On Learning vs. Refutation

Citation	Vadhan, Salil, P. 2017. On Learning vs. Refutation. Proceedings of Machine Learning Research, vol. 65: 1835-1848.
Published Version	http://proceedings.mlr.press/v65/vadhan17a.html
Citable link	http://nrs.harvard.edu/urn-3:HUL.InstRepos:34387110
Terms of Use	This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Open Access Policy Articles, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#OAP

Proceedings of Machine Learning Research vol 65:1–14, 2017

On Learning versus Refutation

Salil P. Vadhan[*] SALIL@SEAS.HARVARD.EDU
School of Engineering & Applied Sciences, Harvard University, Cambridge, Massachusetts, USA,
webpage: http://seas.harvard.edu/~salil

Abstract

Building on the work of Daniely et al. (STOC 2014, COLT 2016), we study the connection between computationally efficient PAC learning and refutation of constraint satisfaction problems. Specifically, we prove that for every concept class $\mathcal{P}$, PAC-learning $\mathcal{P}$ is *polynomially equivalent* to "random-right-hand-side-refuting" ("RRHS-refuting") a dual class $\mathcal{P}^*$, where RRHS-refutation of a class $\mathcal{Q}$ refers to refuting systems of equations where the constraints are (worst-case) functions from the class $\mathcal{Q}$ but the right-hand-sides of the equations are uniform and independent random bits. The reduction from refutation to PAC learning can be viewed as an abstraction of (part of) the work of Daniely, Linial, and Shalev-Schwartz (STOC 2014). The converse, however, is new, and is based on a combination of techniques from pseudorandomness (Yao '82) with boosting (Schapire '90).

In addition, we show that PAC-learning the class of *DNF* formulas is polynomially equivalent to PAC-learning its dual class *DNF**, and thus PAC-learning *DNF* is equivalent to RRHS-refutation of *DNF*, suggesting an avenue to obtain stronger lower bounds for PAC-learning *DNF* than the quasipolynomial lower bound that was obtained by Daniely and Shalev-Schwartz (COLT 2016) assuming the hardness of refuting k-SAT.

1. Introduction

Daniely, Linial, and Shalev-Shwartz (2014) recently developed a beautiful new approach to proving computational hardness results for learning, based on the conjectured hardness of refuting constraint-satisfaction problems (in contrast to the cryptographic hardness assumptions used in earlier work, starting with the work of Valiant (1984); Kearns and Valiant (1994)). This was used to give the first evidence of the hardness of PAC learning *DNF* formulas (Daniely and Shalev-Schwartz, 2016) and of agnostically learning halfspaces (Daniely, 2016), both long-standing open problems in computational learning theory. The hardness result for learning *DNF* formulas by Daniely and Shalev-Schwartz (2016) is based on a k-SAT generalization of Feige's conjecture about hardness of refuting random 3-SAT (Feige, 2002). Specifically, the conjecture is that for every constant c, there is a constant k such that no polynomial-time algorithm can prove that a random k-SAT formula with n^c clauses is unsatisfiable (with high probability).

In this paper, we illuminate the connection between PAC learning and refutation. Specifically, we prove that for every concept class $\mathcal{P}$, PAC-learning $\mathcal{P}$ is polynomially *equivalent* to RRHS-refuting a dual class $\mathcal{P}^*$, where *RRHS-refutation* of a class $\mathcal{Q}$ refers to refuting

[*] Written in part while visiting the Shing-Tung Yau Center and the Department of Applied Mathematics at National Chiao-Tung University in Hsinchu, Taiwan. Supported by NSF grant NSF grant CCF-1420938 and a Simons Investigator Award.

systems of equations where the constraints are (worst-case) functions from the class $\mathcal{Q}$ but the right-hand-sides of the equations are uniform and independent random bits. ("RRHS" stands for "random right-hand sides".) The reduction from refutation to PAC learning can be viewed as an abstraction of (part of) the work of Daniely et al. The converse, however, is new, and is based on a combination of techniques from pseudorandomness (Yao, 1982) with boosting (Schapire, 1990).

In addition, we show that PAC-learning the class of *DNF* formulas is polynomially equivalent to PAC-learning its dual class *DNF**, and thus PAC-learning *DNF* is equivalent to RRHS-refutation of *DNF*. Thus, the result of Daniely and Shalev-Schwartz (2016) can be obtained by reducing the refutation of random k-SAT formula to RRHS-refuting *DNF*. An elegant such reduction is implicit in the work of Daniely, Linial, and Shalev-Shwartz (2014); Daniely and Shalev-Schwartz (2016) and we present it explicitly here.

One limitation of the hardness result of Daniely and Shalev-Schwartz (2016) is that, even under the strongest plausible conjecture about refuting random k-SAT (namely that $n^{\Omega(k)}$ clauses are needed for efficient refutation), at best it provides a quasipolynomial lower bound on the complexity of learning *DNF*, which is quite far from the best known algorithm, which is due to Klivans and Servedio (2004) and runs in time $\exp(\tilde{O}(n^{1/3}))$. Our connection between PAC-learning *DNF* and RRHS-refuting *DNF* suffers no such limitation; if RRHS-refuting *DNF* requires exponential time, then so does PAC-learning *DNF*. Thus, a natural direction for future work is to relate the hardness of RRHS-refuting *DNF* to other, more well-studied refutation problems. In addition, our general connection between learning and refutation can be applied to other classes, and translating ideas and techniques between the two areas can potentially yield additional insights into both learning and refutation.

We also compare hardness of RRHS-refutability and different types of cryptographic hardness assumptions that were shown to imply hardness of PAC learning in the past, to help illuminate the source of the power of the approach of Daniely et al. (2014).

2. Learning vs. Refutation

An evaluation function $\mathrm{Eval} : \{0,1\}^s \times \{0,1\}^t \to \{0,1\}$ gives rise to two dual families of predicates: $\mathcal{P} = \{p_x : \{0,1\}^t \to \{0,1\}\}_{x\in\{0,1\}^s}$ where $p_x(y) = \mathrm{Eval}(x,y)$, and $\mathcal{Q} = \{q_y : \{0,1\}^s \to \{0,1\}\}_{y\in\{0,1\}^t}$ where $q_y(x) = \mathrm{Eval}(x,y)$. We indicate the duality between $\mathcal{P}$ and $\mathcal{Q}$ by writing $\mathcal{Q} = \mathcal{P}^*$ and $\mathcal{P} = \mathcal{Q}^*$.

For example, we will consider $\mathcal{P} = DNF_{s,t}$ be the class of size s DNF formulas on t variables, where size is measured under some fixed encoding of DNF formulas as binary strings. We are typically interested in infinite families of predicates, obtained by varying the parameters s and t, and providing these as input to all of the algorithms. For example, $DNF = \{DNF_{s,t}\}_{s,t\in\mathbb{N}}$ denotes the class of all DNF formulas.

It will be convenient work with the following formulation of PAC learnability.

Definition 1 *$\mathcal{P}$ is* PAC learnable with sample complexity $m = m(s,t)$ *if there is a polynomial-time algorithm $\mathcal{A}$ such that for every $s,t \in \mathbb{N}$, $x \in \{0,1\}^s$ (specifying $p_x \in \mathcal{P}$) and every distribution $\mathcal{D}$ on $\{0,1\}^t$, if we sample $y_1,\ldots,y_{m+1} \leftarrow \mathcal{D}$ (for $m = m(s,t)$), we have*

$$\Pr[\mathcal{A}(1^s, 1^t, (y_1, p_x(y_1)), \ldots, (y_m, p_x(y_m)), y_{m+1}) = p_x(y_{m+1}) \geq 2/3,$$

where the probability is over the choice of the y_i's and the coins of $\mathcal{A}$. We say that $\mathcal{P}$ is PAC learnable *if $m(s,t) \le \mathrm{poly}(s,t)$.*

The inputs 1^s and 1^t are unary inputs to inform the algorithm $\mathcal{A}$ of the size parameters and allow it running time polynomial in these parameters. Note that we can also apply the definition when the sample complexity m is super-polynomial, in which case we allow $\mathcal{A}$ running time $\mathrm{poly}(s,t,m)$. (One could separate the running time and sample complexity into two separate parameters, but we avoid doing so for notational simplicity.) In the more standard formulation of PAC learning, the learning algorithm $\mathcal{A}$ is not asked to simply predict the value of the concept p_x on a single example y_{m+1}, but rather to produce a hypothesis h that approximately agrees with p_x under the distribution $\mathcal{D}$. In that formulation, there are two confidence parameters: an error parameter ϵ that bounds the disagreement between h and p_x, and a confidence parameter δ that bounds the probability that $\mathcal{A}$ outputs a hypothesis with error larger than ϵ. Definition 1 implicitly incorporates both ϵ and δ into the constant $2/3$. In particular, a learner satisfying Definition 1 directly yields a standard PAC learner with error parameter $\epsilon = 1/3 + 1/12 = 5/12$ and $\delta = 1 - 1/12$ (i.e. with probability at least $1/12$, we obtain a hypothesis $h(\cdot) = \mathcal{A}(1^s, 1^t, (y_1, p_x(y_1)), \ldots, (y_m, p_x(y_m)), \cdot)$ that has agreement at least $7/12$ with p_x). Such a learner implies a standard PAC learner (with run time depending polynomially on $1/\epsilon$ and $\log(1/\delta)$) by boosting (Schapire, 1990). (See also Kearns and Valiant (1994, Ch. 4).) Similarly, boosting implies that PAC learnability is equivalent to the following formulation of weak learnability:

Definition 2 *$\mathcal{P}$ is* weakly PAC learnable with advantage $\alpha = \alpha(s,t)$ and sample complexity $m = m(s,t)$ *if there is a polynomial-time algorithm $\mathcal{A}$ such that for every $s, t \in \mathbb{N}$, $x \in \{0,1\}^s$ (specifying $p_x \in \mathcal{P}$) and every distribution $\mathcal{D}$ on $\{0,1\}^t$, if we sample $y_1, \ldots, y_{m+1} \leftarrow \mathcal{D}$ (for $m = m(s,t)$), we have*

$$\Pr[\mathcal{A}(1^s, 1^t, (y_1, p_x(y_1)), \ldots, (y_m, p_x(y_m)), y_{m+1}) = p_x(y_{m+1}) \ge (1+\alpha)/2,$$

for $\alpha = \alpha(s,t)$, where the probability is over the choice of the y_i's and the coins of $\mathcal{A}$. We say that $\mathcal{P}$ is weakly PAC learnable *if $\alpha(s,t) \ge 1/\mathrm{poly}(s,t)$ and $m(s,t) \le \mathrm{poly}(s,t)$.*

Theorem 3 (Schapire (1990)) *If $\mathcal{P}$ is weakly PAC learnable with advantage α and sample complexity m, then $\mathcal{P}$ is PAC learnable with sample complexity $m \cdot \mathrm{poly}(1/\alpha)$.*

Now we turn to defining refutation problems for the dual class $\mathcal{Q}$. Informally, $\mathcal{Q}$ is *RRHS-refutable* (where RRHS stands for "random right-hand side") if we can efficiently refute the satisfiability of systems of equations of the form $q_{y_1}(x) = b_1, \ldots, q_{y_n}(x) = b_n$ when the right-hand side values b_i are chosen uniformly at random.

Definition 4 *$\mathcal{Q}$ is* RRHS-refutable using $n = n(s,t)$ equations *if there is a polynomial-time algorithm $\mathcal{B}$ such that:*

1. *(Soundness) for every $y_1, \ldots, y_n \in \{0,1\}^t$, $b_1, \ldots, b_n \in \{0,1\}$, if the system of equations $p_{y_1}(x) = b_1, \ldots, p_{y_n}(x) = b_n$ is satisfiable, then*

$$\Pr[\mathcal{B}(1^s, 1^t, (y_1, b_1), \ldots, (y_n, b_n)) = 1] \le 1/3.$$

2. *(Completeness) for every $y_1, \ldots, y_n \in \{0,1\}^t$, if we randomly choose $b_1, \ldots, b_n \leftarrow \{0,1\}$, then*
$$\Pr[\mathcal{B}(1^s, 1^t, (y_1, b_1), \ldots, (y_n, b_n)) = 1] \geq 2/3,$$
where the probability is taken over $b_1, \ldots, b_n$ and the coins of $\mathcal{B}$.

We say that $\mathcal{Q}$ is $\mathcal{Q}$ is RRHS-refutable *if $n(s,t) \leq \mathrm{poly}(s,t)$.*

Theorem 5 *Let $\mathcal{P} = \mathcal{Q}^*$ be a family of predicates given by evaluation function* Eval : $\{0,1\}^s \times \{0,1\}^t \to \{0,1\}$. *Then:*

1. *If $\mathcal{P}$ is PAC learnable (with sample complexity m), then $\mathcal{Q}$ is RRHS-refutable (using $n = O(m)$ equations).*

2. *If $\mathcal{Q}$ is RRHS-refutable (using n equations), then $\mathcal{P}$ is PAC learnable (with sample complexity $m = \mathrm{poly}(n)$).*

Proof

1. Let $\mathcal{A}$ be the PAC learner with sample complexity m, set $n = 10m$, and define $\mathcal{B}(1^s, 1^t, (y_1, b_1), \ldots, (y_n, b_n))$ as follows:

   (a) Randomly choose $i_1, \ldots, i_{m+1} \leftarrow [n]$ (with replacement).

   (b) Output 1 iff $\mathcal{A}(1^s, 1^t, (y_{i_1}, b_{i_1}), \ldots, (y_{i_m}, b_{i_m}), y_{i_{m+1}}) \neq b_{i_{m+1}}$.

   For soundness, observe that if the system of equations $q_{y_1}(x) = b_1, \ldots, q_{y_n}(x) = b_n$ is satisfiable by assignment x, then the samples given to $\mathcal{A}$ are iid draws from a distribution of examples (namely the uniform distribution on the multiset $\{y_1, \ldots, y_n\}$) labelled by p_x (since $b_i = q_{y_i}(x) = p_x(y_i)$), and hence $\mathcal{A}$ will output $p_x(y_{i_{m+1}}) = b_{i_{m+1}}$ with probability at least 2/3 (and $\mathcal{B}$ will output 1 with probability at most 1/3, as desired).

   For completeness, observe that if the bits $b_1, \ldots, b_n$ are chosen uniformly and independently at random, then $\mathcal{A}$ has no information about $b_{i_{m+1}}$ unless $i_{m+1} \in \{i_1, \ldots, i_m\}$, which happens with probability at most $m/n = 1/10$. So the probability that $\mathcal{A}$ outputs $b_{i_{m+1}}$ is at most $1/10 + 1/2 = .6$, and hence $\mathcal{B}$ outputs 1 with probability at least .4. The gap between 1/3 and .4 for $\mathcal{B}$ can be amplified to 1/3 and 2/3 by a constant number of repetitions (increasing n by the same constant factor).

2. Let $\mathcal{B}$ be the RRHS-refuter using n equations. We will construct a *weak* PAC learner $\mathcal{A}$ with sample complexity $m = n - 1$ from $\mathcal{B}$ and then apply boosting to obtain a full-fledged PAC learner. Intuitively, the definition of RRHS-refuter means that $\mathcal{B}$ can distinguish the sequence $q_{y_i}(x) = p_x(y_i)$ for $i = 1, \ldots, n$ from a sequence of n independent random bits. Thus, by Yao's equivalence between pseudorandomness and next-bit unpredictability (Yao, 1982), $\mathcal{B}$ can be used to predict $p_x(y_i)$ for a random $i \leftarrow [n]$ from $(p_x(y_1), \ldots, p_x(y_{i-1}))$ with probability noticeably more than 1/2, yielding a weak learner.

   Specifically, we use the following formulation of Yao's result (which shows that next-bit unpredictability implies pseudorandomness):

Lemma 6 (implicit in Yao (1982)) *For every probabilistic algorithm $\mathcal{T} : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}$ (which we think of as a "statistical test" or "distinguisher"), there is a probabilistic algorithm $\mathcal{T}' : \{0,1\}^\ell \times \{0,1\}^{\le n-1} \to \{0,1\}$ (which we think of a "next-bit predictor") whose running time is at most an additive $O(n)$ larger than that of $\mathcal{T}$ and has the following property.*

Suppose that (Y, B) is a random variable distributed arbitrarily on $\{0,1\}^\ell \times \{0,1\}^n$ that $\mathcal{T}$ distinguishes from (Y, C), where C is distributed uniformly on $\{0,1\}^n$ independent of Y, with advantage at least $\alpha > 0$. That is,

$$\Pr[\mathcal{T}(Y, B) = 1] - \Pr[\mathcal{T}(Y, C) = 1] \ge \alpha,$$

where the probabilities are taken over Y, B, C, and the coin tosses of $\mathcal{T}$. Then $\mathcal{T}'$ is a next-bit predictor for B with advantage at least α/n. That is,

$$\Pr[\mathcal{T}'(Y, B_1, B_2, \ldots, B_{I-1}) = B_I] \ge (1 + \alpha/n)/2,$$

where the probability is taken over Y, B, $I \leftarrow [n]$, and the coins of $\mathcal{T}'$.

Specifically, $\mathcal{T}'$ operates as follows: on input $(y, b_1, \ldots, b_{i-1})$, randomly choose $c_i, c_{i+1}, \ldots, c_n \leftarrow \{0,1\}$, run $\mathcal{T}(y, b_1, \ldots, b_{i-1}, c_i, \ldots, c_n)$. If $\mathcal{T}$ outputs 1, then output c_i, else output $\neg c_i$.

To apply Yao's lemma, we take:

- $\ell = n \cdot t$, so $\{0,1\}^\ell = (\{0,1\}^n)^t$,
- $\mathcal{T}((y_1, \ldots, y_n), b_1, \ldots, b_n) = \mathcal{B}(1^s, 1^t, (y_1, b_1), \ldots, (y_n, b_n))$,
- $Y = (Y_1, \ldots, Y_n)$ for n iid samples Y_i from the unknown distribution $\mathcal{D}$ on examples being fed to our learner,
- $B = (p_x(Y_1), \ldots, p_x(Y_n))$, i.e. the labels of the examples under concept p_x, and
- $\alpha = 2/3 - 1/3 = 1/3$.

The fact that $\mathcal{B}$ is a RRHS refuter for $\mathcal{Q}$ implies that the hypothesis of Lemma **??** is satisfied. Note that the construction of the next-bit predictor $\mathcal{T}'$ does not depend on the random variable B, which is important for us since the concept p_x is unknown to our learner. By the lemma, the following is a weak PAC learner (satisfying Definition 2) for $\mathcal{P}$ with advantage at least $\alpha/n = 1/3n$:

(a) Choose $i \leftarrow [n]$. Request $i-1$ labelled examples $(y_1, b_1), \ldots, (y_{i-1}, b_{i-1})$, and $n-i$ unlabelled examples $y_{i+1}, \ldots, y_n$. Let y_i be the challenge example (which $\mathcal{A}$ is supposed to label). (Clearly all this can be simulated by a learner that is simply given $n-1$ labelled examples and the challenge example, but this presentation matches the notation above.)

(b) Choose $c_i, \ldots, c_{n+1}$ uniformly at random.

(c) If $\mathcal{B}(1^s, 1^t, (y_1, b_1), \ldots, (y_{i-1}, b_{i-1}), (y_i, c_i), \ldots, (y_n, c_n)) \ne 1$, then output c_i, otherwise output $\neg c_i$.

By boosting (Theorem 3), we obtain a PAC learner with sample complexity $n \cdot \mathrm{poly}(3n) = \mathrm{poly}(n)$.

■

3. Reductions

For our additional results, regarding the relationship between PAC learnability of *DNF* and refutation, we will use the standard notion of reductions introduced by Pitt and Warmuth (1990) (dubbed "PAC-reducibility" by Kearns and Valiant (1994)).

Definition 7 *Let $\mathcal{P} = \mathcal{Q}^*$ and $\mathcal{P}' = (\mathcal{Q}')^*$ be two classes of predicates given by evaluation functions* $\mathrm{Eval} : \{0,1\}^s \times \{0,1\}^t \to \{0,1\}$ *and* $\mathrm{Eval}' : \{0,1\}^{s'} \times \{0,1\}^{t'} \to \{0,1\}$*, respectively. We say that $\mathcal{P}$* PAC-reduces *to $\mathcal{P}'$, written $\mathcal{P} \leq_{pac} \mathcal{P}'$ if there are polynomials $s' = s'(s,t)$ and $t' = t'(s,t)$, a* $\mathrm{poly}(s,t)$*-time computable function $g = g_{s,t} : \{0,1\}^t \to \{0,1\}^{t'}$ and a (possibly inefficient) function $f = f_{s,t} : \{0,1\}^s \to \{0,1\}^{s'}$ such that for all $s,t \in \mathbb{N}$,*[1] *$x \in \{0,1\}^s, y \in \{0,1\}^t$, we have*

$$\mathrm{Eval}'(f(x), g(y)) = \mathrm{Eval}(x,y).$$

Equivalently $p'_{f(x)}(g(y)) = p_x(y)$, or $q'_{g(y)}(f(x)) = q_y(f(x))$.

Although we allow the function f to be inefficient, we will not take advantage of it in any results. That is, all of the functions f we obtain will be computable in time $\mathrm{poly}(s,t)$. In such a case, the definition of relabelling reduction becomes symmetric between the classes $\mathcal{P}$, $\mathcal{P}'$ and their duals $\mathcal{Q}$, $\mathcal{Q}'$, so we automatically also obtaining a relabelling reduction $\mathcal{Q} \leq_{pac} \mathcal{Q}'$.

Pitt and Warmuth (1990) showed that PAC-reducibility preserves PAC-learnability; we note that it also preserves RRHS-refutability of the dual class (without paying the loss in sample complexity of Theorem 5):

Proposition 8 (Pitt and Warmuth (1990)) *Suppose $\mathcal{Q}^* = \mathcal{P} \leq_{pac} \mathcal{P}' = (\mathcal{Q}')^*$. Then:*

1. *If $\mathcal{P}'$ is PAC-learnable with sample complexity $m' = m'(s',t')$, then $\mathcal{P}$ is PAC-learnable with sample complexity m'.*

2. *If $\mathcal{Q}'$ is RRHS-refutable using $n' = n'(s',t')$ equations, then $\mathcal{Q}'$ is RRHS-refutable with n' equations.*

Proof

1. Let $\mathcal{A}'$ be the PAC learner for $\mathcal{P}'$. Then we can obtain a PAC learner $\mathcal{A}$ for $\mathcal{P}$ by:

   $$\mathcal{A}(1^s, 1^t, (y_1, b_1), \ldots, (y_{m'}, b_{m'}), y_{m'+1}) = \mathcal{A}'(1^{s'}, 1^{t'}, (g(y_1), b_1), \ldots, (g(y_{m'}), b_{m'}), g(y_{m'+1})).$$

   Indeed, if the examples (y_i, b_i) are correctly labelled according to concept $p_x \in \mathcal{P}$, so that $b_i = p_x(y_i)$, then we have $b_i = p'_{f(x)}(g(y_i))$, so the examples $(g(y_i), b_i)$ are correctly labelled according to concept $p'_{f(x)} \in \mathcal{P}'$.

2. Let $\mathcal{B}'$ be the RRHS refuter for $\mathcal{Q}'$. Then we can obtain a RRHS refuter $\mathcal{B}$ for $\mathcal{Q}$ by:

   $$\mathcal{B}(1^s, 1^t, (y_1, b_1), \ldots, (y_{n'}, b_{n'})) = \mathcal{B}'(1^{s'}, 1^{t'}, (g(y_1), b_1), \ldots, (g(y_n), b_n)).$$

[1] The algorithm computing g should get the parameter s in unary, but we omit that from the notation for readability.

> Thus we are transforming the system of equations $q_{y_1}(x) = b_1, \ldots, q_{y_{n'}}(x) = b_{n'}$ into the system of equations $q'_{g(y_1)}(x') = b_1, \ldots, q'_{g(y_{n'})}(x') = b_{n'}$. If there is a satisfying assignment x to the former system, then $x' = f(x)$ is a satisfying assignment to the latter system. And if the right-hand sides of the former system are uniformly random and independent, then the same is true for the latter system.

■

A simple and standard PAC-reduction is the one from DNF to monotone DNF:

Lemma 9 (Kearns et al. (1987)) *DNF* $\leq_{pac}$ *MONDNF and DNF** $\leq_{pac}$ *MONDNF***.*

Proof We simply introduce a new variable for each negative literal. Specifically, set $t' = 2t$, define $g(y)$ to be y concatenated with its bitwise complement $\overline{y}$, and define $f(\varphi(y_1, \ldots, y_t))$ to be the monotone DNF formula $\varphi'(y_1, \ldots, y_{2t})$ where each occurrence of $\neg y_i$ in φ is replaced with the new variable y_{t+i}. The fact that the function f is polynomial time (and not just g) means that this also gives a reduction from *DNF** to *MONDNF**. ■

Less obvious (and crucial for Theorem 11 below) is that *DNF* is PAC-equivalent to its dual:

Proposition 10 *DNF* $\leq_{pac}$ *DNF** $\leq_{pac}$ *DNF.*

Proof We will give a relabelling reduction from *MONDNF*$_{s,t}$ to *DNF*$^*_{s',t'}$ with $s', t' = \text{poly}(s, t)$, with functions f and g that are both polynomial time computable. By Lemma 9, this suffices to prove the proposition.

Assume WLOG that our encoding of monotone DNF formulas is such that a size s formula has at most s terms. Then our function f will map a monotone DNF formula φ on t variables of size at most s to a bitstring x' of length $s' = s \cdot t$, and the function g will map a t-bit assignment y to a monotone DNF formula ψ of size $t' = \text{poly}(s, t)$ on s' variables.

$x' = f(\varphi) \in \{0,1\}^{s \cdot t}$ will be the concatenation of the indicator vectors for the s terms of φ. Specifically $x'_{i,j} = 1$ iff the i'th term of φ contains variable y_j. $g(y)$ will be the DNF formula ψ given by:

$$\psi(z_{1,1,}, \ldots, z_{s,t}) = \bigvee_{i=1}^{s} \bigwedge_{j: y_j = 0} (\neg z_{i,j}).$$

We verify the correctness of this reduction, namely that $\psi(x') = 1$ iff $\varphi(y) = 1$:

$$\begin{aligned}
\psi(x') = 1 \quad &\Leftrightarrow \quad \bigvee_{i=1}^{s} \bigwedge_{j: y_j = 0} (\neg x'_{i,j}) \\
&\Leftrightarrow \quad \bigvee_{i=1}^{s} \bigwedge_{j=1}^{t} (\neg x'_{i,j} \vee y_j) \\
&\Leftrightarrow \quad \bigvee_{i=1}^{s} \bigwedge_{j: x'_{i,j} = 1} y_j \\
&\Leftrightarrow \quad \varphi(y) = 1
\end{aligned}$$

■

Combining Proposition 10 and Theorem 5, we get the following equivalence:

Theorem 11 *DNF is PAC-learnable iff DNF is RRHS-refutable.*

4. Reductions among Refutation Problems

RRHS-refutation of *DNF* differs from more commonly studied refutation of constraint satisfaction problems in several ways:

- In RRHS refutation, the left-hand sides of the constraint equations are worst-case, whereas in typical CSP refutation, they are random.
- In RRHS refutation, the right-hand sides of the constraint equations are random, whereas in typical CSP refutation, they are fixed to be 1 (all the randomness is in the left-hand side).
- In RRHS refutation of DNF, each constraint is described by a polynomial-length formula that can potentially depend on all variables, whereas in typical CSP refutation, they are given by constant-arity constraints (e.g. conjunctions on 3 literals).

For example, the commonly studied problem of refuting k-SAT is defined as follows:

Definition 12 *For functions $k : \mathbb{N} \to \mathbb{N}$ and $n : \mathbb{N} \to \mathbb{N}$, we say that* random k-SAT is refutable *using n equations if there is a polynomial-time algorithm $\mathcal{B}$ such that for every $s \in \mathbb{N}$, if we set $k = k(s)$ and $n = n(s)$, we have:*

1. *(Soundness) for every set of k-way disjunctions $\varphi_1, \ldots, \varphi_n$ on s variables and their negations, if the system of equations $\varphi_1(x) = 1, \ldots, \varphi_n(x) = 1$ is satisfiable, then*

$$\Pr[\mathcal{B}(1^s, \varphi_1, \ldots, \varphi_n) = 1] \leq 1/3.$$

2. *(Completeness) If we choose $\varphi_1, \ldots, \varphi_n$ independently and randomly from the set of all k-way disjunctions on s variables and their negations, then*

$$\Pr[\mathcal{B}(1^s, \varphi_1, \ldots, \varphi_n) = 1] \geq 2/3,$$

where the probability is taken over $\varphi_1, \ldots, \varphi_n$ and the coins of $\mathcal{B}$.

Feige (2002) put forth the hypothesis that random 3-SAT is not refutable with $n(s) = O(s)$ clauses. A natural generalization is the following:

Assumption 13 (Daniely and Shalev-Schwartz (2016)) *For every polynomial $n(s)$, there is a k such that random k-SAT on s variables is not refutable with $n(s)$ clauses.*

We can relate the hardness of refuting random k-SAT to RRHS-refutability of DNF as follows, using the techniques of Daniely and Shalev-Schwartz (2016).

Proposition 14 *If k-DNF formulas on s variables with $m = \lceil 2^k \cdot \ln(4n) \rceil$ terms are RRHS refutable with n equations, then random k-SAT on s variables is refutable using $n' = O(n \cdot m)$ equations.*

Proof By DeMorgan's Law, a RRHS-refuter for k-DNF formulas is equivalent to an RRHS-refuter for k-CNF formulas, so we will assume that we have the latter. Given k-way disjunctions $\varphi_1, \ldots, \varphi_{n \cdot m}$ on s variables, we will generate a sequence $(\psi_1, b_1), \ldots, (\psi_n, b_n)$ of k-CNF formulas ψ_i and bits b_i to feed to our k-CNF refuter. For each $i = 1, \ldots, n$, we will construct ψ_i and b_i as follows:

- With probability $1/2$, set $b_i = 1$ and let ψ_i be the conjunction of the first m disjunctions from $\varphi_1, \ldots, \varphi_{n \cdot m}$ that have not been used yet in constructing $\psi_1, \ldots, \psi_{i-1}$.
- With probability $1/2$, set $b_i = 0$ and let ψ_i be the conjunction of m uniformly random and independent k-way disjunctions.

Notice that if $\varphi_1, \ldots, \varphi_{n \cdot m}$ are random k-way disjunctions (as in the completeness condition for a k-SAT refuter), then the distribution of ψ_i is the same in case $b_i = 1$ as in the case $b_i = 0$. Thus, the b_i's are uniformly random and independent of the ψ_i's, and by completeness, our k-CNF RRHS-refuter will accept with probability at least $2/3$.

For soundness, we argue that if the system of equations $\varphi_1(x) = 1, \ldots, \varphi_{n \cdot m}(x) = 1$ is satisfiable by assignment α, then the system $\psi_1(x) = b_1, \ldots, \psi_n(x) = b_n$ is also satisfiable (by the same assignment α) with high probability. Clearly for the i's where we set $b_i = 1$ and ψ_i to be a conjunction of φ_j's, the satisfying assignment α for the φ_j's will also satisfy $\psi_i(x) = 1 = b_i$. For each i where we set $b_i = 0$, we argue that $\psi_i(\alpha) = 0 = b_i$ with high probability. Indeed, the probability that α satisfies a random k-CNF formula ψ_i with $m = O(2^k \log n)$ clauses is at most $(1 - 2^{-k})^m \leq 1/4n$. So by a union bound, the probability α violates any of the equations is at most 1, and thus our refuter will accept with probability at most $1/4 + 1/3 = 7/12 < 2/3$.

Again, the gap between the completeness probability of $2/3$ and the soundness probability of $7/12$ can be amplified to the $2/3$ vs. $1/3$ gap required by Definition 12 by a constant number of repetitions (increasing n' by the same constant factor). ■

Corollary 15 (Daniely and Shalev-Schwartz (2016)) *If DNF formulas are PAC learnable, then there is a fixed polynomial $n = n(s)$ such that for every constant k, k-SAT is refutable using $O(n(s))$ equations. That is, Assumption 13 is false.*

Proof By Theorem 11, if DNF formulas are PAC learnable, then they are RRHS-refutable. That means there is a fixed polynomial $n_0(s, m)$ such that DNF formulas on s variables with m terms can be refuted using $n_0(s, m)$ equations. Setting $n_1(s) = n_0(s, \log^2 s)$, we see that for every constant k, k-DNF formulas with $m = O(2^k \log n_1(s)) = o(\log^2 s)$ are RRHS-refutable with $n_1(s)$ equations, and hence random k-SAT is refutable with $n_1(s) \cdot m = o(\log^2 s) \cdot n_1(s)$ equations. Setting $n(s) = n_1(s) \cdot \log^2 s$ completes the proof. ■

This gives evidence that there is no *polynomial-time* algorithm for PAC-learning DNF. The fastest known algorithms for learning DNF formulas of size s take time roughly $\exp(s^{1/3})$ and use $\exp(s^{1/3})$ examples (Klivans and Servedio, 2004). Can we give evidence that no subexponential-time algorithm exists under a stronger version of Feige's assumption? A stronger version of Assumption 13 might say that for larger values of k (e.g. $k = s^{\Omega(1)}$), refuting random k-SAT on s variables takes time $2^{\Omega(s)}$ even using $s^{\Omega(k)}$ equations. Our reductions can preserve the time lower bound of $2^{\Omega(s)}$ and the equation/sample-complexity lower bound of $s^{\Omega(k)}$, but notice that the size t of the DNF instance produced by Proposition 14 is larger than 2^k, so our equation/sample-complexity lower bound of $s^{\Omega(k)}$ will not be any better than $t^{\log s}$, which is quasipolynomial in the size of the DNF instance. This is enough to show that PAC learning DNF with a polynomial sample complexity requires exponential time (under the suggested assumption). But it remains open whether we can give evidence that PAC-learning DNF requires exponential time even given a exponential number of samples. Theorem 5 and Proposition 10 show that this is equivalent to giving evidence that RRHS-refuting DNF requires exponential time even given an exponential number of equations.

5. Relation to Cryptographic Hardness

Earlier results on hardness of PAC learning were typically based on cryptographic assumptions. Here we elucidate the relation between RRHS refutability and cryptographic hardness. As already noted in the proof of Theorem 5, RRHS refutability has some connection to pseudorandomness. The most related cryptographic object seems to be that of a *weak pseudorandom function (weak PRF)*:

Definition 16 *Consider a family of functions $\mathcal{P} = \bigcup_s \mathcal{P}_s$ where $\mathcal{P}_s = \{p_x : \{0,1\}^t \to \{0,1\}\}_{x \in \{0,1\}^s}$ is given by a polynomial-time evaluation function* $\mathrm{Eval} : \{0,1\}^s \times \{0,1\}^t \to \{0,1\}$, *with $t = t(s) \leq \mathrm{poly}(s)$. We say that $\mathcal{P}$ is a* weak PRF *family if for every probabilistic polynomial-time algorithm $\mathcal{B}$, every $n = \mathrm{poly}(s)$, and all sufficiently large s, we have either:*

$$\Pr[\mathcal{B}(1^s, (y_1, p_x(y_1)), \ldots, (y_n, p_x(y_n))) = 1] > 1/3,$$

OR

$$\Pr[\mathcal{B}(1^s, (y_1, b_1), \ldots, (y_n, b_n)) = 1] < 2/3,$$

where the probabilities are taken over $x \leftarrow \{0,1\}^s$, $y_1, \ldots, y_t \leftarrow \{0,1\}^t$, $b_1, \ldots, b_n \leftarrow \{0,1\}$, and the coins of $\mathcal{B}$.

The usual definition of pseudorandom functions by Goldreich, Goldwasser, and Micali (1986) was used to give the first hardness result for PAC learning in Valiant's original paper (Valiant, 1984). The above definition of weak PRFs is weaker than the usual definition of PRFs in two respects:

- The inputs y_i to the PRF are chosen uniformly at random rather than adversarially and adaptively chosen by $\mathcal{B}$. (This relaxation was studied by Naor and Reingold (1998) as "indistinguishability under a random sample and random challenge.")

- To violate pseudorandomness, $\mathcal{B}$ needs to achieve a distinguishing advantage greater than $2/3 - 1/3 = 1/3$ (rather than just $1/\mathrm{poly}(s)$), and must do so with specified thresholds of $1/3$ and $2/3$.

Nevertheless, it can be shown that the existence of weak PRFs is equivalent to the existence of one-way functions and hence ordinary PRFs (but these equivalences involve modifying the family of functions).

By definition, if $\mathcal{P}$ is a weak PRF family, then $\mathcal{Q} = \mathcal{P}^*$ cannot be RRHS-refutable. Let's examine the way in which the definition of weak PRF is stronger than the negation of RRHS-refutability (which we will call *RRHS-unrefutability*). If $\mathcal{P}^*$ is *RRHS-unrefutable*, then for every probabilistic polynomial-time algorithm $\mathcal{B}$, for infinitely many s, either soundness or completeness must fail, analogous to the two possibilities in Definition 16. (In this discussion, we restrict to the case that t is polynomially related to s, as in Definition 16, so that we have only one parameter.) Specifically, if soundness fails, there exist $y_1, \ldots, y_n \in \{0,1\}^t$ and $x \in \{0,1\}^s$ such that:

$$\Pr[\mathcal{B}(1^s, (y_1, p_x(y_1)), \ldots, (y_n, p_x(y_n))) = 1] > 1/3,$$

where the probability is taken over the coins of $\mathcal{B}$. If completeness fails, there exist $y_1, \ldots, y_n \in \{0,1\}^t$ such that:

$$\Pr[\mathcal{B}(1^s, (y_1, b_1), \ldots, (y_n, b_n)) = 1] < 2/3,$$

where the probability is taken over the coins of $\mathcal{B}$ and $b_1, \ldots, b_n \leftarrow \{0,1\}$. In both cases, the failure of $\mathcal{B}$ is for a worst-case choice of the inputs $y_1, \ldots, y_n$, and in the case of completeness, it is for a worst-case choice of the PRF key x. Moreover these worst-case choices can depend on the algorithm $\mathcal{B}$ (as can the choice of the infinitely many s on which soundness or completeness fails). In contrast, a weak PRF ensures that $\mathcal{B}$ fails even when these strings are chosen uniformly at random (and this holds for all sufficiently large s).

In the literature on zero-knowledge proofs (Ostrovsky, 1991; Ostrovsky and Wigderson, 1993; Vadhan, 2006; Ong and Vadhan, 2008; Applebaum, Barak, and Xiao, 2008), other weakenings of the definition of pseudorandom functions have been studied, where the functions are also indexed by a string $w \in \{0,1\}^r$ that is worst-case, but is given to the adversary $\mathcal{B}$. One variant is as follows:

Definition 17 *Consider a family of functions $\mathcal{P} = \bigcup_s \mathcal{P}_s$ where $\mathcal{P}_s = \{p_{w,x} : \{0,1\}^t \to \{0,1\}\}_{w \in \{0,1\}^r, x \in \{0,1\}^s}$ for $t, r \leq \mathrm{poly}(s)$ given by a polynomial-time evaluation function* $\mathrm{Eval} : \{0,1\}^r \times \{0,1\}^s \times \{0,1\}^t \to \{0,1\}$. *We say that $\mathcal{P}$ is a* auxiliary-input weak PRF *family if for every probabilistic polynomial-time algorithm $\mathcal{B}$ and every $n = \mathrm{poly}(s)$, there exist infinitely many $s \in \mathbb{N}$ and $w \in \{0,1\}^r$ such that either:*

$$\Pr[\mathcal{B}(1^s, w, (y_1, p_{w,x}(y_1)), \ldots, (y_n, p_{w,x}(y_n))) = 1] > 1/3,$$

OR

$$\Pr[\mathcal{B}(1^s, w, (y_1, b_1), \ldots, (y_n, b_n)) = 1] < 2/3,$$

where the probabilities are taken over $x \leftarrow \{0,1\}^s$, $y_1, \ldots, y_t \leftarrow \{0,1\}^t$, $b_1, \ldots, b_n \leftarrow \{0,1\}$, and the coins of $\mathcal{B}$.

Ostrovsky (1991) introduced a similar notion of *auxiliary-input one-way functions.* In the works of Ostrovsky (1991); Ostrovsky and Wigderson (1993); Ong and Vadhan (2008), it was shown that the existence of zero-knowledge proofs (or even zero-knowledge arguments) for a language outside BPP implies the existence of auxiliary-input one-way functions. Applebaum, Barak, and Xiao (2008) observed that the existence of auxiliary-input one-way functions implies the existence of a auxiliary-input (strong) pseudorandom function family $\mathcal{P}$, and that the latter implies hardness of PAC-learning $\mathcal{P}$. Here, we can directly see that if $\mathcal{P}$ is an auxiliary-input weak PRF family, then $\mathcal{P}^*$ is RRHS-unrefutable (which is equivalent to $\mathcal{P}$ not being PAC-learnable, by Theorem 5). (Here the pair (x, w) should be treated as a single index for function $p_{(x,w)}$. Unlike an auxiliary-input weak PRF adversary, a refuter is not given w, which only makes the task of refutation harder.) Still, the notion of auxiliary-input weak PRFs appears to be substantially stronger than RRHS-unrefutability, due to the worst-case choices of the key x and inputs $y_1, \ldots, y_n$ in the latter. This may explain why Daniely et al. (2014); Daniely and Shalev-Schwartz (2016); Daniely (2016) were able to obtain hardness of PAC-learning results that eluded past work.

In the work of Vadhan (2006), a notion of “instance-dependent one-way functions” was introduced, which is stronger than the auxiliary-input notion above in that the infinite set of hard indices w does not depend on the adversary $\mathcal{B}$. This notion captures the difference between computational zero knowledge and statistical zero knowledge (Vadhan, 2006; Ong and Vadhan, 2008).

It is informative to compare which values are random or worst case in the different notions examined:

	function index (in completeness)	function inputs
RRHS-unrefutability of $\mathcal{P}^*$	worst case and secret	worst case
Ordinary refutability of $\mathcal{P}^*$	worst case and secret	random
(Weak) PRF family $\mathcal{P}$	random and secret	random
Auxiliary-input weak PRF family $\mathcal{P}$	worst-case public part random secret part	random random

Acknowledgments

I thank Boaz Barak, Amit Daniely, Ryan O’Donnell, Rocco Servedio, and Jon Ullman for illuminating conversations, and the anonymous reviewers for helpful corrections and suggestions.

References

Benny Applebaum, Boaz Barak, and David Xiao. On basing lower-bounds for learning on worst-case assumptions. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 211–220. IEEE Computer Society, 2008. ISBN 978-0-7695-3436-7. doi: 10.1109/FOCS.2008.35. URL http://dx.doi.org/10.1109/FOCS.2008.35.

Amit Daniely. Complexity theoretic limitations on learning halfspaces. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 105–117. ACM, 2016. ISBN 978-1-4503-4132-5. doi: 10.1145/2897518.2897520. URL http://doi.acm.org/10.1145/2897518.2897520.

Amit Daniely and Shai Shalev-Schwartz. Complexity theoretic limitations on learning dnf's. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, volume 49 of *JMLR Workshop and Conference Proceedings*, pages 815–830. JMLR.org, 2016. URL http://jmlr.org/proceedings/papers/v49/daniely16.html.

Amit Daniely, Nati Linial, and Shai Shalev-Shwartz. From average case complexity to improper learning complexity. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 441–448. ACM, 2014. ISBN 978-1-4503-2710-7. doi: 10.1145/2591796.2591820. URL http://doi.acm.org/10.1145/2591796.2591820.

Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, pages 534–543. ACM, New York, 2002. doi: 10.1145/509907.509985. URL http://dx.doi.org/10.1145/509907.509985.

Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the Association for Computing Machinery*, 33(4):792–807, 1986. ISSN 0004-5411. doi: 10.1145/6490.6503. URL http://dx.doi.org/10.1145/6490.6503.

Michael Kearns and Leslie Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the Association for Computing Machinery*, 41(1):67–95, 1994. ISSN 0004-5411. doi: 10.1145/174644.174647. URL http://dx.doi.org/10.1145/174644.174647.

Michael J. Kearns, Ming Li, Leonard Pitt, and Leslie G. Valiant. On the learnability of boolean formulae. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 285–295. ACM, 1987. ISBN 0-89791-221-7. doi: 10.1145/28395.28426. URL http://doi.acm.org/10.1145/28395.28426.

Adam R. Klivans and Rocco A. Servedio. Learning DNF in time $2^{\tilde{O}(n^{1/3})}$. *Journal of Computer and System Sciences*, 68(2):303–318, 2004. ISSN 0022-0000. doi: 10.1016/j.jcss.2003.07.007. URL http://dx.doi.org/10.1016/j.jcss.2003.07.007.

Moni Naor and Omer Reingold. From unpredictability to indistinguishability: A simple construction of pseudo-random functions from MACs (extended abstract). In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 267–282. Springer, 1998.

ISBN 3-540-64892-5. doi: 10.1007/BFb0055734. URL http://dx.doi.org/10.1007/BFb0055734.

Shien Jin Ong and Salil Vadhan. An equivalence between zero knowledge and commitments. In *Theory of cryptography*, volume 4948 of *Lecture Notes in Computer Sciences*, pages 482–500. Springer, Berlin, 2008. doi: 10.1007/978-3-540-78524-8_27. URL http://dx.doi.org/10.1007/978-3-540-78524-8_27.

Rafail Ostrovsky. One-way functions, hard on average problems, and statistical zero-knowledge proofs. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference, Chicago, Illinois, USA, June 30 - July 3, 1991*, pages 133–138. IEEE Computer Society, 1991. ISBN 0-8186-2255-5. doi: 10.1109/SCT.1991.160253. URL https://doi.org/10.1109/SCT.1991.160253.

Rafail Ostrovsky and Avi Wigderson. One-way fuctions are essential for non-trivial zero-knowledge. In *Second Israel Symposium on Theory of Computing Systems, ISTCS 1993, Natanya, Israel, June 7-9, 1993, Proceedings*, pages 3–17. IEEE Computer Society, 1993. ISBN 0-8186-3630-0. doi: 10.1109/ISTCS.1993.253489. URL https://doi.org/10.1109/ISTCS.1993.253489.

Leonard Pitt and Manfred K. Warmuth. Prediction-preserving reducibility. *Journal of Computer and System Sciences*, 41(3):430–467, 1990. ISSN 0022-0000. doi: 10.1016/0022-0000(90)90028-J. URL http://dx.doi.org/10.1016/0022-0000(90)90028-J.

Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990. doi: 10.1007/BF00116037. URL http://dx.doi.org/10.1007/BF00116037.

Salil P. Vadhan. An unconditional study of computational zero knowledge. *SIAM Journal on Computing*, 36(4):1160–1214, 2006. ISSN 0097-5397. doi: 10.1137/S0097539705447207. URL http://dx.doi.org/10.1137/S0097539705447207.

Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984. doi: 10.1145/1968.1972. URL http://doi.acm.org/10.1145/1968.1972.

Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982. doi: 10.1109/SFCS.1982.45. URL https://doi.org/10.1109/SFCS.1982.45.