polynomially inferred . This implies that, for any polynomial Q and for sufficiently large n, the probability that A^* predicts correctly for each x_i $(1 \leq i \leq 6P(n))$ is at most $\frac{1}{2} + \frac{1}{Q(n)}$ for $c \in_{rand} \mathcal{C}_n$. Hence, for sufficiently large n, the probability that A^* predicts incorrectly for each x_i ($1 \leq i \leq 6P(n)$) is at least $\frac{1}{2} - \frac{1}{Q(n)} \geq \frac{1}{3}$. On average, algorithm A^* makes at least $\frac{1}{3} \cdot 6P(n) = 2P(n)$ prediction mistakes over the first $6P(n)$ queries. This contradicts the fact that with probability one, A^* makes at most $P(n)$ mistakes over the first $6P(n)$ queries if $c \in_{rand} \mathcal{C}_n$. □

By the above lemma, we know that in order to construct a concept class such that $optM_S(\mathcal{C}_n)$ is superpolynomial, it is sufficient to construct a concept class such that $\mathcal{C}$ is polynomially indistinguishable from Z.

In what follows, we construct a concept class $\mathcal{C}^* = \{\mathcal{C}_n^*\}$ such that $\mathcal{C}^*$ is polynomially indistinguishable from Z and $optM_T(\mathcal{C}_n^*)$ is linear. We begin with some useful notation. For $x \in \{0,1\}^n$, we use $x^{(i)}$ to denote $(x+i) \bmod 2^n$. Given a concept c, we call the sequence $\langle c(x), c(x^{(1)}) \ldots, c(x^{(2^n-1)})\rangle$ (where $x = 0 \cdots 0$) the *label sequence* of c. Note that the label sequence of c is a 0-1 sequence of length 2^n.

Let G be a CSB generator with stretch $2n$, and let $F^* = \{F_n^*\}$ be the Goldreich, Goldwasser, Micali pseudorandom function collection constructed based on G, where $F_n^* = \{f_s\}_{s \in \{0,1\}^n}$. Starting with F^*, we construct $\mathcal{C}^* = \{\mathcal{C}_n^*\}$, together with two intermediate concept classes $L = \{L_n\}$ and $L' = \{L'_n\}$, by the following three-step procedure.

Step 1: Define $L_n = \{l_s\}_{s \in \{0,1\}^n}$, where

$$l_s(x) = \text{the least significant bit of } f_s(x).$$

Step 2: Define $L'_n = \{l'_s\}_{s \in \{0,1\}^n}$, where

$$l'_s(x) = \begin{cases} 0 & \text{if } l_s(x^{(i)}) = 1, \text{ for } i = 0, 1, \ldots, n-1, \\ l_s(x) & \text{otherwise.} \end{cases}$$

Step 3: Define $\mathcal{C}_n^* = \{c_s\}_{s \in \{0,1\}^n}$, where

$$c_s(x) = \begin{cases} 1 & \text{if } x \in \{s, s^{(1)}, \cdots, s^{(n-1)}\}, \\ 0 & \text{if } x \in \{s^{(-1)}, s^{(n)}\}, \\ l'_s(x) & \text{otherwise.} \end{cases}$$

We remark that a somewhat similar construction was used by Amsterdam [1] to distinguish learning by random examples from learning by "experiments" (a certain extended queries). However, his construction does not work for our problem of distinguishing self-directed learning from teacher-directed learning.

We prove in the next two theorems that concept class $\mathcal{C}^*$ constructed above has the desired property.

Theorem 3 *$optM_T(\mathcal{C}_n^*) \leq n$.*

Proof. For any target concept $c_s \in \mathcal{C}_n^*$, we prove that the teacher only needs to present the n labeled examples $\langle s, 1\rangle, \langle s^{(1)}, 1\rangle, \cdots \langle s^{(n-1)}, 1\rangle$ in order to teach c_s. Consider Step 2 of our construction. For each concept, we flip certain 1's to 0's in its label sequence to eliminate all sequences of consecutive 1's of length n or longer. In Step 3, we further modify the label sequence so that (1) there is a unique sequence of consecutive 1's of length n in the label sequence for each concept, and (2) for any given concept, the starting position of its unique sequence of consecutive 1's of length n is different from all of the other concepts. Therefore, the n labeled examples $\langle s, 1\rangle, \langle s^{(1)}, 1\rangle, \cdots \langle s^{(n-1)}, 1\rangle$ uniquely specify c_s. Furthermore, any polynomial-time consistent learner can infer c_s from these n labeled examples. Therefore, a polynomial-time consistent learner will not make more mistakes after seeing the n labeled examples. Thus, $optM_T(\mathcal{C}_n^*) \leq n$. □

Theorem 4 *If one-way functions exist, then for any polynomial P, $optM_S(\mathcal{C}_n^*) > P(n)$ for infinitely many n.*

By Lemma 2, we only need to show that $\mathcal{C}^*$ and $Z = \{Z_n\}$ are polynomially indistinguishable provided that one-way functions exist. (Recall that Z_n is the set of all functions from $\{0,1\}^n$ to $\{0,1\}$.) The indistinguishability will be proved via the next three lemmas.

We define a set of functions $Z' = \{Z'_n\}$ by modifying $Z = \{Z_n\}$. For each $f \in Z_n$, the corresponding $f' \in Z'_n$ is defined as

$$f'(x) = \begin{cases} 0 & \text{if } f(x^{(i)}) = 1, \text{ for } i = 0, 1, \ldots, n-1, \\ f(x) & \text{otherwise.} \end{cases}$$

Note that we modify Z to obtain Z' in the same way as we modify L to obtain L'.

Lemma 5 *Z and Z' are polynomially indistinguishable.*

Proof. Assume for contradiction that Z and Z' are polynomially distinguishable. Then there exist a probabilistic polynomial-time algorithm A and a polynomial Q such that for infinitely many n,

$$|P_n(A, Z) - P_n(A, Z')| \geq \frac{1}{Q(n)}. \qquad (1)$$

Let P be a polynomial such that algorithm A makes at most $P(n)$ oracle calls (to request the value $f(x)$ for chosen x). Since A can distinguish between a function $f \in_{rand} Z_n$ and a function $f' \in_{rand} Z'_n$, A must detect a sequence of n consecutive 1's in the label sequence of

f. Since Z_n contains all functions $f:\{0,1\}^n \to \{0,1\}$, we know that for a fixed $x \in \{0,1\}^n$ and $f \in_{rand} Z_n$,

$$\Pr(f(x) = f(x^{(1)}) = \cdots = f(x^{(n-1)}) = 1) = \frac{1}{2^n}.$$

The probability that A detects a sequence of n consecutive 1's by using at most $P(n)$ queries is less than $\frac{P(n)}{2^n}$. Therefore, for any polynomial Q,

$$|P_n(A,Z) - P_n(A,Z')| < \frac{1}{Q(n)}$$

for sufficiently large n, which contradicts Equation 1. □

Lemma 6 *Z' and L' are polynomially indistinguishable.*

The proof of this lemma is technically the most difficult one, and it relies on the assumption that one-way functions exist. The basic idea, however, is quite simple. In particular, we use a standard cryptographic technique introduced by Yao [26]. Recall that the collection of functions L' is constructed based on CSB generator G. If a probabilistic polynomial-time algorithm A can distinguish between Z' and L', then we can use A to construct another probabilistic polynomial-time algorithm T such that T can distinguish the set of strings generated by G from set R (the set of all possible 0-1 strings). This is a contradiction. A detailed proof is given in the appendix.

Lemma 7 *L' and $\mathcal{C}^*$ are polynomially indistinguishable.*

Proof. Similar to the proof of Lemma 5. □

Proof of Theorem 4. Using Lemmas 5, 6, and 7 and the fact that polynomial indistinguishability is transitive, we can easily prove that $\mathcal{C}^*$ and Z are polynomially indistinguishable. By Lemma 2, $optM_S(\mathcal{C}^*)$ is superpolynomial. □

Remarks

We have seen that for each concept in $\mathcal{C}_n^*$, there exists a small set of labeled examples that contains the "key" information of the concept. However, the set of key examples is hard to find by the smart learner for an unknown target concept, and the learner may have to make a large number of mistakes in self-directed learning. We have also seen that the teacher, who knows the target concept, can easily select and present the key examples to the learner. This phenomenon also occurs in the real world: A knowledgeable teacher can help students to learn faster by providing key points that are sometimes hard to find by the students themselves.

We have shown that concept class $\mathcal{C}^*$ is not learnable in self-directed learning assuming one-way functions exist. This result is stronger than most of the previous non-learnability results that rely on cryptographic assumptions in the following sense: The non-learnability of $\mathcal{C}^*$ is solely inherent in the structure of $\mathcal{C}^*$ and does not depend on having the learner see examples in a way that is less desirable than could have been chosen by itself.

5 Discussions

In this section, we further discuss some properties of concept class $\mathcal{C}^*$. First, we consider concept class $\mathcal{C}^*$ in Littlestone's mistake-bound model [19] and Valiant's distribution-free model [24]. It is not hard to obtain the following non-learnability results from Theorem 4.

Corollary 8 *If one-way functions exist, then concept class $\mathcal{C}^*$ is not learnable in the mistake-bound model.*

Corollary 9 *If one-way functions exist, then concept class $\mathcal{C}^*$ is not learnable in the distribution-free model.*

We next consider some relation between the number of mistakes in self-directed learning and the Vapnik-Chervonenkis dimension of a concept class. Let $\mathcal{C}$ be a concept class over an instance domain X. We say that a finite set $Y \subseteq X$ is *shattered* by $\mathcal{C}$ if $\{c \cap Y \mid c \in \mathcal{C}\} = 2^Y$. The *Vapnik-Chervonenkis dimension* of $\mathcal{C}$ [25], denoted by $\text{VC}(\mathcal{C})$, is defined to be the smallest d for which no set of $d+1$ instances is shattered by $\mathcal{C}$. Note that for any finite concept class $\mathcal{C}$, $\text{VC}(\mathcal{C}) \le \log|\mathcal{C}|$.

Goldman and Sloan [12] studied the relation between $\text{VC}(\mathcal{C})$ and $optM_S(\mathcal{C})$ and presented concept classes for which $\text{VC}(\mathcal{C})$ can be arbitrarily larger than $optM_S(\mathcal{C})$. They also constructed a concept class $\mathcal{C}$ for which $\text{VC}(\mathcal{C})$ = 2 and $optM_S(\mathcal{C}) = 3$. Since this was the only known concept class for which $\text{VC}(\mathcal{C})$ is strictly smaller than $optM_S(\mathcal{C})$, they posed the following question: Is there a concept class $\mathcal{C}$ for which $optM_S(\mathcal{C}) = \omega(\text{VC}(\mathcal{C}))$? The following corollary answers Goldman and Sloan's open question in the affirmative.

Corollary 10 *$optM_S(\mathcal{C}_n^*) = \omega(\text{VC}(\mathcal{C}_n^*))$ assuming that one-way functions exist.*

Proof. By Theorem 4, $optM_S(\mathcal{C}_n^*)$ is superpolynomial in n. Since the number of concepts in $\mathcal{C}_n^*$ is 2^n, we have $\text{VC}(\mathcal{C}_n^*) \le \log|\mathcal{C}_n^*| = n$. □

6 Open problems

As we have pointed out in §1 and §4, for all the natural concept classes studied prior to our work, the number of mistakes is always smaller in self-directed learning than teacher-directed learning. In particular, the smart self-directed learner can always get useful information for the target concept without making many mistakes. It would be interesting to characterize such a property in a rigorous way.

As we have pointed out, our results and most of the previous work rely on cryptographic assumptions to prove the non-learnability of certain concept classes. There has been some recent research in the reverse direction [5]: Provably secure cryptosystems are constructed assuming certain concept classes are hard to learn in the distribution-free model. It would be interesting to construct cryptosystems based on concept classes that are easy to learn in teacher-directed learning but hard to learn in self-directed learning.

Acknowledgement

We would like to thank Silvio Micali for enlightening discussions.

References

[1] J. Amsterdam. *The Valiant Learning Model: Extensions and Assessment.* Master thesis, MIT Department of Electrical Engineering and Computer Science, January 1988.

[2] D. Angluin and M. Kharitonov. When won't membership queries help? In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, pages 444–454, 1991.

[3] D. Angluin and D. K. Slonim. Randomly fallible teachers: Learning monotone DNF with an incomplete membership oracle. *Machine Learning*, 14(1):7–26, January 1994. A preliminary version of this paper appeared in *Proceedings of the 4th Annual ACM Workshop on Computational Learning Theory.*

[4] A. Blum. Separating distribution-free and mistake-bound learning models over the Boolean domain. In *Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Science*, volume I, pages 211–218, 1990.

[5] A. Blum, M. Furst, M. Kearns, and R. Lipton. Cryptographic primitives based on hard learning problems. In *Advances in Cryptology–CRYPTO '93*, pages 278–291, 1993.

[6] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal of Computing*, 13(4):850–863, November 1984.

[7] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

[8] S. Goldman. *Learning Binary Relations, Total Orders, and Read-Once Formulas.* PhD thesis, MIT Department of Electrical Engineering and Computer Science, September 1990. (MIT Laboratory for Computer Science Technical Report MIT/LCS/TR-483, July 1990.)

[9] S. Goldman and M. Kearns. On the complexity of teaching. In *Proceedings of the 4th Annual ACM Workshop on Computational Learning Theory*, pages 303–314, 1991.

[10] S. Goldman and D. Mathias. Teaching a smart learner. In *Proceedings of the 6th Annual ACM Workshop on Computational Learning Theory*, pages 67–76, 1993.

[11] S. Goldman, R. Rivest, and R. Schapire. Learning binary relations and total orders. In *Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science*, pages 46–51, 1989.

[12] S. Goldman and R. Sloan. The power of self-directed learning. Technical Report WUCS-92-49, Washington University in St. Louis, November 1992.

[13] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.

[14] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 12–24, Seattle, 1989.

[15] J. Jackson and A. Tompkins. A computational model of teaching. In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 319–326, 1992.

[16] M. Kearns and L. G. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 433–444, 1989.

[17] M. Kharitonov. Cryptographic hardness of distribution specific learning. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, pages 372–381, 1993.

[18] L. A. Levin. One-way functions and pseudorandom generators. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 363–365, Providence, 1985.

[19] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[20] B. K. Natarajan. On learning boolean functions. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 296–304, 1987.

[21] L. Pitt and M. K. Warmuth. Reductions among prediction problems: On the difficulty of predicting automata (extended abstract). In *3rd IEEE Conference on Structure in Complexity Theory*, pages 60–69, 1988.

[22] S. Salzberg, A. Delcher, D. Heath, and S. Kasif. Learning with a helpful teacher. In *Proceedings of IJCAI-91*, pages 705–711, 1991.

[23] A. Shinohara and S. Miyano. Teachability in computational learning. *New Generation Computing*, 8:337–347, 1991.

[24] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.

[25] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its applications*, XVI(2):264–280, 1971.

[26] A. C. Yao. Theory and applications of trapdoor functions. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pages 80–91, Chicago, 1982.

[27] Y. L. Yin. On learning r-of-t threshold functions. Unpublished manuscript, 1993.

Appendix: Proof of Lemma 6

Assume for contradiction that Z' and L' are polynomially distinguishable. Then there exist a probabilistic polynomial-time algorithm A and a polynomial Q such that for infinitely many n,

$$|P_n(A, Z') - P_n(A, L')| \geq \frac{1}{Q(n)}. \qquad (2)$$

Thus, algorithm A can distinguish between a function $f \in_{rand} L'_n$ and a function $g \in_{rand} Z'_n$ by oracle calls. We next construct a sequence of n oracles that transits smoothly from an oracle O_f for f to an oracle O_g for g.

Consider the computations of A in which A's oracle calls are answered by one of the following algorithms $D_i (i = 0, 1, \ldots, n)$. Let y be a query of A. Recall that $y^{(1)}, \cdots y^{(n-1)}$ are the $n-1$ instances immediately after y in $\{0,1\}^n$ and $y^{(0)}$ is y itself. For $j = 0, 1, \ldots, n-1$, write $y^{(j)}$ as $y_1^{(j)} \cdots y_n^{(j)}$. Algorithm D_i answers A's query y as follows:

```
for $j = 0, 1, \ldots, n-1$
  if the pair $(y_1^{(j)} \cdots y_i^{(j)}, \cdot)$
    has not been stored
  then select a string $r \in_{rand} \{0,1\}^n$
    store the pair $(y_1^{(j)} \cdots y_i^{(j)}, r)$
    compute $b_j = G_{y_{i+1}^{(j)} \cdots y_n^{(j)}}(r)$
  else retrieve the pair $(y_1^{(j)} \cdots y_i^{(j)}, v)$
    compute $b_j = G_{y_{i+1}^{(j)} \cdots y_n^{(j)}}(v)$
if $b_0 = b_1 = \cdots b_{n-1} = 1$
then answer 0
else answer $b_0$
```

Define p_n^i to be the probability that A outputs 1 when n is given as an input and its queries are answered by algorithm D_i, $0 \leq i \leq n$. Then $p_n^0 = P_n(A, L')$ and $p_n^n = P_n(A, Z')$. Hence, Equation 2 is equivalent to $|p_n^0 - p_n^n| \geq \frac{1}{Q(n)}$.

We now use A to construct a probabilistic polynomial-time algorithm T for distinguishing the set of strings generated by CSB generator G from set $R = \bigcup_n R_n$. (Recall that R_n is the set of all possible 0-1 strings of length n.) Let P be a polynomial such that algorithm A makes at most $P(n)$ queries on input n. Algorithm T works in two stages on input n and a set U_n containing $P(n)$ strings each of which has $2n$ bits. In the first stage, T picks $i \in_{rand} \{0, 1, \ldots, n-1\}$. In the second stage, T answers A's queries using set U_n as follows (where y is a query of A):

```
for $j = 0, 1, \ldots, n-1$
  if the pair $(y_1^{(j)} \cdots y_{i+1}^{(j)}, \cdot)$
    has not been stored
  then pick the next string $u = u_0 u_1$ in $U_n$
    store the pairs $(y_1^{(j)} \cdots y_i^{(j)} 0, u_0)$
      and $(y_1^{(j)} \cdots y_i^{(j)} 1, u_1)$
    compute $b_j = G_{y_{i+1}^{(j)} \cdots y_n^{(j)}}(u_\alpha)$, where $\alpha = y_{i+1}^{(j)}$
  else retrieve the pairs $(y_1^{(j)} \cdots y_{i+1}^{(j)}, v)$
    compute $b_j = G_{y_{i+2}^{(j)} \cdots y_n^{(j)}}(v)$
if $b_0 = b_1 = \cdots b_{n-1} = 1$
then answer 0
else answer $b_0$
```

We consider two cases for U_n: (1) U_n consists of $(2n)$-bit strings output by the CSB generator G on random seeds, and (2) U_n consists of randomly selected $(2n)$-bit strings. In case 1, T simulates A with oracle D_i. The probability that T outputs 1 is $\sum_{i=0}^{n-1} (1/n) \cdot p_n^i$. In case 2, T simulates A with oracle D_{i+1}. The probability that T outputs 1 is $\sum_{i=0}^{n-1} (1/n) \cdot p_n^{i+1} = \sum_{i=1}^{n} (1/n) \cdot p_n^i$. For infinitely many n, the probabilities for the two cases differ by at least $(1/n) \cdot |p_n^0 - p_n^n| \geq \frac{1}{nQ(n)}$. Therefore, algorithm T can distinguish the set of strings generated by CSB generator G from set R, which is a contradiction.