From average case complexity to improper learning complexity

Amit Daniely* Nati Linial† Shai Shalev-Shwartz‡

March 9, 2014

Abstract

The basic problem in the PAC model of computational learning theory is to determine which hypothesis classes are efficiently learnable. There is presently a dearth of results showing hardness of learning problems. Moreover, the existing lower bounds fall short of the best known algorithms.

The biggest challenge in proving complexity results is to establish hardness of *improper learning* (a.k.a. representation independent learning). The difficulty in proving lower bounds for improper learning is that the standard reductions from **NP**-hard problems do not seem to apply in this context. There is essentially only one known approach to proving lower bounds on improper learning. It was initiated in [29] and relies on cryptographic assumptions.

We introduce a new technique for proving hardness of improper learning, based on reductions from problems that are hard on average. We put forward a (fairly strong) generalization of Feige's assumption [20] about the complexity of refuting random constraint satisfaction problems. Combining this assumption with our new technique yields far reaching implications. In particular,

- Learning DNF's is hard.
- Agnostically learning halfspaces with a constant approximation ratio is hard.
- Learning an intersection of $\omega(1)$ halfspaces is hard.

*Dept. of Mathematics, The Hebrew University, Jerusalem, Israel
†School of Computer Science and Engineering, The Hebrew University, Jerusalem, Israel.
‡School of Computer Science and Engineering, The Hebrew University, Jerusalem, Israel

1 Introduction

Valiant's celebrated *probably approximately correct* (=PAC) model [44] of machine learning led to an extensive research that yielded a whole scientific community devoted to computational learning theory. In the PAC learning model, a learner is given an oracle access to randomly generated samples $(X, Y) \in \mathcal{X} \times \{0,1\}$ where X is sampled from some *unknown* distribution $\mathcal{D}$ on $\mathcal{X}$ and $Y = h^*(X)$ for some *unknown* function $h^* : \mathcal{X} \to \{0,1\}$. Furthermore, it is assumed that h^* comes from a predefined *hypothesis class* $\mathcal{H}$, consisting of $0,1$ valued functions on $\mathcal{X}$. The learning problem defined by $\mathcal{H}$ is to find a function $h : \mathcal{X} \to \{0,1\}$ that minimizes $\mathrm{Err}_{\mathcal{D}}(h) := \Pr_{X \sim \mathcal{D}}(h(X) \neq h^*(X))$. For concreteness' sake we take $\mathcal{X} = \{\pm 1\}^n$, and we consider the learning problem tractable if there is an algorithm that on input ϵ, runs in time $\mathrm{poly}(n, 1/\epsilon)$ and outputs, w.h.p., a hypothesis h with $\mathrm{Err}(h) \leq \epsilon$.

Assuming $\mathbf{P} \neq \mathbf{NP}$, the status of most basic *computational* problems is fairly well understood. In a sharp contrast, almost 30 years after Valiant's paper, the status of most basic *learning* problems is still wide open – there is a huge gap between the performance of the best known algorithms and hardness results:

- No known algorithms can learn depth 2 circuits, i.e., DNF formulas. In contrast, we can only rule out learning of circuits of depth d, for some unspecified constant d [30]. This result is based on a relatively strong assumption (a certain subexponential lower bound on factoring Blum integers). Under more standard assumptions (RSA in secure), the best we can do is rule out learning of depth $\log n$ circuits [29].

- It is possible to agnostically learn halfspaces (see section 2.1 for a definition of agnostic learning) with an approximation ratio of $O\left(\frac{n}{\log n}\right)$. On the other hand, the best known lower bound only rules out exact agnostic learning ([22], based on [35], under the assumption that the $\tilde{O}\left(n^{1.5}\right)$ unique shortest vector problem is hard).

- No known algorithm can learn intersections of 2 halfspaces, whereas Klivans and Sherstov [35] only rule out learning intersections of polynomially many halfspaces (again assuming that $\tilde{O}\left(n^{1.5}\right)$-uSVP is hard).

The crux of the matter, leading to this state of affairs, has to do with the learner's freedom to return *any* hypothesis. A learner who may return hypotheses outside the class $\mathcal{H}$ is called an *improper learner*. This additional freedom makes such algorithms potentially more powerful than proper learners. On the other hand, this added flexibility makes it difficult to apply standard reductions from **NP**-hard problems. Indeed, there was no success so far in proving intractability of a learning problem based on **NP**-hardness. Moreover, as Applebaum, Barak and Xiao [3] showed, many standard ways to do so are doomed to fail, unless the polynomial hierarchy collapses.

The vast majority of existing lower bounds on learning utilize the crypto-based argument, suggested in [29]. Roughly speaking, to prove that a certain learning problem is hard, one starts with a certain collection of functions, that by assumption are one-way trapdoor permutations. This immediately yields some hard (usually artificial) learning problem. The final step is to reduce this artificial problem to some natural learning problem.

Unlike the difficulty in establishing lower bounds for improper learning, the situation in *proper* learning is much better understood. Usually, hardness of proper learning is proved by showing that it is **NP**-hard to distinguish a realizable sample from an unrealizable sample. I.e., it is hard to tell whether there is some hypothesis in $\mathcal{H}$ which has zero error on a given sample. This, however, does not suffice for the purpose of proving lower bounds on improper learning, because it might be the case that the learner finds a hypothesis (not from $\mathcal{H}$) that does not err on the sample even though no $h \in \mathcal{H}$ can accomplish this. In this paper we present a new methodology for proving hardness of improper learning. Loosely speaking, we show that improper learning is impossible provided that it is hard to distinguish a realizable sample from a *randomly generated* unrealizable sample.

Feige [20] conjectured that random 3-SAT formulas are hard to refute. He derived from this assumption certain hardness of approximation results, which are not known to follow from $\mathbf{P} \neq \mathbf{NP}$. We put forward a (fairly strong) assumption, generalizing Feige's assumption to certain predicates other that 3-SAT. Under this assumption, we show:

1. Learning DNF's is hard.
2. Agnostically learning halfspaces with a constant approximation ratio is hard, even over the boolean cube.
3. Learning intersection of $\omega(1)$ halfspaces is hard, even over the boolean cube.
4. Learning finite automata is hard.
5. Learning parity is hard.

We note that result 4 can be established using the cryptographic technique [29]. Result 5 is often taken as a hardness *assumption*. We also conjecture that under our generalization of Feige's assumption it is hard to learn intersections of even constant number of halfspaces. We present a possible approach to the case of four halfspaces. To the best of our knowledge, these results easily imply most existing lower bounds for improper learning.

1.1 Comparison to the cryptographic technique

There is a crucial reversal of order that works in our favour. To lower bound improper learning, we actually need much less than what is needed in cryptography, where a problem and a distribution on instances are appropriate if they fool *every algorithm*. In contrast, here we are presented with *a concrete* learning algorithms and we devise a problem and a distribution on instances that fail it.

Second, cryptographic assumptions are often about the hardness of number theoretic problems. In contrast, the average case assumptions presented here are about CSP problems. The proximity between CSP problems and learning problems is crucial for our purposes: Since distributions are very sensitive to gadgets, reductions between average case problems are much more limited than reductions between worst case problems.

1.2 On the role of average case complexity

A key question underlying the present study and several additional recent papers is what can be deduced from the average case hardness of specific problems. Hardness on average is crucial for cryptography, and the security of almost all modern cryptographic systems hinges on the average hardness of certain problems, often from number theory. As shown by Kearns and Valiant [29], the very same hardness on average assumptions can be used to prove hardness of improper PAC learning of some hypothesis classes.

Beyond these classic results, several recent works, starting from Feige's seminal work [20], show that average case hardness assumptions lead to dramatic consequences in complexity theory. The main idea of [20] is to consider two possible avenues for progress beyond the classic uses of average hardness: (i) Derive hardness in additional domains, (ii) Investigate the implications of hardness-on-average of other problems. For example, what are the implications of average hardness of 3-SAT? What about other CSP problems?

Feige [20] and then [2, 6] show that average case hardness of CSP problems have surprising implications in hardness of approximation, much beyond the consequences of standard complexity assumptions, or even cryptographic assumptions. Recently, [10] and [17] show that hardness on average of planted clique and 3-SAT have implications in learning theory, in the specific context of computational-sample tradeoffs. In particular, they show that in certain learning tasks (sparse PCA and learning halfspaces over sparse vectors) more data can be leveraged to speed up computation. As we show here, average case hardness of CSP problems has implications even on the hardness of very fundamental tasks in learning theory. Namely, determining the tractability of PAC learning problems, most of which are presently otherwise inaccessible.

2 Preliminaries

2.1 Learning Theory

A *hypothesis class*, $\mathcal{H}$, is a series of collections of functions $\mathcal{H}_n \subset \{0,1\}^{\mathcal{X}_n}$, $n = 1, 2, \ldots$. We often abuse notation and identify $\mathcal{H}$ with $\mathcal{H}_n$. The instance space, $\mathcal{X}_n$, that we consider is either $\mathcal{X}_n = \{\pm 1\}^n$, $\mathcal{X}_n = \{0,1\}^n$ or $\mathcal{X}_n = \{-1, 1, 0\}^n$. Concrete hypothesis classes, such as halfspaces, DNF's etc., are denoted $\text{HALFSPACES}, \text{DNF}$ etc. Also $\mathcal{Z}_n := \mathcal{X}_n \times \{0,1\}$.

Distributions on $\mathcal{Z}_n$ (resp. $\mathcal{Z}_n^m$) are denoted $\mathcal{D}_n$ (resp. $\mathcal{D}_n^m$). *Ensembles* of distributions are denoted by $\mathcal{D}$. That is, $\mathcal{D} = \{\mathcal{D}_n^{m(n)}\}_{n=1}^{\infty}$ where $\mathcal{D}_n^{m(n)}$ is a distributions on $\mathcal{Z}_n^{m(n)}$. We say that $\mathcal{D}$ is a *polynomial ensemble* if $m(n)$ is upper bounded by some polynomial in n.

The error of a hypothesis $h : \mathcal{X}_n \to \{0,1\}$ w.r.t. $\mathcal{D}_n$ on $\mathcal{Z}_n$ is defined as $\text{Err}_{\mathcal{D}_n}(h) = \Pr_{(x,y)\sim\mathcal{D}_n}(h(x) \neq y)$. For a hypothesis class $\mathcal{H}_n$, we define $\text{Err}_{\mathcal{D}_n}(\mathcal{H}_n) = \min_{h\in\mathcal{H}_n} \text{Err}_{\mathcal{D}_n}(h)$. We say that a distribution $\mathcal{D}_n$ is *realizable* by h (resp. $\mathcal{H}_n$) if $\text{Err}_{\mathcal{D}_n}(h) = 0$ (resp. $\text{Err}_{\mathcal{D}_n}(\mathcal{H}_n) = 0$). Similarly, we say that $\mathcal{D}_n$ is *ϵ-almost realizable* by h (resp. $\mathcal{H}_n$) if $\text{Err}_{\mathcal{D}_n}(h) \leq \epsilon$ (resp. $\text{Err}_{\mathcal{D}_n}(\mathcal{H}_n) \leq \epsilon$).

A *sample* is a sequence $S = \{(x_1, y_1), \ldots (x_m, y_m)\} \in \mathcal{Z}_n^m$. The *empirical error* of a hypothesis $h : \mathcal{X}_n \to \{0,1\}$ w.r.t. sample S is $\text{Err}_S(h) = \frac{1}{m}\sum_{i=1}^m 1(h(x_i) = y_i)$. The *empirical error* of a hypothesis class $\mathcal{H}_n$ w.r.t. S is $\text{Err}_S(\mathcal{H}_n) = \min_{h\in\mathcal{H}_n} \text{Err}_S(h)$. We say that a sample S is *realizable* by h if $\text{Err}_S(h) = 0$. The sample S is *realizable* by $\mathcal{H}_n$ if

$\mathrm{Err}_S(\mathcal{H}_n) = 0$. Similarly, we define the notion of ϵ-*almost realizable* sample (by either a hypothesis $h : \mathcal{X}_n \to \{0,1\}$ or a class $\mathcal{H}_n$).

A *learning algorithm*, denoted $\mathcal{L}$, obtains an error parameter $0 < \epsilon < 1$, a confidence parameter $0 < \delta < 1$, a complexity parameter n, and an access to an oracle that produces samples according to unknown distribution $\mathcal{D}_n$ on $\mathcal{Z}_n$. It should output a (description of) hypothesis $h : \mathcal{X}_n \to \{0,1\}$. We say that the algorithm $\mathcal{L}$ *(PAC) learns* the hypothesis class $\mathcal{H}$ if, for every realizable distribution $\mathcal{D}_n$, with probability $\geq 1-\delta$, $\mathcal{L}$ outputs a hypothesis with error $\leq \epsilon$. We say that an algorithm $\mathcal{L}$ *agnostically learns* $\mathcal{H}$ if, for every distribution $\mathcal{D}_n$, with probability $\geq 1-\delta$, $\mathcal{L}$ outputs a hypothesis with error $\leq \mathrm{Err}_{\mathcal{D}_n}(\mathcal{H}) + \epsilon$. We say that an algorithm $\mathcal{L}$ *approximately agnostically learns* $\mathcal{H}$ with approximation ratio $\alpha = \alpha(n) \geq 1$ if, for every distribution $\mathcal{D}_n$, with probability $\geq 1-\delta$, $\mathcal{L}$ outputs a hypothesis with error $\leq \alpha \cdot \mathrm{Err}_{\mathcal{D}_n}(\mathcal{H}) + \epsilon$. We say that $\mathcal{L}$ is *efficient* if it runs in time polynomial in $n, 1/\epsilon$ and $1/\delta$, and outputs a hypothesis that can be evaluated in time polynomial in $n, 1/\epsilon$ and $1/\delta$. We say that $\mathcal{L}$ is *proper* (with respect to $\mathcal{H}$) if it always outputs a hypothesis in $\mathcal{H}$. Otherwise, we say that $\mathcal{L}$ is *improper*.

Let $\mathcal{H} = \{\mathcal{H}_n \subset \{0,1\}^{\mathcal{X}_n} \mid n = 1, 2 \ldots\}$ and $\mathcal{H}' = \{\mathcal{H}'_n \subset \{0,1\}^{\mathcal{X}'_n} \mid n = 1, 2 \ldots\}$ be two hypothesis classes. We say the $\mathcal{H}$ is *realized* by $\mathcal{H}'$ if there are functions $g : \mathbb{N} \to \mathbb{N}$ and $f_n : \mathcal{X}_n \to \mathcal{X}'_{g(n)}$, $n = 1, 2, \ldots$ such that for every n, $\mathcal{H}_n \subset \{h' \circ f_n \mid h' \in \mathcal{H}'_n\}$. We say that $\mathcal{H}$ is *efficiently realized* by $\mathcal{H}'$ if, in addition, f_n can be computed it time polynomial in n. Note that if $\mathcal{H}'$ is efficiently learnable (respectively, agnostically learnable, or approximately agnostically learnable) and $\mathcal{H}$ is efficiently realized by $\mathcal{H}'$, then $\mathcal{H}$ is efficiently learnable (respectively, agnostically learnable, or approximately agnostically learnable) as well.

2.2 Constraints Satisfaction Problems

Let $P : \{\pm 1\}^K \to \{0,1\}$ be some boolean predicate (that is, P is any non-constant function from $\{\pm 1\}^K$ to $\{0,1\}$). A P-*constraint* with n variables is a function $C : \{\pm 1\}^n \to \{0,1\}$ of the form $C(x) = P(j_1 x_{i_1}, \ldots, j_K x_{i_K})$ for $j_l \in \{\pm 1\}$ and K distinct $i_l \in [n]$. The *CSP problem,* $\mathrm{CSP}(P)$, is the following. An instance to the problem is a collection $J = \{C_1, \ldots, C_m\}$ of P-constraints and the objective is to find an assignment $x \in \{\pm 1\}^n$ that maximizes the fraction of satisfied constraints (i.e., constraints with $C_i(x) = 1$). The *value* of the instance J, denoted $\mathrm{VAL}(J)$, is the maximal fraction of constraints that can be simultaneously satisfied. If $\mathrm{VAL}(J) = 1$, we say that J is satisfiable.

For $1 \geq \alpha > \beta > 0$, the problem $\mathrm{CSP}^{\alpha,\beta}(P)$ is the decision promise problem of distinguishing between instances to $\mathrm{CSP}(P)$ with value $\geq \alpha$ and instances with value $\leq \beta$. Denote $\underline{\mathrm{VAL}}(P) = \mathbb{E}_{x \sim \mathrm{Uni}(\{\pm 1\}^K)} P(x)$. We note that for every instance J to $\mathrm{CSP}(P)$, $\mathrm{VAL}(J) \geq \underline{\mathrm{VAL}}(P)$ (since a random assignment $\psi \in \{\pm 1\}^n$ satisfies in expectation $\underline{\mathrm{VAL}}(P)$ fraction of the constraints). Therefore, the problem $\mathrm{CSP}^{\alpha,\beta}(P)$ is non-trivial only if $\beta \geq \underline{\mathrm{VAL}}(P)$. We say that P is *approximation resistant* if, for every $\epsilon > 0$, the problem $\mathrm{CSP}^{1-\epsilon, \underline{\mathrm{VAL}}(P)+\epsilon}(P)$ is **NP**-hard. Note that in this case, unless $\mathbf{P} = \mathbf{NP}$, no algorithm for $\mathrm{CSP}(P)$ achieves better approximation ratio than the naive algorithm that simply chooses a random assignment. We will use even stronger notions of approximation resistance: We say that P is *approximation resistant on satisfiable instances* if, for every $\epsilon > 0$, the problem $\mathrm{CSP}^{1, \underline{\mathrm{VAL}}(P)+\epsilon}(P)$ is **NP**-hard. Note that in this case, unless $\mathbf{P} = \mathbf{NP}$, no algorithm for $\mathrm{CSP}(P)$ achieves better approximation ratio than a random assignment, even

if the instance is guaranteed to be satisfiable. We say that P is *heredity approximation resistant on satisfiable instances* if every predicate that is implied by P (i.e., every predicate $P' : \{\pm 1\}^K \to \{0,1\}$ that satisfies $\forall x,\ P(x) \Rightarrow P'(x)$) is approximation resistant on satisfiable instances. Similarly, we define the notion of *heredity approximation resistance*.

We will consider average case variant of the problem $\mathrm{CSP}^{\alpha,\beta}(P)$. Fix $1 \geq \alpha > \underline{\mathrm{VAL}}(P)$. By a simple counting argument, for sufficiently large constant $C > 0$, the value of a random instance with $\geq C \cdot n$ constraints is about $\underline{\mathrm{VAL}}(P)$, in particular, the probability that a (uniformly) random instance to $\mathrm{CSP}(P)$ with n variables and $\geq Cn$ constraints will have value $\geq \alpha$ is exponentially small. Therefore, the problem of distinguishing between instances with value $\geq \alpha$ and random instances with $m(n)$ constraints can be thought as an average case analogue of $\mathrm{CSP}^{\alpha,\underline{\mathrm{VAL}}(P)+\epsilon}$. We denote this problem by $\mathrm{CSP}^{\alpha,\mathrm{rand}}_{m(n)}(P)$. Precisely, we say that the problem $\mathrm{CSP}^{\alpha,\mathrm{rand}}_{m(n)}(P)$ is easy, if there exists an efficient randomized algorithm, $\mathcal{A}$, with the following properties:

- If J is an instance to $\mathrm{CSP}(P)$ with n variables, $m(n)$ constraints, and value $\geq \alpha$, then
$$\Pr_{\text{coins of } \mathcal{A}}(\mathcal{A}(J) = \text{“VAL}(J) \geq \alpha\text{”}) \geq \frac{3}{4}$$
- If J is a random instance to $\mathrm{CSP}(P)$ with n variables and $m(n)$ constraints then, with probability $1 - o_n(1)$ over the choice of J,
$$\Pr_{\text{coins of } \mathcal{A}}(\mathcal{A}(J) = \text{“}J \text{ is random”}) \geq \frac{3}{4}\ .$$

The problem $\mathrm{CSP}^{\alpha,\mathrm{rand}}_{m(n)}(P)$ will play a central role. In particular, the case $\alpha = 1$, that is, the problem of distinguishing between satisfiable instances and random instances. This problem is also known as the problem of refuting random instances to $\mathrm{CSP}(P)$. A simple observation is that the problem $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P)$ becomes easier as m grows: If $m' \geq m$, we can reduce instances of $\mathrm{CSP}^{1,\mathrm{rand}}_{m'(n)}(P)$ to instances of $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P)$ by simply drop the last $m'(n) - m(n)$ clauses. Note that if the original instance was either random or satisfiable, the new instance has the same property as well. Therefore, a natural metric to evaluate a refutation algorithm is the number of random constraints that are required to guarantee that the algorithm will refute the instance with high probability.

Another simple observation is that if a predicate $P' : \{\pm 1\}^K \to \{0,1\}$ is implied by P then the problem $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P')$ is harder than $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P')$. Indeed, given an instance to $\mathrm{CSP}(P)$, we can create an instance to $\mathrm{CSP}(P')$ by replacing each constraint $C(x) = P(j_1 x_{i_1}, \ldots, j_K x_{j_K})$ with the constraint $C'(x) = P'(j_1 x_{i_1}, \ldots, j_K x_{j_K})$. We note that this reduction preserves both satisfiability and randomness, and therefore establishes a valid reduction from $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P)$ to $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P')$.

2.3 Resolution refutation and Davis Putnam algorithms

A clause is a disjunction of literals, each of which correspond to a distinct variable. Given two clauses of the form $x_i \vee C$ and $\neg x_i \vee D$ for some clauses C, D, the *resolution rule* infer the

clause $C \vee D$. Fix a predicate $P : \{\pm 1\}^K \to \{0,1\}$. A *resolution refutation* for an instance $J = \{C_1, \ldots, C_m\}$ to CSP(P) is a sequence of clauses $\tau = \{T_1, \ldots, T_r\}$ such that T_r is the empty clause, and for every $1 \leq i \leq r$, T_i is either implied by some C_j or resulted from the resolution rule applied on T_{i_1} and T_{i_2} for some $i_1, i_2 < i$. We note that every un-satisfiable instance to CSP(P) has a resolution refutation (of possibly exponential length). We denote by RES(J) the length of the shortest resolution refutation of J.

The length of resolution refutation of random K-SAT instances were extensively studied (e.g., [9], [8] and [7]). Two motivations for these study are the following. First, the famous result of [16], shows that **NP** $\neq$ **CoNP** if and only is there is no propositional proof system that can refute every instance J to K-SAT in length polynomial in $|J|$. Therefore, lower bound on concrete proof systems might bring us closer to **NP** $\neq$ **CoNP**. Also, such lower bounds might indicate that refuting such instances in general, is intractable.

A second reason is that many popular algorithms implicitly produces a resolution refutation during their execution. Therefore, any lower bound on the size of the resolution refutation would lead to the same lower bound on the running time or the algorithm. A widely used and studied refutation algorithms of this kind are Davis-Putnam (DPLL) like algorithms [19]. A DPLL algorithm is a form of recursive search for a satisfying assignment which on CSP input J operates as follows: If J contains the constant predicate 0, it terminates and outputs that the instance is un-satisfiable. Otherwise, a variable x_i is chosen, according to some rule. Each assignment to x_i simplifies the instance J, and the algorithm recurses on these simpler instances.

3 The methodology

We begin by discussing the methodology in the realm of realizable learning, and we later proceed to agnostic learning. Some of the ideas underling our methodology appeared, in a much more limited context, in [17].

To motivate the approach, recall how one usually proves that a class cannot be efficiently *properly* learnable. Given a hypothesis class $\mathcal{H}$, let $\Pi(\mathcal{H})$ be the problem of distinguishing between an $\mathcal{H}$-realizable sample S and one with $\mathrm{Err}_S(\mathcal{H}) \geq \frac{1}{4}$. If $\mathcal{H}$ is efficiently *properly* learnable then this problem is in[1] **RP**: To solve $\Pi(\mathcal{H})$, we simply invoke a proper learning algorithm $\mathcal{A}$ that efficiently learns $\mathcal{H}$, with examples drawn uniformly from S. Let h be the output of $\mathcal{A}$. Since $\mathcal{A}$ properly learns $\mathcal{H}$, we have

- If S is a realizable sample, then $\mathrm{Err}_S(h)$ is small.
- If $\mathrm{Err}_S(\mathcal{H}) \geq \frac{1}{4}$ then, *since* $h \in \mathcal{H}$, $\mathrm{Err}_S(h) \geq \frac{1}{4}$.

This gives an efficient way to decide whether S is realizable. We conclude that if $\Pi(\mathcal{H})$ is **NP**-hard, then $\mathcal{H}$ is not efficiently learnable, unless **NP** = **RP**.

However, this argument does not rule out the possibility that $\mathcal{H}$ is still learnable by an *improper* algorithm. Suppose now that $\mathcal{A}$ efficiently and improperly learns $\mathcal{H}$. If we try to use the above argument to prove that $\Pi(\mathcal{H})$ can be efficiently solved, we get stuck – suppose

[1] The reverse direction is almost true: If the search version of this problem can be solved in polynomial time, then $\mathcal{H}$ is efficiently learnable.

that S is a sample and we invoke $\mathcal{A}$ on it, to get a hypothesis h. As before, if S is realizable, $\mathrm{Err}_S(h)$ is small. However, if S is not realizable, since h not necessarily belongs to $\mathcal{H}$, it still might be the case that $\mathrm{Err}_S(h)$ is small. Therefore, the argument fails. We emphasize that this is not only a mere weakness of the argument – there are classes for which $\Pi(\mathcal{H})$ is **NP**-hard, but yet, they are learnable by an improper algorithm[2]. More generally, Applebaum et al [3] indicate that it is unlikely that hardness of improper learning can be based on standard reductions from **NP**-hard problems, as the one described here.

We see that it is not clear how to establish hardness of improper learning based on the hardness of distinguishing between a realizable and an unrealizable sample. The core problem is that even if S is not realizable, the algorithm might still return a good hypothesis. The crux of our new technique is the observation that if S is *randomly generated* unrealizable sample then even improper algorithm cannot return a hypothesis with a small empirical error. The point is that the returned hypothesis is determined solely by the examples that $\mathcal{A}$ sees and its random bits. Therefore, if $\mathcal{A}$ is an efficient algorithm, the number of hypotheses it might return cannot be too large. Hence, if S is "random enough", it likely to be far from all these hypotheses, in which case the hypothesis returned by $\mathcal{A}$ would have a large error on S.

We now formalize this idea. Let $\mathcal{D} = \{\mathcal{D}_n^{m(n)}\}_n$ be a polynomial ensemble of distributions, such that $\mathcal{D}_n^{m(n)}$ is a distribution on $\mathcal{Z}_n^{m(n)}$. Think of $\mathcal{D}_n^{m(n)}$ as a distribution that generates samples that are far from being realizable by $\mathcal{H}$. We say that it is hard to distinguish between a $\mathcal{D}$-random sample and a realizable sample if there is no efficient randomized algorithm $\mathcal{A}$ with the following properties:

- For every realizable sample $S \in \mathcal{Z}_n^{m(n)}$,

$$\Pr_{\text{internal coins of } \mathcal{A}}(\mathcal{A}(S) = \text{“}\textit{realizable}\text{”}) \geq \frac{3}{4} .$$

- If $S \sim \mathcal{D}_n^{m(n)}$, then with probability $1 - o_n(1)$ over the choice of S, it holds that

$$\Pr_{\text{internal coins of } \mathcal{A}}(\mathcal{A}(S) = \text{“}\textit{unrelizable}\text{”}) \geq \frac{3}{4} .$$

For functions $p, \epsilon : \mathbb{N} \to (0, \infty)$, we say that $\mathcal{D}$ is $(p(n), \epsilon(n))$-*scattered* if, for large enough n, it holds that for every function $f : \mathcal{X}_n \to \{0, 1\}$,

$$\Pr_{S \sim \mathcal{D}_n^{m(n)}}(\mathrm{Err}_S(f) \leq \epsilon(n)) \leq 2^{-p(n)} .$$

Example 3.1 *Let $\mathcal{D}_n^{m(n)}$ be the distribution over $\mathcal{Z}_n^{m(n)}$ defined by taking $m(n)$ independent uniformly chosen examples from $\mathcal{X}_n \times \{0, 1\}$. For $f : \mathcal{X}_n \to \{0, 1\}$, $\Pr_{S \sim \mathcal{D}_n^{m(n)}}\left(\mathrm{Err}_S(f) \leq \frac{1}{4}\right)$ is the probability of getting at most $\frac{m(n)}{4}$ heads in $m(n)$ independent tosses of a fair coin. By Hoeffding's bound, this probability is $\leq 2^{-\frac{1}{8}m(n)}$. Therefore, $\mathcal{D} = \{\mathcal{D}_n^{m(n)}\}_n$ is $\left(\frac{1}{8}m(n), 1/4\right)$-scattered.*

[2]This is true, for example, for the class of DNF formulas with 3 DNF clauses.

Theorem 3.2 *Every hypothesis class that satisfies the following condition is not efficiently learnable. There exists $\beta > 0$ such that for every $c > 0$ there is an (n^c, β)-scattered ensemble $\mathcal{D}$ for which it is hard to distinguish between a $\mathcal{D}$-random sample and a realizable sample.*

Remark 3.3 *The theorem and the proof below work verbatim if we replace β by $\beta(n)$, provided that $\beta(n) > n^{-a}$ for some $a > 0$.*

Proof Let $\mathcal{H}$ be the hypothesis class in question and suppose toward a contradiction that algorithm $\mathcal{L}$ learns $\mathcal{H}$ efficiently. Let $M(n, 1/\epsilon, 1/\delta)$ be the maximal number of random bits used by $\mathcal{L}$ when run on the input n, ϵ, δ. This includes both the bits describing the examples produced by the oracle and "standard" random bits. Since $\mathcal{L}$ is efficient, $M(n, 1/\epsilon, 1/\delta) < \text{poly}(n, 1/\epsilon, 1/\delta)$. Define

$$q(n) = M(n, 1/\beta, 4) + n \ .$$

By assumption, there is a $(q(n), \beta)$-scattered ensemble $\mathcal{D}$ for which it is hard to distinguish a $\mathcal{D}$-random sample from a realizable sample. Consider the algorithm $\mathcal{A}$ defined below. On input $S \in \mathcal{Z}_n^{m(n)}$,

1. Run $\mathcal{L}$ with parameters n, β and $\frac{1}{4}$, such that the examples' oracle generates examples by choosing a random example from S.

2. Let h be the hypothesis that $\mathcal{L}$ returns. If $\text{Err}_S(h) \leq \beta$, output "realizable". Otherwise, output "unrealizable".

Next, we derive a contradiction by showing that $\mathcal{A}$ distinguishes a realizable sample from a $\mathcal{D}$-random sample. Indeed, if the input S is realizable, then $\mathcal{L}$ is guaranteed to return, with probability $\geq 1 - \frac{1}{4}$, a hypothesis $h : \mathcal{X}_n \to \{0, 1\}$ with $\text{Err}_S(h) \leq \beta$. Therefore, w.p. $\geq \frac{3}{4}$ $\mathcal{A}$ will output "realizable".

What if the input sample S is drawn from $\mathcal{D}_n^{m(n)}$? Let $\mathcal{G} \subset \{0, 1\}^{\mathcal{X}_n}$ be the collection of functions that $\mathcal{L}$ might return when run with parameters $n, \epsilon(n)$ and $\frac{1}{4}$. We note that $|\mathcal{G}| \leq 2^{q(n)-n}$, since each hypothesis in $\mathcal{G}$ can be described by $q(n) - n$ bits. Namely, the random bits that $\mathcal{L}$ uses and the description of the examples sampled by the oracle. Now, since $\mathcal{D}$ is $(q(n), \beta)$-scattered, the probability that $\text{Err}_S(h) \leq \beta$ for some $h \in \mathcal{G}$ is at most $|\mathcal{G}|2^{-q(n)} \leq 2^{-n}$. It follows that the probability that $\mathcal{A}$ responds "realizable" is $\leq 2^{-n}$. This leads to the desired contradiction and concludes our proof. □

Next, we discuss analogue theorem to theorem 3.2 for (approximate) agnostic learning. Let $\mathcal{D}$ be a polynomial ensemble and $\epsilon : \mathbb{N} \to (0, 1)$. We say that it is hard to distinguish between a $\mathcal{D}$-random sample and an ϵ-almost realizable sample if there is no efficient randomized algorithm $\mathcal{A}$ with the following properties:

- For every sample $S \in \mathcal{Z}_n^{m(n)}$ that is $\epsilon(n)$-almost realizable,

$$\Pr_{\text{internal coins of } \mathcal{A}}(\mathcal{A}(S) = \text{"}\textit{almost realizable}\text{"}) \geq 3/4 \ .$$

- If $S \sim \mathcal{D}_n^{m(n)}$, then with probability $1 - o_n(1)$ over the choice of S, it holds that

$$\Pr_{\text{internal coins of } \mathcal{A}}(\mathcal{A}(S) = \text{"}\textit{unrelizable}\text{"}) \geq \frac{3}{4} \ .$$

Theorem 3.4 *Let $\alpha \geq 1$. Every hypothesis class that satisfies the following condition is not efficiently agnostically learnable with an approximation ratio of α. For some β and every $c > 0$, there is a $(n^c, \alpha\beta + 1/n)$-scattered ensemble $\mathcal{D}$ such that it is hard to distinguish between a $\mathcal{D}$-random sample and a β-almost realizable sample.*

Remark 3.5 *As in theorem 3.2, the theorem and the proof below work verbatim if we replace α by $\alpha(n)$ and β by $\beta(n)$, provided that $\beta(n) > n^{-a}$ for some $a > 0$.*

Proof Let $\mathcal{H}$ be the hypothesis class in question and suppose toward a contradiction that $\mathcal{L}$ efficiently agnostically learns $\mathcal{H}$ with approximation ratio of α. Let $M(n, 1/\epsilon, 1/\delta)$ be the maximal number of random bits used by $\mathcal{L}$ when it runs on the input n, ϵ, δ. This includes both the bits describing the examples produced by the oracle and the "standard" random bits. Since $\mathcal{L}$ is efficient, $M(n, 1/\epsilon, 1/\delta) < \text{poly}(n, 1/\epsilon, 1/\delta)$. Define,

$$q(n) = M(n, n, 4) + n \ .$$

By the assumptions of the theorem, there is a $(q(n), \alpha\beta + 1/n)$-scattered ensemble $\mathcal{D}$ such that it is hard to distinguish between a $\mathcal{D}$-random sample and a β-almost realizable sample. Consider the following efficient algorithm to distinguish between a $\mathcal{D}$-random sample and a β-almost realizable sample. On input $S \in \mathcal{Z}_n^{m(n)}$,

1. Run $\mathcal{L}$ with parameters $n, 1/n$ and $\frac{1}{4}$, such that the examples are sampled uniformly from S.

2. Let h be the hypothesis returned by the algorithm $\mathcal{L}$. If $\text{Err}_S(h) \leq \alpha\beta + 1/n$, return "almost realizable". Otherwise, return "unrealizable".

Next, we derive a contradiction by showing that this algorithm, which we denote by $\mathcal{A}$, distinguishes between a realizable sample and a $\mathcal{D}$-random sample. Indeed, if the input S is β-almost realizable, then $\mathcal{L}$ is guaranteed to return, with probability $\geq 1 - \frac{1}{4}$, a hypothesis $h : \mathcal{X}_n \to \{0, 1\}$ with $\text{Err}_S(h) \leq \alpha\beta + 1/n$. Therefore, the algorithm $\mathcal{A}$ will return, w.p. $\geq \frac{3}{4}$, "almost realizable".

Suppose now that the input sample S is drawn according to $\mathcal{D}_n$. Let $\mathcal{G} \subset \{0, 1\}^{\mathcal{X}_n}$ be the collection of functions that the learning algorithm $\mathcal{L}$ might return when it runs with the parameters $n, 1/n$ and $\frac{1}{4}$. Note that each hypothesis in $\mathcal{G}$ can be described by $q(n) - n$ bits, namely, the random bits used by $\mathcal{L}$ and the description of the examples sampled by the oracle. Therefore, $|\mathcal{G}| \leq 2^{q(n)-n}$. Now, since $\mathcal{D}$ is $(q(n), \alpha\beta + 1/n)$-scattered, the probability that some function in $h \in \mathcal{G}$ will have $\text{Err}_S(h) \leq \alpha\beta + 1/n$ is at most $|\mathcal{G}|2^{-q(n)} \leq 2^{-n}$. It follows that the probability that the algorithm $\mathcal{A}$ will return "almost realizable" is $\leq 2^{-n}$. □

4 The strong random CSP assumption

In this section we put forward and discuss a new assumption that we call "the strong random CSP assumption" or SRCSP for short. It generalizes Feige's assumption [20], as well as the assumption of Barak, Kindler and Steurer [6]. This new assumption, together with

the methodology described in section 3, are used to establish lower bounds for improper learning. Admittedly, our assumption is strong, and an obvious quest, discussed in the end of this section is to find ways to derive similar conclusions from weaker assumptions.

The SRCSP assumption claims that for certain predicates $P : \{\pm 1\}^K \to \{0,1\}, d > 0$ and $\alpha > 0$, the decision problem $\mathrm{CSP}^{\alpha,\mathrm{rand}}_{n^d}(P)$ is intractable. We first consider the case $\alpha = 1$. To reach a plausible assumption, let us first discuss Feige's assumption, and the existing evidence for it. Denote by $\mathrm{SAT}_3 : \{\pm 1\}^3 \to \{0,1\}$ the 3-SAT predicate $\mathrm{SAT}_3(x_1, x_2, x_3) = x_1 \vee x_2 \vee x_3$.

Assumption 4.1 (Feige) *For every sufficiently large constant $C > 0$, $\mathrm{CSP}^{1,\mathrm{rand}}_{C \cdot n}(\mathrm{SAT}_3)$ is intractable.*

Let us briefly summarize the evidence for this assumption.

- **Hardness of approximation.** Feige's conjecture can be viewed as a strengthening of Hastad's celebrated result [25] that SAT_3 is approximation resistant on satisfiable instances. Hastad's result implies that under $\mathbf{P} \neq \mathbf{NP}$, it is hard to distinguish satisfiable instances to $\mathrm{CSP}(\mathrm{SAT}_3)$ from instances with value $\leq \frac{7}{8} + \epsilon$. The collection of instances with value $\leq \frac{7}{8} + \epsilon$ includes most random instances with $C \cdot n$ clauses for sufficiently large C. Feige's conjecture says that the problem remains intractable even when restricted to these random instances.

  We note that approximation resistance on satisfiable instances is a necessary condition for the validity of Feige's assumption. Indeed, for large enough $C > 0$, with probability $1 - o_n(1)$, the value of a random instance to $\mathrm{CSP}(\mathrm{SAT}_3)$ is $\leq \frac{7}{8} + \epsilon$. Therefore, tractability of $\mathrm{CSP}^{1,\frac{7}{8}+\epsilon}(\mathrm{SAT}_3)$ would lead to tractability of $\mathrm{CSP}^{1,\mathrm{rand}}_{C \cdot n}(\mathrm{SAT}_3)$.

- **Performance of known algorithms.** The problem of refuting random 3-SAT formulas has been extensively studied and a many algorithms were studied. The best known algorithms [21] can refute random instances with $\Omega\left(n^{1.5}\right)$ random constraints. Moreover resolution lower bounds [9] show that many algorithms run for exponential time when applied to random instances with $O\left(n^{1.5-\epsilon}\right)$ constraints.

We aim to generalize Feige's assumption in two aspects – (i) To predicates other than SAT_3, and (ii) To problems with super-linearly many constraints. Consider the problem $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P)$ for some predicate $P : \{\pm 1\}^K \to \{0,1\}$. As above, the intractability of $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P)$ strengthens the claim that P is approximation resistant on satisfiable instances. Also, for $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P)$ to be hard, it is necessary that P is approximation resistant on satisfiable instances. In fact, as explained in section 2.3, if $P' : \{\pm 1\}^K \to \{0,1\}$ is implied by P, then the problem $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P)$ can be easily reduced to $\mathrm{CSP}^{1,\mathrm{rand}}_{m(n)}(P')$. Therefore, to preserve the argument of the first evidence of Feige's conjecture, it is natural to require that P is *heredity* approximation resistant on satisfiable instances.

Next, we discuss what existing algorithms can do. The best known algorithms for the predicate $\mathrm{SAT}_K(x_1, \ldots, x_K) = \vee_{i=1}^K x_i$ can only refute random instances with $\Omega\left(n^{\lfloor \frac{K}{2} \rfloor}\right)$ constraints [15]. This gives some evidence that it becomes harder to refute random instances of $\mathrm{CSP}(P)$ as the number of variables grows. Namely, that many random constraints are

needed to efficiently refute random instances. Of course, some care is needed with counting the "actual" number of variables. Clearly, only *certain* predicates have been studied so far. Therefore, to reach a plausible assumption, we consider the *resolution refutation complexity* of random instances to $\mathrm{CSP}(P)$. And consequently, also the performance of a large class of algorithms, including Davis-Putnam style (DPLL) algorithms.

Davis-Putnam algorithms have been subject to an extensive study, both theoretical and empirical. Due to the central place that they occupy, much work has been done since the late 80's, to prove lower bounds on their performance in refuting random K-SAT formulas. These works relied on the fact that these algorithms implicitly produce a resolution refutation during their execution. Therefore, to derive a lower bound on the run time of these algorithms, exponential lower bounds were established on the resolution complexity of random instances to $\mathrm{CSP}(\mathrm{SAT}_K)$. These lower bounds provide support to the belief that it is hard to refute not-too-dense random K-SAT instances.

We define the 0-*variability,* $\mathrm{VAR}_0(P)$, of a predicate P as the smallest cardinality of a set of P's variables such that there is an assignment to these variables for which $P(x) = 0$, regardless of the values assigned to the other variables. By a simple probabilistic argument, a random $\mathrm{CSP}(P)$ instance with $\Omega\left(n^r\right)$ constraints, where $r = \mathrm{VAR}_0(P)$ is almost surely unsatisfiable with a resolution proof of constant size. Namely, w.p. $1 - o_n(1)$, there are 2^r constraints that are inconsistent, since some set of r variables appears in all 2^r possible ways in the different clauses. On the other hand, we show in section 8 that a random $\mathrm{CSP}(P)$ problem with $O\left(n^{c \cdot r}\right)$ constraints has w.h.p. exponential resolution complexity. Here $c > 0$ is an absolute constant. Namely,

Theorem 4.2 *There is a constant $C > 0$ such that for every $d > 0$ and every predicate P with $\mathrm{VAR}_0(P) \geq C \cdot d$, the following holds. With probability $1 - o_n(1)$, a random instance of $\mathrm{CSP}(P)$ with n variables and n^d constraints has resolution refutation length $\geq 2^{\Omega(\sqrt{n})}$.*

To summarize, we conclude that the parameter $\mathrm{VAR}_0(P)$ controls the resolution complexity of random instances to $\mathrm{CSP}(P)$. In light of the above discussion, we put forward the following assumption.

Assumption 4.3 (SRCSP – part 1) *There is a function $f : \mathbb{N} \to \mathbb{N}$ such that the following holds. Let P be a predicate that is heredity approximation resistant on satisfiable instances with $\mathrm{VAR}_0(P) \geq f(d)$. Then, it is hard to distinguish between satisfiable instances of $\mathrm{CSP}(P)$ and random instances with n^d constraints.*

Next, we motivate a variant of the above assumption, that accommodates also predicates that are not heredity approximation resistant. A celebrated result of Raghavendra [41] shows that under the unique games conjecture [31], a certain SDP-relaxation-based algorithm is (worst case) optimal for $\mathrm{CSP}(P)$, for every predicate P. Barak et al. [6] conjectured that this algorithm is optimal even on random instances. They considered the performance of this algorithm on random instances and purposed the following assumption, which they called the "random CSP hypothesis". Define $\overline{\mathrm{VAL}}(P) = \max_{\mathcal{D}} \mathbb{E}_{x \sim \mathcal{D}} P(x)$, where the maximum is taken over all pairwise uniform distributions[3] on $\{\pm 1\}^K$.

[3] A distribution is *pairwise uniform* if, for every pair of coordinates, the distribution induced on these coordinates is uniform.

Assumption 4.4 (RSCP) *For every $\epsilon > 0$ and sufficiently large $C > 0$, it is hard to distinguish instances with value $\geq \overline{\mathrm{VAL}}(P) - \epsilon$ from random instances with $C \cdot n$ constraints.*

Here we generalize the RCSP assumption to random instances with much more than $C \cdot n$ constraints. As in assumption 4.3, the 0-variability of P serves to quantify the number of random constraints needed to efficiently show that a random instance has value $< \overline{\mathrm{VAL}}(P) - \epsilon$.

Assumption 4.5 (SRSCP - part 2) *There is a function $f : \mathbb{N} \to \mathbb{N}$ such that for every predicate P with $\mathrm{VAR}_0(P) \geq f(d)$ and for every $\epsilon > 0$, it is hard to distinguish between instances with value $\geq \overline{\mathrm{VAL}}(P) - \epsilon$ and random instances with n^d constraints.*

Finally, we define the notion of a SRCSP-hard problem.

Terminology 4.6 *A computational problem is* SRCSP*-hard if its tractability contradicts assumption 4.3 or 4.5.*

4.1 Toward weaker assumptions

The SRCSP assumption is strong. It is highly desirable to arrive at similar conclusions from substantially weaker assumptions. A natural possibility that suggests itself is the SRCSP assumption, restricted to SAT:

Assumption 4.7 *There is a function $f : \mathbb{N} \to \mathbb{N}$ such that for every $K \geq f(d)$, it is hard to distinguish satisfiable instances of $\mathrm{CSP}(\mathrm{SAT}_K)$ from random instances with n^d constraints.*

We are quite optimistic regarding the success of this direction: The lower bounds we prove here use the SRCSP-assumption only for certain predicates, and do not need the full power of the assumption. Moreover, for the hypothesis classes of DNF's, intersection of halfspaces, and finite automata, these predicates are somewhat arbitrary. In [20], it is shown that for predicates of arity 3, assumption 4.5 is implied by the same assumption restricted to the SAT predicate. This gives a hope to prove, based on assumption 4.7, that the SRCSP-assumption is true for predicates that are adequate to our needs.

5 Summary of results

5.1 Learning DNF's

A DNF *clause* is a conjunction of literals. A DNF *formula* is a disjunction of DNF clauses. Each DNF formula over n variables naturally induces a function on $\{\pm 1\}^n$. We define the size of a DNF clause as the number of its literals and the size of a DNF formula as the sum of the sizes of its clauses.

As DNF formulas are very natural form of predictors, learning hypothesis classes consisting of DNF's formulas of polynomial size has been a major effort in computational learning theory. Already in Valiant's paper [44], it is shown that for every constant q, the hypothesis class of all DNF-formulas with $\leq q$ clauses is efficiently learnable. The running time

of the algorithm is, however, exponential in q. We also note that Valiant's algorithm is improper. For general polynomial-size DNF's, the best known result [34] shows learnability in time $\frac{1}{\epsilon} \cdot 2^{\tilde{O}\left(n^{\frac{1}{3}}\right)}$. Better running times (quasi-polynomial) are known under distributional assumptions [36, 38].

As for lower bounds, *properly* learning DNF's is known to be hard [40]. However, proving hardness of improper learning of polynomial DNF's has remained a major open question in computational learning theory. Noting that DNF clauses coincide with depth 2 circuits, a natural generalization of DNF's is circuits of small depth. For such classes, certain lower bounds can be obtained using the cryptographic technique. Kharitonov [30] has shown that a certain subexponential lower bound on factoring Blum integers implies hardness of learning circuits of depth d, for some unspecified constant d. Under more standard assumptions (that the RSA cryptosystem is secure), best lower bounds [29] only rule out learning of circuits of depth $\log(n)$.

For a function $q : \mathbb{N} \to \mathbb{N}$, denote by $\mathrm{DNF}_{q(n)}$ the hypothesis class of functions over $\{\pm 1\}^n$ that can be realized by DNF formulas of size at most $q(n)$. Also, let $\mathrm{DNF}^{q(n)}$ be the hypothesis class of functions over $\{\pm 1\}^n$ that that can be realized by DNF formulas with at most $q(n)$ clauses. Since each clause is of size at most n, $\mathrm{DNF}^{q(n)} \subset \mathrm{DNF}_{nq(n)}$.

As mentioned, for a constant q, the class DNF^q is efficiently learnable. We show that for every super constant $q(n)$, it is SRCSP-hard to learn $\mathrm{DNF}^{q(n)}$:

Theorem 5.1 *If* $\lim_{n\to\infty} q(n) = \infty$ *then learning* $\mathrm{DNF}^{q(n)}$ *is* SRCSP*-hard.*

Since $\mathrm{DNF}^{q(n)} \subset \mathrm{DNF}_{nq(n)}$, we immediately conclude that learning DNF's of size, say, $\leq n\log(n)$, is SRCSP-hard. By a simple scaling argument, we obtain an even stronger result:

Corollary 5.2 *For every* $\epsilon > 0$*, it is* SRCSP*-hard to learn* $\mathrm{DNF}_{n^\epsilon}$*.*

Remark 5.3 *Following the Boosting argument of Schapire [43], hardness of improper learning of a class* $\mathcal{H}$ *immediately implies that for every* $\epsilon > 0$*, there is no efficient algorithm that when running on a distribution that is realized by* $\mathcal{H}$*, guaranteed to output a hypothesis with error* $\leq \frac{1}{2} - \epsilon$*. Therefore, hardness results of improper learning are very strong, in the sense that they imply that the algorithm that just makes a random guess for each example, is essentially optimal.*

5.2 Agnostically learning halfspaces

Let HALFSPACES be the hypothesis class of halfspaces over $\{-1, 1\}^n$. Namely, for every $w \in \mathbb{R}^n$ we define $h_w : \{\pm 1\}^n \to \{0, 1\}$ by $h_w(x) = \mathrm{sign}\left(\langle w, x\rangle\right)$, and let

$$\mathrm{HALFSPACES} = \{h_w \mid w \in \mathbb{R}^n\} \ .$$

We note that usually halfspaces are defined over $\mathbb{R}^n$, but since we are interested in lower bounds, looking on this more restricted class just make the lower bounds stronger.

The problem of learning halfspaces is as old as the field of machine learning, starting with the perceptron algorithm [42], through the modern SVM [45]. As opposed to learning DNF's, learning halfspaces in the realizable case is tractable. However, in the agnostic PAC model,

the best currently known algorithm for learning halfspaces runs in time exponential in n and the best known approximation ratio of polynomial time algorithms is $O\left(\frac{n}{\log(n)}\right)$. Better running times (usually of the form $n^{\text{poly}\left(\frac{1}{\epsilon}\right)}$) are known under distributional assumptions (e.g. [28]).

The problem of *proper* agnostic learning of halfspaces was shown to be hard to approximate within a factor of $2^{\log^{1-\epsilon}(n)}$ [4]. Using the cryptographic technique, improper learning of halfspaces is known to be hard, under a certain cryptographic assumption regarding the shortest vector problem ([22], based on [35]). No hardness results are known for *approximately and improperly* learning halfspaces. Here, we show that:

Theorem 5.4 *For every constant $\alpha \geq 1$, it is* SRCSP*-hard to approximately agnostically learn* HALFSPACES *with an approximation ratio of α.*

5.3 Learning intersection of halfspaces

For a function $q : \mathbb{N} \to \mathbb{N}$, we let $\text{INTER}_{q(n)}$ be the hypothesis class of intersection of $\leq q(n)$ halfspaces. That is, $\text{INTER}_{q(n)}$ consists of all functions $f : \{\pm 1\}^n \to \{0, 1\}$ for which there exist $w_1, \ldots w_k \in \mathbb{R}^n$ such that $f(x) = 1$ if and only if $\forall i, \langle w_i, x\rangle > 0$.

Learning intersection of halfspaces has been a major challenge in machine learning. Beside being a natural generalization of learning halfspaces, its importance stems from *neural networks* [12]. Learning neural networks was popular in the 80's, and enjoy a certain comeback nowadays. A neural network is composed of layers, each of which is composed of nodes. The first layer consists of n nodes, containing the input values. The nodes in the rest of the layers calculates a value according to a halfspace (or a "soft" halfspace obtained by replacing the sign function with a sigmoidal function) applied on the values of the nodes in the previous layer. The final layer consists of a single node, which is the output of the whole network.

Neural networks naturally induce several hypothesis classes (according to the structure of the network). The class of intersection of halfspaces is related to those classes, as it can be realized by very simple neural networks: the class $\text{INTER}_{q(n)}$ can be realized by neural networks with only an input layer, a single hidden layer, and output layer, so that there are $q(n)$ nodes in the second layer. Therefore, lower bounds on improperly learning intersection of halfspaces implies lower bounds on improper learning of neural networks.

Exact algorithms for learning $\text{INTER}_{q(n)}$ run in time exponential in n. Better running times (usually of the form $n^{\text{poly}\left(\frac{1}{\epsilon}\right)}$) are known under distributional assumptions (e.g. [33]). It is known that properly learning intersection of even 2 halfspaces is hard [32]. For improper learning, Klivans and Sherstov [35] have shown that learning an intersection of polynomially many half spaces is hard, under a certain cryptographic assumption regarding the shortest vector problem. Noting that every DNF formula with $q(n)$ clauses is in fact the complement of an intersection of $q(n)$ halfspaces[4], we conclude from theorem 5.1 that intersection of every super constant number of halfsapces is hard.

[4]In the definition of INTER, we considered halfspaces with no threshold, while halfspaces corresponding to DNFs do have a threshold. This can be standardly handled by padding the examples with a single coordinate of value 1.

Theorem 5.5 *If* $\lim_{n\to\infty} q(n) = \infty$ *then learning* $\mathrm{INTER}_{q(n)}$ *is* SRCSP*-hard.*

In section 7.4 we also describe a route that might lead to the result that learning INTER_4 is SRCSP-hard.

5.4 Additional results

In addition to the results mentioned above, we show that learning the class of finite automata of polynomial size is SRCSP-hard. Hardness of this class can also be derived using the cryptographic technique, based on the assumption that the RSA cryptosystem is secure [29]. Finally, we show that agnostically learning parity with any constant approximation ratio is SRCSP-hard. Parity is not a very interesting class from the point of view of practical machine learning. However, learning this class is related to several other problems in complexity [13]. We note that hardness of agnostically learning parity, even in a more relaxed model than the agnostic PAC model (called the random classification noise model), is a well accepted hardness assumption.

In section 8 we prove lower bounds on the size of a resolution refutation for random CSP instances. In section 9 we show that unless the polynomial hierarchy collapses, there is no "standard reduction" from an **NP**-hard problem (or a **CoNP**-hard problem) to random CSP problems.

5.5 On the proofs

Below we outline the proof for DNFs. The proof for halfspaces and parities is similar. For every $c > 0$, we start with a predicate $P : \{\pm 1\}^K \to \{0,1\}$, for which the problem $\mathrm{CSP}^{1,\mathrm{rand}}_{n^c}(P)$ is hard according to the SRCSP-assumption, and reduce it to the problem of distinguishing between a $(\Omega(n^c), \frac{1}{5})$-scattered sample and a realizable sample. Since c is arbitrary, the theorem follows from theorem 3.2.

The reduction is performed as follows. Consider the problem $\mathrm{CSP}(P)$. Each assignment naturally defines a function from the collection of P-constraints to $\{0,1\}$. Hence, if we think about the constraints as instances and about the assignments as hypotheses, the problem $\mathrm{CSP}(P)$ turns into some kind of a learning problem. However, in this interpretation, all the instances we see have positive labels (since we seek an assignment that satisfies as many instances as possible). Therefore, the problem $\mathrm{CSP}^{1,\mathrm{rand}}_{n^c}(P)$ results in "samples" which are not scattered at all.

To overcome this, we show that the analogous problem to $\mathrm{CSP}^{1,\mathrm{rand}}_{n^c}(P)$, where $(\neg P)$-constraints are also allowed, is hard as well (using the assumption on the hardness of $\mathrm{CSP}^{1,\mathrm{rand}}_{n^c}(P)$). The hardness of the modified problem can be shown by relying on the special predicate we work with. This predicate was defined in the recent work of Huang [27], and it has the property of being heredity approximation resistant, even though $|P^{-1}(1)| \leq 2^{O\left(K^{1/3}\right)}$.

At this point, we have an (artificial) hypothesis class which is SRCSP-hard to learn by theorem 3.2. In the next and final step, we show that this class can be efficiently realized by DNFs with $\omega(1)$ clauses. The reduction uses the fact that every boolean function can be expressed by a DNF formula (of possibly exponential size). Therefore, P can be expressed by

a DNF formula with 2^K clauses. Based on this, we show that each hypothesis in our artificial class can be realized by a DNF formula with 2^K clauses, which establishes the proof.

The results about learning automata and learning intersection of $\omega(1)$ halfspaces follow from the result about DNFs: We show that these classes can efficiently realize the class of DNFs with $\omega(1)$ clauses. In section 7.4 we suggest a route that might lead to the result that learning intersection of 4 halfspaces is SRCSP-hard: We show that assuming the unique games conjecture, a certain family of predicates are heredity approximation resistant. We show also that for these predicates, the problem $\mathrm{CSP}^{1,\alpha}(P)$ is **NP**-hard for some $1 > \alpha > 0$. This leads to the conjecture that these predicates are in fact heredity approximation resistant. Conditioning on the correctness of this conjecture, we show that it is SRCSP-hard to learn intersection of 4-halfspaces. This is done using the strategy described for DNFs.

The proof of the resolution lower bounds (section 8) relies on the strategy and the ideas introduced in [24] and farther developed in [7, 8, 9]. The proof that it is unlikely that the correctness of the SRCSP-assumption can be based on **NP**-hardness (section 9) uses the idea introduced in [3]: we show that if an **NP**-hard problem (standardly) reduces to $\mathrm{CSP}^{\alpha,\mathrm{rand}}_{m(n)}(P)$, then the problem has a statistical zero knowledge proof. It follows that $\mathbf{NP} \subset \mathbf{SZKP}$, which collapses the polynomial hierarchy.

6 Future work

We elaborate below on some of the numerous open problems and research directions that the present paper suggests.

6.1 Weaker assumptions?

First and foremost, it is very desirable to draw similar conclusions from assumption substantially weaker than SRCSP (see section 4.1). Even more ambitiously, is it possible to reduce some **NP**-hard problem to some of the problems that are deemed hard by the SRCSP assumption? In section 9, we show that a pedestrian application of this approach is doomed to fail (unless the polynomial hierarchy collapses). This provides, perhaps, a moral justification for an "assumption based" study of average case complexity.

6.2 The SRCSP-assumption

We believe that the results presented here, together with [20, 2, 17, 10] and [6], make a compelling case that it is of fundamental importance for complexity theory to understand the hardness of random CSP problems. In this context, the SRCSP assumption is an interesting conjecture. There are, of course, many ways to try to refute it. On the other hand, current techniques in complexity theory seem too weak to prove it, or even to derive it from standard complexity assumptions. Yet, there are ways to provide more circumstantial evidence in favor of this assumption:

- As discussed in the previous section, one can try to derive it, even partially, from weaker assumptions.

- Analyse the performance of existing algorithms. In section 8 it is shown that no Davis-Putnam algorithm can refute the SRCSP assumption. Also, Barak et al [6] show that the basic **SDP** algorithm [41] cannot refute assumption 4.5, and also 4.3 for certain predicates (those that contain a pairwise uniform distribution). Such results regarding additional classes of algorithms will lend more support to the assumption's correctness.

- Show lower bounds on the proof complexity of random CSP instances in refutation systems stronger than resolution.

For a further discussion, see [6]. Interest in the SRCSP assumption calls for a better understanding of heredity approximation resistance. For recent work in this direction, see [26, 27].

6.3 More applications

We believe that the method presented here and the SRCSP-assumption can yield additional results in learning and approximation. Here are several basic questions in learning theory that we are unable to resolve even under the SRCSP-assumption.

1. Decision trees are very natural hypothesis class, that is not known to be efficiently learnable. Is it SRCSP-hard to learn decision trees?

2. What is the real approximation ratio of learning halfspaces? We showed that it is SRCSP-hard to agnostically learn halfspaces with a constant approximation ratio. The best known algorithm only guarantees an approximation ratio of $\frac{n}{\log n}$. This is still a huge gap. See remark 7.3 for some speculations about this question.

3. Likewise for learning large margin halfspaces (see remark 7.4) and for parity.

4. Prove that it is SRCSP-hard to learn intersections of a constantly many halfspaces. This might be true even for 2 halfspaces. In section 7.4, we suggest a route to prove that intersection of 4 halfspaces is SRCSP-hard.

Besides application to learning and approximation, it would be fascinating to see applications of the SRCSP-assumption in other fields of complexity. It will be a poetic justice if we could apply it to cryptography. We refer the reader to [6] for a discussion. Finding implications in fields beyond cryptography, learning and approximation would be even more exciting.

7 Proofs of the lower bounds

Relying on our general methodology given in section 3, to show that a learning problem is SRCSP-hard, we need to find a scattered ensamble, $\mathcal{D}$, such that it is SRCSP-hard to distinguish between a realizable sample and a $\mathcal{D}$-random sample. We will use the following simple criterion for an ensamble to be scattered.

Proposition 7.1 *Let $\mathcal{D}$ be some distribution on a set $\mathcal{X}$. For even m, let $X_1, \ldots, X_m$ be independent random variables drawn according to $\mathcal{D}$. Consider the sample $S = \{(X_1, 1), (X_2, 0) \ldots, (X_{m-1}, 1), (X_m, 0)\}$. Then, for every $h : \mathcal{X} \to \{0, 1\}$,*

$$\Pr_S \left(\mathrm{Err}_S(h) \leq \frac{1}{5} \right) \leq 2^{-\frac{9}{100}m}$$

Proof For $1 \leq i \leq \frac{m}{2}$ let $T_i = 1[h(X_{2i-1}) \neq 1] + 1[h(X_{2i}) \neq 0]$. Note that $\mathrm{Err}_S(h) = \frac{1}{m} \sum_{i=1}^{\frac{m}{2}} T_i$. Also, the T_i's are independent random variables with mean 1 and values between 0 and 2. Therefore, by Hoeffding's bound,

$$\Pr_S \left(\mathrm{Err}_S(h) \leq \frac{1}{5} \right) \leq e^{-\frac{9}{100}m} \leq 2^{-\frac{9}{100}m} .$$

□

7.1 Learning DNFs

In this section we prove theorem 5.1 and corollary 5.2. We will use the SRCSP assumption 4.3 with Huang's predicate [27]. Let $k \geq 1$ and denote $K = k + \binom{k}{3}$. We index the first k coordinates of vectors in $\{\pm 1\}^K$ by the numbers $1, 2, \ldots, k$. The last $\binom{k}{3}$ coordinates are indexed by $\binom{[k]}{3}$. Let $H_k : \{\pm 1\}^K \to \{0, 1\}$ be the predicate such that $H_k(x) = 1$ if and only if there is a vector y with hamming distance $\leq k$ from x such that, for every $A \in \binom{[k]}{3}$, $y_A = \prod_{i \in A} y_i$. The basic properties of H_k are summarized in the following lemma due to [27].

Lemma 7.2 ([27])

1. *H_k is heredity approximation resistant on satisfiable instances.*
2. *$|H_k^{-1}(1)| = \tilde{O}(K^{1/3})$.*
3. *The 0-variability of H_k is $\geq k$.*
4. *For every sufficiently large k, there exists $y^k \in \{\pm 1\}^K$ such that $H_k(x) = 1 \Rightarrow H_k(y^k \oplus x) = 0$*

Proof 1. and 2. were proved in [27]. 3. is very easy. We proceed to 4. Choose $y^k \in \{\pm 1\}^K$ uniformly at random. By 2., for every $x \in \{\pm 1\}^K$, $\Pr\left(H_k(y^k \oplus x) = 1\right) = 2^{-K + \tilde{O}(K^{1/3})}$. Taking a union over all vectors $x \in H_k^{-1}(1)$, we conclude that the probability that one of them satisfies $H_k(y^k \oplus x) = 1$ is $2^{-K + \tilde{O}(K^{1/3}) + \tilde{O}(K^{1/3})} = 2^{-K + \tilde{O}(K^{1/3})}$. For large enough k, this is less than 1. Therefore, there exists a y^k as claimed. □

Proof (of theorem 5.1) Let $d > 0$ by assumption 4.3 and lemma 7.2, for large enough k, it is SRCSP-hard to distinguish between satisfiable instances to CSP(H_k) and random instances with $m = 2n^d$ constraints. We will reduce this problem to the problem of distinguishing between a realizable sample to $\mathrm{DNF}^{q(n)}$ and a random sample drawn from a $\left(\frac{9}{50}n^d, 1/5\right)$-scattered ensamble $\mathcal{D}$. Since d is arbitrary, the theorem follows from theorem 3.2.

The reduction works as follows. Let y^k be the vector from lemma 7.2. Given an instance

$$J = \{H_k(j_{1,1}x_{i_{1,1}}, \ldots, j_{1,K}x_{i_{1,K}}), \ldots, H_k(j_{m,1}x_{i_{m,1}}, \ldots, j_{m,K}x_{i_{m,K}})\}$$

to CSP(H_k), we will produce a new instance J' by changing the sign of the variables according to y^k in every other constraint. Namely,

$$\begin{aligned} J' \;=\; & \{H_k(j_{1,1}x_{i_{1,1}}, \ldots, j_{1,K}x_{i_{1,K}}), H_k(y^k_1 j_{2,1}x_{i_{2,1}}, \ldots, y^k_K j_{2,K}x_{i_{2,K}}), \ldots \\ & \ldots, H_k(j_{m-1,1}x_{i_{m-1,1}}, \ldots, j_{m-1,K}x_{i_{m-1,K}}), H_k(y^k_1 j_{m,1}x_{i_{m,1}}, \ldots, y^k_K j_{m,K}x_{i_{m,K}})\} \ . \end{aligned}$$

Note that if J is random then so is J'. Also, if J is satisfiable with a satisfying assignment u, then, by lemma 7.2, u satisfies in J' exactly the constraints with odd indices. Next, we will produce a sample $S \in \left(\{\pm 1\}^{2Kn} \times \{0,1\}\right)^m$ from J' as follows. We will index the coordinates of vectors in $\{\pm 1\}^{2Kn}$ by $[K] \times \{\pm 1\} \times [n]$. We define a mapping Ψ from the collection of H_k-constraints to $\{\pm 1\}^{2Kn}$ as follows – for each constraint $C = H_k(j_1 x_{i_1}, \ldots, j_K x_{i_K})$ we define $\Psi(C) \in \{\pm 1\}^{2Kn}$ by the formula

$$(\Psi(C))_{l,b,i} = \begin{cases} -1 & (b,i) = (-j_l, i_l) \\ 1 & \text{otherwise} \end{cases}$$

Finally, if $J' = \{C'_1, \ldots, C'_m\}$, we will produce the sample

$$S = \left\{(\Psi(C'_1), 1), (\Psi(C'_2), 0), \ldots, (\Psi(C'_{m-1}), 1), (\Psi(C'_m), 0)\right\} \ .$$

The theorem follows from the following claim:

Claim 1

1. *If J is a random instance then S is $\left(\frac{9}{100}m, \frac{1}{5}\right)$-scattered.*
2. *If J is a satisfiable instance then S is realizable by a* DNF *formula with $\leq 2^K$ clauses.*

Proposition 7.1 implies part 1. We proceed to part 2. Like every boolean function on K variables, H_k is expressible by a DNF expression of 2^K clauses, each of which contains all the variables. Suppose then that

$$H_k(x_1, \ldots, x_K) = \vee_{t=1}^{2^K} \wedge_{r=1}^{K} b_{t,r} x_r \ .$$

Let $u \in \{\pm 1\}^n$ be an assignment to J. Consider the following DNF formula over $\{\pm 1\}^{2Kn}$

$$\phi_u(x) = \vee_{t=1}^{2^K} \wedge_{r=1}^{K} \wedge_{i=1}^{n} x_{r,(u_i b_{t,r}),i} \ ,$$

where, as mentioned before, we index coordinates of $x \in \{\pm 1\}^{2Kn}$ by triplets in $[K] \times \{\pm 1\} \times [n]$. We claim that for every H_k-constraint C, $\phi_u(\Psi(C)) = C(u)$. This suffices, since if u satisfies J then u satisfies exactly the constraints with odd indices in J'. Therefore, by the definition of S and the fact that $\forall C, \phi_u(\Psi(C)) = C(u)$, ϕ_u realizes S.

Indeed, let $C(x) = H_k(j_1 x_{i_1}, \ldots, j_K x_{i_K})$ be a H_k-constraint. We have

$$\begin{aligned}
\phi_u(\Psi(C)) = 1 &\iff \exists t \in [2^K]\, \forall r \in [K], i \in [n],\ (\Psi(C))_{r,(u_i b_{t,r}),i} = 1 \\
&\iff \exists t \in [2^K]\, \forall r \in [K], i \in [n]\ (u_i b_{t,r}, i) \neq (-j_r, i_r) \\
&\iff \exists t \in [2^K]\, \forall r \in [K],\ u_{i_r} b_{t,r} \neq -j_r \\
&\iff \exists t \in [2^K]\, \forall r \in [K],\ b_{t,r} = j_r u_{i_r} \\
&\iff C(u) = H_k(j_1 u_{i_1}, \ldots, j_K u_{i_K}) = 1\ .
\end{aligned}$$

□

By a simple scaling argument we can prove corollary 5.2.

Proof (of corollary 5.2) By theorem 5.1, it is SRCSP-hard to learn DNF^n. Since $\mathrm{DNF}^n \subset \mathrm{DNF}_{n^2}$, we conclude that it is SRCSP-hard to learn DNF_{n^2}. To establish the corollary, we note that DNF_{n^2} can be efficiently realized by $\mathrm{DNF}_{n^\epsilon}$ using the mapping $f : \{\pm 1\}^n \to \{\pm 1\}^{n^{\frac{2}{\epsilon}}}$ that pads the original n coordinates with $n^{\frac{2}{\epsilon}} - n$ ones. □

7.2 Agnostically learning halfspaces

Proof (of theorem 5.4) Let $\mathcal{H}$ be the hypothesis class of halfspaces over $\{-1, 1, 0\}^n$, induced by ± 1 vectors. We will show that agnostically learning $\mathcal{H}$ is SRCSP-hard. While we defined the class of HALFSPACES over instances in $\{\pm 1\}^n$, proving the hardness of learning $\mathcal{H}$ (which is defined over $\{-1, 1, 0\}^n$) suffices for our needs, since $\mathcal{H}$ can be efficiently realized by HALFSPACES as follows: Define $\psi : \{-1, 1, 0\} \to \{\pm 1\}^2$ by

$$\psi(\alpha) = \begin{cases} (-1,-1) & \alpha = -1 \\ (1,1) & \alpha = 1 \\ (-1,1) & \alpha = 0 \end{cases}\ .$$

Now define $\Psi : \{-1, 1, 0\}^n \to \{\pm 1\}^{2n}$ by

$$\Psi(x) = (\psi(x_1), \ldots, \psi(x_n))\ .$$

Also define $\Phi : \{\pm 1\}^n \to \{\pm 1\}^{2n}$ by

$$\Phi(w) = (w_1, w_1, w_2, w_2, \ldots, w_n, w_n)\ .$$

It is not hard to see that for every $w \in \{\pm 1\}^n$ and every $x \in \{-1, 1, 0\}^n$, $h_w(x) = h_{\Phi(w)}(\Psi(x))$. Therefore, $\mathcal{H}$ is efficiently realized by HALFSPACES.

We will use assumption 4.5 with respect to the majority predicate $\mathrm{MAJ}_K : \{\pm 1\}^K \to \{0, 1\}$. Recall that $\mathrm{MAJ}(x) = 1$ if and only if $\sum_{i=1}^K x_i > 0$. The following claim analyses its relevant properties.

Claim 2 *For every odd K,*

- $\overline{\mathrm{VAL}}(\mathrm{MAJ}_K) = 1 - \frac{1}{K+1}$.
- $\mathrm{VAR}_0(\mathrm{MAJ}_K) = \frac{K+1}{2}$.

Proof It is clear that MAJ_K has $\frac{K+1}{2}$ 0-variability. We show next that $\overline{\mathrm{VAL}}(\mathrm{MAJ}_K) = 1-\frac{1}{K+1}$. Suppose that $K = 2t+1$. Consider the distribution $\mathcal{D}$ on $\{\pm 1\}^K$ defined as follows. With probability $\frac{1}{2t+2}$ choose the all zero vector, and with probability $\frac{2t+1}{2t+2}$ choose a vector uniformly at random among all vectors with $t+1$ ones. It is clear that $\mathbb{E}_{x\sim\mathcal{D}}[\mathrm{MAJ}_K(x)] = 1-\frac{1}{2t+2}$. We claim that $\mathcal{D}$ is pairwise uniform, therefore, $\overline{\mathrm{VAL}}(\mathrm{MAJ}_K) \geq 1-\frac{1}{2t+2}$. Indeed for every distinct $i,j \in [K]$,

$$\Pr_{x\sim\mathcal{D}}\left((x_i,x_j)=(0,1)\right) = \Pr_{x\sim\mathcal{D}}\left((x_i,x_j)=(1,0)\right) = \frac{2t+1}{2t+2}\cdot\frac{t+1}{2t+1}\cdot\frac{t}{2t} = \frac{1}{4}\ ,$$
$$\Pr_{x\sim\mathcal{D}}\left((x_i,x_j)=(1,1)\right) = \frac{2t+1}{2t+2}\cdot\frac{t+1}{2t+1}\cdot\frac{t}{2t} = \frac{1}{4}\ ,$$

and $\Pr_{x\sim\mathcal{D}}\left((x_i,x_j)=(0,0)\right) = \frac{1}{4}$.

Next, we show that $\overline{\mathrm{VAL}}(\mathrm{MAJ}_K) \leq 1-\frac{1}{2t+t}$. Let $\mathcal{D}$ be a pairwise uniform distribution on $\{\pm 1\}^K$. We have $\mathbb{E}_{x\sim\mathcal{D}}\left[\sum_{i=1}^K \frac{x_i+1}{2}\right] = \frac{K}{2}$ therefore, by Markov's inequality,

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\mathop{\mathrm{MAJ}}_{K}(x)\right] = \Pr_{x\sim\mathcal{D}}\left(\mathop{\mathrm{MAJ}}_{K}(x)=1\right) = \Pr_{x\sim\mathcal{D}}\left(\sum_{i=1}^K \frac{x_i+1}{2} \geq t+1\right) \leq \frac{2t+1}{2(t+1)}\ .$$

Since this is true for every pairwise uniform distribution, $\overline{\mathrm{VAL}}(\mathrm{MAJ}_K) \leq \frac{2t+1}{2(t+1)} = 1-\frac{1}{K+1}$. □

Fix $\alpha \geq 1$. We will use theorem 3.4 to show that there is no efficient algorithm that approximately agnostically learns $\mathcal{H}$ with approximation ratio of α, unless the SRCSP assumption is false. Let $c > 1$ and denote $\beta = \frac{1}{10\alpha}$. It suffices to show that there is a polynomial ensamble $\mathcal{D} = \{\mathcal{D}_n^{m(n)}\}_{n=1}^\infty$ that is $(\Omega(n^c), \alpha\beta+\frac{1}{n})$-scattered and it is SRCSP-hard to distinguish between a $\mathcal{D}$-random sample and an β-almost realizable sample.

By assumption 4.5 and claim 2, for large enough odd K, it is SRCSP-hard to distinguish between a random instances of $\mathrm{CSP}(\mathrm{MAJ}_K)$ with $m(n) = n^c$ constraints and instances with value $\geq 1-\beta$. Consider the following ensamble $\mathcal{D} = \{\mathcal{D}_n^{2m(n)}\}_{n=1}^\infty$: pick $m = m(n)$ independent uniform vectors $x_1,\ldots,x_m \in \{x \in \{-1,1,0\}^n \mid |\{i \mid x_i \neq 0\}| = K|\}$. Then, consider the sample $S = \{(x_1,1),(-x_1,0),\ldots,(x_m,1),(-x_m,0)\}$. The theorem follows from the following claim:

Claim 3

- *$\mathcal{D}$ is $(\Omega(n^c), \alpha\beta+\frac{1}{n})$-scattered.*
- *It is* SRCSP*-hard to distinguish between a $\mathcal{D}$-random sample and an β-almost realizable sample.*

Proof The first part follows from proposition 7.1. Next, we show that it is SRCSP-hard to distinguish between a $\mathcal{D}$-random sample and β-almost realizable sample. We will reduce from the problem of distinguishing between a random instance with $m(n)$ constraints and an instance with value $\geq 1-\beta$. Given an instance J with $m(n)$ constraints, we will produce a sample S of $2m(n)$ examples by transforming each constraint into two examples as follows:

for the constraint $C(x) = \mathrm{MAJ}(j_1 x_{i_1}, \ldots, j_K x_{i_K})$ we denote by $u(C) \in \{x \in \{-1,1,0\}^n \mid |\{i \mid x_i \neq 0\}| = K|\}$ the vector whose i_l coordinate is j_l. We will produce the examples $(u(C), 1)$ and $(-u(C), 0)$. It is not hard to see that if J is random then $S \sim \mathcal{D}_n^{2m(n)}$. If the value of J is $\geq 1 - \beta$, indicated by an assignment $w \in \{\pm 1\}^n$, it is not hard to see that $h_w \in \mathcal{H}$ ϵ-almost realizes the sample S. This concludes the proof of the claim. □

Combining all the above we conclude the proof of theorem 5.4. □

Remark 7.3 *What is the real approximation ratio of agnostically learning halfspaces in n dimension? Taking a close look at the above proof, we see that in some sense, by the* SRCSP *assumption with* MAJ_K*, it is hard to agnostically learn halfspaces with approximation ratio of* $\Omega(K)$*. If we let K grow with n (this is* not *allowed by the* SRCSP*-hypothesis), say $K = \frac{1}{100}n$, we can hypothesize that it is hard to agnostically learn halfspaces with approximation ratio of about n. The approximation ratio of the best known algorithms is somewhat better, namely, $\frac{n}{\log(n)}$. But this is not very far from our guess. Therefore, one might hypothesize that the best possible approximation ratio is, say, of the form $\frac{n}{\mathrm{poly}(\log(n))}$. Given a rigorous treatment to the above intuition is left as an open question.*

Remark 7.4 *The problem of* learning large margin halfsapces *is an important variant of the problem of learning halfspaces. Here, we assume that the instance space is the unit ball in $\mathbb{R}^d$. For $1 > \gamma > 0$, the γ-margin error of a hyperplane h is the probability of an example to fall on the wrong side of h or at a distance $\leq \gamma$ from it. The γ-margin error of the best h (with respect to a distribution $\mathcal{D}$) is denoted $\mathrm{Err}_\gamma(\mathcal{D})$. An $\alpha(\gamma)$-approximation algorithm receives γ, ϵ as input and outputs a classifier with error rate $\leq \alpha(\gamma)\,\mathrm{Err}_\gamma(\mathcal{D}) + \epsilon$. Such an algorithm is efficient if it uses $\mathrm{poly}(\frac{1}{\gamma}, \frac{1}{\epsilon})$ samples and runs in time polynomial in the sample size. For a detailed definition, the reader is referred to [18].*

It is not hard to see that the proof of theorem 5.4 shows that it is hard to approximately learn large margin halfspaces with any constant approximation ratio. Taking considerations as in remark 7.3, one might hypothesize that the correct approximation ratio for this problem is about $\frac{1}{\gamma}$. As in the case of learning halfspaces, best known algorithms [37, 11] do just a bit better, namely, they have an approximation ratio of $\frac{1}{\gamma\sqrt{\log(1/\gamma)}}$. Therefore, one might hypothesize that the best possible approximation ratio is $\frac{1}{\gamma\,\mathrm{poly}(\log(1/\gamma))}$. We note that a recent result [18] shows that this is the best possible approximation ratio, if we restrict ourselves to a large class of learning algorithms (that includes SVM with a kernel, regression, Fourier transform and more).

7.3 Learning automata

For a function $q : \mathbb{N} \to \mathbb{N}$, let $\mathrm{AUTO}_{q(n)}$ be the class of functions $h : \{\pm 1\}^n \to \{0,1\}$ that can be realized by a finite automaton with $q(n)$ states.

Theorem 7.5 *For every $\epsilon > 0$, it is* SRCSP*-hard to learn* $\mathrm{AUTO}_{n^\epsilon}$*.*

Note 7.6 *The theorem remains true (with the same proof), even if we restrict to acyclic automata.*

Proof By a simple scaling argument, as in the proof of corollary 5.2, it is enough to show that it is SRCSP-hard to learn AUTO_{n^2+1}. By theorem 5.1, it is SRCSP-hard to learn $\text{DNF}^{\log_2(n)}$. To establish the theorem, we will show that if a function $h : \{\pm 1\}^n \to \{0,1\}$ can be realized by a DNF formula with $\log_2(n)$ clauses, then it can be realized by an automaton with $n^2 + 1$ states.

For simplicity, assume that n is a power of 2. Given a DNF formula R with $k := \log_2(n)$ clauses, we will construct an acyclic automaton as follows. For each variable we will have n states (corresponding to subsets of $[k]$). In addition, we will have a start state. From the start state, the automaton will jump to the state $(1, A)$, where A is the set of the indices of all the clauses in R that are not violated by the value of x_1. After reading x_2 the automaton will jump to the state $(2, A)$, where A is the set of the indices of all the clauses in R that are not violated by the values of x_1 and x_2. In this manner, after reading $x_1, \ldots, x_n$ the automaton will be at the state (n, A), where A is the set of the indices of all the clauses in R that are satisfied by $x_1, \ldots, x_n$. The automaton accepts if and only if $A \neq \emptyset$.

Clearly, this automaton calculates the same function as R. □

7.4 Toward intersection of 4 halfspaces

For $1 \leq l \leq k$ Consider the predicate $T_{k,l} : \{\pm 1\}^k \to \{0,1\}$ such that $T_{k,l}(x) = 1$ if and only if x has at least l ones. For example, $T_{k,1}$ is the SAT predicate, $T_{k,\lfloor \frac{k}{2} \rfloor + 1}$ is the MAJ predicate and $T_{k,k}$ is the AND predicate. Define $P_k : \left(\{\pm 1\}^k\right)^8 \to \{0,1\}$ by

$$P_K(x^1, \ldots, x^8) \quad = \quad \left(\wedge_{j=1}^{4} T_{k,\lceil \frac{k}{2} \rceil - 1}(x^j)\right) \wedge \neg \left(\wedge_{j=5}^{8} T_{k,\lceil \frac{k}{2} \rceil - 1}(x^j)\right) \ .$$

Proposition 7.7 *There is k_0 such that for every odd $k \geq k_0$ we have*

1. *Assuming the unique games conjecture, P_k is heredity approximation resistant.*
2. *For some constant $1 > \alpha > 0$, it is* **NP***-hard to distinguish between satisfiable instances to $\text{CSP}(P_k)$ and instances with value $\leq \alpha$.*

Proof We start with part 1. By [5], it suffices to show that there is a pairwise uniform distribution that is supported in $P_k^{-1}(1)$. Denote $Q(x^1, \ldots x^4) = \wedge_{j=1}^{4} T_{k,\lceil \frac{k}{2} \rceil - 1}(x^j)$ and $R(x^1, \ldots x^4) = \neg \left(\wedge_{j=1}^{4} T_{k,\lceil \frac{k}{2} \rceil - 1}(x^j)\right)$. Note that if $\mathcal{D}_Q$ is a pairwise uniform distribution that is supported in $Q^{-1}(1)$ and $\mathcal{D}_R$ is a pairwise uniform distribution that is supported in $R^{-1}(1)$, then $\mathcal{D}_Q \times \mathcal{D}_R$ is a pairwise uniform distribution that is supported in $P_k^{-1}(1)$. Therefore, it suffices to show that such $\mathcal{D}_Q$ and $\mathcal{D}_R$ exist.

We first construct $\mathcal{D}_Q$. Let $\mathcal{D}_k$ be the following distribution over $\{\pm 1\}^k$ – with probability $\frac{1}{k+1}$ choose the all-one vector and with probability $\frac{k}{k+1}$, choose at random a vector with $\lceil \frac{k}{2} \rceil - 1$ ones (uniformly among all such vectors). By the argument of claim 2, $\mathcal{D}_k$ is pairwise uniform. Clearly, the distribution $\mathcal{D}_Q = \mathcal{D}_k \times \mathcal{D}_k \times \mathcal{D}_k \times \mathcal{D}_k$ over $\left(\{\pm 1\}^k\right)^4$ is a pairwise uniform distribution that is supported in $Q^{-1}(1)$.

Next, we construct $\mathcal{D}_R$. Let k_0 be large enough so that for every $k \geq k_0$, the probability that a random vector from $\{\pm 1\}^k$ will have more than $\lceil \frac{k}{2} \rceil$ minus-ones is $\geq \frac{3}{8}$ (it is easy to

see that this probability approaches $\frac{1}{2}$ as k approaches ∞. Therefore, such k_0 exists). Now, let $Z \in \{0,1\}^4$ be a random variable that satisfies:

- $Z_1, \ldots, Z_4$ are pairwise independent.
- For every $1 \leq i \leq 4$, $\Pr(Z_i = 1) = \frac{3}{8}$.
- $\Pr(Z = (0,0,0,0)) = 0$.

In a moment, we will show that a random variable with the above properties exists. Now, let $B \subset \{\pm 1\}^k$ be a set with $|B| \geq \frac{3}{8} \cdot 2^k$ such that every vector in B has more than $\lceil \frac{k}{2} \rceil$ minus-ones. Consider the distribution $\mathcal{D}_R$ of the random variable $(X^1, \ldots, X^4) \in \left(\{\pm 1\}^k\right)^4$ sampled as follows. We first sample Z, then, for $1 \leq i \leq 4$, if $Z_i = 1$, we choose X^i to be a random vector B and otherwise, we choose X^i to be a random vector B^c.

We note that since $Z_1, \ldots, Z_4$ are pairwise independent, $X^1, \ldots, X^4$ are pairwise independent as well. Also, the distribution of X^i, $1 = 1, \ldots, 4$ is uniform. Therefore, $\mathcal{D}_R$ is pairwise uniform. Also, since $\Pr(Z = (0,0,0,0)) = 0$, with probability 1, at least one of the X^i's will have more than $\lceil \frac{k}{2} \rceil$ minus-ones. Therefore, $\mathcal{D}_R$ is supported in $R^{-1}(1)$.

It is left to show that there exists a random variable $Z \in \{0,1\}^4$ as specified above. Let Z be the random variable defined as follows:

- With probability $\frac{140}{192}$ Z is a uniform vector with a single positive coordinate.
- With probability $\frac{30}{192}$ Z is a uniform vector with 2 positive coordinates.
- With probability $\frac{22}{192}$ Z is a uniform vector with 4 positive coordinates.

Clearly, $\Pr(Z = (0,0,0,0)) = 0$. Also, for every distinct $1 \leq i, j \leq 4$ we have

$$\Pr(Z_i = 1) = \frac{140}{192} \cdot \frac{1}{4} + \frac{30}{192} \cdot \frac{1}{2} + \frac{22}{192} = \frac{3}{8}$$

and

$$\Pr(Z_i = 1, Z_j = 1) = \frac{30}{192} \cdot \frac{1}{6} + \frac{22}{192} = \left(\frac{3}{8}\right)^2 .$$

Therefore, the other two specifications of Z hold as well.

We proceed to part 2. The reduction is quite simple and we only sketch it. By adding dummy variables, it is enough to prove that it is NP-hard to distinguish between satisfiable instances of $\mathrm{CSP}(T_{k,\lceil \frac{k}{2} \rceil - 1})$ and instances with value $\leq \alpha$ for some constant $0 < \alpha < 1$. We will show somewhat stronger property, namely, that if $1 \leq l \leq k-2$, then for some $0 < \alpha < 1$, it is NP-hard to distinguish between satisfiable instances of $\mathrm{CSP}(T_{k,l})$ and instances with value $\leq \alpha$.

We will reduce from the problem of distinguishing between satisfiable instances to 3-SAT and instances with value $\leq \frac{8}{9}$. This problem is NP-hard [25]. Given an instance J to 3-SAT, we will produce an instance $R(J)$ to $\mathrm{CSP}(T_{k,l})$ as follows. Its variables would be the variables of J together with some new variables. For every constraint $C(x) = j_1 x_{i_1} \vee j_2 x_{i_2} \vee j_3 x_{i_3}$ in J, we will add $k + l - 1$ new variables $x_1^C, \ldots, x_k^C$ and $y_4^C, \ldots, x_{l+2}^C$. These new variables will be used only in the new clauses corresponding to C. We will introduce the following

constraints: we add the constraint $T_{k,l}(j_1 x_{i_1}, j_2 x_{i_2}, j_3 x_{i_3}, y_4^C, \ldots, y_{l+2}^C, -x_{l+3}^C, \ldots, -x_k^C)$. Also, for every $(j_1, \ldots, j_k) \in \{\pm 1\}^k$ with at most $(k-l)$ minus-ones we will add the constraint $T_{k,l}(j_1 x_1^C, \ldots, j_k x_k^C)$.

If J is satisfiable, then $R(J)$ is satisfiable as well: simply set all new variables to 1. On the other hand, if $\mathrm{VAL}(J) \leq \frac{8}{9}$, then it is not hard to see that for every assignment to $R(J)$'s variables, for at least $\frac{1}{9}$ of J's clauses, at least one of the new clauses corresponding to it will be unsatisfied. Since we introduce $\leq 2^k$ constraints in $R(J)$ for each constraint in J, we conclude that $\mathrm{VAL}(R(J)) \leq 1 - 2^{-k}\frac{1}{9}$. Therefore, the theorem holds with $\alpha = 1 - 2^{-k}\frac{1}{9}$. □

Conjecture 7.8 *P_k is heredity approximation resistant on satisfiable instances.*

Theorem 7.9 *Assuming conjecture 7.8, it is* SRCSP*-hard to learn* INTER_4

Proof (sketch) The proof goes along the same lines of the proof of theorem 5.4. We will prove SRCSP-hardness for learning intersections of two halfspaces over $\{-1, 1, 0\}^n$, induced by ± 1 vectors. As in the proof of theorem 5.4, SRCSP-hardness of learning intersections of two halfspaces over the boolean cube follows from this.

Fix $d > 0$. It is not hard to check that $\mathrm{VAR}_0(P_k) \geq \lceil \frac{k}{2} \rceil - 2$. Therefore, by conjecture 7.8 and assumption 4.3, for large enough odd k, it is SRCSP-hard to distinguish between a random instance to $\mathrm{CSP}(P_k)$ with n^d constraints and a satisfiable instance. We will reduce from this problem to the problem of distinguishing between a realizable sample and a random sample that is $(\Omega(m^d), \frac{1}{5})$-scattered. Since d is arbitrary, the theorem follows.

Given an instance J, we produce two examples for each constraint: for the constraint

$$\begin{aligned} C(x) \quad = \quad & \left(\wedge_{q=1}^{4} T_{k,\lceil \frac{k}{2} \rceil - 1}(j_{q,1} x_{i_{q,1}}, \ldots, j_{q,k} x_{i_{q,k}}) \right) \\ & \wedge \neg \left(\wedge_{q=5}^{8} T_{k,\lceil \frac{k}{2} \rceil - 1}(j_{q,1} x_{i_{q,1}}, \ldots, j_{q,k} x_{i_{q,k}}) \right) \end{aligned}$$

we will produce two examples in $\{-1, 1, 0\}^{4n} \times \{0, 1\}$, each of which has exactly $4k$ non zero coordinates. The first is a positively labelled example whose instance is the vector with the value $j_{q,l}$, $1 \leq q \leq 4, 1 \leq l \leq k$ in the $n(q-1) + i_{q,l}$ coordinate. the second is a negatively labelled example whose instance is the vector with the value $j_{q,l}$, $5 \leq q \leq 8, 1 \leq l \leq k$ in the $n(q-5) + i_{q,l}$ coordinate.

It is not hard to see that if J is satisfiable then the produced sample is realizable by intersection of four halfspaces: if $u \in \{\pm 1\}^n$ is a satisfying assignment then the sample is realized by the intersection of the 4 halfspaces $\sum_{i=1}^{n} u_i x_{n(q-1)+i} \geq -1, \quad q = 1, 2, 3, 4$. On the other hand, by proposition 7.1, if J is random instance with n^d constraints, then the resulting ensamble is $(\Omega(n^d), \frac{1}{5})$ scattered.

□

7.5 Agnostically learning parity

For convenience, in this section the domain of hypotheses will be $\{0, 1\}^n$ and the domain of predicates will be $\{0, 1\}^K$ (instead of $\{\pm 1\}^n$ and $\{\pm 1\}^K$). For every $S \subset [n]$ define $\chi_S : \{0, 1\}^n \to \{\pm 1\}$ by $\chi_S(x) = \oplus_{i \in S} x_i$. Let PARITY be the hypothesis class consisting of all functions χ_S, $S \subset [n]$.

Theorem 7.10 *For every constant $\alpha \geq 1$, it is* SRCSP*-hard to approximately agnostically learn* PARITY *with an approximation ratio of α.*

Proof Let $P_K : \{0,1\}^K \to \{0,1\}$ be the parity predicate. That is, $P_K(x) = \oplus_{i=1}^K x_i$. We first show that for $K \geq 3$, $\overline{\mathrm{VAL}}(P_K) = 1$. Indeed, a pairwise uniform distribution which is supported in $P_K^{-1}(1)$ is the following – choose $(x_1, \ldots, x_{K-1})$ uniformly at random and then choose x_K so that $P_K(x) = 1$. Second, it is clear that $\mathrm{VAR}_0(P_K) = K$. Therefore, by assumption 4.5, for every $\beta > 0$ and every d, for sufficiently large K, it is SRCSP-hard to distinguish between instances to $\mathrm{CSP}(P_K)$ with value $\geq 1-\beta$ and random instances with m^d constraints. Note that with the convention that the domain of P_K is $\{0,1\}^K$, the constraints of instances to $\mathrm{CSP}(P_K)$ are of the form $C(x) = x_{i_1} \oplus \ldots \oplus x_{i_K}$ or $C(x) = x_{i_1} \oplus \ldots \oplus x_{i_K} \oplus 1$.

We will reduce from the aforementioned problem to the problem of distinguishing between β-almost realizable sample and $\mathcal{D}$-random sample for a distribution $\mathcal{D}$ which is $\left(\Omega\left(n^d\right), \frac{1}{4}\right)$-scattered. Since both β and d are arbitrary, the theorem follows from theorem 3.4.

Given an instance J to $\mathrm{CSP}(P_K)$, for each constraint $C(x) = x_{i_1} \oplus \ldots \oplus x_{i_K} \oplus b$ we will generate an example (u_C, y_C) where u_C is the vector with ones precisely in the coordinates $i_1, \ldots, i_K$ and $y_C = b$. It is not hard to verify that if J is a random instance with m^d constraints then the generated sample is $\left(\Omega\left(n^d\right), \frac{1}{4}\right)$-scattered. On the other hand, assume that the assignment $\psi \in \{0,1\}^n$ satisfies $1-\beta$ fraction of the constraints. Consider the hypothesis χ_S where $S = \{i \mid x_i = 1\}$. We have $\chi_S(x_C) = \oplus_{i \in S}(u_C)_i = \oplus_{q=1}^K \psi_{i_q}$. Therefore, ψ satisfies C if and only if χ_S is correct on (u_C, y_C). Since ψ satisfies $1-\beta$ fraction of the constraints, the generated sample is β-almost realizable. □

8 Resolution lower bounds

In this section we prove theorem 4.2. Let $P : \{0,1\}^K \to \{0,1\}$ be some predicate. Let $\tau = \{T_1, \ldots, T_r\}$ be a resolution refutation for a $\mathrm{CSP}(P)$ instance J. A basic parameter associated with τ is the *width*. The *width* of a clause is the number of literals it contains, and the *width* of τ is $\mathrm{width}(\tau) := \max_{1 \leq i \leq r} \mathrm{width}(T_i)$. We also define the width of an unsatisfiable instance J to $\mathrm{CSP}(P)$ as the minimal width of a resolution refutation of J. Ben-Sasson and Wigderson [9] have shown that if an instance to $\mathrm{CSP}(P)$ has a short resolution refutation, then it necessarily has a narrow resolution refutation. Namely,

Theorem 8.1 ([9]) *Let J be an unsatisfiable instance to* $\mathrm{CSP}(P)$*. The length of every resolution refutation for J is at least $2^{\Omega\left(\frac{\mathrm{width}^2(J)}{n}\right)}$.*

Theorem 4.2 now follows from theorem 8.1 and the following two lemmas.

Lemma 8.2 *Let J be an unsatisfiable instance to* $\mathrm{CSP}(P)$*. Assume that for every subset I of l constraints from J, most of the constraints in I have $\geq K - \mathrm{VAR}_0(P) - 1$ variables that do not appear in any other constraint in I. Then* $\mathrm{width}(J) \geq \frac{l}{6}$.

Proof Let $\tau = \{T_1, \ldots, T_r\}$ be a resolution refutation to J. Define $\mu(T_i)$ as the minimal number μ such that T_i is implied by μ constraints in J.

Claim 4

1. $\mu(\emptyset) > l$.

2. *If* T_i *is implied by* T_{i_1}, T_{i_2}, $i_1, i_2 < i$ *then* $\mu(T_i) \leq \mu(T_{i_1}) + \mu(T_{i_2})$.

Proof The second property clearly holds. To prove the first property, suppose toward a contradiction that $\mu(\emptyset) \leq l$. It follows that there are $t \leq l$ constraints $I \subset J$ that implies the empty clause, i.e., it is impossible to simultaneously satisfy all the constraints in I. By the assumption of the lemma, it is possible to choose an ordering $I = \{C_1, \ldots, C_t\}$ such that for every $1 \leq i \leq t$, C_i contains at least $K - \mathrm{VAR}_0(P) - 1$ variables that do not appear in $C_1, \ldots, C_{i-1}$. Indeed, let us simply take C_t to be a clause that contains at least $K - \mathrm{VAR}_0(P) - 1$ variables that do not appear in the clauses in $I \setminus \{C_t\}$. Then, choose C_{t-1} in the same way from $I \setminus \{C_t\}$ and so on. Now, let $\psi \in \{\pm 1\}^n$ be an arbitrary assignment that satisfies C_1. By the definition of 0-variability, it is possible to change the values of the variables appearing in C_2 but not in C_1 to satisfy also C_2. We can continue doing so till we reach an assignment that satisfies $C_1, \ldots, C_t$ simultaneously. This leads to the desired contradiction. □

By the claim, and the fact that $\mu(C) = 1$ for every clause that is implied by one of the constraints of J, we conclude that there is some T_i with $\frac{l}{3} \leq \mu = \mu(T_j) \leq \frac{2l}{3}$. It follows that there are μ constraints $C_1, \ldots, C_\mu$ in J that imply T_j, but no strict subset of these clauses implies T_j. For simplicity, assume that these constraints are ordered such that for every $1 \leq i \leq \frac{\mu}{2}$, C_i contains at least $K - \mathrm{VAR}_0(P) - 1$ variables that do not appear in the rest of these constraints. The proof of the lemma is established by the following claim

Claim 5 *For every* $1 \leq i \leq \frac{\mu}{2}$, T_j *contains a variable appearing only is* C_i.

Proof Assume toward a contradiction that the claim does not hold for some $1 \leq i \leq \frac{\mu}{2}$. Since no strict subset of $C_1, \ldots, C_\mu$ imply T_j, there is an assignment $\psi \in \{\pm 1\}^n$ such that for every $i' \neq i$, $C_{i'}(\psi) = 1$ but $T_j(\psi) = 0$. Since $C_1, \ldots, C_\mu$ imply T_j, we must have $C_i(\psi) = 0$. Now, by the definition of 0-variability, we can modify the values of the $K - \mathrm{VAR}_0(P) - 1$ variables that appear only in C_i to have a new assignment $\psi' \in \{\pm 1\}^n$ with $C_i(\psi') = 1$. Since T_j and the rest of the constraints do not contain these variables, we conclude that still for every $i' \neq i$, $C_{i'}(\psi) = 1$ and $T_j(\psi) = 0$. This contradicts the fact that $C_1, \ldots, C_\mu$ imply T_j. □

□

The next lemma shows that the condition in lemma 8.2 holds w.h.p. for a suitable random instance. For the sake of readability, it is formulated in terms of sets instead of constraints.

Lemma 8.3 *Fix integers* $k > r > d$ *such that* $r > \max\{17d, 544\}$. *Suppose that* $A_1, \ldots, A_{n^d} \in \binom{[n]}{k}$ *are chosen uniformly at random. Then, with probability* $1 - o_n(1)$, *for every* $I \subset [n^d]$ *with* $|I| \leq n^{\frac{3}{4}}$ *for most* $i \in I$ *we have* $|A_i \setminus \cup_{j \in I \setminus \{i\}} A_j| \geq k - r$.

Proof Fix a set I with $2 \leq t \leq n^{\frac{3}{4}}$ elements. Order the sets in I arbitrarily and also order the elements in each set arbitrarily. Let $X_1, \ldots, X_{kt}$ be the following random variables: X_1 is the first element in the first set of I, X_2 is the second element in the first set of I and so on till the k'th element of the last set of I.

Denote by R_i $1 \leq i \leq kt$ the indicator random variable of the event that $X_i = X_j$ for some $j < i$. We claim that if $\sum R_i < \frac{tr}{4}$, the conclusion of the lemma holds for I. Indeed,

let $J_1 \subset I$ be the set of indices with $R_i = 1$, $J_2 \subset I$ be the set of indices i with $R_i = 0$ but $X_i = X_j$ for some $j > i$ and $J = J_1 \cup J_2$. If the conclusion of the lemma does not hold for I, then $|J| \geq \frac{tr}{2}$. If in addition $|J_1| = \sum R_i < \frac{tr}{4}$ we must have $|J_2| > \frac{tr}{4} > |J_1|$. For every $i \in J_2$, let $f(i)$ be the minimal index $j > i$ such that $X_i = X_j$. We note that $f(i) \in J_1$, therefore f is a mapping from J_2 to J_1. Since $|J_2| > |J_1|$, $f(i_1) = f(i_2)$ for some $i_1 < i_2$ in J_2. Therefore, $X_{i_1} = X_{f(i_1)} = X_{i_2}$ and hence, $R_{i_2} = 1$ contradicting the assumption that $i_2 \in J_2$.

Note that the probability that $R_i = 1$ is at most $\frac{tk}{n}$. This estimate holds also given the values of $R_1, \ldots, R_{i-1}$. It follows that the probability that $R_i = 1$ for every $i \in A$ for a particular $A \subset I$ with $|A| = \lceil \frac{rt}{4} \rceil$ is at most $\left(\frac{tk}{n}\right)^{\frac{rt}{4}}$. Therefore, for some constants $C', C > 0$ (that depend only on d and k), the probability that J fails to satisfy the conclusion of the lemma is bounded by

$$\begin{aligned}
\Pr\left(\sum R_i \geq \frac{tr}{4}\right) &\leq \binom{tk}{\lceil \frac{tr}{4} \rceil} \left(\frac{tk}{n}\right)^{\frac{tr}{4}} \\
&\leq 2^{C \cdot t} \left(\frac{tk}{n}\right)^{\frac{tr}{4}} \\
&\leq 2^{C' \cdot t} \left(\frac{t}{n}\right)^{\frac{tr}{4}}
\end{aligned}$$

The second inequality follows from Stirling's approximation. Summing over all collections I of size t we conclude that for some $C'' > 0$, the probability that the conclusion of the lemma does not hold for some collection of size t is at most

$$\binom{n^d}{t} 2^{C' \cdot t} \left(\frac{t}{n}\right)^{\frac{tr}{4}} \leq n^{dt - \frac{1}{16}tr} \cdot 2^{C' \cdot t} \leq n^{-\frac{1}{272}tr} \cdot 2^{C' \cdot t} \leq n^{-2t} \cdot 2^{C' \cdot t} \leq C'' \frac{1}{n}$$

Summing over all $2 \leq t \leq n^{\frac{3}{4}}$, we conclude that the probability that the conclusion of the lemma does not hold is at most $C'' n^{-\frac{1}{4}} = o_n(1)$. □

9 On basing the SRCSP assumption on NP-Hardness

Fix a predicate $P : \{\pm 1\}^K \to \{0,1\}$ and let $1 \geq \alpha > \underline{\mathrm{VAL}}(P)$. Let $L \subset \{0,1\}^*$ be some language. We say that L can be efficiently reduced to the problem of distinguishing between random instances to CSP(P) with Cn constraints and instances with value $\geq \alpha$, if there is an efficient probabilistic Turing machine that given $x \in \{0,1\}^n$, acts as follows: for some function $f : \mathbb{N} \to \mathbb{N}$,

- If $x \in L$ then $M(x)$ is an instance to CSP(P) with $f(n)$ variables, $C \cdot f(n)$ constraints and value $\geq \alpha$.
- If $x \notin L$ then $M(x)$ is a random instance to CSP(P) with $f(n)$ variables and $C \cdot f(n)$ constraints.

Theorem 9.1 *For every sufficiently large constant $C > 0$, the following holds. Assume that the language $L \subset \{0,1\}^*$ can be efficiently reduced to the problem of distinguishing between random instances to $\mathrm{CSP}(P)$ with $m(n) \geq Cn$ constraints and instances with value $\geq \alpha$. Then, L has a statistical zero knowledge proof.*

Corollary 9.2 *For every sufficiently large constant $C > 0$, the following holds. Assume that there is a reduction from an either an* **NP***-hard or* **CoNP***-hard problem to the problem of distinguishing between random instances to $\mathrm{CSP}(P)$ with $m(n) \geq Cn$ constraints and instances with value $\geq \alpha$. Then, the polynomial hierarchy collapses.*

Proof (of corollary 9.2) Under the conditions of the corollary, by theorem 9.1, we have **NP** $\subset$ **SZKP** or **CoNP** $\subset$ **SZKP**. Since **SZKP** is closed under taking complement [39], in both cases, **NP** $\subset$ **SZKP**. Since **SZKP** $\subset$ **CoAM** [1], we conclude that **NP** $\subset$ **CoAM**, which collapses the polynomial hierarchy [14]. □

Proof (of theorem 9.1) Let $C > 0$ be a constant large enough so that, with probability $\geq \frac{1}{2}$, a random instance to $\mathrm{CSP}(P)$ with Cn constraints will have value $\leq \alpha$.

Consider the following problem. The input is a circuit $\Psi : \{0,1\}^n \to \{0,1\}^m$ and a number t. The instance is a YES instance if the entropy[5] of Ψ, when it acts on a uniform input sampled from $\{0,1\}^n$, is $\leq t-1$. The instance is a NO instance if this entropy is $\geq t$. By [23] this problem is in **SZKP**. To establish the proof, we will show that L can be reduced to this problem.

Assume that there is a reduction from the language L to the problem of distinguishing between random instances to $\mathrm{CSP}(P)$ with $m(n) \geq Cn$ constraints and instances with value $\geq \alpha$. Let M and f be a Turing machine and a function that indicate that. By a standard argument, it follows that there is an efficient deterministic Turing machine M' that given $x \in \{0,1\}^n$ produces a circuit Ψ whose input is $\{0,1\}^{g(n)}$ for some polynomially growing function and whose output is an instance to $\mathrm{CSP}(P)$, such that, for a uniformly randomly chosen input $z \in \{0,1\}^{g(n)}$,

- If $x \in L$ then $\Psi(z)$ is a (possibly random) satisfiable instance to $\mathrm{CSP}(P)$ with $f(n)$ variables and $m(f(n))$ constraints.
- If $x \notin L$ then $\Psi(z)$ is a random instance to $\mathrm{CSP}(P)$ with $f(n)$ variables and $m(f(n))$ constraints.

Since the number of instances to $\mathrm{CSP}(P)$ with $m(f(n))$ constraints is $\left(\binom{f(n)}{K}2^K\right)^{m(f(n))}$, in the second case, the entropy of Ψ is $q(n) := m(f(n))\log_2\left(\binom{f(n)}{K}2^K\right)$. On the other hand, in the first case, the entropy is at most the entropy of a random instance to $\mathrm{CSP}(P)$ with $m(f(n))$ constraints and value $\geq \alpha$. By the choice of C, the number of such instances is at most half of the total number of instances with $m(f(n))$ constraints. Therefore, the entropy of Ψ is at most $m(f(n))\log_2\left(\binom{f(n)}{K}2^K\right) - 1 = q(n) - 1$. Hence, using M', we can reduce L to the problem mentioned in the beginning of the proof. □

[5] We consider the standard Shannon's entropy with bits units.

Acknowledgements: Amit Daniely is a recipient of the Google Europe Fellowship in Learning Theory, and this research is supported in part by this Google Fellowship. Nati Linial is supported by grants from ISF, BSF and I-Core. Shai Shalev-Shwartz is supported by the Israeli Science Foundation grant number 590-10. We thank Sangxia Huang for his kind help and for valuable discussions about his paper [27]. We thank Guy Kindler for valuable discussions.

References

[1] William Aiello and Johan Hastad. Statistical zero-knowledge languages can be recognized in two rounds. *Journal of Computer and System Sciences*, 42(3):327–345, 1991.

[2] Michael Alekhnovich. More on average case vs approximation complexity. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 298–307. IEEE, 2003.

[3] B. Applebaum, B. Barak, and D. Xiao. On basing lower-bounds for learning on worst-case assumptions. In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 211–220. IEEE, 2008.

[4] Sanjeev Arora, László Babai, Jacques Stern, and Z Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. In *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on*, pages 724–733. IEEE, 1993.

[5] Per Austrin and Elchanan Mossel. Approximation resistant predicates from pairwise independence. *Computational Complexity*, 18(2):249–271, 2009.

[6] Boaz Barak, Guy Kindler, and David Steurer. On the optimality of semidefinite relaxations for average-case and generalized constraint satisfaction. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 197–214. ACM, 2013.

[7] Paul Beame and Toniann Pitassi. Simplified and improved resolution lower bounds. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 274–282. IEEE, 1996.

[8] Paul Beame, Richard Karp, Toniann Pitassi, and Michael Saks. On the complexity of unsatisfiability proofs for random k-cnf formulas. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 561–571. ACM, 1998.

[9] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrowresolution made simple. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 517–526. ACM, 1999.

[10] Quentin Berthet and Philippe Rigollet. Computational lower bounds for sparse pca. In *COLT*, 2013.

[11] A. Birnbaum and S. Shalev-Shwartz. Learning halfspaces with the zero-one loss: Time-accuracy tradeoffs. In *NIPS*, 2012.

[12] Christopher M Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[13] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

[14] Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for np problems. *SIAM Journal on Computing*, 36(4):1119–1159, 2006.

[15] Amin Coja-Oghlan, Colin Cooper, and Alan Frieze. An efficient sparse regularity concept. *SIAM Journal on Discrete Mathematics*, 23(4):2000–2034, 2010.

[16] Stephen A Cook and Robert A Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979.

[17] Amit Daniely, Nati Linial, and Shai Shalev-Shwartz. More data speeds up training time in learning halfspaces over sparse vectors. In *NIPS*, 2013.

[18] Amit Daniely, Nati Linial, and Shai Shalev-Shwartz. The complexity of learning halfspaces using generalized linear methods. *Arxiv preprint arXiv:1211.0616 v3*, 2013.

[19] Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, 1962.

[20] Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 534–543. ACM, 2002.

[21] Uriel Feige and Eran Ofek. Easily refutable subformulas of large random 3cnf formulas. In *Automata, languages and programming*, pages 519–530. Springer, 2004.

[22] V. Feldman, P. Gopalan, S. Khot, and A.K. Ponnuswami. New results for learning noisy parities and halfspaces. In *In Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, 2006.

[23] Oded Goldreich and Salil Vadhan. Comparing entropies in statistical zero knowledge with applications to the structure of szk. In *Computational Complexity, 1999. Proceedings. Fourteenth Annual IEEE Conference on*, pages 54–73. IEEE, 1999.

[24] Armin Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–308, 1985.

[25] Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001.

[26] Sangxia Huang. Approximation resistance on satisfiable instances for predicates strictly dominating parity. 2012.

[27] Sangxia Huang. Approximation resistance on satisfiable instances for predicates with few accepting inputs. In *STOC*, 2013.

[28] Adam Tauman Kalai, Adam R Klivans, Yishay Mansour, and Rocco A Servedio. Agnostically learning halfspaces. *SIAM Journal on Computing*, 37(6):1777–1805, 2008.

[29] Michael Kearns and Leslie G. Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. In *STOC*, pages 433–444, May 1989.

[30] Michael Kharitonov. Cryptographic hardness of distribution-specific learning. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 372–381. ACM, 1993.

[31] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775. ACM, 2002.

[32] Subhash Khot and Rishi Saket. On the hardness of learning intersections of two halfspaces. *Journal of Computer and System Sciences*, 77(1):129–141, 2011.

[33] Adam R Klivans and Ryan O'Donnell. Learning intersections and thresholds of halfspaces. In *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, pages 177–186. IEEE, 2002.

[34] Adam R Klivans and Rocco Servedio. Learning dnf in time $2^{O(n^{1/3})}$. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 258–265. ACM, 2001.

[35] Adam R. Klivans and Alexander A. Sherstov. Cryptographic hardness for learning intersections of halfspaces. In *FOCS*, 2006.

[36] Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. In *FOCS*, pages 574–579, October 1989.

[37] P.M. Long and R.A. Servedio. Learning large-margin halfspaces with more malicious noise. In *NIPS*, 2011.

[38] Yishay Mansour. An $o(n \log \log n)$ learning algorithm for dnf under the uniform distribution. *Journal of Computer and System Sciences*, 50(3):543–550, 1995.

[39] Tatsuaki Okamoto. On relationships between statistical zero-knowledge proofs. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 649–658. ACM, 1996.

[40] L. Pitt and L.G. Valiant. Computational limitations on learning from examples. *Journal of the Association for Computing Machinery*, 35(4):965–984, October 1988.

[41] Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 245–254. ACM, 2008.

[42] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958. (Reprinted in *Neurocomputing* (MIT Press, 1988).).

[43] R.E. Schapire. The strength of weak learnability. In *FOCS*, pages 28–33, October 1989.

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

[45] V. N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.