PAC-Learning in the Presence of One-sided Classification Noise

Hans Ulrich Simon
Fakultät für Mathematik, Ruhr-Universität Bochum
D-44780 Bochum, Germany

Abstract

We derive an upper and a lower bound on the sample size needed for PAC-learning a concept class in the presence of one-sided classification noise. The upper bound is achieved by the strategy "Minimum One-sided Disagreement". It matches the lower bound (which holds for any learning strategy) up to a logarithmic factor. Although "Minimum One-sided Disagreement" often leads to NP-hard combinatorial problems, we show that it can be implemented quite efficiently for some simple concept classes like, for example, unions of intervals, axis-parallel rectangles, and TREE$(2, n, 2, k)$ which is a broad subclass of 2-level decision trees. For the first class, there is an easy algorithm with time bound $O(m \log m)$. For the second-one (resp. the third-one), we design an algorithm that applies the well-known UNION-FIND data structure and has an almost quadratic time bound (resp. time bound $O(n^2 m \log m)$).

1 Introduction

The classification noise variant of the PAC model (Valiant 1984) was introduced by (Angluin and Laird 1988). It is known that $\Omega\left(\frac{d-1}{\varepsilon(1-2\eta)^2)}\right)$ examples are needed for learning a concept class of VC-dimension d in this model (Simon 1996). A matching upper bound (up to a logarithmic factor) is found in the book by (Laird 1988). It is achieved by the strategy "Minimum Disagreement" which returns a hypothesis that disagrees as seldom as possible with the (possibly corrupted) labels of the instances in the empirical sample. The interest in this learning model was pushed considerably by the work of (Kearns 1998) who introduced the model of learning from statistical queries (SQ-model) and proved the following. First, any algorithm that works in the SQ-model can be efficiently simulated by an algorithm that works in the classification noise variant of the PAC model. Second, almost every efficient PAC-learning algorithm can be rewritten as an efficient algorithm in the SQ-model.

The classification noise in the model introduced by (Angluin and Laird 1988) is *two-sided*, i.e., positive as well as negative examples in the sample may obtain a wrong label. If the wrong label can be assigned to one type of examples only, say to the negative-ones, we obtain *one-sided classification noise* — the main topic of this paper. PAC-learning in the presence of one-sided classification noise is a model of theoretical and practical interest. As shown by (Blum and Kalai 1998), any algorithm that PAC-learns a class $\mathcal{C}$ in the presence of one-sided classification noise can be transformed (without much loss of efficiency) into an algorithm that *PAC-learns $\mathcal{C}$ from multiple-instance examples*. The latter model, call it Multiple Instance Learning or MIL for short, was introduced by (Dietterich, Lathrop, and Lozano-Pérez 1997). MIL is the appropriate model for several learning applications as they occur, for example, in drug design (Dietterich, Lathrop, and Lozano-Pérez 1997), image classification (Maron and Ratan 1998), web index page recommendation (Zhou, Jiang, and Li 2005), and text categorization (Andrews 2007). On top of the motivation that comes from MIL, one-sided errors within the labeling arise naturally when the training data for the learner are prepared in a way where the same "default-label" is assigned to any sample point with an unclear classification.

It was noticed by several researchers already in an early stage of learning theory that relatively simple and low-dimensional classification-rules (e.g., axis-parallel rectangles (Weiss and Kapouleas 1989; Weiss, Galen, and Tadepalli 1990; Weiss and Kulikowski 1990), unions of intervals (Holte 1993) or 2-level decision trees (Auer, Holte, and Maass 1995)) can be quite successful on benchmark data sets provided that these rules are given in terms of the (few) most relevant attributes. It is precisely for this reason that (Maass 1994) and (Auer, Holte, and Maass 1995) presented efficient implementations of "Minimum Disagreement" for unions of k intervals, axis-parallel rectangles and for TREE$(2, n, p, k)$. These algorithms run in time $O(m(k^2+\log m))$, $O(m^2 \log m)$, and $O(k^2 n^2 m \log m)$, respectively, where m denotes the sample size. The main results in this paper are as follows:

- "Minimum One-sided Disagreement" (a variant of "Minimum Disagreement" that is tailored to one-sided classification noise) can be implemented so as to run faster than the algorithms for "Minimum Disagreement" by (Maass 1994) and (Auer, Holte, and Maass 1995): we achieve time bound $O(m \log m)$ for unions of k intervals, an almost quadratic time bound for axis-parallel rectangles, and time bound $O(n^2 m log m)$ for the class TREE$(2, n, 2, k)$.

- "Minimum One-sided Disagreement" learns from few examples: the number of examples needed almost matches with a corresponding lower bound.

2 Definitions, Notations and Facts

For $m \geq 1$, we define $[m] = \{1, \ldots, m\}$. For a set Z, 2^Z denotes the corresponding power-set (set of all subsets of Z). Let $\mathcal{R} \subseteq 2^Z$ be a family of subsets of Z. We say that a set $A \subseteq Z$ is *shattered by* $\mathcal{R}$ if, for any $B \subseteq A$, there exists a set $R \in \mathcal{R}$ such that $B = A \cap R$. The *VC-dimension* of $\mathcal{R}$ is ∞ if there exist arbitrarily large sets that are shattered by $\mathcal{R}$, and the cardinality of the largest set shattered by $\mathcal{R}$ otherwise.

Prerequisites from Probability Theory

Assume now that $\mathcal{R}$ (or a σ-algebra containing $\mathcal{R}$) is equipped with a probability measure P. For any $m \geq 1$, P^m denotes the corresponding product measure. For $R \in \mathcal{R}$ and $S = (z_1, \ldots, z_m) \in Z^m$, we define $\hat{P}_S(R)$ as $|i \in [m] : z_i \in R\}|/m$. In other words, $m\hat{P}_S(R)$ counts how often R is hit by components of S. Clearly, $\hat{P}_S(R)$ approaches $P(R)$ when S is chosen at random according to P^m and m is getting large. In learning theory, R may be a hypothesis that is chosen *in dependence of* S (e.g., a hypothesis that fits the data in S as good as possible). In this case, the relation between $P(R)$ and $\hat{P}_S(R)$ is more delicate and its analysis requires concentration bounds that hold uniformly *for all members of* $\mathcal{R}$. In the sequel, we remind the reader to some well-known facts in this context.

The first result that we mention, Theorem 2.1, gives an answer to the following question: how large do we have to choose a random sample of points such that, with high probability, *any* set of probability mass greater than ε is hit by at least one sample point?

Theorem 2.1 ((Blumer et al. 1989)) *Let d be the VC-dimension of $\mathcal{R} \subseteq 2^Z$, and let $\mathcal{R}_\varepsilon \subseteq \mathcal{R}$ be the subclass of all sets $R \in \mathcal{R}$ such that $P(R) > \varepsilon$. Then, for any $0 < \delta, \varepsilon < 1$ and any*

$$m \geq \max\left\{\frac{4}{\varepsilon}\log\frac{2}{\delta}, \frac{8d}{\varepsilon}\log\frac{13}{\varepsilon}\right\} \ ,$$

the following holds:

$$P^m\{S : (\exists R \in \mathcal{R}_\varepsilon : \hat{P}_S(R) = 0)\} \leq \delta$$

The next three results use the notation from Theorem 2.1. The first of them is concerned with the danger that, for some $R \in \mathcal{R}$, $\hat{P}_S(R)$ significantly underestimates $P(R)$:

Theorem 2.2 ((Blumer et al. 1989)) *For any $0 < \delta, \varepsilon, \gamma < 1$ and any*

$$m \geq \max\left\{\frac{8}{\gamma^2\varepsilon}\ln\frac{8}{\delta}, \frac{16d}{\gamma^2\varepsilon}\ln\frac{16}{\gamma^2\varepsilon}\right\} \ ,$$

the following holds:

$$P^m\{S : (\exists R \in \mathcal{R}_\varepsilon : \hat{P}_S(R) \leq (1-\gamma)P(R))\} \leq \delta$$

We are not aware of a result that is directly concerned with the dual danger of having an empirical overestimation of the true probability by a factor $1 + \gamma$ (or more). However such a result, see Corollary 2.4 below, can easily be derived from the following (more general) result:

Theorem 2.3 ((Haussler 1992)) *For any $0 < \alpha, \delta < 1$, any $0 < \nu \leq 8$, and any*

$$m \geq \frac{8}{\alpha^2\nu}\left(2d\ln\frac{8e}{\alpha\nu} + \ln\frac{8}{\delta}\right) \ , \tag{1}$$

the following holds:

$$P^m\left\{S : \left(\exists R \in \mathcal{R} : \frac{|\hat{P}_S(R) - P(R)|}{\nu + \hat{P}_S(R) + P(R)} > \alpha\right)\right\} \leq \delta \tag{2}$$

Corollary 2.4 *For any $0 < \delta, \varepsilon < 1$, any $0 < \gamma < 3$, and any*

$$m \geq \frac{8(3+\gamma)^2}{\gamma^2\varepsilon}\left(2d\ln\frac{8e(3+\gamma)}{\gamma\varepsilon} + \ln\frac{8}{\delta}\right)$$

the following holds:

$$P^m\{S : (\exists R \in \mathcal{R}_\varepsilon : \hat{P}_S(R) \geq (1+\gamma)P(R))\} \leq \delta$$

Proof: The probability in (2) upper-bounds the probability of the event E_1 given by

$$\hat{P}_S(R) - P(R) > \alpha(\nu + \hat{P}_S(R) + P(R)),$$

which is equivalent to

$$(1-\alpha)\frac{\hat{P}_S(R)}{P(R)} > 1 + \alpha + \frac{\alpha\nu}{P(R)} \ .$$

Setting $\nu = \varepsilon < P(R)$, we see that the probability of E_1 upper-bounds the probability of the event E_2 given by

$$\frac{\hat{P}_S(R)}{P(R)} \geq \frac{1+2\alpha}{1-\alpha} = 1 + \frac{3\alpha}{1-\alpha} \ .$$

Note that $\frac{3\alpha}{1-\alpha} = \gamma$ for $\alpha = \frac{\gamma}{3+\gamma}$. Thus in order to bound the probability for $\frac{\hat{P}_S(R)}{P(R)} \geq 1 + \gamma$ by δ, it suffices to set $\nu = \varepsilon$, $\alpha = \frac{\gamma}{3+\gamma}$, and to choose m according to (1). This leads us to Corollary 2.4. •

An easy monotonicity argument shows that, for m chosen as in Corollary 2.4, the following holds:

$$P^m\{S : (\exists R \in \mathcal{R} : \hat{P}_S(R) \geq (1+\gamma)\max\{\varepsilon, P(R))\} \leq \delta \tag{3}$$

Prerequisites from Learning Theory

In learning theory a family $\mathcal{C} \subseteq 2^X$ of subsets of X is called a *concept class over domain* X. Members $f \in \mathcal{C}$ are sometimes viewed as functions from X to $\{0, 1\}$ (with the obvious one-to-one correspondence between these functions and subsets of X). An algorithm $\mathcal{A}$ is said to *(properly) PAC-learn $\mathcal{C}$ with sample size $m(\varepsilon, \delta)$* if, for any $0 < \varepsilon, \delta < 1$, any (so-called) target concept $f \in \mathcal{C}$, and any domain distribution P the following holds:

1. If $\mathcal{A}$ is applied to a (so-called) sample $S = (x_1, f(x_1)), \ldots, (x_m, f(x_m))$, it returns (a "natural" representation of) a hypothesis $h \in \mathcal{C}$.

2. If $m = m(\varepsilon, \delta)$ and the instances $x_1, \ldots, x_m$ in S are drawn at random according to P^m, then, with probability at least $1 - \delta$, $P\{x : h(x) = 0 \wedge f(x) = 1\} + P\{x : h(x) = 1 \wedge f(x) = 0\} \leq \varepsilon$.

We briefly note that in the original definition of PAC-learning (Valiant 1984) $m(\varepsilon, \delta)$ is required to be polynomially bounded in $1/\varepsilon, 1/\delta$ and $\mathcal{A}$ has to be polynomially time-bounded. In this paper, we do obtain polynomial bounds on the sample size, but these bounds are proved for a strategy that cannot always be implemented in polynomial time. In Section 4, however, we will come back to the issue of efficiency.

In the presence of *classification noise with noise-rate* η, the task for the learner is made harder by corrupting some of the labels in the sample. For any instance x_i in the sample, we flip a coin with bias η for showing "heads". If the coin shows "heads", we set $b_i = 1 - f(x_i)$; otherwise we set $b_i = f(x_i)$. The learner then obtains as input the corrupted sample $S = (x_1, b_1), \ldots, (x_m, b_m)$ (and still has to satisfy the same success criterion as before). We can think of S as being drawn at random according to $P^m_{f,\eta}$ where, for any P-measurable set A, $P_{f,\eta}\{(x, f(x)) : x \in A\} = (1-\eta) \cdot P(A)$ and $P_{f,\eta}\{(x, 1 - f(x)) : x \in A\} = \eta \cdot P(A)$.

A *semi-random model* of classification-noise is the following (more malicious) variant of the model just described. The coin with bias η is flipped m-times and the instances x_i for which the coin showed "heads" are marked. Then an adversary of the learner decides whether the labels of the marked instances are flipped or not. At first glance it looks as if a delivery of the true label instead of the flipped-one could be to the advantage of the learner only. This, however, is spurious thinking. There do exist learning strategies that rely on the statistics resulting from flipping the label at the precise rate η. These algorithms can easily be fooled by an adversary. In other words: algorithms that work fine in the semi-random model of classification noise can be considered more robust against noise.

In the model with *one-sided classification noise*, the label of a negative example is flipped with probability η (as before), but the labels of the positive examples are not touched. Again one can consider the semi-random variant of this model.

The notion "Minimum Disagreement" refers to a learning strategy that returns a hypothesis with a smallest number of disagreements with the (possibly corrupted) labels in the sample.

In this paper, the notion "Minimum One-sided Disagreement" refers to a strategy that returns a hypothesis which minimizes the number of disagreements with the 1-labels in the sample S subject to the constraint of having perfect agreement with the 0-labels in S. This strategy is reasonable when learning takes place in the presence of one-sided classification noise.

3 Tight Bounds on the Sample Size

We begin this section with the analysis of "Minimum One-sided Disagreement":

Theorem 3.1 *Let $\mathcal{C}$ be a concept class of VC-dimension d. "Minimum One-sided Disagreement" PAC-learns $\mathcal{C}$ in the presence of one-sided classification noise from*

$$O\left(\frac{1}{\varepsilon(1-\eta)}\left(d \ln \frac{1}{\varepsilon(1-\eta)} + \ln \frac{1}{\delta}\right)\right) \tag{4}$$

examples.

Proof: Assume that the sample size m has order (4) of magnitude (with sufficiently large constants). Let $S = (x_1, b_1), \ldots, (x_m, b_m)$ be a sample drawn at random according to $P^m_{f,\eta}$, where f denotes the target concept. The following statements are the main building stones in the proof:

Claim 1: With a probability of at least $1 - \delta/3$, the following holds for any $h \in \mathcal{C}$ with one-sided disagreement on S:

$$P\{x : h(x) = 1 \wedge f(x) = 0\} \leq \frac{\varepsilon}{5} \tag{5}$$

Claim 2: With probability at least $1 - \delta/3$, the following holds for any $h \in \mathcal{C}$ such that $P\{x : h(x) = 0 \wedge f(x) = 1\} > 4\varepsilon/5$:

$$\hat{P}_S\{x : h(x) = 0 \wedge f(x) = 1\} \geq \frac{1}{2} P\{x : h(x) = 0 \wedge f(x) = 1\} \tag{6}$$

Claim 3: With probability at least $1 - \delta/3$, the following holds for any $h \in \mathcal{C}$ such that $P\{x : h(x) = 1 \wedge f(x) = 0\} > \varepsilon/5$:

$$\hat{P}_S\{x : h(x) = 1 \wedge f(x) = 0\} \leq 2P\{x : h(x) = 1 \wedge f(x) = 0\} \tag{7}$$

Claim 4: The hypothesis returned by "Minimum One-sided Disagreement", say h_S, satisfies the following condition:

$$\hat{P}_S\{x : h_S(x) = 0 \wedge f(x) = 1\} \leq \hat{P}_S\{x : h_S(x) = 1 \wedge f(x) = 0\} \ . \tag{8}$$

Before proving the claims, we show how they are applied. Note that, with probability at least $1 - \delta$, the inequalities (5), (6), (7), are valid. It therefore suffices to show that these inequalities together with (8) imply that h_S errs with probability at most ε. We analyze the two types of errors separately. According to Claim 1, $P\{x : h_S(x) = 1 \wedge f(x) = 0\} \leq \varepsilon/5$. Assume for sake of contradiction that $P\{x : h_S(x) = 0 \wedge f(x) = 1\} > 4\varepsilon/5$ so that h_S satisfies (6). This leads to a contradiction as follows:

$$P\{x : h_S(x) = 0 \wedge f(x) = 1\} \overset{(6)}{\leq} 2\hat{P}_S\{x : h_S(x) = 0 \wedge f(x) = 1\} \overset{(8)}{\leq} 2\hat{P}_S\{x : h_S(x) = 1 \wedge f(x) = 0\} \overset{(3)}{\leq} \frac{4\varepsilon}{5}$$

In the final inequality, we made use of Claim 1, and we applied (3) with $\gamma = 1$ and $\varepsilon/5$ in the role of ε. The proof of the theorem can now be accomplished by the verification of the claims. As for Claim 1, we observe first that the following holds for any $h \in \mathcal{C}$:

$$P\{x : h(x) = 1 \wedge f(x) = 0\} = \frac{1}{1-\eta} \cdot P_{f,\eta}\{(x,0) : h(x) = 1\} \ . \quad (9)$$

We apply Theorem 2.1 with the following set-up of the relevant parameters:

- $Z = \{(x,b) \in X \times \{0,1\} : b \geq f(x)\}$, where condition $b \geq f(x)$ reflects our assumption that the labels of positive examples are not affected with noise.
- $\mathcal{R} \subseteq 2^Z$ is the family of sets of the form $\{(x,0) : h(x) = 1\}$ for some $h \in \mathcal{C}$.
- P from Theorem 2.1 is identified here with $P_{f,\eta}$.
- ε from Theorem 2.1 is identified here with $\varepsilon(1-\eta)/5$.

The VC-dimension of $\mathcal{R}$ equals the VC-dimension of $\{h \backslash f : h \in \mathcal{C}\}$ and is therefore upper-bounded by d. It follows now from Theorem 2.1 that the sample size m is large enough so that, with probability at least $1 - \delta/3$, any hypothesis h with $P_{f,\eta}\{(x,0) : h(x) = 1\} > \varepsilon(1-\eta)/5$ is hit at least once by an example labeled 0 in S. Since such a hypothesis has not one-sided disagreement, we can conclude that any hypothesis h with one-sided disagreement on S satisfies $P_{f,\eta}\{(x,0) : h(x) = 1\} \leq \varepsilon(1-\eta)/5$. According to (9), this implies that $P\{x : h(x) = 1 \wedge f(x) = 0\} \leq \varepsilon/5$.
In order to verify Claim 2, we observe first that

$$P\{x : h(x) = 0 \wedge f(x) = 1\} = P_{f,\eta}\{(x,1) : h(x) = 0 \wedge f(x) = 1\} \ .$$

Thus it suffices to show that $\hat{P}_S\{h(x) = 0 \wedge f(x) = 1\} \geq \frac{1}{2} P_{f,\eta}\{(x,1) : h(x) = 0 \wedge f(x) = 1\}$ holds with probability at least $1 - \delta/3$, which can easily be verified by means of Theorem 2.2 and the following set-up of the relevant parameters:

- $Z = \{(x,1) \in X \times \{1\} : f(x) = 1\}$.
- $\mathcal{R} \subseteq 2^Z$ is the family of all sets of the form $\{(x,1) : h(x) = 0 \wedge f(x) = 1\}$.
- $P_{f,\eta}$ plays the role of P in Theorem 2.2.
- ε in Theorem 2.2 is replaced here by $4\varepsilon/5$, and γ is set to $1/2$.

It is easy to fill in the missing details.
The verification of Claim 3 is similar to the verification of Claim 2. One can first observe that $P\{x : h(x) = 1 \wedge f(x) = 0\} = P_{f,\eta}\{(x,b) : h(x) = 1 \wedge f(x) = 0 \wedge b \in \{0,1\}\}$ and then apply Corollary 2.4. Again, it is easy to fill in the missing details.
We finally verify Claim 4. Note first that $|\{i \in [m] : h_S(x_i) = 0 \wedge b_i = 1\}| \leq |\{i \in [m] : f(x_i) = 0 \wedge b_i = 1\}|$ since "Minimum One-sided Disagreement" minimizes the number of 0-labels that are assigned to examples having label 1 in the sample. It follows that $|\{i \in [m] : h_S(x_i) = 0 \wedge f(x_i) = 1 \wedge b_i = 1\}| \leq |\{i \in [m] : f(x_i) = 0 \wedge h_(x_i) = 1 \wedge b_i = 1\}|$. Since $h_S(x_i) = f(x_i) = 0$ for all $i \in [m]$ such that $b_i = 0$, we conclude that

$$\overbrace{|\{i \in [m] : h_S(x_i) = 0 \wedge f(x_i) = 1\}|}^{= m \cdot \hat{P}_S\{x : h_S(x) = 0 \wedge f(x) = 1\}} \leq \underbrace{|\{i \in [m] : f(x_i) = 0 \wedge h_S(x_i) = 1\}|}_{= m \cdot \hat{P}_S\{x : h_S(x) = 1 \wedge f(x) = 0\}} \ ,$$

which proves (8). •

We briefly note that the proof of Theorem 3.1, after some minor modifications, shows that (4) upper-bounds the number of examples needed by "Minimum One-sided Disagreement" even in the semi-random model.

Here comes the (almost) matching lower bound on the sample size (valid for any algorithm).

Theorem 3.2 *Let $\mathcal{C}$ be a concept class of VC-dimension d. $\mathcal{C}$ cannot be PAC-learned in the presence of one-sided classification noise from fewer than $\Omega\left(\frac{d-1}{\varepsilon(1-\eta)}\right)$ examples.*

Proof: The proof is similar to the proof for the corresponding lower bound in the noise-free setting (Ehrenfeucht et al. 1989). Let $t = d - 1$, and let $X_0 = \{x_0, x_1, \ldots, x_t\}$ be a set of size d that is shattered by $\mathcal{C}$. Assign probability $1 - 4\varepsilon$ to x_0 and distribute the remaining probability mass, 4ε, equally among the points $x_1, \ldots, x_t$. The target concept is chosen at random by assigning label 1 to x_0 and by flipping a perfect coin independently t times in order to determine the labels for $x_1, \ldots, x_t$, respectively. Let the sample size m be upper-bounded by $\frac{t}{16\varepsilon(1-\eta)}$. It suffices to show that, with a probability of at least $1/2$, the error of the learner, averaged over the 2^t possible target concepts, is at least ε. To this end, let $S = (x'_1, b_1), \ldots, (x'_m, b_m)$ with $x'_1, \ldots, x'_m \in X_0$ be the random sample, and let Y be the random variable that counts the number of sample points with label 0, i.e., $Y = |\{i \in [m] : b_i = 0\}$. Since positive examples are always presented with label 1, and negative examples are labeled 0 with probability $1 - \eta$, it follows that $\mathbb{E}[Y] \leq 4\varepsilon(1-\eta)m \leq t/4$. According to Markov's inequality, $Y \leq t/2$ with a probability of at least $1/2$. If this happens, a learner cannot do better than randomly guessing the labels of the $t/2$ (or more) points that did not occur with label 0 in the sample. Thus, with a probability of at least $1/2$, the average error of the learner is at least ε (= half of probability mass of $t/2$ points from $\{x_1, \ldots, x_t\}$). •

4 Efficient Learners for Simple Classes

In this section, we show that "Minimum One-sided Disagreement" can be implemented quite efficiently for some simple (but arguably important) concept classes. There will be no loss of efficiency when the input S contains items of the form $(x, w, b) \in X \times \mathbb{R}^+ \times \{0,1\}$ so that a disagreement on (x,b) is penalized by w. For this reason, we deal with such "weighted labeled samples" throughout this section, and the objective is to minimize the total "weighted one-sided disagreement" on the input sample S.

Theorem 4.1 *For the concept class consisting of unions of k (or less) intervals of the real line, "Minimum One-sided Disagreement" has an implementation with time bound $O(m \log m)$.*

Proof: Let $S = (x_1, w_1, b_1), \ldots, (x_m, w_m, b_m)$ denote the weighted labeled sample that serves as input. The algorithm for "Minimum One-sided Disagreement" is based on sorting and proceeds as follows:

1. Sort S in increasing order of x_i and return the sorted sequence, say $S' = (x'_1, w'_1, b'_1), \ldots, (x'_m, w'_m, b'_m)$ such that $x'_1 \leq \cdots \leq x'_m$.
2. Find (as few as possible) sub-intervals of $[x'_1, x'_m]$, say $I_1, \ldots, I_l$, that do not include any instance occurring in S' with label 0 but cover the remaining instances in S'. For $j = 1, \ldots, l$, let W_j denote the total weight of all items from S' that belong to I_j.
3. Sort the l intervals in decreasing order of W_j and return the union of the $\min\{k, l\}$ first intervals in this ordering.

The second step can be implemented in linear time by one pass through S'. The run-time is dominated by the two calls of the sorting procedure. •

The proof of the next result is an application of the well-known UNION-FIND data structure. This data structure is used for the administration of a collection of pairwise disjoint sets and supports the operations FIND (given an element, return the set to which it belongs) and UNION (given two sets, return their union). It is well-known that, given a partition of n elements, a sequence of m UNION- or FIND-operations can be performed in "almost" linear time. Here the word "almost" hides an additional factor $\alpha(n)$ that is a close relative of the inverse of the Ackermann function. Since $\alpha(n) \leq 4$ for all $n \leq 16^{512}$, this factor can be considered as a small constant in practice. For more details about the UNION-FIND data structure, the reader is referred to any standard book about Efficient Algorithms (e.g. (Cormen et al. 2009)).

Theorem 4.2 *Let $\mathcal{B}_2$ denote the class of axis-parallel rectangles in the Euclidean plane (also called two-dimensional boxes). For this concept class, "Minimum One-sided Disagreement" has an implementation with an almost quadratic time bound.*

Proof: Let $S = ((x_1, y_1), w_1, b_1), \ldots, ((x_m, y_m), w_m, b_m)$ be the given weighted labeled sample. In the sequel, we think of S as a list that is sorted lexicographically according to y in decreasing order. Clearly, this list can be produced in $O(m \log m)$ steps. Let y_{max} (resp. y_{min}) be the largest (resp. smallest) y-coordinate of an instance in S. Let y' be a (plane-sweep) variable that ranges from y_{max} to y_{min} and takes the values attained by y-coordinates of points in S in between. Let $S_{\uparrow}(y')$ denote the initial part of the list S containing all points from S whose y-coordinate is at least y'. Let $T = T(y')$ denote a binary search tree that contains the elements of $S_{\uparrow}(y')$ and is organized with respect to the x-coordinates of these elements. T is initialized by $T(y_{max})$ and, as y' decreases and the set $S_{\uparrow}(y')$ becomes larger, is extended incrementally (thereby maintaining a balance criterion if we want to). Since every sample point is inserted into T only once, the total number of steps required for the administration of T is bounded by $O(m^2)$ (or even by $O(m \log m)$ when we decided before to keep the search tree balanced). Let us now consider a fixed value of y'. The theorem is obvious from the following

Claim 5: Given the data structures mentioned above, a best box (i.e., a box with minimum weighted one-sided disagreement) among the boxes with one-sided disagreement and with a bottom line at level y' can be computed in almost linear time.

The key observation is that $T(y')$ can be used to initialize a UNION-FIND data structure which enables us to efficiently find a best box in the sense of Claim 5. To this end, we traverse $T(y')$ in inorder (in direction from small to large x-coordinates) and, on the way, compute the following partition $S_1, \ldots, S_l$ of $S_{\uparrow}(y')$:

- Every point in S_i is to the left of every point in S_{i+1}, i.e., the partition $S_1, \ldots, S_l$ is induced by a left-to-right partition of the plane into vertical stripes.
- For every S_i exactly one of the following conditions holds:
  - S_i contains positive examples only (set of "positive type").
  - S_i contains negative examples only (set of "negative type").
  - S_i contains both types of examples (set of "mixed type") and all examples in S_i share the same x-coordinate (i.e., every set of a mixed type is located on a vertical line).
- If S_i is of positive (resp. negative) type, then S_{i+1} is not, i.e., sets of positive (resp. negative) type extend to left and right as much as possible.

Let $W_1(i)$ (resp. $W_0(i)$) denote the total weight of positive (resp. negative) examples in S_i. We initialize a UNION-FIND data structure with the partition $S_1, \ldots, S_l$. Specifically, we use the tree-implementation with path-compression (see Chapter 21.4 of (Cormen et al. 2009)) so that every set S_i is stored in a tree T_i. We furthermore connect the roots of the trees $T_1, \ldots, T_l$, so as to form a doubly linked list, and we store the values $W_0(\cdot), W_1(\cdot)$ at the respective roots. It is easy to see that the described initialization of the UNION-FIND data structure can be done within $O(m)$ steps by means of an inorder-traversal of $T(y')$. For any finite set $M \subset \mathbb{R}^2$, let $\langle M \rangle$ denote the smallest box containing M. It is easy to check that a best box among the ones with the bottom-line at level y' and the top-line at level y_{max} must be among the boxes $\langle S_i \rangle$ such that $i \in [l]$ and S_i is of positive type. Claim 5 is now proved by using another (plane-sweep) variable y'' that ranges from y_{max} to y' so that, for every fixed value of y'', the UNION-FIND data structure enables us to efficiently perform a comparable search through all (promising) boxes with the bottom-line at level y' and the top-line at level y''. To this end, we analyze what happens when y'' is decreased to the next possible

value, say from y''_{old} to y''_{new}. Let $S^{=}(y'')$ be the segment of the list S_y that contains the examples whose y-coordinate equals y''. When we assign the value y''_{new} to the variable y'', we have to (virtually) eliminate the elements of $S^{=}(y''_{old})$ from the UNION-FIND data structure and to perform the resulting updates. Specifically, we do the following for every element $((x, y), w, b)$ in $S^{=}(y''_{old})$:

1. FIND the set, say S_i, that contains $((x, y), w, b)$. Retrieve the neighbors of S_i, say $S_{i'}$ and $S_{i''}$, in the doubly linked list.
2. Decrement $W_b(i)$ by w.
3. If $W_b(i) = W_{1-b}(i) = 0$ (i.e., S_i is empty), then delete S_i from the doubly linked list. If $S_{i'}$ and $S_{i''}$ are of the same non-mixed type, then apply the operation UNION to them.
4. If $W_b(i) = 0$ and $W_{1-b}(i) \neq 0$ (so that the type of S_i changes from "mixed" to either "positive" or "negative"), then apply the operation UNION to $S_{i'}$ (resp. $S_{i''}$) and S_i provided that $S_{i'}$ (resp. $S_{i''}$) is of the same type as S_i.
5. If UNION operations have taken place, update the information in the records associated with the roots of the affected trees accordingly.

For a fixed value of y', let $S^*_{y'}$ denote the set of positive type with the largest total weight that was build within the UNION-FIND data structure during the loop which decreases y'' from its start value y_{max} to its final value y'. It is easy to see that $\langle S^*_{y'} \rangle$ is a best box in the sense of Claim 5. Clearly, $S^*_{y'}$ can be computed on the way by keeping track of the currently "heaviest" set of positive type (the "champion") and its total weight. The run-time is dominated by the administration of the UNION-FIND data structure. For every fixed y', the operations FIND and UNION are called at most m times, respectively. This yields Claim 5. •

The proof of the following result (given for sake of completeness) is completely analogous to the proof of a similar result in (Maass 1994):

Corollary 4.3 *"Minimum One-sided Disagreement" can implemented in almost quadratic time for the concept class of unions of two disjoint axis-parallel rectangles.*

Proof: Given $S = ((x_1, y_1), w_1, b_1), \ldots, ((x_m, y_m), w_m, b_m)$, let us assume that a "best" union of two disjoint boxes B_1, B_2 has the property that B_1, B_2 can be separated by a horizontal line, say at level y^*. (The case where they can be separated by a vertical line is similar.) We use the notation from the proof of Theorem 4.2. For every fixed value of y', we can find a best box $B_\uparrow(y')$ for $S_\uparrow(y')$ in almost linear time. For reasons of symmetry, we find a best box $B_\downarrow(y)$ for $S_\downarrow(y') = \{((x_i, y_i), w_i, b) \in S : y_i < y'\}$ in the same amount of time. When y' (which takes at most m values between y_{max} and y_{min}) reaches the lowest level above y^*, the algorithm will find the two boxes that form a best union. Clearly, the whole procedure still runs in almost quadratic time. •

By combining arguments from the proofs of the preceding two theorems, we can show that "Minimum One-sided Disagreement" can be solved efficiently for 2-level decision trees. Details follow.

Let $y' \in \mathbb{R}$ and let $\mathcal{I} = (I_1, \ldots, I_k)$ and $\mathcal{J} = (J_1, \ldots, J_k)$ be two collections of pairwise disjoint intervals being ordered from left to right, respectively. The function $h_{y',\mathcal{I},\mathcal{J}} : \mathbb{R}^2 \to \{0, 1\}$ is defined by setting $h_{y',\mathcal{I},\mathcal{J}}(x) = 1$ if and only if

either $y < y'$ and $x \in \cup_{\ell=1}^{k} I_\ell$ **or** $y \geq y'$ and $x \in \cup_{\ell=1}^{k} J_\ell$.

The concept class $\mathcal{T}_{2,k}$ consisting of all concepts of this form corresponds to 2-level decision trees of the following kind:

- At the root node, it is tested whether a continuous attribute y is below a threshold y'.
- At the nodes on level 1 (the two children of the root), it is tested whether a continuous attribute x falls into a union of k intervals (where the interval collections associated with the left and the right child of the root, respectively, can be chosen independently from each other).

Let $S = \{((x_1, y_1), w_1, b_1), \ldots, ((x_m, y_m), w_m, b_m)\}$ denote a weighted labeled sample. Let $Y = \{y_1, \ldots, y_m\}$. Given a threshold $y' \in Y$, S partitions into the following two sets:

$$\begin{aligned} S_\downarrow(y') &= \{((x_i, y_i), w_i, b_i) : y_i < y'\} \\ S_\uparrow(y') &= \{((x_i, y_i), w_i, b_i) : y_i \geq y'\} \end{aligned}$$

Once we have committed ourselves to a partition of S into $S_\downarrow(y')$ and $S_\uparrow(y')$, the y-coordinates of the sample points become irrelevant. For example, in order to minimize the weighted one-sided disagreement on $S_\downarrow(y')$, we have to find k vertical stripes (corresponding to k intervals on the x-axis) that include a maximum total weight of positive examples in $S_\downarrow(y')$ subject to the constraint of excluding all negative examples in $S_\downarrow(y')$. For fixed y', this problem can be cast as Minimum One-sided Disagreement for the concept class "Union of (up to) k Intervals". But what causes trouble is that the search for the best y' is intertwined with the search for the best collections of intervals on the x-axis.

Let $d_\downarrow(y')$ denote the minimum weighted one-sided disagreement that can be achieved on $S_\downarrow(y')$ by a union of (up to) k vertical stripes. Let $d_\uparrow(y')$ denote the corresponding quantity for $S_\uparrow(y')$.

Lemma 4.4 *With the above notations the following holds. There is a procedure that, on input S, runs for $O(m \log m)$ steps and returns the sequences $(d_\downarrow(y'))_{y' \in Y}$ and $(d_\uparrow(y'))_{y' \in Y}$.*

Proof: For reasons of symmetry, we may restrict ourselves to the procedure that returns the sequence $(d_\downarrow(y'))_{y' \in Y}$. Given S, we build up two lists, say S_x and S_y. S_x contains the items of S sorted according to increasing values of x whereas S_y contains the elements of S sorted according to decreasing values of y. Clearly, both list can be produced in $O(m \log m)$ steps. In addition, the following data structures are used:

1. A UNION-FIND data structure $\mathcal{P}$ that maintains a partition of $S_\downarrow(y')$, say $S_1, \ldots, S_l$. This partition is build in analogy to the partition of $S_\uparrow(y')$ that was described within the proof of Theorem 4.2. Specifically, $\mathcal{P}$ is induced by a left-to-right partition of the plane into vertical stripes, and the sets $S_1, \ldots, S_l$ are of type either "positive" (also called type 1), "negative" (also called type 0) or "mixed". All sample points in a set of mixed type share the same x-coordinate, whereas the stripes induced by non-mixed sets extend to left and right as much as possible. For ease of later reference, the vertical stripes associated with $S_1, \ldots, S_l$ are denoted $\langle S_1 \rangle, \ldots, \langle S_l \rangle$, respectively. We use the well-known list-implementation of $\mathcal{P}$ (see Chapter 21.2 of (Cormen et al. 2009)) so that we may think of every set S_i as a list. A FIND-operation is then executed in constant time and m UNION-operations take time $O(m \log m)$. The heads of the lists $S_1, \ldots, S_l$ are connected by a doubly linked list, and with every $i \in [l]$ and $b = 0, 1$, we associate the total weight $W_b(i)$ of all sample points in S_i that are labeled b.
2. PRIORITY QUEUES $\mathcal{Q}_1$ and $\mathcal{Q}_2$ that, in combination, contain exactly the items $(i, W_1(i))$ such that $W_1(i) > 0$ and $W_0(i) = 0$ (i.e., the items corresponding to sets of positive type): $\mathcal{Q}_1$ contains the items $(i, W_1(i))$ with the (up to) k largest W_1-values (breaking ties arbitrarily), and $\mathcal{Q}_2$ contains the remaining-ones. Both priority queues are organized according to the weights $W_1(i)$ with highest priority and according to i with second priority. $\mathcal{Q}_1$ supports the operations INSERT, DELETE and MIN whereas $\mathcal{Q}_2$ supports the operations INSERT, DELETE and MAX so that every single operation can be executed in $O(\log m)$ steps.
3. A global variable C whose value equals the total W_1-weight of the items which are stored in $\mathcal{Q}_2$.

The above invariance properties of our data structures make sure that the following holds:

- The minimum weighted one-sided disagreement $d_\downarrow(y')$ that can be achieved on $S_\downarrow(y')$ by a union of (up to) k vertical stripes is achieved by the union of the stripes $\langle S_i \rangle$ such that $(i, W_1(i))$ is stored in $\mathcal{Q}_1$.
- Thus, $d_\downarrow(y')$ equals the total W_1-weight of the items that are stored in $\mathcal{Q}_2$, which coincides with the current value of the global variable C.

The variable y' runs from $y_{max} = 1 + \max Y$ to $y_{min} = \min Y$, attaining all values of the y-coordinates of items in S_y in between. Initially, $y' = y_{max}$ so that $S_\downarrow(y')$ coincides with the full sample S. It is easy to see that, given the list S_x, the data structures $\mathcal{P}$, $\mathcal{Q}_1$ and $\mathcal{Q}_2$ can be initialized within $O(m \log m)$ steps. As y' becomes smaller and smaller, more and more items of S disappear from $S_\downarrow(y')$. The central part of the proof is the analysis of the updates that are caused by a single item $((x, y), w, b)$ which leaves $S_\downarrow(y')$. These updates (except for the updates of C) are as follows:

1. FIND the set, say S_i, that contains $((x, y), w, b)$.
2. Decrement $W_b(i)$ by w, which can be seen as a virtual removal of $((x, y), w, b)$ from S_i.
3. If $W_b(i) = W_{1-b}(i) = 0$ (i.e., S_i is empty), then do the following:
   (a) If $b = 1$, then DELETE the item (i, w) from $\mathcal{Q}_1$ and from $\mathcal{Q}_2$. (Note that w is the W_1-value of i within either $\mathcal{Q}_1$ or $\mathcal{Q}_2$, and note that the operation DELETE has no effect on the priority queue which does *not* contain the item (i, w).)
   (b) Delete the head of S_i from the doubly linked list. If the former neighbors to the left and right of S_i, say $S_{i'}$ and $S_{i''}$, are of the same non-mixed type $b' \in \{0, 1\}$, then do the following:
   Update 1: Apply the operation UNION to i' and i'' so that $S_{i'} \leftarrow S_{i'} \cup S_{i''}$ and change the doubly linked list accordingly.
   Update 2: If $b' = 1$, then DELETE the two items $(i', W_1(i')), (i'', W_1(i''))$ from $\mathcal{Q}_1$ and from $\mathcal{Q}_2$, and INSERT $(i', W_1(i') + W_1(i''))$ into $\mathcal{Q}_2$.
   Update 3: $W_{b'}(i') \leftarrow W_{b'}(i') + W_{b'}(i'')$.
4. If $W_b(i) = 0$ and $W_{1-b}(i) \neq 0$ so that the type of S_i changes from "mixed" to $b' = 1-b$, then do the following:
   (a) If $b' = 1$, then DELETE $(i, W_1(i) + w)$ from $\mathcal{Q}_1$ and from $\mathcal{Q}_2$. INSERT $(i, W_1(i))$ into $\mathcal{Q}_2$.
   (b) Retrieve the neighbors of S_i in the doubly linked list, say these are the sets $S_{i'}$ and $S_{i''}$.
   (c) If $S_{i'}$ (resp. $S_{i''}$) is of the same type as S_i, then perform the Updates 1, 2 and 3 described above where the role of i'' (resp. the role of i') is taken by i.
5. If $b = 1$, $W_b(i) \neq 0$ and $W_{1-b}(i) = 0$ (so that S_i is not empty and of positive type, then DELETE $(i, W_1(i) + w)$ from $\mathcal{Q}_1$ and $\mathcal{Q}_2$ and INSERT $(i, W_1(i))$ into $\mathcal{Q}_2$.
6. In order to restore the property that $\mathcal{Q}_1$ contains the (up to) k items with the largest W_1-values, do the following:
   (a) While $\mathcal{Q}_1$ contains fewer than k items and $\mathcal{Q}_2$ is non-empty, the item with the maximal key in $\mathcal{Q}_2$ is moved from $\mathcal{Q}_2$ to $\mathcal{Q}_1$ (at the expense of one operation MAX, one DELETION from $\mathcal{Q}_2$ and one INSERTION into $\mathcal{Q}_1$).
   (b) While the minimal key in $\mathcal{Q}_1$ is smaller than the maximal key in $\mathcal{Q}_2$, the items occupying these keys are swapped (at the expense of one operation MIN, one operation MAX, two DELETIONS and two INSERTIONS).

In order to keep the specification of the updates simple, we did not explicitly mention the updates of the global variable C. These updates are given implicitly by the modifications of $\mathcal{Q}_2$: whenever an item $(i, W_1(i))$ is DELETED from (resp. INSERTED into) $\mathcal{Q}_2$, C must be updated according to $C \leftarrow C - W_1(i)$ (resp. $C + W_1(i)$). The key observation, which concludes the proof of Lemma 4.4, is that every single item of S_y causes updates that can be executed with constantly many of the supported operations (UNION, FIND, INSERT, DELETE, MIN, MAX) plus little additional overhead. •

We are now ready to state the final main results in this section:

Theorem 4.5 *For the concept class $\mathcal{T}_{2,k}$, "Minimum One-sided Disagreement" has an implementation with time bound $O(m \log m)$.*

Proof: We can proceed as follows:

1. Run the procedure mentioned in Lemma 4.4 on input S and obtain the sequences $(d_\downarrow(y'))_{y' \in Y}$ and $(d_\uparrow(y'))_{y' \in Y}$.
2. Pick a minimizer $y^* \in Y$ of $d_\downarrow(y') + d_\uparrow(y')$.
3. Project the weighted samples $S_\downarrow(y^*)$ and $S_\uparrow(y^*)$ on the x-axis (in the obvious manner), which yields two weighted samples on the real line, say $\tilde{S}_\downarrow(y^*)$ and $\tilde{S}_\uparrow(y^*)$, respectively.
4. Run the algorithm mentioned in Theorem 4.1 on input $\tilde{S}_\downarrow(y^*)$ and obtain k intervals $\mathcal{I} = (I_1, \ldots, I_k)$ whose union produces one-sided disagreement of weight $d_\downarrow(y^*)$ on $\tilde{S}_\downarrow(y^*)$ (so that the corresponding vertical stripes produce the one-sided disagreement of the same weight on $S_\downarrow(y^*)$).
5. Proceed analogously for $\tilde{S}_\uparrow(y^*)$ and obtain k intervals $\mathcal{J} = (J_1, \ldots, J_k)$ whose union produces one-sided disagreement of weight $d'(y^*)$ on $\tilde{S}_\uparrow(y^*)$.
6. Return hypothesis $h_{y^*,\mathcal{I},\mathcal{J}}$.

The whole procedure clearly runs in $O(m \log m)$ steps. •

The class $\mathcal{T}_{2,k}$ is a close relative of the class TREE$(2, n, p, k)$ that is considered in the paper by (Auer, Holte, and Maass 1995), but the following features of TREE$(2, n, p, k)$ make it slightly different:

- The two attributes y, x that are used at levels 0 and 1 of the decision tree are chosen from n possible attributes (which blows up the time bound by factor n^2).
- In addition to the continuous attributes there can be attributes with finite range (so-called "categorical attributes").
- There is a constant number $p \geq 2$ of classification labels. Furthermore, sample points can have missing attribute values.

Exploiting the fact that categorical attributes are easier to handle than continuous-ones, it is easy to see that our algorithm for $\mathcal{T}_{2,k}$ can be modified so that the following holds:

Corollary 4.6 *For the concept class TREE$(2, n, 2, k)$, "Minimum One-sided Disagreement" has an implementation with time bound $O(n^2 m \log m)$.*

5 Some Closing Remarks:

The dual one-sided disagreement: What happens if we want to find a hypothesis that perfectly classifies all positive examples and makes as few mistakes as possible on the negative-ones? We briefly note that this problem is trivial for axis-parallel hyper-rectangles because the best hypothesis is simply the unique smallest hyper-rectangle that contains all positive examples. For the other two concept classes, the problem has the same computational complexity as before. The key observation for proving this is that a union of (up to) k intervals is optimal iff it excludes the (up to) $k-1$ "negative intervals" with the largest total weights (among the "negative intervals" located in between "positive intervals"). This makes the dual problem sort of "isomorphic" to the problem that we discussed before.

Two-sided disagreement harder to handle than one-sided disagreement: In this paper "Minimum One-sided Disagreement" could be implemented more efficiently than "Minimum Disagreement". This no accident: any algorithm for the latter problem can be used to solve the former simply by setting the weights of negative examples in S to a sufficiently large value, respectively.
The following result by (Blum and Kalai 1998) is similar in spirit: any algorithm that PAC-learns a class $\mathcal{C}$ in the presence of two-sided classification noise can be transformed (without much loss of efficiency) into an algorithm that PAC-learns class $\mathcal{C}$ in the presence of one-sided classification noise.

The agnostic setting: It is well-known that a hypothesis class $\mathcal{H}$ is PAC-learnable in the (so-called) agnostic setting (with no a-priori assumptions about the data) iff the corresponding Minimum Disagreement Problem for $\mathcal{H}$ can be solved in polynomial time (Kearns, Schapire, and Sellie 1994). The completely analogous remark holds for "Minimum One-sided Disagreement" and "Agnostic PAC-learning with One-sided Empirical Error".

Learning from multiple-instance examples (MIL): We finally would like to add some more remarks about MIL. In this model, a so-called "r-bag" with r instances is labeled 1 iff it contains at least one positive example. The learner obtains a labeled sample of bags and should return a good classification-rule for bags. In the original model, it is assumed that the r instances in the bag are chosen according to the product distribution P^r. If we instead allow an arbitrary distribution on bags (so that the instances in a bag may exhibit statistical dependencies), the learning problem becomes harder. It follows from recent results of (Sabato and Tishby 2011) that an algorithm for "Minimum One-sided Disagreement" can be transformed (without much loss of efficiency) into an algorithm that is successful in the hard version of the MIL model. The learning problem for the hard version of the MIL model is known to be NP-hard for axis-parallel hyper-rectangles of variable dimension (according to a result by (Auer, Long, and Srinivasan 1998))[1] and for Euclidean halfspaces of variable dimension (according to results by (Diochnos, Sloan, and Turan 2011). On the positive side, our algorithms for the classes "Unions of k Intervals", "Axis-parallel Rectangles", $\mathcal{T}_{2,k}$ and TREE$(2, n, 2, k)$ can be used as sub-routines to solve the hard version of the corresponding MIL problem.

[1] The same paper presents quite efficient algorithms for learning hyper-rectangles in the classical MIL model.

References

Andrews, S. J. D. 2007. *Learning from Ambiguous Examples*. Ph.D. Dissertation, Brown University.

Angluin, D., and Laird, P. 1988. Learning from noisy examples. *Machine Learning* 2(4):343–370.

Auer, P.; Holte, R. C.; and Maass, W. 1995. Theory and applications of agnostic PAC-learning with small decision trees. In *Proceedings of the 12th International Conference on Machine Learning*, 21–29.

Auer, P.; Long, P. M.; and Srinivasan, A. 1998. Approximating hyper-rectangles: Learning and pseudorandom sets. *Journal of Computer and System Sciences* 57(3):376–388.

Blum, A., and Kalai, A. 1998. A note on learning from multiple-instance examples. *Machine Learning* 30(1):23–29.

Blumer, A.; Ehrenfeucht, A.; Haussler, D.; and Warmuth, M. K. 1989. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association on Computing Machinery* 36(4):929–965.

Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; and Stein, C. 2009. *Introduction to Algorithms*. MIT Press.

Dietterich, T. G.; Lathrop, R. H.; and Lozano-Pérez, T. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence* 89(1–2):31–71.

Diochnos, D. I.; Sloan, R. H.; and Turan, G. 2011. On multiple-instance learning learning of halfspaces. http://www.math.uic.edu/ diochnos/research/publications/dst-MIL-Halfspaces.pdf (submitted).

Ehrenfeucht, A.; Haussler, D.; Kearns, M.; and Valiant, L. 1989. A general lower bound on the number of examples needed for learning. *Information and Computation* 82(3):247–261.

Haussler, D. 1992. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation* 100(1):78–150.

Holte, R. C. 1993. Very simple classification rules perform well on most commonly used datasets. *Machine Learning* 11(1):63–90.

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

Kearns, M. 1998. Efficient noise-tolerant learning from statistical queries. *Journal of the Association on Computing Machinery* 45(6):983–1006.

Laird, P. 1988. *Learning from Good and Bad Data*. Boston: Kluwer Academic Publishers.

Maass, W. 1994. Efficient agnostic pac-learning with simple hypotheses. In *Proceedings of the 7th Annual Conference on Computational Learning Theory*, 67–75.

Maron, O., and Ratan, A. L. 1998. Multiple-instance learning for natural scene classification. In *Proceedings of the 15th International Conference on Machine Learning*, 341–349.

Sabato, S., and Tishby, N. 2011. Multi-instance learning with any hypothesis class. arXiv:1107.2021v1 [cs.LG].

Simon, H. U. 1996. General bounds on the number of examples needed for learning probabilistic concepts. *Journal of Computer and System Sciences* 52(2):239–255.

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

Weiss, S. M., and Kapouleas, I. 1989. An empirical comparison of pattern recognition, neural nets, and machine learning classification methods. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, 781–787.

Weiss, S. M., and Kulikowski, C. 1990. *Computer Systems that Learn: Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems*. Morgan Kaufmann.

Weiss, S. M.; Galen, R. S.; and Tadepalli, P. 1990. Maximizing the predictive value of production rules. *Artificial Intelligence* 45(1–2):47–71.

Zhou, Z.-H.; Jiang, K.; and Li, M. 2005. Multi-instance learning based web mining. *Applied Intelligence* 22(2):135–147.