PExact = Exact Learning

Dmitry Gavinsky
Department of Computer Science
University of Calgary
Calgary, Alberta, Canada, T2N 1N4
e-mail: gavinsky@cpsc.ucalgary.ca

Avi Owshanko
Departments of Computer Science
Technion
Haifa, Israel, 32000
e-mail: avshash@cs.technion.ac.il

Abstract

The Probably Exact model (PExact) is a relaxation of the Exact model, introduced in by Bshouty. In this paper, we show that the PExact model is equivalent to the Exact model.

We also show that in the Exact model, the adversary (oracle) gains no additional power from knowing the learners' coin tosses a-priory.

1 Introduction

In this paper we examine the *Probably Exact* (PExact) model introduced by Bshouty in [5] (called PEC there). This model lies between Valiant's PAC model [12] and Angulin's Exact model [1].

We show that the PExact model is equivalent to the Exact model, thus extending the results by Bshouty et. al. [8] who showed the PExact model is stronger than the PAC model (under the assumption that one way functions exist), as well as that the deterministic Exact model (where the learning algorithm is deterministic) is equivalent to the deterministic PExact model.

The PExact model is a variant of the Exact model, in which each counterexample to an equivalence query is drawn according to a distribution, rather than maliciously chosen. The main advantage of the PExact model is that the teacher is not an adversary. For achieving lower bounds in the Exact model, (like those given by Bshouty in [5]), we must consider a malicious adversary with unbounded computational power that actively adapts its behavior. On the other hand, in the PExact model the only role of the adversary is to choose a target and a distribution. After that the learning algorithm starts learning without any additional adversarial influence.

For removing randomness from the PExact model, we introduce a new variation of the model introduced by Ben-David et. al. in [3]. We call this the *Ordered Exact* (OExact) model. This model is similar to the PExact model, where instead of a distribution function we have an ordered set. Each time the OExact oracle gets an equivalence query, it returns the lowest indexed counterexample, instead of randomly or maliciously choosing one.

Another model we consider in this work is the random-PExact model, introduced by Bshouty and Gavinsky [7]. The random-PExact model is a relaxation of the PExact model that allows the learner to use random hypotheses. We will show that for every algorithm

A that uses some restricted random hypothesis for efficiently learning the concept class C in the random-PExact model, there exists an algorithm ALG that efficiently learns C in the Exact model.

In additional we show that the adversary does not gain any additional power by knowing all coin tosses in advance. In other words, we show that offline-Exact learning = Exact learning.

In [8] Bshouty et al. showed that Exact-learnable $\Rightarrow$ PExact-learnable $\Rightarrow$ PAC-learnable. Based on Blum construction [4] they also showed that under the standard cryptographic assumptions (that one-way functions exist), PExact-learnable $\neq$ PAC-learnable. In [7], Bshouty and Gavinsky showed that under polybit distributions, PExact-learnable = PAC-learnable. In this work we will exploit the exponential probabilities to show that PExact-learnable $\Rightarrow$ Exact-learnable.

Another model residing between the PAC model and the PExact model is the PAExact model introduced by Bshouty et al. in [8]. The PAExact model is similar to the PExact model, but allows the learner some exponentially small final error (as opposed to the exact target identification required in PExact). Bshouty and Gavinsky [7] showed that PAExact-learnable = PAC-learnable using boosting algorithms based on [11] and [10]. In [6], Bshouty improves the error factor and gives a more simple algorithm for boosting process.

The following chart indicates relations between the models.

$$\begin{array}{ccc} Exact & & PAExact \\ \| & \overset{\Rightarrow}{\not\Leftarrow} & \| \\ PExact & & PAC \end{array}$$

We note that this work represents results independently obtained by the authors. This joint publication has evolved from a manuscript by Avi Owshanko; the other author's original manuscript [9] may be found at his web page.

2 Preliminaries

In the following we formally define the models we use. We will focus on exact learning of concept classes. In this setting, there exists some learning algorithm A with the goal of exactly identifying some target concept t out of the concept class C over a domain X. In this paper we consider only finite and countable infinite domains X. The learner A has full knowledge of the domain X and of the concept class C, but does not have any a-priory knowledge about the target class t. As each concept $t \in C$ is a subset of the domain X, we will refer to it as a function $t : X \to \{0, 1\}$.

For learning the target concept, the learner can ask some teacher (also referred to as an oracle) several kinds of queries about the target. The teacher can be regarded as an adversary with unlimited computational power and full knowledge of all that the learner knows. The adversary must always answer queries honestly, though it may choose the worst (correct) answer. If the adversary knows in advance all the learner's coin tosses, we call the adversary an *offline adversary* and call the model an *offline-model*.

In this paper we will focus on efficient learning under several models. Whenever we write *efficient learning* of some target t with success probability δ, we mean that the learning algorithm receives the answer "Equivalent" after time polynomial in $size_C(t)$, $\log(1/\delta)$ and b (the size of the longest answer that the teacher returns).

We now give the formal definitions of Exact learning [12], PExact learning [5] and a new model we denote OExact (which is a variation over the model considered in [3]).

We say that a concept class C is *learnable* in some model if there exists some algorithm A such that for every target $t \in C$, and each confidence level δ, A efficiently learns t with the help of the teacher, with success probability greater than $1-\delta$. We say that a learner is *random* if it uses coin tosses and *deterministic* otherwise.

In the ***Exact*** model, the learner A supplies the adversary with some hypothesis h (such that h can be computed efficiently for every point x in X) and the adversary either says "Equivalent", or returns a counterexample, $x \in X$ such that $t(x) \neq h(x)$.

In the ***PExact*** (probably exact) model, the PExact teacher holds some probability distribution D over X, as well as the target $t \in C$. Both the target and the distribution functions are determined before the learning process starts and stay fixed for the duration of the learning process. The learner can supply the teacher with some hypothesis h and the teacher either returns "Equivalent" (when $Pr_D[h(x) \neq t(x)] = 0$), or returns some counterexample x. The counterexample is randomly chosen, under the distribution D induced over all erroneous points $x \in X$ (that is $h(x) \neq t(x)$).

In the ***OExact*** (ordered exact) model, the OExact oracle holds some finite well ordered set $S \subseteq X$. For each query of the algorithm A , the OExact oracle returns $x \in S$ where x is the smallest member of S such that $h(x) \neq t(x)$. For every member $x \in S$, we let $Ord(S,x)$ denotes the number of elements in S that are smaller than x (for example, for x_0 the smallest member of S, $Ord(S,x_0)=0$).

For the PExact model, There exists some relaxed variation of the PExact model, denoted ***random-PExact***, introduced by Bshouty and Gavinsky [7]. In this setting, the algorithm A may use a random hypothesis. A random hypothesis $h_r : X \times R \to \{0,1\}$ is a function such that for every input $x_0 \in X$ it randomly uniformly chooses $r_0 \in R$ and returns $h_{r_0}(x_0)$. As before, the teacher may either answer "Equivalent" (when $\forall x \in X : Pr_D[h_r(x) \neq t(x)] = 0$) or returns some counterexample x. For choosing the counterexample, the teacher keeps randomly choosing points x in X according to the distribution D until the first point such that $h_r(x) \neq t(x)$. For the Exact (OExact) model, the adversary returns some (the smallest) point $x \in X$ ($x \in S$) such that $Pr[h_r(x) \neq t(x)] > 0$.

We will also use the following inequality:

Theorem 1 (Chernoff inequality) *Let $Y_1, Y_2, \ldots, Y_n$ be n independent random variables such that for $1 \leq i \leq n$, $\mathbf{Pr}[Y_i = 1] = p_i$, where $0 < p_i < 1$. Then, for $Y = \Sigma_{i=1}^{n} Y_i$, $\mu = \mathbf{E}[Y] = \Sigma_{i=1}^{n} p_i$, and $0 < \lambda \leq 1$,*

$$Pr[Y < (1-\lambda)\mu] < e^{-\mu\lambda^2/2}$$

3 The Learning Algorithm

In this section we introduce a scheme relying on majority vote to turn every algorithm A that efficiently learns a concept class C in the PExact model into an algorithm ALG that can learn C in the Exact model.

We will rely on the fact that you can fool most of the people some of the time, or some of the people most of the time, but you can never fool most of the people most of the time. Consider some algorithm A where for every target $t \in C$, there exists some bound T, such that A makes no more than T mistakes with some probability p. When we run two copies of A , the probability that both make mistakes on the same T points (in the same order) is p^2. When running k copies of A , the probability that all make mistakes on the same points is p^k. But this fact is not enough for building a new algorithm, because it is not enough for us to know that there is a possible error, but we need to label every point correctly. Hence we need to have that the number of points such that more than half the running copies of A mislabel is bounded by some factor of T. We will prove that if A is an efficient PExact algorithm, then there exists some such (efficient) bound T for every target $t \in C$, and that the number of errors is no more than $4T$.

Because the learner does not know the target t in advance, it must find this bound T dynamically, using a standard doubling technique — each iteration doubling the allowable mistakes number (and the number of copies of A) until successfully learning t. The full algorithm can be viewed in figure 1

We start by showing that A is an efficient learning algorithm in the OExact model. That way, we can remove the element of randomness that is inherent to the PExact model.

Lemma 2 *If A learns every target t in C using less than $T(t)$ steps, with the aid of a PExact teacher with confidence greater than* 0.95*, then there exists an algorithm A', (a copy of A), that learns every target t in C using less than $T(t)$ steps, with the aid of an OExact teacher with confidence greater than* 0.9.

Proof: In this proof we build for every well ordered set S and every target $t \in C$ a step probability function D_S that will force the PExact oracle to behave the same as the OExact oracle (with high probability).

We will run both algorithms A and A' in parallel, where both use the same random strings (when they are random algorithms). Let k be the size of S and let l denotes $T(t)$. We define the probability distribution D_S as follows (recall that $Ord(S, x)$ denotes the number of elements in S that are smaller than x).

$$D_S(x) = \begin{cases} 0 & x \notin S \\ \frac{(40l+2)^{-Ord(S,x)}}{\Sigma_{i=1}^{k}(40l+2)^{-i}} & x \in S \end{cases}$$

Consider the case that both A and A' ask their teachers some equivalence query using the same hypothesis h. Let x be the counterexample that the OExact teacher returns to A'. By definition of the OExact model, x is the smallest counterexample in S. The probability that the PExact teacher returns to A a counterexample y such that $y \neq x$ (and $Ord(S, y) > Ord(S, x)$) is less than

$$\sum_{j=Ord(S,x)+1}^{k} \frac{(40l+2)^{-j}}{\Sigma_{i=1}^{k}(40l+2)^{-i}} < \frac{2}{40l+2} \cdot \frac{(40l+2)^{-Ord(S,x)}}{\Sigma_{i=1}^{k}(40l+2)^{-i}} = \frac{D_S(x)}{20l+1}$$

Hence, the PExact oracle returns the lowest indexed counterexample x with probability greater than $1 - \frac{1}{20l}$.

```
ALG
 1. Init $k \leftarrow 4$
 2. *Do*
 3.     Init $\mathcal{P} \leftarrow \emptyset$, $count \leftarrow 0$
 4.     Let $\mathcal{A} \leftarrow \{\mathcal{A}_\infty, \mathcal{A}_\in, \dots, \mathcal{A}_{||}\}$ [where each $A_i$ is a copy of $A$]
 5.     Init each copy $A_i$ in $\mathcal{A}$.
 6.     While ($count < k$)
 7.         Run each copy $A_i \in \mathcal{A}$ until it asks an equivalence query,
              terminates, or executed more than $k/4$ steps.
 8.         Remove from $\mathcal{A}$ all copies $A_i$ that either terminated unsuccessfully
              or executed more than $k/4$ steps.
 9.         If there exists some copy $A_i \in \mathcal{A}$ asking an equivalence query with
              an hypothesis $h_i$ that is not consistent with $\mathcal{P}$
10.             Let $(y, label(y), c)$ be a counterexample with the lowest index $c$.
11.             Give $(y, label(y))$ as a counterexample to $A_i$
12.             Jump back to step 7
13.         End If
14.         Let $h = majority\{h_1, h_2, \dots, h_k\}$
              [where $h_i$ is $A_i$'s hypothesis at this point].
15.         Let $x \leftarrow EQ(h)$. If $x =$ "Equivalent", return $h$ as the answer
              else, Add $(x, label(x), count)$ to $\mathcal{P}$
16.         Let $count \leftarrow count + 1$
17.     End While
18.     Let $k \leftarrow 2k$
19. While the hypothesis $h$ is not equivalent to the target.
```

Figure 1: The learning algorithm

We can conclude that the PExact and the OExact teachers return the same answer with probability greater than $1 - \frac{1}{20l}$, and the probability for l such consequent answers is greater than

$$\left(1 - \frac{1}{20l}\right)^l \approx e^{-1/20} > 0.95.$$

Because both A and A' hold the same random string, they will both behave the same (ask the same queries) until the first time that the teachers give different answers. On the other hand, A learns t using less than $T(t)$ steps with confidence of 0.95. So we can conclude that with confidence greater than $0.95 \cdot 0.95 > 0.9$, A' learns t in the OExact model using less than $T(t)$ steps. ■

Our next step is to show that if A is an efficient OExact learning algorithm, then ALG learns C in the Exact model.

Lemma 3 *Let X be a finite domain. If A learns every class t in C using less than $T(t)$ steps, with the aid of an OExact teacher with confidence level greater than 0.9, then ALG learns every t in C with the aid of an offline-exact teacher, with probability greater than $1 - \delta$ using less than*

$$O((\log(1/\delta) + T(t)\log(|X|) + \log(|C|))^2)$$

steps.

Proof: Let l denotes $T(t)$ and let $m \geq m_0 = 20(\ln(1/\delta) + 4l\ln(|X|) + \ln(|C|))$. Consider running $3m$ copies of the learning algorithm A , over some given ordered set S of size $4l$. We shall calculate the probability that m of these copies need more than l steps to exact learn t.

Using Chernoff inequality (1), we have $n = 3m$, $\mu = 0.9 \cdot 3m = 2.7m$, and $\lambda > 0.2$:

$$Pr[Y < 2m] < e^{-2.7m\cdot(0.2)^2/2} < e^{-0.05m} \leq \frac{\delta}{|X|^{4l} \cdot |C|}. \quad (1)$$

Next we define the following property:
property I: *The probability that there exists some target $t \in C$ and some ordered set S of size $4l$ such that more than m copies of A will need more than l steps to learn t is less than δ.*

The reasoning behind this claim is as follows. Assume that all $3m$ copies of A have a sequence of random bits. We let the adversary know these random bits and look for some target $t \in C$ and some ordered set S that will cause more than m copies to fail. The number of possible target concepts $t \in C$ is $|C|$ and the number of possible ordered sets is less than $|X|^{4l}$. On the other hand, the probability for some set to cause more than m copies to fail for some target t is less than $\frac{\delta}{|X|^{4l}\cdot|C|}$ by (1). Hence the probability for the existence of such a bad target t and ordered set S is less than

$$\frac{\delta}{|X|^{4l} \cdot |C|} \cdot |X|^{4l} \cdot |C| = \delta.$$

and property I holds.

We now consider ALG 's main loop (steps 6-17 in figure 1) when $6m_0 \geq k \geq 3m_0$ (ALG reaches this loop after after $O(k^2)$ steps, unless it already received the answer "Equivalent"). Assume that ALG receives $4l$ counterexamples in this loop (recall that $4l < k$).

Note that this set of counterexamples defines an ordered set S of size $4l$ (we order the counterexamples chronologically). Because each such counterexample is given to at least half the currently running copies of A , at least m copies of A received at least l counterexamples (or executed more than $k/4 > l$ steps). But property I states that there exists such a set of counterexamples with probability smaller than δ.

So we conclude that with probability greater than $1-\delta$, ALG learns t in the Exact model when $6m_0 \geq k$, where the number of steps is bounded by

$$O(m_0^2) = O((\log(1/\delta) + T(t)\log(|X|) + \log(|C|))^2).$$

■

Our next step is to remove the size of the domain X and the concept class C from the complexity analysis.

Lemma 4 *If A learns every class t in C using less than $T(t)$ steps, with the aid of an OExact teacher with confidence level greater than* 0.9*, then ALG learns every t in C with the aid of an offline-exact teacher, with probability greater than $1-\delta$ using less than*

$$O((\log(1/\delta) + T(t)(size(t) + b))^2)$$

steps, where b is the size of the longest counterexample that the teacher returns.

Proof: For some set Q, we let Q^b denotes all members of Q that are represented by no more than b bits. By definition, $|Q^b| < 2^{b+1}$. By lemma 3, there exists some constant c, such that for every finite domain X, ALG learns every t in C with the aid of an offline-exact teacher with probability greater than $1-\delta$ using less than $c \cdot (\log(1/\delta) + T(t)\log(|X|) + \log(|C|))^2$ steps.

Let us consider the case that the longest counterexample b, or the size of the target t ($size_C(t)$) is at least 2^i and less than 2^{i+1}. We let d denotes 2^i. So we have that $d < size(t)+b$. Applying lemma 3, we get that ALG learns t with probability greater than $1-\delta/d$, using less than

$$\begin{aligned} & c \cdot (\log(1/\delta) + T(t)\log(|X|) + \log(|C|))^2 \\ < \quad & c \cdot (\log(1/\delta) + T(t)\log(d) + log(d))^2 \\ = \quad & c \cdot (\log(1/\delta) + (T(t)+1)(size(t)+b))^2 \end{aligned}$$

steps. Hence, the probability to find some $d = 2^i$ such that ALG will be forced to use more than $c \cdot (poly(size(t)) \cdot \log^2(d/\delta) \cdot 16d^4)$ steps is less than:

$$1 - \prod_{i=1}^{\infty}(1 - \frac{\delta}{2^i}) \leq \sum_{i=1}^{\infty}(\frac{\delta}{2^i}) \leq \delta$$

and the lemma holds. ■

At this point we can conclude that:

Theorem 5 *PExact = offline-Exact learning.*

Proof: This theorem immediately follows from lemmas 2 and 4. In lemma 2 we showed that every algorithm A that efficiently learns the class C in the PExact model with probability greater than 0.95 also efficiently learns C in the OExact model with probability greater than 0.9. In lemma 4 we showed that if A efficiently learns C in the OExact model with probability greater than 0.9, the algorithm ALG efficiently learns C in the offline-Exact model with any needed confidence level $1 - \delta$. On the other hand, Bshouty et. al. [8] already showed that $Exact \Rightarrow PExact$. Hence the theorem holds. ■

An additional interesting result following immediately from theorem 5 is:

Corollary 6 *Exact = offline-Exact learning.*

4 Handling the Random Model

We now show that if A is an efficient algorithm for learning C in the random-PExact model and if A follows some constraints, then ALG learns C in the Exact model. Namely, we will show that if we can efficiently determine for every hypothesis h_r that A produces and for every $x \in X$ whether $0 < E[h_r(x)] < 1$ or not, then if A learns C in the random-PExact model, ALG learns C in the Exact model. As in the previous section, we start by showing that random-PExact = OExact.

Lemma 7 *If A efficiently learns C in the random-PExact model with probability greater than* 0.95*, then A efficiently learns C in the OExact model with probability greater than* 0.9*.*

Proof: This proof is similar to that of Lemma 2. For every target $t \in C$ and every order $S \in X$ we build a step distribution function that will force the random-PExact oracle to behave in the same way as the OExact oracle.

Let k be the size of S and assume that that A needs $l = poly(size(t))$ Consider running A for l steps in the OExact model until A executes l steps (or terminates successfully). Let h_r^i denotes A's hypothesis after the i's step. Because the number of steps is bounded by l, there exists some $0 < \lambda < 1$ such that for all members $x \in S$ and all steps $0 \leq i \leq l$,

$$(E[h_r^i(x)] = 0) \vee (E[h_r^i(x)] = 1) \vee (\lambda < E[h_r^i(x)] < 1 - \lambda).$$

Using this value λ, We define the probability distribution D_S as follows

$$D_S(x) = \begin{cases} 0 & x \notin S \\ \left(\frac{\lambda}{40l+2}\right)^{Ord(S,x)} \cdot \frac{1}{\Sigma_{i=1}^{k}\left(\frac{\lambda}{40l+2}\right)^{i}} & x \in S \end{cases}$$

For every x member of S , We let $Y(x) \subset S$ denotes all members of S larger than x in the order S. By definition of D_S, we have

$$D_S(x) > \frac{\lambda}{20l+1} \cdot \Sigma_{y \in Y(x)} D_S(y).$$

From this point on, the proof is similar to that of Lemma 2. The probability to receive the smallest possible x as the counterexample in the random-PExact model under the probability distribution D_S is (at least) $\frac{1}{20l+1}$, and the probability that the random-PExact oracle behaves the same as the OExact oracle for all l steps is greater than 0.95. So we conclude that A learns C in the OExact model with probability greater than 0.9. ■

After we showed that random-PExact = OExact, we can apply the same proofs as in the previous section to receive the following result:

Theorem 8 *If A efficiently learns C in the random-PExact model, and if for every hypothesis h_r that A holds and every $x \in X$ we can (efficiently) determine whether $0 < E[h_r(x)] < 1$ or not, then ALG efficiently learns C in the Exact model.*

Proof: The proof is similar to that of the theorem 5. We can still emulate the way that the OExact oracle behaves, because for every hypothesis h_r and every $x \in X$ we can efficiently determine whether $0 < E[h_r(x)] < 1$ or not. When h_r can assign x both values, we can give x as a counterexample. Otherwise, we can choose any random string r (for example all bits are zero) and calculate the value of $h_r(x)$. Also note that if x is a counterexample for ALG , then at least half of the running copies of A can receive x as a counterexample. So we can use both lemmas 2 and 4. The rest of the proof is similar. ■

5 Conclusions and Open Problems

In this paper we showed that PExact = Exact learning, thus allowing the use of a model without an adaptive adversary, in order to prove computational lower bounds. We also showed that a limited version of random-PExact is equivalent to that of the Exact model. An interesting question left open is whether the random-PExact is strictly stronger than the Exact model or not (assuming that $P \neq NP$).

The second result we gave is that even when the adversary knows all the learner's coin tosses in advance (the offline-Exact model), it does not gain any additional computational power. This results also holds when the learner has the help of a membership oracle, but it is not known whether this still holds when the membership oracle is limited, such as in [2].

References

[1] D. Angluin. Queries and concept learning. *Machine Learning*, 75(4):319-342, 1988.

[2] D. Angluin and D. Slonim. (1994). Randomly Fallible Teachers: Learning Monotone DNF with an Incomplete Membership Oracle. *Machine Learning*, 14:7-26.

[3] Shai Ben-David, Eyal Kushilevitz, Yishay Mansour. Online Learning versus Offline Learning. *Machine Learning* 29(1): 45-63, 1997.

[4] A. Blum. Separating distribution-free and mistake-bound learning models over the boolean domain. *SIAM Journal on Computing 23(5)*, pp. 990-1000, 1994.

[5] N. H. Bshouty. Exact learning of formulas in parallel. *Machine Learning* 26, pp. 25-41, 1997.

[6] N. H. Bshouty. A Booster for the PAExact Model.

[7] N. H. Bshouty, D. Gavinsky. PAC = PAExact and other Equivalent Models in Learning. *Proceedings of the 43th Annual Symposium on Foundations of Computer Science*, pp. 167-176, 2002.

[8] N. H. Bshouty, J. Jackson, C. Tamon. Exploring learnability between exact and PAC. *Proceedings of the 15th Annual Conference on Computational Learning Theory*, 2002.

[9] D. Gavinsky. Exact = PExact. 2004.
http://pages.cpsc.ucalgary.ca/∼gavinsky/papers/papers.html

[10] Y. Mansour and D. McAllester, Boosting using Branching Programs, Proceedings of the 13th Annual Conference on Computational Learning Theory, pp. 220-224, 2000.

[11] R. E. Schapire, The strength of weak learnability, *Machine Learning, 5(2)* pp. 197-227, 1990.

[12] L. G. Valiant. (1984) A theory of the learnable. *communications of the ACM, 27:1134-1142*.