Learning Hurdles for Sleeping Experts

Varun Kanade*
EECS, University of California
Berkeley, CA, USA
vkanade@eecs.berkeley.edu

Thomas Steinke†
SEAS, Harvard University
Cambridge, MA, USA
tsteinke@fas.harvard.edu

September 1, 2012

Abstract

We study the online decision problem where the set of available actions varies over time, also called the *sleeping experts* problem. We consider the setting where the performance comparison is made with respect to the *best ordering* of actions in hindsight. In this paper, both the payoff function and the availability of actions is adversarial. Kleinberg et al. (2008) gave a computationally efficient no-regret algorithm in the setting where payoffs are stochastic. Kanade et al. (2009) gave an efficient no-regret algorithm in the setting where action availability is stochastic.

However, the question of whether there exists a *computationally efficient* no-regret algorithm in the adversarial setting was posed as an open problem by Kleinberg et al. (2008). We show that such an algorithm would imply an algorithm for PAC learning DNF, a long standing important open problem. We also consider the setting where the number of available actions is restricted, and study its relation to agnostic learning monotone disjunctions over examples with bounded Hamming weight.

1 Introduction

In online decision problems, a decision-maker must choose one of n possible *actions*, in each of the total T rounds. Based on her choice, the decision-maker receives a payoff in the range $[0, 1]$. In the *full information* or *expert* setting, at the end of each round, the decision-maker sees the payoff corresponding to each of the possible actions. In the *bandit* setting, she only observes the reward of the action that she chose. The goal of the decision maker is to maximize her total payoff across T rounds, or as is common in the *non-stochastic* setting, to minimize her *regret* with respect to a class of strategies. The regret of the decision-maker is defined as the difference between the payoff she would have received by following the best strategy in hindsight from the class and the payoff that she actually received. There is vast literature studying the online experts problem. The optimal strategy for a decision-maker is to randomly choose an action biasing the choice by exponentially weighted cumulative payoff scores. (See for example the Hedge algorithm due to Freund and Schapire [1995].)

In this paper, we focus on the so-called *sleeping experts* problem. In this problem, there are a total of n actions, and at each round t, a subset, S^t, is available to the decision-maker. The

*This work was completed while the author was at Harvard University supported in part by NSF grants CCF-04-27129 and CCF-09-64401

†Supported in part by the Lord Rutherford Memorial Research Fellowship and NSF grant CCF-1116616

class of strategies we compare against is the set of *rankings* over the n total actions. Each ranking induces a simple strategy for the online decision problem: pick the highest-ranked available action. As a motivating example, consider the problem of choosing an advertisement to display alongside a search query. Of all the ads that match the particular keyword, only a subset might be available for displaying because of budget, geographical or other constraints. In this case, we would want the decision-making algorithm to compare well against the best (in hindsight) hypothetical ranking on the ads.

Our work focuses on the fully non-stochastic setting, where both the set of available actions and their payoffs are decided by an adversary[1]. In this paper, we consider the case of an *oblivious adversary*, *i.e.* one that does not observe the actual (random) choices made by the decision-maker. Since our results show computational difficulties in designing efficient no-regret algorithms, they are equally applicable to the more challenging case of an adaptive adversary. An algorithm that selects an action, a^t, at time step, t, is efficient, if it makes its choice (possibly using history) in time that is polynomial in n. An algorithm is said to be a *no-regret* algorithm if its regret is $O(\text{poly}(n)T^{1-\delta})$ for some constant $\delta > 0$. An informal statement of our main result is:

Theorem 1. *If there exists a computationally efficient no-regret algorithm for the sleeping experts problem (with respect to ranking strategies), then the class of polynomial size* DNFs *is PAC-learnable under arbitrary distributions.*

In contrast to the above result, if computational efficiency is not a concern, it is easy to see that the Hedge algorithm [Freund and Schapire, 1995] achieves regret $O(\sqrt{n\log(n)T})$, by treating each of the $n!$ rankings as a (meta-)expert. This observation was made by Kleinberg et al. [2008], who also showed that when the class of online algorithms is restricted to those that select actions by sampling over rankings and *without* observing the set of available actions S^t, there is no efficient no-regret algorithm unless RP = NP. However, this is a severe restriction and whether there exists an efficient no-regret algorithm without such restrictions was posed by Kleinberg et al. as an open question. Our result shows that such an algorithm would imply an algorithm for PAC-learning DNFs under arbitrary distributions, a long standing important open problem [Valiant, 1984]. The best known algorithm for PAC-learning DNFs is due to Klivans and Servedio [2001] and takes time $2^{\tilde{O}(n^{1/3})}$. In fact, we show that the sleeping experts problem is at least hard as agnostic learning disjunctions, the best known algorithm for which takes time $2^{\tilde{O}(\sqrt{n})}$ [Kalai et al., 2005].

Our Contributions

We use show that the sleeping experts problem is at least as hard as a PAC learning problem, which is widely believed to be hard. As far as we are aware, computational hardness assumptions have not been previously used to show lower bounds on regret in experts/bandits problems[2]. Lower bounds in the literature are usually based on information theoretic arguments (such as predicting coin tosses). In the sleeping experts setting, the information-theoretic lower bound of $\Omega(\sqrt{n\log(n)T})$ can indeed be achieved if computational efficiency is not a concern.

We also consider a computationally easy restriction of the sleeping experts problem, namely where the number of available actions is at most some constant, k. While computational efficiency is easy to achieve in this case, the problem of achieving the right regret rate in terms of n is open. For $k \geq 3$, the only algorithm we know is a trivial one, which achieves regret $O(\sqrt{n^k T})$. However,

[1] No-regret algorithms are known for the case when either the payoffs or action availabilities are stochastic; these are discussed in the related works section.

[2] In the case of online learning of concepts (such as linear separators), such lower bounds have been shown before (see e.g. Shalev-Shwartz et al. [2010])

running Hedge, which is computationally inefficient, achieves regret $O(\sqrt{nT})$. For $k = 2$, a recent result of Hazan et al. [2012] shows that achieving nearly optimal regret by a computationally efficient algorithm is possible. We show the restriction of the sleeping experts problem is at least as hard as the problem of agnostic learning monotone disjunctions under distributions with support over points having Hamming weight at most $k - 1$.

Related Work

The most relevant related work to ours is that of Kleinberg et al. [2008] and Kanade et al. [2009]. Kleinberg et al. showed that in the setting where payoffs are stochastic (i.e. are drawn from a fixed distribution on each round and independently for each action) and action availability is adversarial, there exists an efficient no-regret algorithm that is essentially information-theoretically optimal. Kanade et al. gave an efficient no-regret algorithm in the setting when the payoffs are set by an oblivious adversary, but the action availability is decided stochastically, *i.e.* a subset $S \subseteq [n]$ of available actions is drawn according to a fixed distribution at each time step. In contrast, our results in this paper show that an adversarial coupling between action availability and payoffs makes the problem much harder.

The set of available experts may be thought of as context information at each time step, and hence allows for encoding learning problems (in our case agnostic learning of disjunctions). We believe that such techniques can be applied to show hardness results of other contextual experts/bandits problems as well. It is interesting to note in certain cases the opposite approach also works, *i.e.* when the learning problem corresponding to the contextual experts problem admits an efficient algorithm, it can be converted to an efficient no-regret online algorithm [Langford and Zhang, 2007, Beygelzimer et al., 2011, Dudik et al., 2011]. When not in the contextual experts/bandits setting, it is often possible to compete against a class of exponentially many experts using a computationally efficient algorithm (see Cesa-Bianchi and Lugosi [2006] Chap. 5).

In earlier literature, different versions of the sleeping experts problems have been considered by Freund et al. [1997] and Blum and Mansour [2007]. Our results are not applicable to their settings, and in fact computationally efficient no-regret algorithms are known in those settings.

2 Preliminaries

We begin by describing the sleeping experts problem and introduce notation used in this paper. In Section 2.2, we describe the online and batch agnostic learning settings and state the result relating online and batch learning.

2.1 Setting and Notation

Let $A = \{a_1, \ldots, a_n\}$ be the set of actions. Let T be the total number of time steps for the online decision problem. In the sleeping experts setting, at time step, t, a subset, $S^t \subseteq A$, of actions is available, from which the decision-maker picks an action, $a^t \in S^t$. Let $p^t : S^t \to [0, 1]$ be the payoff function, and for any action, a, let $p^t(a)$ denote the payoff associated with action, a, at time step, t. At the end of round, t, the entire payoff function, p^t, is revealed to the decision-maker. The total payoff of the decision-maker across T rounds is simply:

$$P_{\text{DM}} = \sum_{t=1}^{T} p^t(a^t)$$

When choosing an action $a^t \in S^t$, at time step t, the decision-maker may use history to guide her choice. If the adversary cannot see any of the choices of the decision-maker, we say that the adversary is *oblivious*. An *adaptive* adversary can see the past choices of the decision-maker and may then decide the payoff function and action availability. In this paper, we only consider the oblivious adversarial setting, since the hardness of designing no-regret algorithms against oblivious adversaries also applies to the case of (stronger) adaptive adversaries. Also, we only consider the full information setting, since the bandit setting is strictly harder.

The set of *strategies* that the decision-maker has to compete against is defined by the set of *rankings* over the actions. Let Σ_A denote the set of all $n!$ possible rankings over the n total actions. Given a particular ranking, $\sigma \in \Sigma_A$, the corresponding strategy is to play the highest ranked available action according to σ. For subset, $S \subseteq A$, of available actions, let $\sigma(S) \in A$ denote the action in S which is ranked highest according to σ. Thus, the payoff obtained by playing according to strategy σ is:

$$P_\sigma = \sum_{t=1}^{T} p^t(\sigma(S^t))$$

The quantity of interest is the *regret* of the decision-maker with respect to the class of ranking strategies. The regret is defined as the difference between the payoff that would have been obtained by playing according to the *best ranking strategy in hindsight* and the actual payoff received by the decision maker. Thus,

$$\text{Regret}_{\text{DM}} = \max_{\sigma \in \Sigma_A} P_\sigma - P_{\text{DM}}$$

We say that an algorithm is a no-regret algorithm, if, by playing according to the algorithm, the decision-maker can achieve regret $O(p(n)T^{1-\delta})$, where $p(n)$ is a polynomial in n and $\delta \in (0, 1/2]$. Furthermore, we say that such an algorithm is *computationally efficient*, if at each time step, t, given the set, S^t, of available actions (and possibly using history), it selects an action, $a^t \in S^t$, in time polynomial in n.

2.2 Agnostic Learning

In this section, we define online and batch agnostic learning. Let X be an instance space and n be a parameter that captures the representation size of X (*e.g.* $X = \{0,1\}^n$ or $X = \mathbb{R}^n$).

Online Agnostic Learning

The definition of online agnostic learning used here is slightly different from those previously used in the literature (see Ben-David et al. [2009]), but is essentially equivalent. Our definition simplifies the presentation of our results.

An online agnostic learning algorithm observes examples one at a time; at time step, t, it sees example, $\mathbf{x}^t$, makes a prediction, $\hat{y}^t \in \{0,1\}$ (possibly using history), and then observes y^t. Let $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$ be a sequence of length T, where $\mathbf{x}^t \in X$ and $y^t \in \{0,1\}$. We consider the oblivious adversarial setting, where the sequence, s, may be fixed by an adversary, but is fixed ahead of time, *i.e.* without observing the past predictions made by the online learning algorithm. We define error of an online agnostic learning algorithm, $\mathcal{A}$, with respect to a sequence, $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$, as

$$\text{err}_s(\mathcal{A}) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{I}(\hat{y}^t \neq y^t),$$

where $\mathbb{I}$ is the indicator function. For any boolean function, $f : X \to \{0,1\}$, we can define error of f with respect to the sequence, $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$, as

$$\mathrm{err}_s(f) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{I}(f(\mathbf{x}^t) \neq y^t).$$

For a concept class, C, of boolean functions over X, online agnostic learnability of C is defined as[3]:

Definition 2 (Online Agnostic Learning)**.** *We say that a concept class, C, over X is online agnostically learnable if there exists an online agnostic learning algorithm, $\mathcal{A}$, that for all T, for all example sequences, $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$, makes predictions, $\hat{y}^1, \ldots, \hat{y}^T$, such that*

$$\mathrm{err}_s(\mathcal{A}) \leq \min_{f \in C} \mathrm{err}_s(f) + O(p(n)/T^{\zeta}),$$

for some polynomial, $p(n)$, and $\zeta \in (0, 1/2]$. Furthermore, the running time of $\mathcal{A}$ at each time step must be polynomial in n. We say that $\mathcal{A}$ has an average regret bound *$O(p(n)/T^{\zeta})$.*

Batch Agnostic Learning

We also give a definition of (batch) agnostic learning (see Haussler [1992], Kearns et al. [1994]). For a distribution, D, over $X \times \{0,1\}$ and any boolean function, $f : X \to \{0,1\}$, define

$$\mathrm{err}_D(f) = \Pr_{(x,y)\sim D}[f(x) \neq y].$$

Definition 3 ((Batch) Agnostic Learning Kearns et al. [1994])**.** *We say that a concept class, C, is (batch) agnostically learnable, if there exists an efficient algorithm that for every $\epsilon, \delta > 0$ and for every distribution, D, over $X \times \{0,1\}$, with access to random examples from D, with probability at least $1 - \delta$ outputs a hypothesis, h, such that*

$$\mathrm{err}_D(h) \leq \min_{f \in C} \mathrm{err}_D(f) + \epsilon.$$

The running time of the algorithm is polynomial in $n, 1/\epsilon, 1/\delta$ and h is polynomially evaluable. The sample complexity *of the algorithm is the number of examples used.*

Online to Batch Conversion

In most learning settings, it is well-known that batch learning is no harder than online learning. Theorem 4 follows more or less directly from the work of Littlestone [1989] and Cesa-Bianchi et al. [2004], but we provide a proof in Appendix A for completeness. Roughly speaking after an online to batch conversion, the sample complexity of the resulting batch algorithm is the number of time steps required to make the average regret $O(\epsilon)$.

Theorem 4. *If a concept class C is online agnostically learnable with regret bound $O(p(n)/T^{\zeta})$ then it is (batch) agnostically learnable. Furthermore the sample complexity for (batch) agnostic learning is $O((p(n)/\epsilon)^{1/\zeta} + 1/\varepsilon^4 + \log^2(1/\delta) + (1/(\zeta\varepsilon^2))\log(n/(\varepsilon\delta)))$.*

[3]The definition assumes that the online algorithm is deterministic; one may instead also allow a randomized algorithm that achieves low regret with high probability over its random choices. But, the guarantee must hold with respect to all sequences.

3 Sleeping Experts Problem

In this section, we show that the sleeping experts problem is at least as hard as online agnostic learning of disjunctions. Theorem 4 then implies that the sleeping expert problem is at least as hard as batch agnostic learning disjunctions. The best known algorithm for (batch) agnostic learning disjunctions is due to Kalai et al. [2005] and has time and sample complexity $2^{\tilde{O}(\sqrt{n})}$. Klivans and Sherstov [2007] show that this bound is essentially tight for the class of algorithms that approximate disjunctions using linear combinations of real-valued functions. Also, it is known that agnostic learning of disjunctions implies PAC learning of DNF expressions (see Kearns et al. [1994], Kalai et al. [2009])[4], thus proving Theorem 1. The problem of PAC learning DNF expressions, proposed by Valiant [1984], has remained an important open problem in computational learning theory for over 25 years.

Recall that in this paper, for the sleeping experts setting, the action availability and payoff functions are set by an oblivious adversary. First, we define the notation used in this section. Let $X = \{0,1\}^n$ and let DISJ denote the class of disjunctions over X. Let $\mathbf{x} = x_1 \cdots x_n \in X$; for each bit x_i we define two actions O_i (corresponding to $x_i = 1$) and Z_i (corresponding to $x_i = 0$). We define an additional action $\perp$. Thus, the set of actions is $A = \{\perp, O_1, Z_1, \ldots, O_i, Z_i, \ldots, O_n, Z_n\}$.

Suppose there exists an algorithm, Alg, for the sleeping experts problem that achieves regret $O(p(n)T^{1-\delta})$ for some polynomial, $p(n)$, and $\delta \in (0, 1/2]$. We use Alg to construct an online learning algorithm, DISJ-Learn (see Fig. 1), for online agnostic learning DISJ that has average regret bound $O(p(n)/T^{\delta})$. The instance, $\mathbf{x}^t$, received by the online disjunction learning algorithm, is used to define the set of available actions at round t for the sleeping experts problem. And the label, y^t, is used to define the payoffs.

Proposition 5. *Suppose there exists an efficient algorithm for the sleeping experts problem with regret $O(p(n)T^{1-\delta})$, then there exists an efficient online agnostic algorithm for learning disjunctions with average regret bound $O(p(n)/T^{\delta})$.*

In order to prove Proposition 5, we need to use Lemma 6.

Lemma 6. *Let $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$ be any sequence of examples from $X \times \{0,1\}$. Let $A = \{\perp, O_1, Z_1, \ldots, O_n, Z_n\}$ and Σ_A be the set of rankings over A. Let S^t and p^t be as defined in Fig. 1. Then*

$$\min_{f \in \mathsf{DISJ}} \mathrm{err}_s(f) + \frac{1}{T} \max_{\sigma \in \Sigma_A} P_\sigma = 1,$$

where P_σ is the payoff achieved by playing the sleeping experts problem according to ranking strategy σ.

Proof. Let σ be a ranking over the set of actions, $A = \{\perp, O_1, Z_1, \ldots, O_n, Z_n\}$. For any two actions, $a_1, a_2 \in A$, define $a_1 \prec_\sigma a_2$ to mean that a_1 is ranked higher than a_2 according to the ranking σ. For a ranking, σ, define a disjunction, f_σ, as

$$f_\sigma = \bigvee_{i: O_i \prec_\sigma \perp} x_i \vee \bigvee_{i: Z_i \prec_\sigma \perp} \bar{x}_i.$$

If, for some i, both $O_i \prec_\sigma \perp$ and $Z_i \prec_\sigma \perp$, then $f_\sigma \equiv 1$. Note that several permutations may map to the same disjunction, since only which O_i and Z_i are ranked above $\perp$ is important, not their

[4] Actually, agnostically learning conjunctions implies PAC learning DNF, but because of the duality between conjunctions and disjunctions, an agnostic learning algorithm for learning disjunctions also implies an algorithm for learning conjunctions.

ranking relative to each other. We will show that

$$\mathrm{err}_s(f_\sigma) + \frac{1}{T} P_\sigma = 1. \tag{1}$$

Consider some vector, $\mathbf{x}^t \in \{0,1\}^n$, and let $S^t \subseteq A$ be the corresponding subset of available actions (see Fig. 1). Then, observe that $f_\sigma(\mathbf{x}^t) = 0$ if and only if $\sigma(S^t) = \perp$. If the true label $y^t = 1$, f_σ suffers error $1 - f_\sigma(\mathbf{x}^t)$ and $\sigma(S^t)$ receives payoff $f_\sigma(\mathbf{x}^t)$. If the true label $y^t = 0$, then f_σ suffers error $f_\sigma(\mathbf{x}^t)$ and $\sigma(S^t)$ receives payoff $1 - f_\sigma(\mathbf{x}^t)$. Summing over $(\mathbf{x}^t, y^t)$ in the sequence s, we get (1). But, this also completes the proof of the lemma, since for every disjunction, g, there exists a ranking, π, such that $g = f_\pi$, namely the ranking where the actions corresponding to literals occurring in g (O_i or Z_i depending on whether x_i or $\bar{x}_i$ appears in g) are placed first, followed by $\perp$, followed by the rest of the actions. □

Proof of Proposition 5. Let $s = \langle (\mathbf{x}^t, y^t) \rangle_{t=1}^T$ be any sequence of examples from $X \times \{0,1\}$ for the problem of online agnostic learning DISJ. Suppose there is an efficient algorithm, Alg, for the sleeping experts problem with regret $O(p(n)T^{1-\delta})$. Then, we claim that Algorithm DISJ-Learn (Fig. 1) has average regret $O(p(n)/T^\delta)$.

Let the total set of actions be $A = \{\perp, O_1, Z_1, \ldots, O_n, Z_n\}$ and Σ_A be the set of rankings over A. Let the payoff functions, p^t, and the set of available actions, S^t, be as defined in Fig. 1. Let σ^* be the best ranking in hindsight, *i.e.* $\sigma^* = \mathrm{argmax}_{\sigma \in \Sigma_A} \sum_{t=1}^T P_\sigma$. Also, let f^* be the best disjunction with respect to the sequence, s, *i.e.* $f^* = \mathrm{argmin}_{f \in \mathsf{DISJ}} \mathrm{err}_s(f)$.

Note that $\hat{y}^t$ is the prediction made by DISJ-Learn using the action selected by Alg. At any given round, the payoff received by Alg is $1 - \mathbb{I}(\hat{y}^t \neq y^t)$ (if Alg picks $\perp$, then the payoff is $1 - y^t$ and $\hat{y}^t = 0$; otherwise, the payoff is y^t and $\hat{y}^t = 1$). Hence, summing over all rounds,

$$\frac{1}{T} P_{\mathsf{Alg}} = 1 - \mathrm{err}_s(\text{DISJ-Learn})$$

Now, the proof follows immediately from Lemma 6, since $1 = \min_{f \in \mathsf{DISJ}} \mathrm{err}_s(f) + (1/T) \max_{\sigma \in \Sigma_A} P_\sigma$, and hence from the above equation we get,

$$\begin{aligned} \mathrm{err}_s(\text{DISJ-Learn}) - \min_{f \in \mathsf{DISJ}} \mathrm{err}_s(f) &= \frac{1}{T} \left(\max_{\sigma \in \Sigma_A} P_\sigma - P_{\mathsf{Alg}} \right) \\ &= O(p(n)/T^\delta). \end{aligned}$$

□

3.1 Restricted Number of Actions

We consider a restricted version of the sleeping experts problem, where the number of available actions at each round is at most k, and we refer to this as the *k-sleeping experts problem*. For constant k, we note that it is easy to achieve regret $O(\sqrt{n^k T})$ by an algorithm that uses $O(n^k)$ time in each round. The algorithm is the following: For each possible set of available actions, we run an independent experts algorithm, such as Hedge. For each subset, $S \subseteq [n]$, with $|S| \leq k$, let T_S be the number of time steps for which S was the set of available actions. Then the regret on those rounds obtained by running Hedge is $O(\sqrt{\log(|S|) T_S})$. Since, $\sum_{S \subseteq [n], |S| \leq k} T_S = T$ (the total number of rounds), the total regret is,

$$\sum_{S \subseteq [n], |S| \leq k} O(\sqrt{\log(|S|) T_S}) = O(\sqrt{n^k \log(k) T}),$$

since the number of subsets of size at most k is $O(n^k)$.

Algorithm. DISJ-Learn: Online Agnostic Learning Disjunctions

Input: Alg - the algorithm for sleeping experts problem.

For $t = 1, \ldots, T$,

1. Receive example $\mathbf{x}^t$. Define $S^t = \{\perp\} \cup \{O_i \mid x_i^t = 1\} \cup \{Z_i \mid x_i^t = 0\}$.
2. Give S^t as the set of available actions to Alg. Let Alg choose a^t.
3. If $a^t = \perp$, then set $\hat{y}^t = 0$, else set $\hat{y}^t = 1$.
4. Observe y^t. Define $p^t(\perp) = 1 - y^t$ and $p^t(a) = y^t$ for all other actions $a \in S^t \setminus \{\perp\}$. Return p^t as the payoff function to Alg.

Figure 1: Algorithm for online agnostically learning DISJ.

Gambling Problem

In the special case when $k = 2$, the corresponding sleeping experts problem is the so called *gambling* problem [Abernethy, 2010]. This is because one may equivalently view the problem as predicting a winner of a match between 2 players. The regret is compared with the best ranking of the players in hindsight. Recently, Hazan et al. [2012] gave an efficient no-regret algorithm that achieves regret $O(\sqrt{n \log^3(n) T})$ for this problem, which improves upon the trivial guarantee of $O(\sqrt{n^2 T})$.

3.1.1 Reduction from agnostic learning of monotone disjunctions

For a vector, $\mathbf{x} \in \{0,1\}^n$, let $\|\mathbf{x}\|_H$ denote the Hamming weight of $\mathbf{x}$, *i.e.* the number of '1' bits in $\mathbf{x}$. Let $X_k = \{\mathbf{x} \in \{0,1\}^n \mid \|\mathbf{x}\|_H \leq k\}$ be the subset of the boolean cube containing all points having Hamming weight at most k. Let MON-DISJ be the concept class of monotone disjunctions over $\{0,1\}^n$, *i.e.* those disjunctions that do not have any negated variables. We consider agnostic learning MON-DISJ when the input is restricted to X_k, *i.e.* each example has Hamming weight at most k. Our reduction in Proposition 5 can be easily modified to show that the $(k+1)$-sleeping experts problem is at least as hard as online agnostic learning MON-DISJ over X_k.

Proposition 7. *Suppose that there exists an efficient algorithm for the $(k+1)$-sleeping experts problem with regret bound R, then there exists an efficient algorithm for online agnostic learning MON-DISJ over X_k with average regret bound R/T.*

The proof of Proposition 7 follows from Lemma 8 and is very similar to the proof of Proposition 5.

Lemma 8. *Let $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$ be the sequence of examples with $(\mathbf{x}^t, y^t) \in X_k \times \{0,1\}$, let $A = \{1, \ldots, n\} \cup \{\perp\}$ and S^t and p^t be the set of available actions and payoffs as defined in Figure 2. Then,*

$$\min_{f \in \text{MON-DISJ}} \text{err}_s(f) + \frac{1}{T} \max_{\sigma \in \Sigma_A} P_\sigma = 1$$

where Σ_A is the set of rankings over the set A and P_σ is the payoff received by playing according to σ on each round.

Proof. Let σ be a ranking over the set of actions, $A = \{\perp, 1, 2, \ldots, n\}$. As in the proof of Lemma 6, for actions, $a_1, a_2 \in A$, define $a_1 \prec_\sigma a_2$ to mean that a_1 is ranked higher than a_2 according to the

Algorithm: MON-DISJ-Learn: Online Agnostic Learning Monotone Disjunctions over X_k

Input: Alg - the algorithm for the $(k+1)$-sleeping experts problem

For $t = 1, \ldots, T$,

1. Receive example $\mathbf{x}^t$. Define $S^t = \{\perp\} \cup \{i \mid x_i^t = 1\}$.
2. Give S^t as the set of available actions to Alg. Let Alg choose a^t.
3. If $a^t = \perp$, then set $\hat{y}^t = 0$, else set $\hat{y}^t = 1$.
4. Observe y^t. Define $p^t(\perp) = 1 - y^t$ and $p^t(a) = y^t$ for all other actions $a \in S^t \setminus \{\perp\}$. Return p^t as the payoff vector to Alg.

Figure 2: Algorithm for online agnostic learning of MON-DISJ over X_k

ranking σ. For a ranking, σ, define a disjunction, f_σ, as

$$f_\sigma = \bigvee_{i: i \prec_\sigma \perp} x_i.$$

Note that, in this case, f_σ is a monotone disjunction. As in the proof of Lemma 6, it only matters which actions among $\{1, \ldots, n\}$ are ranked above $\perp$, not the relative order among them. We show that,

$$\mathrm{err}_s(f_\sigma) + \frac{1}{T} P_\sigma = 1 \tag{2}$$

Consider some vector, $\mathbf{x}^t \in X_k$, and let $S^t \subseteq A$ be the corresponding set of available actions (see Fig. 2). Observe that $|S^t| = \|\mathbf{x}^t\|_H + 1$, and hence $|S^t| \leq k + 1$. Then, note that $f_\sigma(\mathbf{x}^t) = 0$ if and only if $\sigma(S^t) = \perp$. If the true label is $y^t = 1$, f_σ suffers error $1 - f_\sigma(\mathbf{x}^t)$ and $\sigma(S^t)$ receives payoff $f_\sigma(\mathbf{x}^t)$. If the true label is $y^t = 0$, then f_σ suffers error $f_\sigma(\mathbf{x}^t)$ and $\sigma(S^t)$ receives payoff $1 - f_\sigma(\mathbf{x}^t)$. Summing over $(\mathbf{x}^t, y^t)$ in the sequence s, we get (2). But, this also completes the proof of the Lemma, since for every disjunction, g, there exists a ranking, π, such that $g = f_\pi$, namely the ranking where the actions corresponding to literals occurring in g are placed first, followed by $\perp$, followed by the rest of the actions. □

Theorem 4 now implies that an efficient algorithm for the $(k+1)$-sleeping experts problem gives an efficient algorithm for batch agnostic learning MON-DISJ, when the input is restricted to X_k. In particular, let D be a distribution over $X_k \times \{0, 1\}$, then an agnostic learner for MON-DISJ, is required to output a hypothesis, h, such that

$$\mathrm{err}_D(h) = \Pr_{(x,y)\sim D}[h(x) \neq y] \leq \min_{f \in \text{MON-DISJ}} \mathrm{err}_D(f) + \epsilon.$$

In *proper* agnostic learning, it is required that the output hypothesis, h, be in MON-DISJ, *i.e.* h is itself a monotone disjunction. For any $k \geq 2$, proper agnostic learning MON-DISJ over X_k is NP-hard, even for constant accuracy parameter ϵ. This follows via a simple reduction from vertex cover. The proof of Proposition 9 is provided in Appendix B. (The proof is based on a similar result by Kearns et al. [1994]).

Proposition 9. *There exists a constant $\epsilon_0 > 0$, such that there is no efficient algorithm for* proper *agnostic learning MON-DISJ over instance space X_k to accuracy ϵ_0, unless* NP = RP.

3.1.2 Improper Agnostic Learning MON-DISJ

While proper learning is NP-hard, it is trivial to improperly learn MON-DISJ over X_k, for any constant k. The trivial algorithm works as follows: For any distribution D over $X_k \times \{0,1\}$, the algorithm draws $\Theta(n^k)$ examples from D. The output hypothesis simply memorizes all the examples and outputs the majority label on any example. (If some example was not seen in the random draw, the label is predicted randomly.) It is easy to show that such a hypothesis is an agnostic learner for *any* class of functions, not just MON-DISJ. In fact, that this is the learner that would be produced as a result of the reduction from the trivial algorithm for the $(k+1)$-sleeping experts problem.[5] However, observe that the sample complexity of the trivial efficient algorithm is $\Theta(n^k)$, whereas the sample complexity of (inefficient) proper learning is only $O(n)$ (for constant ϵ). This follows the from the fact that the VC-dimension of the class of monotone disjunctions is n.

The question of interest is whether there is an efficient improper learning algorithm with sample complexity better than $O(n^k)$. In Appendix C, we show that when $k = 2$, the techniques of Hazan et al. [2012] can be used to achieve near optimal sample complexity: We achieve average regret $O(\sqrt{n\log(n)/T})$ for this problem using their matrix prediction algorithm. This approach does not readily generalize to $k \geq 3$, which remains open.

Acknowledgments

We would like to thank Elad Hazan, Adam Kalai, Satyen Kale, Tal Moran, Shai Shalev-Shwartz, Justin Thaler, Jonathan Ullman, Salil Vadhan, and Leslie Valiant for helpful discussions.

References

J. Abernethy. Can we learn to gamble efficiently? (open problem). In *COLT*, 2010.

S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45:501–555, May 1998. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/278298.278306. URL `http://doi.acm.org/10.1145/278298.278306`.

S. Ben-David, D. Pál, and S. Shalev-Shwartz. Agnostic online learning. In *COLT*, 2009.

A. Beygelzimer, J. Langford, L. Li, L. Reyzin, and R. E. Schapire. Contextual bandit algorithms with supervised learning guarantees. In *Artificial Intelligence and Statistics (AISTATS)*, 2011.

A. Blum and Y. Mansour. From external to internal regret. *Journal of Machine Learning Research*, 8:1307–1324, 2007.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, 2004.

M. Dudik, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang. Efficient optimal learning for contextual bandits. In *Uncertainty in Artificial Intelliegence (UAI)*, 2011.

[5] Observe that for the $(k+1)$-sleeping experts problem in our reduction, regret $\Theta(\sqrt{n^k T})$ (instead of $\Theta(\sqrt{n^{k+1}T})$) can be achieved since $\perp$ is always available. Thus, the number of different subsets that may be seen is only $O(n^k)$ and not $O(n^{k+1})$.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the second European conference on computational learning theory*, 1995.

Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, 1997.

M. R. Garey and D. S. Johnson. *Computers and Intractability*. W.H. Freeman and Co., New York, NY, USA, 1979. ISBN 0-7167-1044-7.

J. Håstad. Some optimal inapproximability results. *J. ACM*, 48:798–859, July 2001. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/502090.502098. URL `http://doi.acm.org/10.1145/502090.502098`.

D. Haussler. Decision theoretic generalizations of the pac model for neural net and other learning applications. *Information and Computation*, 100:78–150, 1992.

E. Hazan, S. Kale, and S. Shalev-Shwartz. Near-optimal algorithms for online matrix prediction. In *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23, pages 38.1–38.13, Edinburgh, Scotland, 2012. JMLR.

A. T. Kalai, A. R. Klivans, Y. Mansour, and R. A. Servedio. Agnostically learning halfspaces. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, 2005.

A. T. Kalai, V. Kanade, and Y. Mansour. Reliable agnostic learning. In *COLT*, 2009.

V. Kanade, B. McMahan, and B. Bryan. Sleeping experts and bandits with stochastic action availability and adversarial rewards. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pages 272–279, 2009.

M. J. Kearns, R. E. Schapire, and L. M. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994.

R. Kleinberg, A. Niculescu-Mizil, and Y. Sharma. Regret bounds for sleeping experts and bandits. *Machine learning*, pages 1–28, 2008.

A. R. Klivans and R. A. Servedio. Learning dnf in time. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, STOC '01, pages 258–265, New York, NY, USA, 2001. ACM. ISBN 1-58113-349-9. doi: http://doi.acm.org/10.1145/380752.380809. URL `http://doi.acm.org/10.1145/380752.380809`.

A. R. Klivans and A. Sherstov. A lower bound for agnostically learning disjunctions. In *Proceedings of the 22nd Annual Conference on Learning Theory*, 2007.

J. Langford and T. Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. In *Neural Information Processing Systems (NIPS)*, 2007.

N. Littlestone. From on-line to batch learning. In *Proceedings of the second annual workshop on computational learning theory*, 1989.

S. Shalev-Shwartz, O. Shamir, and K. Sridharan. Learning kernel-based halfspaces with the zero-one loss. In *Proceedings of the 23rd Annual Conference on Learning Theory*, 2010.

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

A On-line to Batch Learning

We prove Theorem 4 using the following lemma.

Lemma 10. *Let $\mathcal{A}$ be an online agnostic learning algorithm for a concept class $\mathcal{C}$ over X with regret bound $O(p(n)/T^{\zeta})$. We run $\mathcal{A}$ for T steps on $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$. At each step $\mathcal{A}$ can be interpreted as a hypothesis H_s^t which computes $\hat{y}^t = H_s^t(\mathbf{x}^t)$.*

Then we can choose $T = O\left(p(n)/\varepsilon\right)^{1/\zeta} + O\left(1/\varepsilon^4 + \log^2(1/\delta)\right)$ such that the following holds.

Let D be a distribution over $X \times \{0,1\}$. Take a sequence $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$ of T examples from D. Let $\langle H^t\rangle_{t=1}^T$ be the hypotheses produced by $\mathcal{A}$ running on s. Then, with probability $1-\delta$ over the choice of s, there exists t^* such that $\operatorname{err}_D(H_s^{t^*}) \leq \min_{f\in C} \operatorname{err}_D(f) + \varepsilon$.

Lemma 10 allows us to convert an online agnostic learning algorithm into a hypothesis, which we can use for (batch) agnostic learning.

Proof. Let $Q_s^t = \sum_{t'\leq t} \operatorname{err}(H_s^{t'})$. Then, clearly, $\langle Q_s^t\rangle_{t=1}^T$ is a submartingale. Moreover, induction on T gives

$$\begin{aligned}\mathbb{E}_s[\operatorname{err}_s(\mathcal{A})] &= \mathbb{E}_s\left[\frac{1}{T}\sum_{t=1}^{T} \mathbb{I}(H_s^t(\mathbf{x}^t) \neq y^t)\right] \\ &= \mathbb{E}_s\left[\frac{1}{T}\sum_{t=1}^{T} \operatorname{err}_D(H_s^t)\right] = \mathbb{E}_s\left[\frac{Q_s^T}{T}\right]. \end{aligned} \tag{3}$$

We will now use standard bounds to show that (i) the expectation (3) is close to (or better than) the optimal error and that (ii) Q_s^T is close to its expectation with high probability. It follows that at least one hypothesis $H_s^{t^*}$ must have error close to (or better than) an optimal concept.

(i) We have

$$\begin{aligned}\mathbb{E}_s[\operatorname{err}_s(\mathcal{A})] &\leq \mathbb{E}_s[\min_{f\in\mathcal{C}} \operatorname{err}_s(f)] + O(p(n)/T^{\zeta}) \\ &\leq \min_{f\in\mathcal{C}} \operatorname{err}_D(f) + O(p(n)/T^{\zeta}). \end{aligned} \tag{4}$$

The first inequality follows from $\mathcal{A}$ being an online agnostic learning algorithm. The second inequality follows from the fact that $\mathbb{E}_s[\min_{f\in\mathcal{C}} \operatorname{err}_s(f)] \leq \min_{f\in\mathcal{C}} \mathbb{E}_s[\operatorname{err}_s(f)]$.

(ii) Noting that $Q_s^t \leq Q_s^{t+1} \leq Q_s^t + 1$, Azuma's inequality gives

$$\Pr_s\left[Q_s^T \geq \mathbb{E}[Q_s^T] + T^{1-\alpha}\right] \leq \exp\left(-T^{1-2\alpha}/2\right). \tag{5}$$

Combining (3), (4), and (5), we have

$$\begin{aligned}\Pr_s\left[\frac{1}{T}\sum_{t=1}^{T} \operatorname{err}_D(H_s^t) \geq \min_{f\in\mathcal{C}} \operatorname{err}_D(f) + T^{-\alpha} + O(p(n)/T^{\zeta})\right] \\ \leq \exp\left(-T^{1-2\alpha}/2\right).\end{aligned}$$

So we can choose $\alpha = 1/4$ and

$$T = \max\left\{(2/\varepsilon)^4, O(2p(n)/\varepsilon)^{1/\zeta}, (2\log(1/\delta))^2\right\}$$

to ensure that, with probability $1-\delta$,

$$\min_{t=1}^{T} \mathrm{err}_D(H_s^t) \le \frac{1}{T}\sum_{t=1}^{T} \mathrm{err}_D(H_s^t) \le \min_{f\in\mathcal{C}} \mathrm{err}_D(f) + \varepsilon.$$

□

Proof of Theorem 4. Let $\mathcal{A}$ be an online agnostic learning algorithm for a concept class $\mathcal{C}$ over X with regret bound $O(p(n)/T^{\zeta})$. Fix $\varepsilon, \delta > 0$ and a distribution D over $X \times \{0,1\}$. Choose $T = O\left(p(n)/\varepsilon\right)^{1/\zeta} + O\left(1/\varepsilon^4 + \log^2(1/\delta)\right)$ as in Lemma 10. We sample $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^{T}$ from D and run $\mathcal{A}$ on s. Now we have a sequence of hypotheses $\langle H^t\rangle_{t=1}^{T}$. With probability $1-\delta/2$ over the choice of s, at least one hypothesis $H_s^{t^*}$ satisfies $\mathrm{err}_D(H^{t^*}) \le \min_{f\in\mathcal{C}} \mathrm{err}_D(f) + \varepsilon/2$. All that remains is to identify one such hypothesis.

Take T' samples $s' = \langle(\mathbf{x}^{t'}, y^{t'})\rangle_{t'=1}^{T'}$ from D. By the Chernoff bound, for any $f : X \to \{0,1\}$,

$$\Pr_{s'}\left[|\mathrm{err}_{s'}(f) - \mathrm{err}_D(f)| \ge \varepsilon/2\right] \le 2e^{-T'\varepsilon^2/16}.$$

Let $T' = (16/\varepsilon^2)\log(4T/\delta)$. Then

$$\Pr_{s'}\left[\forall t \;\; |\mathrm{err}_{s'}(H_s^t) - \mathrm{err}_D(H_s^t)| < \varepsilon/2\right] \ge 1-\delta/2.$$

So we can estimate the accuracy of each hypothesis and identify a good one. Thus we take $T+T'$ samples and, with probability $1-\delta$, we can find a good hypothesis. □

B Hardness of Agnostic Learning MON-DISJ

In order to prove Proposition 9, we require the following result, which states that finding a constant factor approximation to vertex cover is NP-hard (for some constant), even when the number of edges is linear in the number of vertexes and the size of the minimum vertex cover is linear in the size of the graph.

Proposition 11. *Given a graph $G = (V, E)$, let C_G denote a minimum vertex cover of G. There exist universal constants k_1, k_2, ϵ_1, such that if $\mathcal{G} = \{G = (V, E) \mid |E| \le k_1|V|, |V| \le k_2|C_G|\}$, given $G \in \mathcal{G}$, it is NP-hard to find a vertex cover $C' \subseteq V$ of G, such that $|C'| \le (1+\epsilon_1)|C_G|$.*

We prove Proposition 11 later. We now give a proof of Proposition 9.

Proof of Proposition 9. Let $G = (V, E)$ be an arbitrary graph from the family $\mathcal{G}$ defined in Proposition 11. For each $v \in V$, let $\mathbf{x}_v = (0, \ldots, 1, \ldots, 0)$, where only the v^{th} position of $\mathbf{x}_v$ is 1 and the remaining positions are 0. For every edge, $e = (i, j) \in E$, let $\mathbf{x}_e = (0, \ldots, 1, \ldots, 1, \ldots, 0)$, where only the i^{th} and j^{th} position of $\mathbf{x}_e$ are 1 and the rest are 0. Let D_G be a distribution over $X_k \times \{0,1\}$ that is uniform over the set $\{(\mathbf{x}_v, 0) \mid v \in V\} \cup \{(\mathbf{x}_e, 1) \mid e \in E\}$, *i.e.* each point has weight $1/(|V| + |E|)$.

Let $C \subseteq V$ be a vertex cover of G. Then, let $f_C = \bigvee_{i\in C} x_i$. Observe that $\mathrm{err}_{D_G}(f_C) = |C|/(|V| + |E|)$. (For any edge $e = (i, j)$, $f_C(\mathbf{x}_e) = 1$ since either i or j is in C, and for any $v \notin C$, $f_C(\mathbf{x}_v) = 0$.)

Suppose $g = \bigvee_{i\in K} x_i$ is some monotone disjunction. We show that given g, we can easily construct a vertex cover of graph, G, of size $\mathrm{err}_{D_G}(g)(|V| + |E|)$. We construct the vertex cover as follows: For all $i \in K$, where $g = \bigvee_{i\in K} x_i$, put i in the vertex cover. Note that $g(\mathbf{x}_i) = 1$ if $i \in K$

and 0 otherwise. Thus, the total contribution to the error of g from the points corresponding to the vertices's is $|K|/(|V|+|E|)$. For every edge e, such that $g(\mathbf{x}_e)=0$, add a vertex corresponding to one of the end-points of e to the vertex cover under construction. Observe, that any edge for which $g(\mathbf{x}_e)=1$, was already covered by some vertex in K. Thus, for every error of g we have added at most 1 vertex to the cover. Thus, the size of the vertex cover is at most $\mathrm{err}_{D_G}(g)(|V|+|E|)$.

Suppose there is an algorithm for proper agnostic learning MON-DISJ. Then, let g be the monotone disjunction returned when run on distribution D_G. Let C' be the vertex cover constructed using g. Let C be a minimum vertex cover of G and let f_C be the monotone disjunction corresponding to C. We showed that $\mathrm{err}_{D_G}(f_C)=|C|/(|V|+|E|)$. The agnostic learning guarantee is that $\mathrm{err}_{D_G}(g)\leq \mathrm{err}_{D_G}(f_C)+\epsilon_0$. Thus, $|C'|/(|V|+|E|)\leq \mathrm{err}_{D_G}(g)\leq |C|/(|V|+|E|)+\epsilon_0$. But, since $|C|\geq |V|/k_2)$ and $|E|\leq k_1|V|)$, we have $|C'|\leq |C|(1+\epsilon_1)$, for some suitable choice of constant $\epsilon_0>0$. However, this is contradicts Proposition 11. Therefore, there must be some constant ϵ_0 such that proper learning MON-DISJ to accuracy ϵ_0 is not possible unless RP = NP. □

We use the following version of the PCP theorem as the basis of our proof of Proposition 11.

Theorem 12 (Arora et al. [1998], Håstad [2001] Theorem 2.24)**.** *Let L be a language in* NP *and x be a string. There is a universal constant $c<1$ such that, we can in time polynomial in $|x|$ construct a 3CNF formula $\phi_{x,L}$ such that if $x\in L$ then $\phi_{x,L}$ is satisfiable while if $x\notin L$, $\phi_{x,L}$ is at most c-satisfiable. Furthermore, each variable appears exactly 5 times.*

We show that the reduction of *sparse* 3SAT to vertex cover produces sparse graphs with linear-sized minimal covers.

Lemma 13. *Let ϕ be a 3CNF formula with n variables and m clauses, in which each variable appears at most $d\geq 2$ times. Then there exists a simple undirected graph G_ϕ with $2n+3m$ vertexes and $n+6m$ edges such that*

(i) if ϕ has an assignment of variables that satisfies $m-k$ clauses, then G_ϕ has a vertex cover of size $n+2m+k$, and

(ii) if G_ϕ has a vertex cover of size $n+2m+k$, then ϕ has an assignment that satisfies at least $m-k\lfloor d/2\rfloor$ clauses.

Moreover, the graph G_ϕ and the correspondence between assignments and vertex covers can be computed in uniform polynomial time in n and m.

Proof. This reduction comes from Garey and Johnson [1979] Theorem 3.3. Let ϕ be a 3CNF formula. For each variable x_i create two vertexes v_{x_i} and $v_{\overline{x_i}}$ and an edge $(v_{x_i}, v_{\overline{x_i}})$ between them. For each clause $c_i=x^*_{i_1}\vee x^*_{i_2}\vee x^*_{i_3}$ (where $x^*_{i_k}$ is either x_{i_k} or $\overline{x_{i_k}}$ for $1\leq k\leq 3$), create three vertexes $v_{c_i,1}$, $v_{c_i,2}$, and $v_{c_i,3}$ and make them a triangle with edges $(v_{c_i,1},v_{c_i,2})$, $(v_{c_i,2},v_{c_i,3})$, and $(v_{c_i,3},v_{c_i,1})$. Then, for $1\leq k\leq 3$, connect $v_{c_i,k}$ to $v_{x^*_{i_k}}$. Figure 3 gives an example of this reduction.

(i) Let x be an assignment to the variables of ϕ that satisfies $m-k$ clauses. The corresponding vertex cover is as follows. Firstly, for each variable x_i, if x_i is true, include v_{x_i} in the vertex cover; otherwise include $v_{\overline{x_i}}$. For each unsatisfied clause c_i include $v_{c_i,1}$, $v_{c_i,2}$, and $v_{c_i,3}$. For each satisfied clause $c_i=x^*_{i_1}\vee x^*_{i_2}\vee x^*_{i_3}$, choose $1\leq k\leq 3$ such that $x^*_{i_k}$ is true and include $v_{c_i,k'}$ for $k'\neq k$. Clearly this covers all edges and has size $n+2m+k$.

(ii) Let S be a vertex cover of G_ϕ of size $n+2m+k$. For each variable x_i either $v_{x_i}\in S$ or $v_{\overline{x_i}}\in S$, as the edge $(v_{x_i},v_{\overline{x_i}})$ must be covered. Likewise, for each clause c_i, two of $v_{c_i,1}$, $v_{c_i,2}$, and $v_{c_i,3}$ must be in S, as the triangle $\{(v_{c_i,1},v_{c_i,2}),(v_{c_i,2},v_{c_i,3}),(v_{c_i,3},v_{c_i,1})\}$ must be covered. This accounts for $n+2m$ elements of S. So we identify k "extra" elements.

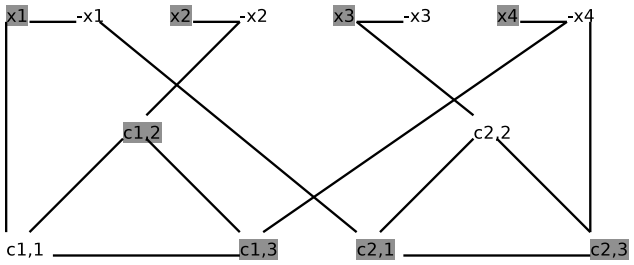

Figure 3: Vertex cover instance corresponding to 3CNF $(x_1 \vee \overline{x_2} \vee \overline{x_4}) \wedge (\overline{x_1} \vee x_3 \vee \overline{x_4})$ with an optimal vertex cover corresponding to $x_1 = x_2 = x_3 = 1$ highlighted.

We can assume that all extra elements are of the form $v_{c_i,k}$: Suppose instead that $v_{x_i}, v_{\overline{x_i}} \in S$. Since x_i and $\overline{x_i}$ appear at most d times, we can choose one—say, x_i^*—that appears at most $\lfloor d/2 \rfloor$ times. We remove $v_{x_i^*}$ from S and insert all $v_{c_i,k}$ where $x_{i_k}^* = x_i$. Clearly all the edges that $v_{x_i^*}$ covered are now covered by the newly inserted vertexes—that is, we moved the vertex cover to the other end of each edge covered by $v_{x_i^*}$. This process grows the number of extra elements by a factor of at most $\lfloor d/2 \rfloor$, so $|S| \leq n + 2m + k\lfloor d/2 \rfloor$.

The assignment is computed as follows. If $v_{x_i} \in S$, then x_i is set to true. Otherwise, if $v_{\overline{x_i}} \in S$, then x_i is set to false. Each extra element of S corresponds to an unsatisfied clause, so at most $k\lfloor d/2 \rfloor$ clauses are unsatisfied. □

Proposition 11 follows immediately from Proposition 14.

Proposition 14. *Vertex cover is inapproximable, even for graphs with a number of edges linear in the number of vertexes and minimal vertex covers linear in the number of vertexes. Formally, there exist universal constants $c_1, c_2 > 0$, and $\varepsilon > 0$ such that the following holds. Consider graphs G with n vertexes and m edges, where $m = c_1 n$. Moreover, either (i) G has a vertex cover of size $c_2 m$ or (ii) no vertex cover of size $(1+\varepsilon)c_2 m$. If there exists a polynomial-time algorithm for deciding which of (i) or (ii) holds, then* $\mathsf{NP} = \mathsf{P}$.

Proof. This follows from Theorem 12 and Lemma 13. Firstly, Theorem 12 shows that to prove that $\mathsf{NP} = \mathsf{P}$ it suffices to distinguish in polynomial time satisfiable 3CNF formulas with $5n = 3m$, where n is the number of variables, m is the number of clauses, and each variable appears at most $d = 5$ times, from ones where at most cm clauses can be satisfied, where $c < 1$ is a universal constant. Secondly, Lemma 13 shows that such 3CNF formulas can be converted into graphs with $n' = 2n + 3m = 7n$ vertexes and $m' = n + 6m = 11n = (11/7)n'$ edges. Satisfiable 3CNF formulas become graphs with vertex covers of size $n + 2m = (13/33)m'$.

Conversely, suppose that S is a vertex cover of size $(1+\varepsilon)(13/33)m'$. By Lemma 13 part (ii), this translates into an assignment satisfying at least $m - \varepsilon(13/33)m'\lfloor 5/2 \rfloor = (1-(26/5)\varepsilon)m$ clauses. So, if $1 - (26/5)\varepsilon > c$, then at least cm clauses are satisfied. □

C Learning Disjunctions on Inputs of Hamming Weight Two

We consider the problem of online learning monotone disjunctions when the input is restricted to X_2, *i.e.* points having Hamming weight at most 2. We show that there is an efficient algorithm for solving this problem with average regret $O(\sqrt{n \log(n)/T})$, where n is the size of the inputs and T is the number of rounds. The optimal average regret is $O(\sqrt{n/T})$ (for the inefficient algorithm)

and the average regret of the trivial (efficient) algorithm (learning the answer to each of the $O(n^2)$ possible inputs separately) is $O(\sqrt{n^2/T})$.

We obtain this result using the online matrix prediction algorithm of Hazan et al. [2012]. The online matrix prediction problem for a set $\mathcal{W} \subseteq [-1,1]^{n \times n}$ has T rounds, which proceed sequentially as follows:

For each round t:

1. The world presents indexes $i^t, j^t \in [n]$.
2. The algorithm answers $\hat{y}^t \in [-1,1]$.
3. The world presents the correct answer $y^t \in [-1,1]$.

The total loss of the algorithm is $L = \sum_{t=1}^{T} |\hat{y}^t - y^t|$. The regret of the algorithm is

$$\text{Regret} := \mathbb{E}\left[L - \min_{W \in \mathcal{W}} \sum_t |W_{i^t,j^t} - y^t|\right].$$

That is, we compare the loss of the algorithm to that of predictions made using a fixed matrix in $\mathcal{W}$.

The result of Hazan et al. [2012] is as follows.

Theorem 15. *Let $\mathcal{W}$ be a set of symmetric matrices in $[-1,1]^{n \times n}$ such that every $W \in \mathcal{W}$ is (β, τ)-*decomposable*, i.e. $W = P - N$, where P and N are symmetric positive semidefinite matrices with trace(P) + trace$(N) \leq \tau$ and, for all i, $P_{i,i} \leq \beta$ and $N_{i,i} \leq \beta$. Then there exists an efficient algorithm for online matrix prediction for $\mathcal{W}$ with*

$$\textit{Regret} \leq O(\sqrt{\tau \beta \log(n) T}).$$

Our reduction represents a monotone disjunction $f(x) = \bigvee_{i \in S} x_i$ as a matrix, W^f, where any input in X_2 corresponds to one entry in the matrix. An input $\mathbf{x}_{i,j} \in X_2$ with 1s in positions i and j corresponds to the entry $W^f_{i,j} = f(\mathbf{x}_{i,j})$ in the matrix. Equivalently, let

$$\mathcal{W} := \left\{W \in \{0,1\}^{n \times n} : \exists S \subseteq [n] \;\; (W_{i,j} = 1 \iff i \in S \vee j \in S)\right\}.$$

We have the following bound on the (β, τ)-decomposition of $\mathcal{W}$, which we prove later.

Lemma 16. *The set $\mathcal{W}$ of symmetric matrices is $(2n, 1)$-decomposable.*

Given an online matrix prediction algorithm, Alg, our online monotone disjunction learning algorithm, MON-DISJ$_2$-Learn, proceeds as described in Figure 4.

Theorem 17. *There exists an efficient algorithm for online learning of monotone disjunctions over X_2 with average regret $O(\sqrt{n \log(n)/T})$, where n is the size of the inputs and T is the number of rounds.*

Proof. We use the algorithm, MON-DISJ$_2$-Learn, with Alg being the algorithm from Theorem 15. Let $s = \langle(\mathbf{x}^t, y^t)\rangle$ be a sequence over $X_2 \times \{0,1\}$. Let $f(x) = \bigvee_{i \in S} x_i$ be a monotone disjunction

Algorithm. MON-DISJ$_2$-Learn: Online Learning of Monotone Disjunctions over X_2
Input: Alg - the algorithm for online matrix prediction.
For $t = 1, \ldots, T$,

1. Receive example $\mathbf{x}^t = \mathbf{x}_{i^t, j^t}$.
2. Give (i^t, j^t) as indexes to Alg. Let Alg choose $\hat{y}^t \in [-1, 1]$.
3. Let $\hat{y}^t_* = 1$ with probability $\max(\hat{y}^t, 0)$ and 0 otherwise. Output $\hat{y}^t_*$.
4. Observe y^t and pass this to Alg.

Figure 4: Algorithm for online agnostically learning monotone disjunctions over X_2.

such that $\mathrm{err}_s(f) = \min_{f' \in \text{MON-DISJ}} \mathrm{err}_s(f')$.

$$\begin{aligned} T \cdot (\mathrm{err}_s(\text{MON-DISJ}_2\text{-Learn}) - \mathrm{err}_s(f)) &= \mathbb{E}\left[\sum_{t=1}^{T} |\hat{y}^t_* - y^t| - |f(x^t) - y^t|\right] \\ &= \sum_{t=1}^{T} |\max(\hat{y}^t, 0) - y^t| - |f(x^t) - y^t| \\ &\leq \sum_{t=1}^{T} |\hat{y}^t - y^t| - |W^f_{i^t, j^t} - y^t| \\ &\leq \mathrm{Regret}(\text{Alg}), \end{aligned}$$

where the expectation is taken over the random choices of the algorithm, MON-DISJ$_2$-Learn. Then, by Theorem 15 and Lemma 16, $\mathrm{Regret}(\text{Alg}) \leq O(\sqrt{n \log(n) T})$, and hence $\mathrm{err}_s(\text{MON-DISJ}_2\text{-Learn}) \leq \mathrm{err}_s(f) + O(\sqrt{n \log(n)/T})$. □

All that remains to prove is Lemma 16.

Proof of Lemma 16. Let $W \in \mathcal{W}$ and let $S \subseteq [n]$ be such that $W_{i,j} = 1 \iff i \in S \vee j \in S$. Without loss of generality, $S = \{1 \cdots k\}$ and

$$W = \begin{pmatrix} 1^{k \times k} & 1^{k \times (n-k)} \\ 1^{(n-k) \times k} & 0^{(n-k) \times (n-k)} \end{pmatrix}.$$

Let

$$W = P - N = \begin{pmatrix} 1^{k \times k} & 1^{k \times (n-k)} \\ 1^{(n-k) \times k} & 1^{(n-k) \times (n-k)} \end{pmatrix} - \begin{pmatrix} 0^{k \times k} & 0^{k \times (n-k)} \\ 0^{(n-k) \times k} & 1^{(n-k) \times (n-k)} \end{pmatrix}.$$

Clearly P and N are symmetric positive semidefinite matrices with trace at most n and maximum entry at most 1, as required. □