Fast Co-Training under Weak Dependence via Stream-Based Active Learning

Ilias Diakonikolas [*1] **Mingchen Ma** [*1] **Lisheng Ren** [*1] **Christos Tzamos** [*1 2]

Abstract

Co-training is a classical semi-supervised learning method which only requires a small number of labeled examples for learning, under reasonable assumptions. Despite extensive literature on the topic, very few hypothesis classes are known to be provably efficiently learnable via co-training, even under very strong distributional assumptions. In this work, we study the co-training problem in the stream-based active learning model. We show that a range of natural concept classes are efficiently learnable via co-training, in terms of both label efficiency and computational efficiency.

We provide an efficient reduction of co-training under the standard assumption of weak dependence, in the stream-based active model, to online classification. As a corollary, we obtain efficient co-training algorithms with *error independent* label complexity for every concept class class efficiently learnable in the mistake bound online model. Our framework also gives co-training algorithms with label complexity $\tilde{O}(d\log(1/\epsilon))$ for any concept class with VC dimension d, though in general this reduction is not computationally efficient. Finally, using additional ideas from online learning, we design the first efficient co-training algorithms with label complexity $\tilde{O}(d^2\log(1/\epsilon))$ for several concept classes, including unions of intervals and homogeneous halfspaces.

1. Introduction

Supervised learning — the task of learning using a large pool of random labeled examples — is a standard machine learning paradigm that has reached substantial maturity. Even in the most basic binary classification setting, learning a hypothesis with 0-1 error ϵ requires at least $\Omega(1/\epsilon)$ labeled examples. In a number of modern machine learning applications, the scale of the problem is usually very large, which makes it very costly to obtain enough labeled data to fulfill the training requirements. The bottleneck caused by the scarcity of labeled data is especially pronounced in applications such as webpage classification, spam detection, or training large language models, where examples need to be labeled by human experts. As a result, several models and techniques have been developed to learn with fewer labeled examples. These include label propagation (Yarowsky, 1995), transductive learning (Joachims et al., 1999), and active learning (Dasgupta, 2005). Here we focus on co-training (Blum & Mitchell, 1998), one of the classical methods in the area of semi-supervised learning.

Co-training is a method used when every single example x can be partitioned into two views x_1, x_2, such that each view can solely determine the label of x. The high-level idea of co-training is that instead of trying to learn a single hypothesis h using labeled examples, one tries to find two hypotheses h_1 and h_2 in each view that maximize their agreement with each other over a large pool of unlabeled examples, and are also consistent with a small pool of labeled examples. The framework of co-training has attracted substantial interest in both theory and application; see, e.g., (Dasgupta et al., 2001; Abney, 2002; Kumar & Daumé, 2011; Balcan & Blum, 2010; Liu et al., 2014; Park & Zhang, 2003; Collins & Singer, 1999; Song et al., 2020; Li et al., 2021).

The power of co-training in classification problems, according to (Abney, 2002; Balcan & Blum, 2010; Blum & Mansour, 2017), is that under suitable assumptions any pair of hypotheses (h_1, h_2) that has large agreement and a balanced prediction over a pool of unlabeled examples must be close to either the target concept or its negation. Given this observation, $O(\log(1/\delta))$ random labeled examples suffice to break the symmetry with probability $1-\delta$. That is, *information-theoretically* the label complexity of co-training does not depend on the accuracy parameter ϵ.

Despite its low label-complexity, known co-training approaches typically suffer from the perspective of computational efficiency. Specifically, even under very strong distributional assumptions, very few concept classes are known

[*]Equal contribution [1]Department of Computer Sciences, University of Wisconsin-Madison, Madison, USA [2]University of Athens and Archimedes AI, Athens, Greece. Correspondence to: Mingchen Ma <mingchen@cs.wisc.edu>, Lisheng Ren <lren29@wisc.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

to be efficiently learnable via co-training. Intuitively, this holds because the agreement maximization method (mentioned above) is in general computationally intractable. In fact, many concept classes that are efficiently learnable in the vanilla supervised setting are not known to be efficiently learnable via co-training (with low label complexity).

The prototypical example illustrating this phenomenon is the class of halfspaces — one of the most fundamental concept classes in machine learning. As pointed out in (Blum & Mansour, 2017), although finding a consistent halfspace is computationally easy, solving the agreement maximization problem for halfspaces is NP-hard in general. Given this obstacle, co-training algorithms are usually designed using heuristics (so that they are implementable in practice).

For the class of halfspaces, the first efficient co-training algorithm was given by (Blum & Mitchell, 1998) under the conditional independence assumption, postulating that (x_1, x_2) are drawn independently given the label. That work reduces the co-training problem to the problem of learning under random classification noise (Blum et al., 1998). As we will explain in Section 1.2, such an algorithm breaks down immediately if we relax the conditional independence assumption to the more realistic λ-weak dependence assumption (Abney, 2002). The latter assumption posits that, given the label, with probability λ, x_1, x_2 are drawn independently; and with probability $1 - \lambda$ they are drawn in an arbitrarily correlated way.

(Blum & Mansour, 2017) gave the first efficient co-training algorithm in the weak dependence model for large margin halfspaces under additional distributional assumptions. Notably, although they relaxed the conditional independence assumption, (Blum & Mansour, 2017) required several additional distributional assumptions (on top of large margin). Their algorithm is based on convex programming and appears difficult to extend to other hypothesis classes. In fact, whether the class of halfspaces is efficiently learnable (in the distribution-free model) under the weak dependence assumption was posed as an open problem in their work and remains open. This motivates the following question:

Is there a natural algorithmic template to design co-training algorithms that achieve both label and computational efficiency for a broad class of concept classes?

Our key observation is that if we work in a slightly stronger learning model, where the labeled examples are obtained via adaptive label queries, we can design efficient co-training algorithms for a range of hypothesis classes (under the weak dependence assumption) with low label complexity. Formally, we work in the standard stream-based active learning model (Freund et al., 1997), defined below.

Definition 1.1. (Stream-Based Active Learning) A learning problem (X, H) contains X, the space of examples, and H, the hypothesis class over X. Each $h \in H$ is a Boolean function over X that labels each $x \in X$ by $h(x) \in \{\pm 1\}$. Let h^* be an unknown target hypothesis over X and let D be an unknown distribution over X. A learner $\mathcal{A}$ receives a stream of unlabeled examples $\{x^{(t)}\}_{t=1}^m$ drawn i.i.d. from D. When $x^{(t)}$ arrives, $\mathcal{A}$ must make an irrevocable decision whether to query (and observe) the true label $h^*(x^{(t)})$ or not. After a single pass over $\{x^{(t)}\}_{t=1}^m$, $\mathcal{A}$ outputs a hypothesis $\hat{h}$. We say that $\mathcal{A}$ learns H if for all $\epsilon, \delta > 0$ and $h^* \in H$ with probability $1 - \delta$, $\mathrm{err}(\hat{h}) := \Pr_{x \sim D}(\hat{h} \neq h^*(x)) < \epsilon$.

The model of Definition 1.1 is considered simple and practical, as it captures a variety of important real-world applications (e.g., recommendation systems and digital marketing).

Interestingly, a number of experimental works have studied co-training in the active learning setting; see, e.g. (Muslea et al., 2000; 2006; Farouk Abdel Hady & Schwenker, 2010)). In more detail, these papers focused on the *pool-based* active model — a data access model that typically requires more memory and is stronger than the streaming-based active model we study here (note that the streaming-based setting corresponds to the pool-based setting with pool of size one). Despite this extensive interest, no provable results were previously established in either of these models.

1.1. Our Contributions

Black-Box Reduction to Online Learning Classic co-training algorithms can be viewed as making queries in the following way. They set up some budget L, query the first L examples in the data stream, and do not make additional queries. With such a query strategy, every queried example plays the same role in the learning process — because after seeing L labeled examples, classic co-training algorithms will not make queries anymore. However, there are some examples that can provide more information. Suppose that we have two hypotheses (h_1, h_2) and over some example $x = (x_1, x_2)$, $h_1(x_1) \neq h_2(x_2)$; then at least one of the h_i makes a wrong prediction on x. If we make a query on x and update the h_i that makes a mistake, then hopefully h_i will get closer to the target hypothesis. On the other hand, mistake-bound online learning (Blum, 2005) is a well-studied field that designs prediction algorithms that minimize the total number of incorrect predictions over a stream of examples (mistake-bound model). A natural idea is to use an online learning algorithm as a subroutine to design co-training algorithms. This potentially provides us with a way to control the number of queries while maintaining computational efficiency.

Building on this observation, in Theorem 3.2, we give a black-box reduction from the problem of co-training with label queries (Definition 1.1) to the problem of online learning. Consider a co-training problem (X, H) that

satisfies λ-weak dependence. If the hypothesis class H can be learned in the mistake-bound model by an online learner $\mathcal{A}$ with mistake bound M, then we can use $\mathcal{A}$ to design a co-training algorithm that makes $\tilde{O}(M/\alpha)$ label queries — importantly, the label complexity does not depend on the accuracy parameter ϵ — over a stream of $\tilde{O}(\mathrm{poly}(M, 1/\alpha, 1/(\lambda\epsilon)))$ unlabeled examples, and with high probability learns a hypothesis with error at most ϵ. Here $\alpha := \min\{\Pr_{x\sim D}(y(x) = 1), \Pr_{x\sim D}(y(x) = -1)\}$. In particular, the running time of the co-training algorithm in each iteration is the same as that of the online learner $\mathcal{A}$.

As a corollary, any hypothesis class that is known to be efficiently learnable in the mistake-bound online learning model, can be efficiently learned via co-training with *error independent* label complexity. In particular, for the class of γ-margin halfspaces, the perceptron algorithm implies a fast halfspace co-training algorithm with only $\tilde{O}(1/(\gamma^2\alpha))$ label queries. In comparison, the prior work (Blum & Mansour, 2017) gives a co-training algorithm (without queries) for γ-margin halfspaces with additional restrictions; namely, it requires homogeneity and zero-mean marginal distribution. Our algorithm does not impose further assumptions on the marginal distribution and runs in linear time per iteration.

Co-Training beyond Finite Mistake Bound The learnability of a hypothesis class in the mistake-bound model is characterized by its Littlestone dimension (Littlestone, 1988). On the other hand, many simple hypothesis classes, such as unions of intervals, have finite VC-dimension but infinite Littlestone dimension. This suggests that in general we cannot hope to achieve error-independent label complexity for arbitrary VC classes via our reduction. Intriguingly, as an implicit corollary of our approach, we show the following (Theorem 4.1): for any hypothesis class with VC dimension d, we can obtain a co-training algorithm (under λ-weak dependence) that makes $\tilde{O}(d\log(1/(\lambda\epsilon))/\alpha)$ label queries over a stream of $\mathrm{poly}(d, 1/(\lambda\epsilon), 1/\alpha)$ unlabeled examples. Our argument combines the technique of constructing ϵ-covers with unlabeled examples and the well-known Halving algorithm. This label query upper bound still achieves an exponential improvement on the label complexity, compared to the standard $\tilde{O}(d/\epsilon)$ label complexity of passive supervised learning. We note however that the running time of such a generic algorithm is not polynomial in general. Finally, we remark that the label complexities of Theorem 3.2 and Theorem 4.1 have a very mild dependence on the parameter λ. This is especially beneficial, since in practical settings, λ is usually taken to be a very small quantity (since λ-weak dependence is a strong assumption).

Since it is in general impossible to design a single computationally efficient learning algorithm that works for every hypothesis class, in the third part of the work we focus on designing efficient co-training algorithms for two concrete classes — k-unions of intervals and homogeneous halfspaces (both have infinite Littlestone dimension and have label complexity $\Omega(1/\epsilon)$ in the standard active learning model (Dasgupta, 2005)). For the class of k-unions of intervals, we give an efficient co-training learner with label complexity $\tilde{O}(k^2\log(1/(\lambda\epsilon))/(\lambda\alpha))$. For the class of homogeneous halfspaces (without a margin assumption), we show the following: assuming that the marginal distribution on one view is approximately symmetric, we given an efficient co-training learner with label complexity $\tilde{O}(d^2\log(1/\epsilon))$. The analysis of our algorithms might be of independent interest.

1.2. Related Work

Theoretic Analysis for Co-Training The theory of co-training has been developed since the late 90s, (Blum & Mitchell, 1998; Abney, 2002; Dasgupta et al., 2001; Balcan & Blum, 2010; Darnstädt et al., 2014; Balcan et al., 2004; Blum & Mansour, 2017). Unlike our algorithmic framework, previous algorithmic templates are designed in a "boosting" style. That is, they use labeled examples to train a weak hypothesis h_1 over X_1 and use h_1 to label examples in X_2; finally, they use learning algorithms with very strong guarantees to learn a good hypothesis over X_2 using the "unreliable" labels. For example, in (Blum & Mitchell, 1998), given a weak hypothesis h_1 in X_1, they label a random (x_1, x_2) by $h_1(x_1)$; under 1-weak dependence, the label of a random x_2 can be seen as corrupted by random classification noise so that they can use a robust learner to find a good hypothesis over X_2. However, once we relax the assumption to λ-weak dependence, such a method fails immediately, because without having a good enough initial hypothesis h_1, the above labeling method could make some x_2 have adversarially corrupted labels, thus making it impossible to recover the target hypothesis in X_2.

Label Complexity of Active Learning Our work could be viewed as learning in the active learning model with additional co-training assumptions. Active learning is another learning model which targets reducing the label complexity of supervised learning. Unlike co-training, which has error-independent label complexity (information-theoretically), it has been pointed out by (Dasgupta, 2005; Hanneke, 2012; Hanneke & Yang, 2015) that active learning in the worst case has the same label complexity as passive learning. Even for simple classes such as unions of intervals and halfspaces in more than 2 dimensions (even with large margin), there exist distributions for which any learning algorithm that outputs a hypothesis with error $O(\epsilon)$ must make $\Omega(1/\epsilon)$ label queries. Thus, designing efficient learning algorithms with $o(1/\epsilon)$ label complexity is both important and challenging.

Application of Online Learning to Other Models Online learning is a well-studied model that focuses on how to learn in a sequential adversarial setting. Although fully

adversarial learning is unrealistic in practice, the idea of online learning usually has important applications in other learning problems, such as boosting algorithms (Freund & Schapire, 1997), learning in games (Fudenberg & Levine, 1998), and reinforcement learning (Agarwal et al., 2019). Our work also applies the idea of online learning to a seemingly unrelated task, that of co-training. A seemingly similar application of online learning is the well-known "online to PAC" conversion (Littlestone, 1989). Specifically, if a class H can be learned in the mistake-bound model with M mistakes, we can convert the online learner to a PAC learner with sample complexity $N = O(\frac{M}{\epsilon}\log(\frac{M}{\delta}))$. That is, one needs to use N *labeled examples* during the learning process (because in each round the online learner must receive feedback and know the true label of the example used in every single iteration). On the other hand, the focus of Theorem 3.2 is the number of *labeled examples* used in the training process. The importance of the guarantee in Theorem 3.2 is that, with the help of adaptivity, the resulting label complexity $\tilde{O}(M/\alpha)$ is *independent of the accuracy parameter* ϵ.

Active Learning for Co-Training Finally, we point out that the idea of combining co-training and pool-based active learning was previously empirically studied in a number of works (Muslea et al., 2000; 2006; Farouk Abdel Hady & Schwenker, 2010). Several learning and query strategies have been empirically shown to be useful in obtaining hypotheses with low error — to the best of our knowledge, our work is the first one that introduces a reduction from co-training to online learning and provides provable guarantees for both label complexity and computational complexity. Last, we mention that (Wang & Zhou, 2008) claim that a different pool-based active learning strategy can be applied to obtain co-training algorithms with $\tilde{O}(d\log(1/\epsilon))$ label complexity for any hypothesis class with VC dimension d. However, as explained in Appendix D, the distributional assumption made in their work contradicts the co-training assumption, i.e., no distribution satisfies their requirements.

2. Notations and Preliminaries

Definition 2.1. (Co-Traing Assumption) We say a learning problem (X, H) satisfies the co-training assumption if it satisfies the following requirement. There are spaces of examples X_1, X_2 such that each $x \in X$ can represented as the form (x_1, x_2), where $x_1 \in X_1, x_2 \in X_2$. Furthermore, there are hypothesis classes H_1 over X_1 and H_2 over X_2 such that for every $h \in H$ there exist $h_1 \in H_1$ and $h_2 \in H_2$ such that $h_1(x_1) = h(x) = h_2(x_2)$ for every example x.

For simplicity, if we do not specify it when we say (X, H) satisfies the co-training assumption, we mean $X = X_1 = X_2$ and $H = H_1 = H_2$.

Let (X, H) be a learning problem that satisfies the co-training assumption and let D be the marginal distribution over X. For $i \in [2]$, we denote by D_i the marginal distribution of D over X_i. For $y \in \{\pm 1\}$, denote by D^y the distribution of x conditioned on $h^*(x) = y$ and denote by α_y the probability that a random example x has label y and $\alpha := \min\{\alpha_1, \alpha_{-1}\}$. For $y \in \{\pm 1\}$ and $i \in [2]$, denote by D_i^y the marginal distribution of x_i of distribution D^y.

Definition 2.2. (Weak Dependence Assumption) Let (X, H) be a learning problem that satisfies the co-training assumption. Let D be the marginal distribution over X. For $\lambda \in [0, 1]$, we say that distribution D satisfies λ-weak dependence if for every $y \in \{\pm 1\}$ and for every $(x_1, x_2) \in D^y$, $D^y(x_1, x_2) \geq \lambda D_1^y(x_1) D_2^y(x_2)$.

Let $f = (f_1, f_2)$ be a pair of hypotheses, where $f_1 : X_1 \to \{\pm 1\}$ and $f_2 : X_2 \to \{\pm 1\}$. We define the unlabeled error of f as $\mathrm{unl}(f) := \Pr_{x\sim D}(f_1(x_1) \neq f_2(x_2))$. Sometimes we will also treat a hypothesis pair $f = (f_1, f_2)$ itself as a hypothesis. That is, for some $x = (x_1, x_2)$, if for some $y \in \{\pm 1\}$, $f_1(x_1) = f_2(x_2) = y$, then $f(x) = y$. Otherwise, f assigns an arbitrary label to x. Based on this, we define the distance and error of hypotheses. Let $h \in H$ be some hypothesis. We define the distance between f and h as $d(f, h) := 1 - \Pr_{x\sim D}(f_1(x_1) = f_2(x_2) = h(x))$. In particular $\mathrm{err}(f) := d(f, h^*)$. Throughout the paper, we will also use the notation $\hat{\mathrm{unl}}, \hat{d}, \hat{err}$ to denote the corresponding empirical quantity when evaluated at some sample set S.

In this paper, we are also interested in learning geometric concepts when $X = \mathbb{R}^d$. We will use $\langle\cdot,\cdot\rangle$ to denote the inner product and use $\|\cdot\|$ to denote the ℓ_2 norm. Let $H = \{\mathrm{sign}(\langle w^*, x\rangle) \mid w^* \in \mathbb{R}^d\}$ be the class of half spaces in $\mathbb{R}^d$. A marginal distribution D satisfies the γ-margin assumption if $\frac{|\langle w^*, x\rangle|}{\|w^*\|\|x\|} \geq \gamma$ holds with probability 1, where w^* is the target. When (X, H) satisfies the co-training assumption, we say the distribution D satisfies γ-margin assumption if for $i \in [2]$, the margin assumption over X_i satisfies the γ-margin assumption. Finally, we state the following lemma, which will be heavily used in this paper.

Lemma 2.3. *[Restatement of Lemma 13 in (Blum & Mansour, 2017)] Let (X, H) be a learning problem that satisfies the co-training assumption and D be the marginal distribution over X that satisfies λ-weak dependence. Let $f = (f_1, f_2)$ be a pair of hypotheses over X. For every $\epsilon \in (0, 1)$, if $\mathrm{unl}(f) < \epsilon$, then at least one of the following holds. $d(f, 1) < 4\epsilon/\lambda, d(f, -1) < 4\epsilon/\lambda, \mathrm{err}(f) < 4\epsilon/\lambda, \mathrm{err}(-f) < 4\epsilon/\lambda$.*

3. Co-Training via Online Classification

Here we present our first main result which shows that with the power of label queries, we can efficiently transform an online classification algorithm into a co-training algorithm,

Algorithm 1 REDUCTION($\mathcal{A}_1, \mathcal{A}_2$) (Efficient Black-Box Reduction from Co-Training to Online Learning)

Input: Online learning algorithm $\mathcal{A}_i$ for H_i, $i \in \{1, 2\}$, sample access to distribution D, a label oracle, accuracy parameter $\epsilon \in (0, 1)$, confidence parameter $\delta \in (0, 1)$
Output: A hypothesis $\hat{h}$ with error $\mathrm{err}(\hat{h}) < \epsilon$ with probability at least $1 - \delta$
for $t = 1, 2, \ldots$ **do**
  Compute h_i^t, the hypothesis used by $\mathcal{A}_i$ in round t.
  Draw example $x^{(t)} = (x_1^{(t)}, x_2^{(t)}) \sim D$
  if $h_1^t(x_1^{(t)}) \neq h_2^t(x_2^{(t)})$ **then**
    Query the label of $x^{(t)}$ and enter the next round.
  if $\mathcal{A}_1, \mathcal{A}_2$ have continuously agreed on $n = \mathrm{poly}(1/(\lambda\epsilon), \log(M/\delta))$ rounds **then**
    if $\forall y \in \{\pm 1\}$, more than $\Omega(\epsilon)$ fraction of the agreed examples are labeled by y **then**
      Return $\hat{h} = (h_1^t, h_2^t)$
    else
      Query the label of $x^{(t)}$ and enter the next round if $h^*(x^{(t)}) \neq h_1^t(x^{(t)})$
  Delete $x^{(t)}$ from the memory and enter the next round

using very few label queries. To start with, we remind the reader of the mistake-bound model for online learning.

Definition 3.1. (Mistake Bound Online Learning Model) Let X be a space of example and H be a class of hypothesis over X. Let $h^* \in H$ be the unknown target hypothesis. An online classification algorithm $\mathcal{A}$ works in the following way. In round t, $\mathcal{A}$ maintains some hypothesis h^t, some $x^{(t)}$ selected by an adversary is presented to $\mathcal{A}$ and $\mathcal{A}$ makes a prediction $h^t(x^{(t)})$ for the label $h^*(x^{(t)})$ and sees $h^*(x^{(t)})$. When $h^t(x^{(t)}) \neq h^*(x^{(t)})$, we say $\mathcal{A}$ makes a mistake. We say $\mathcal{A}$ runs in time T with a mistake bound M for hypothesis class H if for every $h^* \in H$ and every possible sequence of examples, $\mathcal{A}$ makes at most M mistakes and makes each update in time T.

Theorem 3.2. *Let (X, H) be a learning problem that satisfies the co-training assumption, and let D be a distribution over X that is λ-weak dependence. If there exists an online learning algorithm $\mathcal{A}_1$ for H_1 and $\mathcal{A}_2$ for H_2 such that $\mathcal{A}_1, \mathcal{A}_2$ run in time T with a mistake bound M, then there is a learning algorithm $\mathcal{A}$ such that for $\epsilon, \delta \in (0, 1)$, it draws $m = \tilde{O}(\mathrm{poly}(M, 1/\alpha, 1/(\lambda\epsilon)))$ unlabeled examples, makes $\tilde{O}(M/\alpha)$ label queries runs in time $O(Tm)$ and outputs a hypothesis $\hat{h}$ such that with probability at least $1 - \delta$, $\mathrm{err}(\hat{h}) < \epsilon$.*

Before proving Theorem 3.2, we give an overview of the intuition behind Algorithm 1. Algorithm 1 runs $\mathcal{A}_1, \mathcal{A}_2$ simultaneously over the two views. When a random example $x^{(t)}$ arrives, we use $\mathcal{A}_1, \mathcal{A}_2$ to predict its label. If the predictions are different, we query its label and make an update because one of the two algorithms is guaranteed to make a mistake. If the predictions are the same, we check if the two hypotheses used by $\mathcal{A}_1, \mathcal{A}_2$ have agreed on many examples. By Lemma 2.3, we know that if this is the case, then the current hypothesis pair (h_1^t, h_2^t) is close to either $\pm h^*$ or ± 1. In particular, if (h_1^t, h_2^t) does not label too many examples by $+1$ or by -1, then this pair of hypotheses must be close to the target and we can safely output it. If (h_1^t, h_2^t) is very close to a constant hypothesis, then we can simply continuously request labels of examples because in expectation after $O(1/\alpha)$ rounds we will see an example where both algorithms make a mistake. When we do not make a query label for $x^{(t)}$ or the queried label agrees with both predictions of $\mathcal{A}_1, \mathcal{A}_2$, we simply delete this example from the memory and run $\mathcal{A}_1, \mathcal{A}_2$ as if $x^{(t)}$ has not appeared, so that they do not change the hypothesis. We notice that after making $2M$ updates, Algorithm 1 is guaranteed to stop because both $\mathcal{A}_1, \mathcal{A}_2$ will not make any more mistakes, and to make one update we only need to make $O(1/\alpha)$ queries. Thus, Algorithm 1 only makes $O(M/\alpha)$ label queries over a stream of unlabeled examples. Furthermore, the running time of Algorithm 1 is the same as that of $\mathcal{A}_1, \mathcal{A}_2$, since in every round it only requests labels or makes updates using $\mathcal{A}_1, \mathcal{A}_2$.

Proof of Theorem 3.2. We start by showing the correctness of Algorithm 1. Notice that during the execution of Algorithm 1, the hypothesis pair (h_1^t, h_2^t) will not change until some example x is queried and at least one of h_i^t, $i \in [2]$, makes a mistake on x. Now we consider a fixed pair of hypotheses $h^t = (h_1^t, h_2^t)$. Let S be a set of $n = \mathrm{poly}(1/\epsilon, \log(M/\delta))$ unlabeled examples drawn i.i.d. from D. By Hoeffding's inequality, we have $\Pr_{S\sim D^n}\left(\left|\hat{\mathrm{unl}}(h^t) - \mathrm{unl}(h^t)\right| \geq \frac{\lambda\epsilon}{4}\right) \leq O(\frac{\delta}{M})$. Thus, with probability at least $1 - O(\delta/M)$, unless $\mathrm{unl}(h^t) < \lambda\epsilon$, h^t will not continuously agree on n randomly drawn examples. By Lemma 2.3, we know that h^t is ϵ-close to either $\pm h^*$ or ± 1. For $y \in \{\pm 1\}$, denote by $b_y^t := \Pr_{x\sim D}(h^t(x) = y)$. By Hoeffding's inequality again, for a set of n randomly unlabeled samples S, we have $\Pr_{S\sim D^n}\left(\left|\frac{\sum_{x\in S} 1(h^t(x)=y)}{n} - b_y^t\right| \geq \epsilon\right) \leq O(\frac{\delta}{M})$. In particular, if h^t is close to $\pm h^*$, then for every $y \in \{\pm 1\}$, we have $\Pr_{x\sim D}(h^t(x) = y) \geq \alpha - \epsilon \geq \alpha/2$. Thus, with probability at least $1 - O(\delta/M)$, for every $y \in \{\pm 1\}$, more than $\Omega(\epsilon)$ fraction of the agreed examples are labeled by y and h^t will be output. On the other hand, if h^t is close to ± 1, then with probability at least $1 - O(\delta/M)$, $\exists y \in \{\pm 1\}$ such that h^t only labels $O(\epsilon)$ fraction of them to be y, and thus h^t will not be output. We remark that here, for simplicity, we assume when h^t is close to $\pm h^*$, it is close to h^*, because we can check this by testing the empirical error of h^* over $O(\log(1/\delta))$ random labeled examples, and this will not affect the performance of our algorithm.

So far, our analysis works for a fixed hypothesis pair. To finish the proof of the correctness, we will show that throughout the execution of Algorithm 1, there are at most $2M$ hypothesis pairs and at least one of them is close to h^*. Denote by $x^{(t_1)}, \ldots, x^{(t_k)}$ the sequence of examples, where either $\mathcal{A}_1$ or $\mathcal{A}_2$ makes an update throughout the execution of Algorithm 1. We notice that these examples are the only examples that are in memory when we use $\mathcal{A}_i$ to compute hypothesis h_i^t for $i \in [2]$. In other words, for $i \in [2]$, the performance of $\mathcal{A}_i$ during the execution of Algorithm 1 is the same as that of $\mathcal{A}_i$ when it runs over the sequence of the examples $x^{(t_1)}, \ldots, x^{(t_k)}$. According to Algorithm 1, for each example in this sequence, either $\mathcal{A}_1$ or $\mathcal{A}_2$ will make a mistake. Since both $\mathcal{A}_1$ and $\mathcal{A}_2$ have a mistake bound of M, if $\mathcal{A}_i$ makes M mistakes in the sequence then after that the hypothesis computed by $\mathcal{A}_i$ will not make any mistakes. According to Algorithm 1, for each example in this sequence, either $\mathcal{A}_1$ or $\mathcal{A}_2$ will make a mistake. Thus, the length of the sequence is at most $2M$, and only at most $2M$ pairs of hypotheses are used by Algorithm 1. Furthermore, if $\mathcal{A}_1$ and $\mathcal{A}_2$ both make M mistakes, then after querying $x^{(t_k)}$, both $\mathcal{A}_1$ and $\mathcal{A}_2$ will not make any mistakes; thus, there must be at least one pair of hypotheses that is close to h^*. Now, if a hypothesis pair h^t is close to h^*, then with probability $1 - O(\delta/M)$, it will be output. If a hypothesis pair h^t is not close to h^t, then with probability $1 - O(\delta/M)$ it will not be output. By the union bound, we know that with probability at least $1 - O(\delta)$, Algorithm 1 will output a hypothesis $\hat{h}$ with error at most ϵ. This finishes the proof of correctness.

We next upper bound the label complexity and the number of unlabeled examples used by Algorithm 1. We start with the label complexity. We make a label query when either h^t disagrees on some unlabeled example or h^t is close to ± 1. In the first case, we make a single query. In the second case, without loss of generality, we assume $d(h^t, 1) < \epsilon$. Since a random example has a probability at least α to be negative, we know that with probability at least $\alpha - \epsilon \geq \alpha/2$, it will be misclassified by h^t. In expectation, after making $O(1/\alpha)$ label queries, we will see an example misclassified by h^t and update h^t. By Markov's inequality, no matter which case we are in, with probability at least $1 - O(\delta/M)$, we will make at most $\tilde{O}(1/\alpha)$ label queries to make an update. As we discussed above, throughout the execution of Algorithm 1, we will in total make $2M$ updates, which implies the with probability at least $1 - O(\delta)$ the total number of label queries made by Algorithm 1 is at most $\tilde{O}(M/\alpha)$. On the other hand, the total number of unlabeled examples is the label complexity plus the number of examples used to estimate $\mathrm{unl}(h^t)$, which is $n = O(\mathrm{poly}(1/(\lambda\epsilon)), \log(M/\delta))$ between a single update. Thus, the total number of unlabeled examples used by Algorithm 1 is $m = O(\mathrm{poly}(M, 1/\alpha, 1/(\lambda\epsilon), \log 1/\delta))$. Finally, the running time of Algorithm 1 directly follows the fact that it runs in time $O(T)$ for every single iteration.

□

As corollaries of Theorem 3.2, we obtain the first efficient co-training algorithms for a broad class of hypotheses under the weak dependence assumption. One of the most interesting results is on learning margin halfspaces under weak dependence. Before this work, the only known efficient co-training halfspaces algorithm (Blum & Mansour, 2017) under the weak dependence assumption, not only assumes D satisfies the margin assumption but also makes non-trivial structural assumptions over D. Furthermore, the algorithm is fairly complicated and needs to solve polynomially many large convex programs. On the contrary, based on Algorithm 1, we give the first *linear time* co-training algorithm for learning margin halfspaces under weak dependence *without making any assumption on the marginal distribution*.

Corollary 3.3. *Let $X = S^{d-1}$ and H be the class of halfspaces over X. Assume (X, H) satisfies the co-training assumption. Let D be any distribution over X that satisfies λ-weak dependence and γ-margin assumption. There is an algorithm such that for $\delta, \epsilon \in (0, 1)$, with probability at least $1 - \delta$ it draws $m = \tilde{O}(\mathrm{poly}(1/\gamma, 1/\alpha, 1/(\lambda\epsilon)))$ unlabeled examples from D makes $\tilde{O}(1/(\gamma^2\alpha))$ label queries, runs in $O(dm)$ time, and outputs a hypothesis $\hat{h}$ with $\mathrm{err}(\hat{h}) < \epsilon$.*

Proof. It is well-known that if a sequence of example $(x^{(t)})_{t=1}^{\infty} \subseteq \mathbb{R}^d$ satisfies the margin assumption $\frac{\langle w^*, x^{(t)} \rangle}{\|w^*\| \left\|x^{(t)}\right\|} \geq \gamma$, then the perception update $w^{(t+1)} = w^{(t)} + y(x^{(t)})x^{(t)}$, when $x^{(t)}$ is predicted incorrectly gives an online classification algorithm with mistake bound $O(1/\gamma^2)$. Each update takes time $O(d)$ to implement. Corollary 3.3 follows directly by Theorem 3.2. □

We also summarize additional implications of Theorem 3.2 in Table 1. We point out that before this work no efficient co-training algorithms were known for these classes under weak dependence.

Hypothesis Class	Label Complexity	Time
Disjunctions	$\tilde{O}(n/\alpha)$	$O(n)$
Conjunctions	$\tilde{O}(n/\alpha)$	$O(n)$
LDeep-Decision List	$\tilde{O}(nL/\alpha)$	$O(nL)$
k-Term-DNF	$\tilde{O}(n^{O(k)}/\alpha)$	$n^{O(k)}$
k-CNF	$\tilde{O}(n^{O(k)}/\alpha)$	$n^{O(k)}$

Table 1. Label Complexity and Running Time per Iteration for Several Hypothesis Classes over Boolean Domain $\{0, 1\}^n$.

We notice that there is a dependence on α, the bias of the target hypothesis, in our label complexity. However,

information-theoretically, such a dependence is not necessary. The dependence on α is not unique to our algorithmic framework. We point out that though not formally stated, such a dependence also implicitly exists in previous algorithmic frameworks such as (Blum & Mitchell, 1998). These frameworks assume a weakly useful hypothesis is trained over a small pool of randomly drawn labeled examples L. A weakly useful hypothesis is a hypothesis h such that for every $y \in \{\pm 1\}$ an example with label y has probability at most $1/2 - \beta$ that is labeled incorrectly for some constant $\beta > 0$. In particular, unless the hypothesis class has a good structure if $|L| < o(1/\alpha)$, then every labeled example in L will have the same label and such a weakly useful hypothesis cannot be obtained. It is unclear if such a dependence is necessary to obtain a computationally efficient algorithm, and we leave it as an important open question.

4. Co-Training beyond Finite Mistake Bound

Although Algorithm 1 shows that many hypotheses classes can be efficiently learned via co-training with only an error-independent number of label queries, it requires the existence of an online learning algorithm with finite mistake bound. However, not every hypothesis class has such a property. In this section, we show that the idea of online learning still leads to co-training algorithms for hypothesis classes with finite VC dimension (even if they are not online learnable under the mistake-bound model) with low label complexity.

4.1. Learning General VC-Classes in Exponential Time

We first show that the idea of using online learning information theoretically still exponentially improves the label complexity of learning a general VC class over the label complexity in the passive learning setting (the detailed proof for Theorem 4.1 is deferred to Appendix A).

Theorem 4.1. *Let (X, H) be a learning problem that satisfies the co-training assumption, where $VC(H) = d$. Let D be a distribution over X that satisfies λ-weak dependence. There is an (exponential time) algorithm such that for $\epsilon, \delta \in (0,1)$, it draws $m = \mathrm{poly}(d, 1/(\lambda\epsilon), \log(1/\delta))$ unlabeled examples, makes $\tilde{O}(d \log(1/(\lambda\epsilon))/\alpha)$ many label queries, and outputs a hypothesis $\hat{h}$ with error $\mathrm{err}(\hat{h}) < \epsilon$.*

Theorem 4.1 could be seen as an implicit corollary of Theorem 3.2. It is well known that if H is finite then under the mistake bound model, the Halving algorithm can learn H with $O(\log(|H|))$ mistakes. Although H in general is infinite, according to (Hanneke & Yang, 2015), we can draw $\mathrm{poly}(d/\eta)$ unlabeled examples to construct an η-cover C for H with size $\tilde{O}((d/\eta)^d)$. This guarantees that for all $h \in H$ there exists $c \in C$ such that $d(h, c) < \eta$. Importantly, if we set $\eta = \mathrm{poly}(\epsilon, \lambda, d)$, then with probability at least $1/2$, there exists $c^* \in C$ that agrees with h^* on every unlabeled example we use. Thus, we can simply implement Algorithm 1 with the Halving algorithm by assuming the target hypothesis is in C. In expectation after repeating Algorithm 1 several times, we finally learn a good hypothesis with label complexity $\tilde{O}(\log|C|/\alpha) = \tilde{O}(d \log(1/(\lambda\epsilon))/\alpha)$.

Since the Halving algorithm in general is not computationally efficient, in the rest of the paper, we will focus on developing efficient co-training algorithms for two concrete hypothesis classes: k-unions of intervals halfspaces (both have $\Omega(1/\epsilon)$ label complexity in the standard active model).

4.2. Co-Training k-Unions of Intervals

Theorem 4.2. *Let (X, H) be a learning problem that satisfies the co-training assumption, where $X = \mathbb{R}$ and H is the class of k-union of intervals over $\mathbb{R}$. i.e. for each $h \in H$, there exists k intervals I_i, $i \in [k]$ such that $h(x) = 1$ if and only if $x \in \bigcup_{i\in[k]} I_i$. Let D be a distribution over X that satisfies λ-weak dependence. There is an algorithm such that for $\epsilon, \delta \in (0,1)$, with probability $1-\delta$ it draws $m = \mathrm{poly}(k, 1/(\lambda\epsilon), \log(1/\delta))$ unlabeled examples, makes $\tilde{O}(k^2 \log(1/(\lambda\epsilon)) + k/\alpha)$ many label queries, runs in $\mathrm{poly}(m)$ time and outputs a hypothesis $\hat{h}$ with error $\mathrm{err}(\hat{h}) < \epsilon$.*

We present Algorithm 2, a sketch version of our learning algorithm (due to space limitations). Algorithm 4, the detailed version and the proof of Theorem 4.2, can be found in Appendix B. Here we give the intuition behind Algorithm 2. For simplicity, we assume $\lambda = 1$. Initially, Algorithm 2 will sample $\tilde{O}(k/\alpha)$ examples and query their labels. We use these examples to construct an initial hypothesis (h_1^0, h_2^0) over the two views. We show in Claim B.1, by the VC inequality, any hypothesis class that is consistent with these labeled examples will be far from the constant hypothesis. Throughout the execution of Algorithm 2, we will make a small modification for h_i^t when it makes an incorrect prediction on $x_i^{(t)}$. Since h_i^t is a union of at most k intervals, the region where positive examples are misclassified by h_i^t is also a union of $O(k)$ intervals. For simplicity, we denote this region by $\cup_j R_{ij}^t$. According to the 1-dependent assumption we made, when some $x_i^{(t)}$ with a positive label is misclassified by h_i^t, we can treat it as a random example sampled from D_i^+ conditioned on $\cup_j R_{ij}^t$. Together with the modification method used by Algorithm 2, we can show that if such $x_i^{(t)} \in R_{ij}^t$, then with constant probability $\Pr(R_{ij}^t)$ will drop by a constant factor. This implies that after seeing such an example $x_i^{(t)}$ with constant probability, $\mathrm{err}_+(h_i^t)$ will drop by a factor of $(1 - 1/O(k))$. A similar analysis holds if a negative example $x_i^{(t)}$ is misclassified. This intuitively gives that after making $O(k \log(1/(\lambda\epsilon)))$ mistakes

Algorithm 2 LEARNK-INTERVAL (Efficient co-training k intervals)

Input: A sample access to D, a label query oracle, accuracy parameter $\epsilon \in (0,1)$, confidence parameter $\delta \in (0,1)$
Output: A hypothesis $\hat{h}$ with error $\mathrm{err}(\hat{h}) < \epsilon$ with probability at least $1-\delta$
$n = \mathrm{poly}(1/(\lambda\epsilon), \log(1/\delta))$
Draw $\tilde{O}(k/\alpha)$ examples from D and query their labels
Over X_1, X_2, construct initial hypothesis pair (h_1^1, h_2^1), write it as $h_i^1 = \bigcup_{j=1} I_{ij}^1$. ($I_{ij}^1$ ordered by position)
$\mathrm{COUNT}_T \leftarrow 0$, $\mathrm{COUNT}_U \leftarrow 0$
for $t = 1, 2, \ldots$ **do**
  Draw example $x^{(t)} = (x_1^{(t)}, x_2^{(t)}) \sim D$
  if $\mathrm{COUNT}_T < n$ **then**
    Increase COUNT_U by 1 , if $h_1^t(x_1^{(t)}) \neq h_2^t(x_2^{(t)})$.
    Increase COUNT_T by 1 and enter next round
  if $\mathrm{COUNT}_T \geq n$ and $\frac{\mathrm{COUNT}_U}{\mathrm{COUNT}_T} < \frac{\lambda\epsilon}{8}$ **then**
    Return $\hat{h} = (h_1^t, h_2^t)$.
  {Check if the current hypothesis is good enough}
  if $h_1^t(x_1^{(t)}) \neq h_2^t(x_2^{(t)})$ **then**
    Query the label of $x^{(t)}$
    if $y(x^{(t)}) = 1$ and is misclassified by h_i^t **then**
      Find j such that $x_i^{(t)}$ lies between I_{ij}^t and $I_{i(j+1)}^t$ and modify h_{ij}^t as follows;
      Enlarge one of $I_{ij}^t, I_{i(j+1)}^t$ by setting the boundary as $x_i^{(t)}$ if the modification does not cause inconsistency, otherwise, create a new interval $\{x_i^{(t)}\}$.
    if $y(x^{(t)}) = 0$ and is misclassified by h_i^t **then**
      Find the interval I_{ij}^t that contains $x^{(t)}$ and modify h_i^t by dividing I_{ij}^t into two intervals.
    $\mathrm{COUNT}_T \leftarrow 0$, $\mathrm{COUNT}_U \leftarrow 0$.

$\mathrm{err}(h_i^t)$ will be roughly $O(\lambda\epsilon)$ and, according to Lemma 2.3, it will be output. However, such a direct analysis is not fully correct, because when we make a modification that tends to decrease the $\mathrm{err}_+(h_i^t)$, $\mathrm{err}_-(h_i^t)$ might increase due to the modification, and similarly $\mathrm{err}_+(h_i^t)$ might increase when we try to decrease $\mathrm{err}_-(h_i^t)$. We show in Appendix B that by employing the structure of the hypothesis class, our update method can ensure that these bad events happen at most $O(k)$ times before we output a good hypothesis, and thus we can still achieve an exponential improvement on the label complexity.

4.3. Co-Training Homogeneous Halfspaces

Finally, we design an efficient algorithm for learning homogeneous halfspaces (without a margin assumption) under weak independence and approximate symmetric assumption. In Corollary 3.3, the algorithm works only when the target halfspace has a margin γ, while in this section, we do not make such an assumption. As an alternative, we require that one of the marginal distributions (without loss of generality D_1 of x_1) satisfies an approximate reflective symmetry assumption. Namely, we need $D_1(x_1)/D_1(-x_1)$ to be always bounded between $[\alpha, 1/\alpha]$ for some $\alpha \leq 1$. Under this assumption, we have the following theorem (the detailed proof is deferred to Appendix C).

Theorem 4.3. *Let (X, H) be a learning problem where $X = \mathbb{R}^d \times \mathbb{R}^d$ and H be the class of homogeneous halfspaces that satisfies the co-training assumption. Let D be a distribution over X that satisfies λ-weak dependence and let D_1 be the marginal distribution of x_1 satisfies the α-reflective symmetry, namely, for any $x_1 \in X_1$, $D(x_1)/D(-x_1) \in [\alpha, 1/\alpha]$ for some $\alpha \leq 1$. There is an algorithm such that for $\epsilon, \delta \in (0,1)$, with probability at least $1-\delta$ it draws $m = \mathrm{poly}(d, 1/\epsilon, 1/\alpha, 1/\lambda)$ unlabeled examples, makes $\tilde{O}(d^2 \log(1/\epsilon))$ label queries and returns a hypothesis $\hat{h}$ such that $\mathrm{err}(\hat{h}) < \epsilon$.*

We give the intuition behind Theorem 4.3 here. For simplicity, we assume $\alpha = 1$. Let us assume that for $i \in [2]$, marginal distribution D_i in the statement also satisfies the following property, for every vector $w_i \in X_i$, with probability at least β, a random example drawn from D_i has a margin at least γ with respect to w_i. This implies that if we can learn some w_i that can correctly classify every example x_i that has a margin γ with respect to it, then we can correctly label β fraction of the distribution D_i. Fortunately, under the mistake bound, a modified version of the perceptron algorithm developed by (Blum et al., 1998) can learn such a w_i with $\mathrm{poly}(1/\gamma)$ mistakes. Thus, by implementing the modified perceptron algorithm via Algorithm 1, we are able to use $\mathrm{poly}(1/\gamma)$ label queries to learn such w_i up to a small error. This suggests if after removing the region $\{x_i \in X_i \mid |\langle w_i, x_i\rangle| \geq \gamma\}$ from D_i, the rest of the distribution still satisfies the assumed property, then after running the method recursively $\tilde{O}(1/\beta)\log(1/\epsilon)$ times there is at ϵ fraction of D_i we cannot correctly classify.

Although the above margin property is not satisfied by every distribution, Forster's transform (Forster, 2002; Hardt & Moitra, 2013; Diakonikolas et al., 2021; 2023b) (see Fact C.1) provides us with a way to overcome the difficulties. Roughly speaking, for every distribution D_i, we learn with a pool of unlabeled examples in polynomial time, a subspace V of dimension k, and a non-linear transform $f_A(\cdot)$ such that (1) $\Pr_{x_i \sim D_i}(x_i \in V) \geq \Omega(k/d)$, (2) for every $w_i \in V$ and a random example x_i drawn from $D_i \mid V$, with probability at least $\Omega(1/k)$, $f_A(x_i)$ has a margin $\Omega(1/\sqrt{k})$ (see Fact C.2). With the help of Forster's transform, with $\mathrm{poly}(d)$ label queries, we are able to classify at least $\Omega(1/d)$ fraction of D_i correctly using the margin perception algorithm in Fact C.3. In particular, we will show in Appendix C that after deleting the region we have classified so far, the

remaining distribution still satisfies the reflective symmetry assumption. Thus, after $O(d\log(1/\epsilon))$ rounds, only $O(\epsilon)$ fraction of the D_i has not been classified, which implies that we have learned a good enough hypothesis.

Acknowledgement

Ilias Diakonikolas was supported by NSF Medium Award CCF-2107079, NSF Award CCF-1652862 (CAREER), a Sloan Research Fellowship, and a DARPA Learning with Less Labels (LwLL) grant. Lisheng Ren was supported by NSF Award CCF-1652862 (CAREER). Mingchen Ma and Christos Tzamos were supported by NSF Award CCF-2144298 (CAREER).

Impact Statement

This work presents theoretical results on certain topics of co-training and active learning. The goal is to advance the field of Machine Learning. We do not feel there is any potential societal consequence that needs to be specifically highlighted.

References

Abney, S. Bootstrapping. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 360–367, 2002.

Agarwal, A., Jiang, N., and Kakade, S. M. Reinforcement learning: Theory and algorithms. 2019.

Balcan, M.-F. and Blum, A. A discriminative model for semi-supervised learning. *Journal of the ACM (JACM)*, 57(3):1–46, 2010.

Balcan, M.-F., Blum, A., and Yang, K. Co-training and expansion: Towards bridging theory and practice. *Advances in neural information processing systems*, 17, 2004.

Blum, A. On-line algorithms in machine learning. *Online algorithms: the state of the art*, pp. 306–325, 2005.

Blum, A. and Mansour, Y. Efficient co-training of linear separators under weak dependence. In *Conference on Learning Theory*, pp. 302–318. PMLR, 2017.

Blum, A. and Mitchell, T. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 92–100, 1998.

Blum, A., Frieze, A., Kannan, R., and Vempala, S. A polynomial-time algorithm for learning noisy linear threshold functions. *Algorithmica*, 22:35–52, 1998.

Collins, M. and Singer, Y. Unsupervised models for named entity classification. In *1999 Joint SIGDAT conference on empirical methods in natural language processing and very large corpora*, 1999.

Darnstädt, M., Simon, H. U., and Szörényi, B. Supervised learning and co-training. *Theoretical Computer Science*, 519:68–87, 2014.

Dasgupta, S. Coarse sample complexity bounds for active learning. *Advances in neural information processing systems*, 18, 2005.

Dasgupta, S., Littman, M., and McAllester, D. Pac generalization bounds for co-training. *Advances in neural information processing systems*, 14, 2001.

Diakonikolas, I., Kane, D., and Tzamos, C. Forster decomposition and learning halfspaces with noise. *Advances in Neural Information Processing Systems*, 34:7732–7744, 2021.

Diakonikolas, I., Kontonis, V., Tzamos, C., and Zarifis, N. Self-directed linear classification. In *The Thirty Sixth Annual Conference on Learning Theory*, pp. 2919–2947. PMLR, 2023a.

Diakonikolas, I., Tzamos, C., and Kane, D. M. A strongly polynomial algorithm for approximate forster transforms and its application to halfspace learning. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pp. 1741–1754, 2023b.

Dunagan, J. and Vempala, S. A simple polynomial-time rescaling algorithm for solving linear programs. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pp. 315–320, 2004.

Farouk Abdel Hady, M. and Schwenker, F. Combining committee-based semi-supervised learning and active learning. *Journal of Computer Science and Technology*, 25(4):681–698, 2010.

Forster, J. A linear lower bound on the unbounded error probabilistic communication complexity. *Journal of Computer and System Sciences*, 65(4):612–625, 2002.

Freund, Y. and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Freund, Y., Seung, H. S., Shamir, E., and Tishby, N. Selective sampling using the query by committee algorithm. *Machine learning*, 28:133–168, 1997.

Fudenberg, D. and Levine, D. K. *The theory of learning in games*, volume 2. MIT press, 1998.

Hanneke, S. Activized learning: Transforming passive to active with improved label complexity. *The Journal of Machine Learning Research*, 13(1):1469–1587, 2012.

Hanneke, S. and Yang, L. Minimax analysis of active learning. *J. Mach. Learn. Res.*, 16(1):3487–3602, 2015.

Hardt, M. and Moitra, A. Algorithms and hardness for robust subspace recovery. In *COLT 2013*, pp. 354–375, 2013.

Joachims, T. et al. Transductive inference for text classification using support vector machines. In *Icml*, volume 99, pp. 200–209, 1999.

Kumar, A. and Daumé, H. A co-training approach for multi-view spectral clustering. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 393–400, 2011.

Li, S., Wang, W., Li, W.-T., and Chen, P. Multi-view representation learning with manifold smoothness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8447–8454, 2021.

Littlestone, N. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning*, 2:285–318, 1988.

Littlestone, N. From on-line to batch learning. In *Proceedings of the second annual workshop on Computational learning theory*, pp. 269–284, 1989.

Liu, W., Li, Y., Lin, X., Tao, D., and Wang, Y. Hessian-regularized co-training for social activity recognition. *PLoS One*, 9(9):e108474, 2014.

Muslea, I., Minton, S., and Knoblock, C. A. Selective sampling with redundant views. In *AAAI/IAAI*, pp. 621–626, 2000.

Muslea, I., Minton, S., and Knoblock, C. A. Active learning with multiple views. *Journal of Artificial Intelligence Research*, 27:203–233, 2006.

Park, S.-B. and Zhang, B.-T. Large scale unstructured document classification using unlabeled data and syntactic information. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 88–99. Springer, 2003.

Song, J., Lanka, R., Yue, Y., and Ono, M. Co-training for policy learning. In *Uncertainty in Artificial Intelligence*, pp. 1191–1201. PMLR, 2020.

Wang, W. and Zhou, Z.-H. On multi-view active learning and the combination with semi-supervised learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 1152–1159, 2008.

Yarowsky, D. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pp. 189–196, 1995.

Supplemental Material

A. Label Efficient Co-Training of General VC-Classes via Online Learning

In this section, we prove Theorem 4.1, which shows the idea of online learning can even lead to co-training algorithms for any VC class with a low label complexity.

To start with, we want to remind readers of the notion of ϵ-cover and Halving Algorithms, which will play crucial roles in the proof of Theorem 4.1.

Definition A.1 (ϵ-cover)**.** Let H be a class of hypotheses over an example space X. Let D be a distribution over X. An ϵ-cover of H under distribution D is a finite set of hypotheses C over X such that for every $h \in H$, there exists some $c \in C$ such that $d(h, c) < \epsilon$.

Given a hypothesis class H with VC dimension d, and a sample oracle to D, according to Lemma 21 in (Hanneke & Yang, 2015), we can construct an ϵ cover for H with high probability via the following procedure. Draw $m = \tilde{O}(d/\epsilon)$ unlabeled examples from D, for each possible labeling method l over these m examples, select one $h \in H$ that agrees with l over these m examples and add it to C. In particular, by Sauer's lemma, $|C| \leq O(m^d)$ and $\log(|C|) \leq \tilde{O}(d \log(1/\epsilon))$.

Definition A.2 (Halving Algorithm)**.** Let H be a finite set of hypotheses over example space X. In the mistake-bound online learning model, a halving algorithm works in the following way. Let $H_t \subseteq H$ be the set of hypotheses that are consistent with the sequence of labeled examples $(x_1, y_1), \ldots, (x_{t-1}, y_{t-1})$ seen so far. When x_t arrives, the Halving algorithm predicts x_t by $\operatorname{argmax}_{y \in \{\pm 1\}} \{|\{h \in H_t \mid h(x_t) = y\}| \mid y \in \{\pm 1\}\}$.

It is well known that the Halving algorithm has a mistake bound $O(\log(|H|))$. In particular, if the sequence of labeled examples is not consistent with any $h \in H$, after making $O(\log(|H|))$ mistakes, we can verify this.

Given the background of ϵ-cover and Halving algorithm, we are ready to present the proof of Theorem 4.1.

Proof of Theorem 4.1. We will show Algorithm 3 is an algorithm that satisfies the statement of Theorem 4.1. We first show the correctness of Algorithm 3. Consider a single round of Algorithm 3. There are two cases to consider.

In the first case, the learner manually stops Algorithm 1 and checks the quality of $h = (h_1, h_2)$ that is used by $\mathcal{A}_1, \mathcal{A}_2$ by sampling a set of unlabeled examples S from D.

Let S be a set of $n = \text{poly}(1/(\lambda\epsilon), \log(1/\delta))$ unlabeled examples drawn i.i.d. from D. By Hoeffding's inequality, we have

$$\Pr_{S \sim D^n}\left(\left|\hat{\text{unl}}(h) - \text{unl}(h)\right| \geq \frac{\lambda\epsilon}{8}\right) \leq 2e^{\left(-\frac{\lambda^2\epsilon^2 n}{64}\right)} \leq \exp(-\Omega(\frac{\log 1/\delta}{\epsilon})). \tag{1}$$

For $y \in \{\pm 1\}$, denote by $b_y := \Pr_{x \sim D}(h(x) = y)$. By Hoeffding's inequality again, for a set of n randomly unlabeled samples S, we have

$$\Pr_{S \sim D^n}\left(\left|\frac{\sum_{x \in S} 1(h(x) = y)}{n} - b_y\right| \geq \epsilon\right) \leq \text{poly}(\delta). \tag{2}$$

This implies that if h has an error $\Omega(\epsilon)$ or is close to a constant hypothesis then with a high probability it will not be returned.

In the second case, before we manually stop Algorithm 1, it returns a pair of hypotheses $h = (h_1, h_2)$. Recall the stopping condition in Algorithm 1, h_1, h_2 need to agree on $\text{poly}(1/(\lambda\epsilon))$ unlabeled examples and not to label too many of these examples by $y \in \{\pm 1\}$, which is an even stronger condition of return. Thus if h is not $\epsilon-$close to the target hypothesis, then it will not be returned.

Combining the two cases together, we know that if a pair of hypotheses $h = (h_1, h_2)$ is returned in a single round, then with probability $1 - \text{poly}(\delta)$, h has an error at most ϵ.

To finish the proof of the correctness, it suffices to show that each round, Algorithm 3 will terminate with constant probability. If this is true, then high probability, that after running $O(\log(1/\delta))$ rounds, Algorithm 3 will terminate and the output hypothesis is guaranteed to have error ϵ. Since the online learning algorithm, we use in Algorithm 1 is the Halving algorithm over the η-cover C_i. The mistake bound of the Halving algorithm is $\log(|C_i|) = \tilde{O}(d \log(1/\eta))$. We know from Algorithm 1

Algorithm 3 CO-HALVING(H) (Co-training VC classes via Halving)

Input: A sample access to D, a label query oracle, accuracy parameter $\epsilon \in (0,1)$, confidence parameter $\delta \in (0,1)$
Output: A hypothesis h with error $\mathrm{err}(h) < \epsilon$ with probability at least $1-\delta$
Repeat the following process until a pair of hypotheses $h = (h_1, h_2)$ is output
For $i \in [2]$, draw a set of n_i unlabeled examples $S_i \sim D^{(i)}$, where $n_i = \mathrm{poly}(d/(\lambda\epsilon))$.
For $i \in [2]$, construct η-cover C_i of H over X_i using S_i. (η is chosen as $\mathrm{poly}(\lambda\epsilon/d)$ for some large degree polynomial)
Let $\mathcal{A}_i$ be the halving algorithm over C_i, for $i \in [2]$.
Run REDUCTION($\mathcal{A}_1, \mathcal{A}_2$) by assuming $h_i^* \in C_i$
if For $i \in [2]$, at most one hypothesis in C_i is consistent with all previous label queries during running REDUCTION($\mathcal{A}_1, \mathcal{A}_2$). **then**
  Stop REDUCTION($\mathcal{A}_1, \mathcal{A}_2$) and output the hypothesis pair $h = (h_1, h_2)$ that is currently used by $\mathcal{A}_1, \mathcal{A}_2$
  Draw S, a set of $n = \mathrm{poly}(1/\lambda\epsilon, \log(1/\delta))$ unlabeled examples from D and compute $\hat{\mathrm{unl}}(h)$ using S.
  if $\hat{\mathrm{unl}}(h) < O(\lambda\epsilon)$ and $\forall y \in \{\pm 1\}$, more than $\Omega(\epsilon)$ fraction of the agreed examples are labeled by y **then**
    Return $h = (h_1, h_2)$.
else
  Return the hypothesis output by $(\mathcal{A}_1, \mathcal{A}_2)$.

that every time we use Algorithm 1, the number of unlabeled examples we use is $m = \mathrm{poly}(\log(|C_i|), 1/\alpha, 1/(\lambda\epsilon))$. This implies that if we choose $\eta = \mathrm{poly}(\lambda\epsilon/d)$ for some large degree polynomial, then we can ensure $m\eta < 1/4$. In particular, let h_i^* be the target hypothesis in X_i, the guarantee of η-cover says there exists some $c_i^* \in C_i$ such that $d(c_i^*, h_i^*) < \eta$. By Markov's inequality, we know that with probability at least $1/2$, for $i \in [2]$, each x of these m unlabeled examples satisfies $h_i^*(x) = c_i^*(x)$. According to the Halving algorithm, when we manually stop Algorithm 1, we exactly find $(h_1, h_2) = (c_1^*, c_2^*)$, which has an error at most $O(\eta) = o(\lambda\epsilon)$. In this case, (1) and (2) imply that with high probability (h_1, h_2) will be output. Thus, each round, Algorithm 3 will terminate with a constant probability. This finishes the proof of the correctness.

Finally, we bound the label complexity of Algorithm 3, this follows directly from the proof of the correctness. As discussed above the halving algorithm has a mistake bound $\log(|C_i|) = \tilde{O}(d\log(1/(\lambda\epsilon)))$. In each round, after making $\tilde{O}(d\log(1/(\lambda\epsilon)))$ updates, we are guaranteed to mannually stop Algorithm 1. Since in the worst case to make an update we need to make $\tilde{O}(1/\alpha)$ queries, we know the number of queries we make in a single round is $\tilde{O}(d\log(1/(\lambda\epsilon))/\alpha)$. This finishes the proof of the label complexity since each round will terminate will a constant probability.

□

B. Efficient Co-Training of k-Interval

In this section, we will prove Theorem 4.2. We give Algorithm 4, a detailed version of Algorithm 2. To avoid confusion, we will first list some notations that we will use in the proof.

For $i \in [2]$, we denote by $h_i^* = \cup_j I_{ij}^*$ the target hypothesis over X_i, where for each $j \in [k]$, I_{ij}^* is one of the interval that defines h_i^*. For simplicity, the index j is ordered according to the relative positions of these intervals.i.e If $1 \le j < g \le k$, then I_{ij}^* lies on the left-hand side of I_{ig}^*. For each interval I_{ij}^*, we will denote by l_{ij}^* and r_{ij}^* the left boundary and the right boundary of I_{ij}^*. Since Algorithm 4 runs in rounds, in round t, we will denote by $\hat{l^{*t}}_{ij}$ and $\hat{r^{*t}}_{ij}$ the left boundary and the right boundary of the convex hull of the examples in I_{ij}^* that has been queried by Algorithm 4 before round $t+1$.

Since the concept h_i^* is a union of k intervals, we know that the region in $\mathbb{R}$, where examples are labeled negative by h_i^* is a union of at most $k+1$ intervals. Similarly, we will use N_{ij}^* to denote these "negative" intervals ordered from left to right.

We will also define similar notations for the hypothesis h_i^t maintained in each round. Since for each time step t and for each $i \in [2]$, h_i^t is defined by a union of several intervals. We will denote by $\cup_j I_{ij}^t$ these intervals, also ordered from left to right.

Finally, for every t and for each $i \in [2]$, we decompose the error of $\mathrm{erf}\ h_i^t$ in the following way

$$\mathrm{err}(h_i^t) = \alpha_+ \mathrm{err}_+(h_i^t) + \alpha_- \mathrm{err}_-(h_i^t),$$

where for $y \in \{\pm 1\}$, $\mathrm{err}_y(h_i^t) = \Pr_{x_i \sim D_i^y}(h_i^t(x_i) \neq y)$.

Algorithm 4 LEARNK-INTERVAL (Efficient co-training k intervals)

Input: A sample access to D, a label query oracle, accuracy parameter $\epsilon \in (0,1)$, confidence parameter $\delta \in (0,1)$
Output: A hypothesis $\hat{h}$ with error $\mathrm{err}(\hat{h}) < \epsilon$ with probability at least $1-\delta$
Draw $\tilde{O}(k/\alpha)$ examples from D and query their labels. Over X_1, X_2, construct initial hypothesis pair (h_1^1, h_2^1) as follows.
For $i \in [2]$, let $Q_i \subseteq X_i$ be the queried examples. Repeat the following procedure until no positive examples are in Q_i
1. Find x_i^a, the first positive example in Q_i and x_i^b the smallest negative example in Q_i such that $x_i^- > x_i^+$
2. Denote by I, the set of positive examples in Q_i between x_i^a and x_i^b (including x_i^a).
3. Create a new interval $\mathrm{conv}(I)$ for h_i^1 and delete I from Q_i

Write $h_i^1 = \bigcup_{j=1} I_{ij}^1$. ($I_{ij}^1$ ordered by position from left to right)
$\mathrm{COUNT}_T \leftarrow 0$, $\mathrm{COUNT}_U \leftarrow 0$
$n = \mathrm{poly}(1/(\lambda\epsilon), \log(1/\delta))$
for $t = 1, 2, \ldots$ **do**
  Draw example $x^{(t)} = (x_1^{(t)}, x_2^{(t)}) \sim D$
  if $\mathrm{COUNT}_T < n$ **then**
    $\mathrm{COUNT}_U \leftarrow \mathrm{COUNT}_U + 1$, if $h_1^t(x_1^{(t)}) \neq h_2^t(x_2^{(t)})$.
    $\mathrm{COUNT}_T \leftarrow \mathrm{COUNT}_T + 1$ and enter next round
  if $\mathrm{COUNT}_T \geq n$ and $\frac{\mathrm{COUNT}_U}{\mathrm{COUNT}_T} < \frac{\lambda\epsilon}{8}$ **then**
    Return $\hat{h} = (h_1^t, h_2^t)$.
  {Check if the current hypothesis is good enough}
  if $h_1^t(x_1^{(t)}) \neq h_2^t(x_2^{(t)})$ **then**
    Query the label of $x^{(t)}$
    if $y(x^{(t)}) = 1$ and is misclassified by h_i^t **then**
      Find j such that $x_i^{(t)}$ lies between I_{ij}^t and $I_{i(j+1)}^t$ and modify h_{ij}^t as follows:
      1. If $\mathrm{conv}(I_{ij}^t \cup \{x_i^{(t)}\})$ contains no negative examples queried before, $I_{ij}^t \leftarrow \mathrm{conv}(I_{ij}^t \cup \{x_i^{(t)}\})$, otherwise;
      2. If $\mathrm{conv}(I_{i(j+1)}^t \cup \{x_i^{(t)}\})$ contains no negative examples queried before, $I_{i(j+1)}^t \leftarrow \mathrm{conv}(I_{i(j+1)}^t \cup \{x_i^{(t)}\})$;
      3. Otherwise, create $\{x_i^{(t)}\}$ as a new interval for h_i^t.
    if $y(x^{(t)}) = 0$ and is misclassified by h_i^t **then**
      Find the interval I_{ij}^t that contains $x^{(t)}$ and modify h_i^t by dividing I_{ij}^t into two intervals
    $\mathrm{COUNT}_T \leftarrow 0$, $\mathrm{COUNT}_U \leftarrow 0$

Based on these notations, we present several structural results of learning k-intervals.

We start showing that our initialized hypothesis is not close to any constant hypothesis.

Claim B.1. [Property of the initial hypothesis] For $i \in [2]$, let S be a set of $\tilde{O}(k/\alpha)$ examples drawn randomly from D_i. With probability at least $1-\delta$, any $h_i \in H$ that correctly labels S satisfies $d(h_i, +1) \geq \alpha/2$ and $d(h_i, -1) \geq \alpha/2$.

Proof of Claim B.1. Without loss of generality, we assume that $\Pr_{x\sim D_i}(h_i^*(x) = +1) = \alpha$. We first show that with probability at least $1-\delta$, for $y \in \{\pm 1\}$, there are $\tilde{O}(k)$ examples with label y. The sampling process for S can be understood in the following way. We first draw a Bernoulli random variable with parameter α, if we get 1 then we sample some $x \sim D_i^+$, otherwise, we sample some $x \sim D_i^-$. In expectation, each round we will sample α example from D_i^+. Since $|S| = \tilde{O}(k/\alpha)$, Hoeffding's inequality implies that with probability at least $1-\delta$, at least $\alpha/2$ fraction of S are drawn from D_i^+ and no more than $3\alpha/2$ fraction of S are drawn from D_i^+, which also implies that at least $\alpha/2$ fraction of S are drawn from D_i^- since $1-\alpha \geq \alpha$. Thus for $y \in \{\pm 1\}$, there are $\tilde{O}(k)$ examples in S with label y. Given this happens, we apply VC-inequality over distribution D_i^+ and D_i^-. That is to say, for $y \in \{\pm 1\}$, with probability at least every hypothesis $h_i \in H$ that correctly labels S has an error at most $1/3$ with respect to h^* over D_i^y. This suffices to show $d(h_i, y) \geq \alpha/2$ for $y \in \{\pm 1\}$. Because assuming $d(h_i, y) < \alpha/2$, then a random example drawn from D_i with have a probability at least $1-\alpha/2$ to be labeled by y. However,

$$\Pr_{x\sim D_i}(h_i(x) = 1) = \alpha \Pr_{x\sim D_i^+}(h_i(x) = 1) + (1-\alpha)\Pr_{x\sim D_i^-}(h_i(x) = 1) \leq \alpha + (1-\alpha)/3 < 1 - \alpha/2.$$

$$\Pr_{x\sim D_i}(h_i(x) = -1) = \alpha \Pr_{x\sim D_i^+}(h_i(x) = -1) + (1-\alpha)\Pr_{x\sim D_i^-}(h_i(x) = 1) \leq \alpha/3 + 1 - \alpha < 1 - \alpha/2,$$

which gives a contradiction. □

Claim B.2. [Structure of the hypothesis] For every $t \geq 1$, $i \in [2]$, hypothesis $h_i^t = \cup_j I_{ij}^t$ maintained by Algorithm 4 satisfies the following properties

- For each j, the endpoints of I_{ij}^t are defined by two positive examples $x_i^{(t_1)}, x_i^{(t_2)}$ that were queried by Algorithm 4.
- For each j, if some example $x_i^{(t_1)} \in I_{ij}^t$ was queried by Algorithm 4, then $x_i^{(t_1)}$ is a positive example.
- For each j, there is at least one negative example and no positive example that was queried by Algorithm 4 in the area between intervals I_{ij}^t and $I_{i(j+1)}^t$.

In particular, if h_i^* is defined by ℓ intervals, then the above properties ensure each h_i^t is also defined by at most ℓ intervals.

Proof of Claim B.2. We prove Claim B.2 by induction. Let $I_{i1}^*, \ldots, I_{i\ell}^*$ be the intervals that define h_i^*. Let $S_i^0 \subseteq X_i$ be the initial set of examples that we use to construct the initial hypothesis $h_i^0 = \cup_j I_{ij}^0$. The three properties in the statement of Claim B.2 are clearly satisfied by h_i^0 due to Algorithm 4. Now suppose the three properties are satisfied by h_i^t, we show they are also satisfied by h_i^{t+1}. We only need to consider the case when h_i^{t+1} is obtained based on an example $x_i^{(t)}$ that is misclassified by h_i^t, because otherwise h_i^t will not be changed.

If some negative example $x_i^{(t)}$ is predicted positive by h_i^t, then it must be the case that for some j, $x_i^{(t)} \in I_{ij}^t$. By induction $x_i^{(t)}$ is the first negative example in I_{ij}^t that is queried by Algorithm 4. Let $x_i^{(t_1)} < x_i^{(t)} < x_i^{(t_2)} \in I_{ij}^t$ be two positive examples that are closest to $x_i^{(t)}$ and were queried by Algorithm 4. To obtain h_{ij}^{t+1}, Algorithm 4 will cut I_{ij}^t into two intervals and set up $x_i^{(t_1)}, x_i^{(t_2)}$ to be endpoints of the new intervals. So the first two properties still hold. Furthermore, $x_i^{(t)}$ is the only queried point between the two intervals, so the third property also holds.

If some positive example $x_i^{(t)}$ is predicted negative by h_i^t, then such an $x_i^{(t)}$ must lie between some I_{ij}^t and $I_{i(j+1)}^t$. According to Algorithm 4, h_i^t could be modified in different ways. In the first case, one of the two intervals, assuming this is I_{ij}^t without loss of generality, will enlarge its boundary to $x_i^{(t)}$. When this happens, there must be no negative examples queried between $x_i^{(t)}$ and the original boundary of I_{ij}^t. By induction, we know the three properties still hold. In the second case, we insert $\{x_i^{(t)}\}$ itself as a new interval, because enlarging either I_{ij}^t or $I_{i(j+1)}^t$ will cause inconsistency. By induction, the three properties still hold.

Finally, we show that once the three properties hold, the hypothesis h_i^t cannot be defined by too many intervals. We notice that each interval I_{ij}^t is created if it touches some target interval I_{ig}^* (some $x_i^{(t)} \in I_{ig}^* \cap I_{ij}^t$ is queried) . But on the other hand, each target interval I_{ig}^* cannot have intersections with more than one I_{ij}^t because otherwise, I_{ig}^* will contain some negative example. Since there are at most ℓ target intervals to be learned, we know that as long as the three properties hold, h_i^t is defined by at most ℓ intervals.

□

Given the structure of the hypothesis maintained by Algorithm 4, we will next give some structural results for $\mathrm{err}_-(h_i^t)$ and $\mathrm{err}_+(h_i^t)$. We will start with the negative error $\mathrm{err}_-(h_i^t)$.

Claim B.3. For $t \geq 1, i \in [2]$ and for every j, if $x_i^{(t)} \in N_{ij}^*$, a negative example is queried by Algorithm 4 then for every $t' > t$, $h_i^{t'}$ will not misclassify any example in N_{ij}^*.

Proof of Claim B.3. Without loss of generality, we assume $x_i^{(t)} \in N_{ij}^*$ is the first example queried in N_{ij}^* by Algorithm 4. We first notice that for every $g \geq 0$, the boundaries of I_{ig}^t cannot lie in N_{ij}^* by the first property in the statement of Claim B.2. In other words, N_{ij}^* must either be contained in some interval I_{ig}^t or be contained in the area between some intervals I_{ig}^t and $I_{i(g+1)}^t$. By the second property in the statement of Claim B.2, for each g, I_{ig}^t will not contain a queried example with a negative label, which implies that after querying $x_i^{(t)}$, N_{ij}^* cannot be contained in any I_{ig}^t and must lie in between two intervals I_{ig}^t and $I_{i(g+1)}^t$. This implies that after round t, every example in N_{ij}^* will be predicted as negative.

□

Given Claim B.3, we will analyze how $\mathrm{err}_-(h_i^t)$ will change if a negative example misclassified by h_i^t is queried.

By the first property of Claim B.2, we know that each for every t and for every $j \in [k]$, there must be some $g \in [k]$ such that N_{ij}^* must either lie entirely in some interval I_{ig}^t or line between two intervals I_{ig}^t and $I_{i(g+1)}^t$.

In the first case, h_i^t will predict every example in N_{ij}^* incorrectly and thus N_{ij}^* will contribute $\Pr_{x\sim D_i^-}(x \in N_{ij}^*)$ to $\mathrm{err}_-(h_i^t)$. For simplicity, we denote by N_i^t the set of N_{ij}^* of this kind. In the second case, h_{ij}^t will not predict any example in N_{ij}^* incorrectly and N_{ij}^* contributes 0 to $\mathrm{err}_-(h_i^t)$. Thus,

$$\mathrm{err}_-(h_i^t) = \sum_{N_{ij}^* \in N_i^t} \Pr_{x\sim D_i^-}(x \in N_{ij}^*) \,.$$

By Claim B.3, we know that in a single round if h_i^t makes a mistake at N_{ij}^*, then after that round, N_{ij}^* will be perfectly classified then $\mathrm{err}_-(h_i^t)$ drops additively by $\Pr_{x\sim D_i^-}(x \in N_{ij}^*)$. Given this, in each round, we define a random variable B_i^t as the indicator of the event that $\mathrm{err}_-(h_i^{(t+1)}) \le (1 - 1/(2k))\mathrm{err}_-(h_i^{(t+1)})$.

On the other hand, however, we point out that although when some $x_i^{(t)} \sim D_{-i}$ is found to be misclassified $\mathrm{err}_-(h_i^t)$ will not increase, it would be the case that $\mathrm{err}_+(h_i^t)$ increases instead. This is because example $x_i^{(t)}$ would cut some interval I_{ij}^t into two pieces, in which cases some area contained in I_{ij}^t would not be predicted positive by $h_i^{(t+1)}$. Let $\tilde{B_i}^t$ be the event that $err_+(h_i^t)$ increases after round t. We notice that $\tilde{B_i}^t$ will happen at most $k+1$ times because it will happen only when a negative example $x_i^{(t)}$ is queried and by Claim B.3, such events would happen only at most $k+1$ times. For simplicity, we denote by $(B_i')^t := B_i^t + \tilde{B_i}^t$. The above discussion gives the following claim for the change of $\mathrm{err}_-(h_i^*)$.

Claim B.4. For $t \ge 1$ and $i \in [2]$,

$$\Pr_{x^{(t)}\sim D}\left((B_i')^t = 1 \mid x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = 1, h^*(x^{(t)}) = -1\right) \ge \frac{\lambda}{8}.$$

Proof of Claim B.4. We will first show the following fact. Let $x_i \sim D_i^-\,|_{x\in N_i^t}$, with probability at least $1/4$, $x_i \in N_{ij}^*$ with $\Pr_{x\sim D_i^-|_{x\in N_i^t}}(x \in N_{ij}^*) \ge 1/(2k)$. We prove this by contradiction. Assuming instead the event happens with a probability less than $1/4$, since there are at most $k+1$ different N_{ij}^*, we have

$$1 = \sum_{N_{ij}^* \in N_i^t} \Pr_{x\sim D_i^-|_{x\in N_i^t}}(x \in N_{ij}^*) < \frac{1}{4} + \frac{k+1}{2k} \le 1, \tag{3}$$

which gives a contradiction. In particular, if $x_i^{(t)}$ falls into some N_{ij}^* that satisfies the above condition, and we query it, then it must have $(B_t') = 1$. Thus, we have

$$\begin{aligned}
&\Pr_{x^{(t)}\sim D}\left((B_i')^t = 1 \mid x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = 1, h^*(x^{(t)}) = -1\right)\\
&= \frac{\Pr_{x^{(t)}\sim D}\left((B_i')^t = 1, x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = 1, h^*(x^{(t)}) = -1\right)}{\Pr_{x^{(t)}\sim D}\left(x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = -1, h^*(x^{(t)}) = 1\right)}\\
&\ge \frac{\Pr_{x^{(t)}\sim D^-}\left((B_i')^t = 1, x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = 1, h^*(x^{(t)}) = -1\right)}{\Pr_{x^{(t)}\sim D_i^-}\left(h_i^t(x^{(t)}) = 1\right)}\\
&= \frac{\Pr_{x^{(t)}\sim D^-}\left((B_i')^t = 1, h_i^t(x^{(t)}) = 1, h_{3-i}^t(x^{(t)}) = -1\right)}{\Pr_{x^{(t)}\sim D_i^-}\left(h_i^t(x^{(t)}) = 1\right)}\\
&\ge \frac{\lambda \Pr_{x^{(t)}\sim D^-}\left((B_i')^t = 1, h_i^t(x^{(t)}) = 1\right)}{2\Pr_{x^{(t)}\sim D_i^-}\left(h_i^t(x^{(t)}) = 1\right)} = \frac{\lambda}{2}\Pr_{x_i\sim D_i^-|_{x\in N_i^t}}((B_i')^t) \ge \frac{\lambda}{8}\,.
\end{aligned}$$

Here the second inequality is followed by the λ-weak dependence and Claim B.1 and the last inequality is followed by the property we proved at the beginning.

□

We next analyze $\mathrm{err}_+(h_i^t)$ in a similar way.

Claim B.5. For $t \geq 1$, $i \in [2]$ and $j \in [k]$, let $x \sim D_i$ conditioned on $x \in [l_{ij}^*, \hat{l^{*t}}_{ij}]$. With probability at least $1/2$, we have

$$\Pr_{x' \sim D_i|_{x' \in I_{ij}^*}}(x' \in [l_{ij}^*, x])) \leq \frac{1}{2} \Pr_{x' \sim D_i|_{x' \in I_{ij}^*}}(x' \in [l_{ij}^*, \hat{l^{*t}}_{ij}]),$$

where the randomness only comes from x. The same statement holds if we replace $l_{ij}^*, \hat{l^{*t}}_{ij}$ by $r_{ij}^*, \hat{r^{*t}}_{ij}$ accordingly.

Proof of Claim B.5. Denote by x_m the median of the interval $[l_{ij}^*, \hat{l^{*t}}_{ij}]$ under the distribution $D_i\ |_{[l_{ij}^*, \hat{l^{*t}}_{ij}]}$. That is $\Pr_{x \sim D_i|_{[l_{ij}^*, \hat{l^{*t}}_{ij}]}}(x < x_m) = 1/2$. For any $x < x_m$, we have

$$\Pr_{x' \sim D_i|_{x' \in I_{ij}^*}}(x' \in [l_{ij}^*, x])) = \Pr_{x' \sim D_i|_{[l_{ij}^*, \hat{l^{*t}}_{ij}]}}(x' < x) \Pr_{x' \sim D_i|_{x' \in I_{ij}^*}}(x' \in [l_{ij}^*, \hat{l^{*t}}_{ij}]) \leq \frac{1}{2} \Pr_{x' \sim D_i|_{x' \in I_{ij}^*}}(x' \in [l_{ij}^*, \hat{l^{*t}}_{ij}]).$$

Thus, with probability at least $1/2$ we will shrink the probability mass of $[l_{ij}^*, \hat{l^{*t}}_{ij}]$ by a factor of 2.

□

The analysis of the change of $\mathrm{err}_+(h_i^t)$ will be based on Claim B.5. We first introduce the notation of type 1 error and type 2 error. Consider a fixed I_{ij}^*. By the third property of Claim B.2, we know that for each $t \geq 0$, each I_{ij}^* can only touch at most one I_{ig}^t. In other words, each I_{ij}^* either entirely lies in some I_{ig}^t or cross boundaries of one I_{ig}^t or lie in between two intervals I_{ig}^t and $I_{i(g+1)}^t$.

In the first case, I_{ij}^* contributes no error. In the second case, I_{ij}^* can cross either one boundary of some interval I_{ig}^t or two boundaries of some interval I_{ig}^*. We say such I_{ij}^* has a type 1 error and the error it contributes to $\mathrm{err}_+(h_i^t)$ is

$$E^1(I_{ij}^*) = \Pr_{x \sim D_i^+}([l_{ij}^*, \hat{l_{ij}^{*t}}]) + \Pr_{x \sim D_i^+}([\hat{r_{ij}^{*t}}, r_{ij}^*])$$

(and only one of the two terms if only the left or right boundary crosses the boundary of some I_{ig}^t).

In the third case, we say I_{ij}^* has type 2 error and it contributes $\mathrm{err}_+(h_i^t)$, $E^2(I_{ij}^*) = \Pr_{x \sim D_i^+}(I_{ij}^*)$ because no examples in this region has been queried.

As we discussed above, type 1 error and type 2 error are defined by at most $2k$ region, each of which is an interval. For simplicity, for each t and $i \in [2]$, we will use R_{ij}^t, $j \in [2k]$ to denote these intervals, ordered from left to right and we use R_i^t to denote their union. Based on these notations, we will discuss how the err_+ changes when a positive example $x^{(t)}$ is queried. If $x_i^{(t)} \in R_{ij}^t$ for some j such that R_{ij}^t causes type 1 error, then according to Claim B.5, we know that with a constant probability, the error contribution of R_{ij}^t will drop by a factor of 2. To capture such progress made by Algorithm 4, we define A_i^t to be indicator of the event $\mathrm{err}_+(h_i^{(t+1)}) \leq (1 - 1/(8k))\mathrm{err}_+(h_i^{(t+1)})$. If $x_i^{(t)} \in R_{ij}^t$ for some j such that R_{ij}^t causes type 2 error, the error contribution of R_{ij}^t will not change but the type of error will instead become type 1. Similarly, define $\bar{A_i}^t$ to be the indicator of the event that a queried example falls into some region R_{ij}^t that causes a type 2 error. In particular, $\bar{A_i}^t$ can happen at most k times, because as long as an example in some I_{ij}^* is queried, I_{ij}^* will never cause any type 2 error. We also observe that after we query some positive example $\mathrm{err}_-(h_i^t)$ might also increase because by enlarging some I_{ij}^t, I_{ij}^t might cover some N_{ij}^* where no negative example has been queried. We define such an event to be $\tilde{A_i}^t$ and similarly, $\tilde{A_i}^t$ can also happen at most $k+1$ times. Now, let $(A_i')^t := A_i^t + \tilde{A_i}^t + \bar{A_i}^t$. The main structural result for $\mathrm{err}_+(h_i^t)$ can be summarized via the following claim.

Claim B.6. For $t \geq 1$ and $i \in [2]$,

$$\Pr_{x^{(t)} \sim D}\left((A_i')^t \geq 1 \mid x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = -1, h^*(x^{(t)}) = 1\right) \geq \frac{\lambda}{16}.$$

The proof of Claim B.6 is similar to that of Claim B.4.

Proof of Claim B.6. Notice that let $x_i \sim D_i^+ \,|_{x \in R_i^t}$, with probability at least $1/4$, $x_i \in R_{ij}^t$ such that $\Pr_{x \sim D_i^+|_{x \in R_i^t}}(x \in R_{ij}^t) \geq 1/(4k)$. This observation can be proved in the same way as (3). When this happens, if R_{ij}^t causes type 1 error, we know from Claim B.5, with another probability of $1/2$, the contribution of R_{ij}^t will shrink by a factor of 2. If R_{ij}^t causes type 2 error, we know that $\tilde{A_i}^t = 1$. Based on this, we have

$$\begin{aligned}
&\Pr_{x^{(t)} \sim D}\left((A_i')^t \geq 1 \mid x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = -1, h^*(x^{(t)}) = 1\right) \\
&= \frac{\Pr_{x^{(t)} \sim D}\left((A_i')^t \geq 1, x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = -1, h^*(x^{(t)}) = 1\right)}{\Pr_{x^{(t)} \sim D}\left(x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = -1, h^*(x^{(t)}) = 1\right)} \\
&\geq \frac{\Pr_{x^{(t)} \sim D^+}\left((A_i')^t \geq 1, x^{(t)}\text{is queried}, h_i^t(x^{(t)}) = -1, h^*(x^{(t)}) = 1\right)}{\Pr_{x^{(t)} \sim D_i^+}\left(h_i^t(x^{(t)}) = -1\right)} \\
&= \frac{\Pr_{x^{(t)} \sim D^-}\left((A_i')^t \geq 1, h_i^t(x^{(t)}) = -1, h_{3-i}^t(x^{(t)}) = 1\right)}{\Pr_{x^{(t)} \sim D_i^-}\left(h_i^t(x^{(t)}) = -1\right)} \\
&\geq \frac{\lambda \Pr_{x^{(t)} \sim D^-}\left((A_i')^t \geq 1, h_i^t(x^{(t)}) = -1\right)}{2\Pr_{x^{(t)} \sim D_i^+}\left(h_i^t(x^{(t)}) = -1\right)} = \frac{\lambda}{2} \Pr_{x_i \sim D_i^+|_{x \in R_i^t}}((A_i')^t) \geq \frac{\lambda}{16}.
\end{aligned}$$

Here the second inequality holds because of the λ-weak dependence and Claim B.1. □

Proof of Theorem 4.2. We first show the correctness of the algorithm. Notice that every time we obtain a new hypothesis pair h^t, it will not be changed before we query the next example when at least one of h_i^t will make a mistake on that example.

Now we consider a fixed pair of hypotheses $h^t = (h_1^t, h_2^t)$. Upon we obtain h^t, we maintain two counters, COUNT_T, which counts the total number of examples we have seen since the last update, and COUNT_U, the number of examples that are predicted differently by h^t. We say h^t fails to pass the test, if h^t is not output when $\mathrm{COUNT}_T > n$.

Notice that by Algorithm 4, each time we makes a modification, h_i^t will not misclassify any example $x_i^{(t')}$ that we have queried previously. By Claim B.1, we know that with high probability $\forall t$ and $\forall i \in [2]$, we have $d(h_i^t, 1) \geq \alpha/2$ and $d(h_i^t, -1) \geq \alpha/2$. We know from Lemma 2.3 that if $d(h^t, h^*) > \epsilon$ then we must have $\mathrm{unl}(h^t) > \lambda\epsilon/4$ because h^t is guaranteed to be far from any constant hypothesis.

Let S be a set of $n = \mathrm{poly}(1/(\lambda\epsilon), \log(1/\delta))$ unlabeled examples drawn i.i.d. from D. By Hoeffding's inequality, we have

$$\Pr_{S \sim D^n}\left(\left|\hat{\mathrm{unl}}(h^t) - \mathrm{unl}(h^t)\right| \geq \frac{\lambda\epsilon}{8}\right) \leq 2e^{\left(-\frac{\lambda^2\epsilon^2 n}{64}\right)} \leq \exp(-\Omega(\frac{\log 1/\delta}{\epsilon})).$$

This implies that if $\mathrm{err}(h^t) > \epsilon$, with high probability $\hat{\mathrm{unl}}(h^t) > \lambda\epsilon/8$ and will fail to pass the test. On the other hand, if $\mathrm{unl}(h^t) < \lambda\epsilon/16$, then with high probability $\hat{\mathrm{unl}}(h^t) < \lambda\epsilon/8$ and will pass the test and returned. To finish showing the correctness of Algorithm 4 it remains to show that the method of updating hypotheses used by Algorithm 4 can quickly drop $\mathrm{unl}(h_i^t)$ to $O(\lambda\epsilon)$ so that a good hypothesis will be output finally. If we can prove this then we can also directly obtain the label complexity and the number of unlabeled examples used by Algorithm 4.

For $i \in [2]$ and $y \in \{\pm 1\}$, we define $(e_i^y)^t$ be the indicator of the event $h_i^t(x_i^{(t)}) \neq y, h_{3-i}^t(x_i^{(t)}) = y$ and define $(e_m)^t$ be the indicator of the event $\mathrm{argmax}_{i,y}\{\Pr_{x^{(t)} \sim D}((e_i^y)^t | i \in [2], y \in \{\pm 1\})\}$. In time round t, if $x^{(t)}$ is queried, we say such a query

is good if $(e_m)^t = 1$ and the following condition is satisfied. Suppose $(e_m)^t$ is achieved by $(e_i^y)^t$. If $y = +$, then a good query requires $(A_i')^t \geq 1$. If $y = -$, then a good query requires $(B_i')^t \geq 1$. Notice that

$$\Pr_{x^{(t)}\sim D}((e_m)^t \mid x^{(t)} \text{ is queried}) = \frac{\Pr_{x^{(t)}\sim D}((e_m)^t)}{\sum_{i,y}\Pr_{x^{(t)}\sim D}((e_i^y)^t)} \geq \frac{1}{4}. \tag{4}$$

(5) together with Claim B.4 and Claim B.6 implies that every time we make a query, with probability at least $\lambda/64$, such a query is good. We claim that if at some point T we have made $\Omega(k^2\log(1/(\lambda\epsilon)))$ good queries, then before time T there must be some time t such that $\mathrm{err}(h^t) < O(\lambda\epsilon)$. Notice that during the learning process, when events $\tilde{A_i}^t, \bar{A_i}^t, \tilde{B_i}^t$ happen, $\mathrm{err}(h^t)$ might increase. In fact, they are the only events that can make $\mathrm{err}(h^t)$ increase. However, according to Claim B.2 and Claim B.3, these events happen deterministically at most $O(k)$ times. This implies if we have made $\Omega(k^2\log(1/(\lambda\epsilon)))$ good queries, then there must be $\Omega(k\log(1/(\lambda\epsilon)))$ successive good query such that $\tilde{A_i}^t, \bar{A_i}^t, \tilde{B_i}^t$ do not happen at the time we make the good queries. In this case, after each good query has been made, $\Pr_{x^{(t)}\sim D}((e_m)^t)$ deterministically drop by a factor of $(1 - 1/O(k))$. Thus, after these good queries have been made we must have

$$\mathrm{unl}(h^t) \leq 4\Pr_{x^{(t)}\sim D}((e_m)^t) \leq O(\lambda\epsilon),$$

in which case h^t will be output. Notice that

$$\Pr_{x^{(t)}\sim D}((e_m)^t \mid x^{(t)} \text{ is queried}) = \frac{\Pr_{x^{(t)}\sim D}((e_m)^t)}{\sum_{i,y}\Pr_{x^{(t)}\sim D}((e_i^y)^t)} \geq \frac{1}{4}\,. \tag{5}$$

Notice that (5) together with Claim B.4 and Claim B.6 implies that every time we make a query, with probability at least $\lambda/64$, such a query is good. Hoeffding's inequality implies that if we make $\tilde{O}(k^2\log(1/(\lambda\epsilon))/\lambda)$ queries after the initialization step, then with probability $1 - \mathrm{poly}(\delta)$, a constant fraction of them are good queries. This finishes the proof of Theorem 4.2.

□

C. Learning Homogeneous Halfspaces

In this section, we provide the algorithm and proof establishing Theorem 4.3. We will use $h_w : \mathbb{R}^d \to \{\pm 1\}$ for a unit vector w to denote the homogeneous LTF defined as $h_w(x) = \mathrm{sign}(\langle w, x\rangle)$ and use $x \sim_u S$ to denote that x is sampled uniformly at random from a set S. The high-level idea of the algorithm is the following. Instead of directly learning on the original domain $X = X_1 \times X_2$ where the halfspaces do not have "margins", we will apply the technique of Forster decomposition on each domain X_1 and X_2. Such a technique gives a transformation mapping from X to $X' = X_1' \times X_2'$. After this transformation, at least $1/(4d)$ fraction of the examples will be correctly classified by halfspaces with $1/\left(2\sqrt{d}\right)$ margins on either X_1' or X_2'. Then our algorithm uses a modified version of the perceptron algorithm, so that after at most roughly $O(d\log(d))$ many queries to the oracle, we learn a halfspace on X_1 that will correctly classify those $1/(4d)$ fraction of examples that has margins (this corresponds to a partial classifier in the original space X). We removed those regions that have been classified by this partial classifier from our input distribution D by doing rejection sampling. We repeat this process $d\log(1/\epsilon)$ times until there is only $\epsilon/2$ mass of the input distribution remaining. Then we output a decision list of all the partial classifiers we get from each iteration.

We start by introducing the following fact for Forster transformation from (Diakonikolas et al., 2023b).

Fact C.1 (Proposition 7.1 from (Diakonikolas et al., 2023b))**.** There is an algorithm that given a multiset S of n points in $\mathbb{R}_*^d$ and $\epsilon > 0$, runs in time polynomial of dn/ϵ, and with high probability returns a subspace $V \subseteq \mathbb{R}^d$ with $V \neq 0$ and a rank $\dim(V)$ linear transformation $A : V \to \mathbb{R}^{\dim(V)}$, such that for the function $f_A(x) \stackrel{\text{def}}{=} Ax/\left\|Ax\right\|_2$,

1. $|S \cap V| \geq (n/d)\dim(V)$.
2. The eigenvalues of $\frac{1}{X\cap V}\sum_{x\in X\cap V} f_A(x)(f_A(x))^\intercal$ are in $[(1-\epsilon)/\dim(V), (1+\epsilon)/\dim(V)]$.

We note that for any unit vector $w \in \mathbb{R}^d$ and its corresponding LTF h_w, there must be a unit vector $w' \in \mathbb{R}^{\dim(V)}$ where $w' = (A^{-1})^\intercal w/\|(A^{-1})^\intercal w\|_2$ such that for any $x \in V$, $h_{w'}(f_A(x)) = h_w(x)$. Namely, Forster decomposition preserves

the function class of homogeneous LTFs. Furthermore, if we choose ϵ to be at most a sufficiently small constant, since $\mathbf{E}_{x\sim U(X\cap V)}\left[\langle w, f_A(x)\rangle^2\right]$ is close to 1. It is easy to see that at least $|X\cap V|/(4\dim(V)) \geq |X|/(4d)$ many points in X satisfies $\langle w, f_A(x)\rangle \geq 1/\left(2\sqrt{d}\right)$ for any unit vector w. The above observations are summarized by the following facts.

Fact C.2. Given a multiset S of n samples of $x \in \mathbb{R}^d$ and let f_A be the Forster decomposition from Fact C.1 performed on the samples in S for ϵ at most a sufficiently small constant. Conditioned on the algorithm in Fact C.1 succeeds, the following holds:

1. Let w be any unit vector in $\mathbb{R}^d$, there is a unit vector $w' \in \mathbb{R}^{\dim(V)}$ such that $h_w(x) = h_{w'}(f_A(x))$ for any $x \in V$.
2. For any unit vector $w \in \mathbb{R}^{\dim(V)}$, there is at least $1/(4d)$ fraction of the sample in S satisfies $x \in V$ and $|\langle w, f_A(x)\rangle| \geq 1/\left(2\sqrt{d}\right)$;

Proof. To prove the first property, let $w' = (AA^\intercal)^{-1}Aw/\|(AA^\intercal)^{-1}Aw\|_2$. It is easy to see that for any $x \in V$,

$$h_{w'}(f_A(x)) = \mathrm{sign}(\langle w', f_A(x)\rangle) = \mathrm{sign}(w^\intercal A^\intercal (AA^\intercal)^{-1}Ax) = \mathrm{sign}(w^\intercal x) = h_w(x)\ ,$$

where the second from last equality follows from $x \in V$.

For the second property, notice that

$$\mathop{\mathbf{E}}_{x\sim_u S\cap V}[\langle w, f_A(x)\rangle^2] = w^\intercal \mathop{\mathbf{E}}_{x\sim_u S\cap V}[f_A(x)(f_A(x))^\intercal]w \in \left[\frac{3}{4\dim(V)}, \frac{5}{4\dim(V)}\right]\ .$$

We suppose there is less than $1/(4d)$ fraction of the sample in S satisfies $x \in V$ and $|\langle w, f_A(x)\rangle| \geq 1/\left(2\sqrt{d}\right)$ and prove by contradiction. From Fact C.1, we have $|S\cap V| \geq (\dim(V)/d)|S|$, therefore,

$$\Pr_{x\sim_u S\cap V}\left(|\langle w, f_A(x)\rangle| \geq 1/\left(2\sqrt{d}\right)\right) \leq 1/(4\dim(V))\ .$$

Since $\langle w, f_A(x)\rangle^2$ is bounded between $[0,1]$, we have

$$\mathop{\mathbf{E}}_{x\sim_u S\cap V}[\langle w, f_A(x)\rangle^2] \leq 1/(4\dim(V)) + \frac{1-1/(4\dim(V))}{4d} < \frac{3}{4\dim(V)}\ ,$$

contradiction. This proves the second property. □

Given a multiset S of examples $(x_1, x_2) \in X_1 \times X_2$ from the co-training problem. The algorithm will apply the above Forster decomposition on both the set of x_1 and x_2. This gives two mappings $f_{A_1} : V_1 \to \mathbb{R}^{\dim(V_1)}$ and $f_{A_2} : V_2 \to \mathbb{R}^{\dim(V_2)}$ where $V_1, V_2 \neq 0$ are subspaces of $\mathbb{R}^d$. Then we will use the margin perceptron algorithm (a variant of which is shown in (Dunagan & Vempala, 2004)) described in the following fact. The version we use here is from (Diakonikolas et al., 2023a).

Fact C.3. [Lemma 16 of (Diakonikolas et al., 2023a)] Let $w^*, w^{(0)} \in \mathbb{R}^d$ be unit vectors such that $\langle w^*, w^{(0)}\rangle \geq \alpha$ for some $\alpha > 0$ and let $x_1, x_2, \cdots$ be any sequence of unit vectors in $\mathbb{R}^d$. Assume the following: $w^{(t+1)} \leftarrow w^{(t)} - x^{(t)}\langle x^{(t)}, w^{(t)}\rangle$ and let $t_0 \in \mathbb{Z}_+$ so that for all $t \in \mathbb{Z}_+$ with $t \leq t_0$, $|\langle x^{(t)}, w^{(t)}\rangle| \geq \beta\|w^{(t)}\|_2$ and $(\langle w^{(t)}, x^{(t)}\rangle)(\langle w^*, x^{(t)}\rangle) < 0$. Then, $t_0 \leq (2/\beta^2)\log(1/\alpha)$.

Now we are ready to describe the pseudocode (see Algorithm 5) for the algorithm in Theorem 4.3.

We now prove the main result of this section, Theorem 4.3.

Proof for Theorem 4.3. We first prove the correctness of the subroutine Algorithm 6. Notice that due to Fact C.2, we know there must be a pair of unit vectors w_1^* and w_2^* such that for all $(x_1, x_2) \in E$, $h_{w_1^*}(x_1) = h_{w_2^*}(x_2) = h^*(x_1, x_2)$, where h^* is the true classifier. Then notice that for random unit vector initialization w_1, w_2, there is $\Omega(1)$ probability that $\langle w_1, w_1^*\rangle = \Omega(1/d)$ and $\langle w_2, w_2^*\rangle = \Omega(1/d)$. Since we will try $c\log(T/\delta)$ many pairs of random initialization, with probability at least $1-\delta/(10T)$, one pair of them will satisfy the correlation. Therefore, given both correlations are $\Omega(1/d)$, according to Fact C.3, we can only make at most $O(d\log(d))$ many updates until all samples in E satisfy the margin

Algorithm 5 Co-training Halfspaces without Margin with Label Queries

Input: Let (X, H) be a learning problem where $X = \mathbb{R}^d \times \mathbb{R}^d$ and H be the class of homogeneous LTFs satisfies the co-training assumption with h^* being the true concept. Let D be a distribution over X that satisfies λ-weak dependence and let D_1 be the marginal distribution of x_1 satisfies the α-reflective symmetry, namely, for any $x_1 \in X_1$, $D(x_1)/D(-x_1) \in [\alpha, 1/\alpha]$ for some $\alpha \leq 1$. The algorithm is given sample access to distribution D and access to a label oracle. Let ϵ be the target accuracy and δ be the target failure probability.

Output: The algorithm uses at most $\tilde{O}(d^2 \log(\epsilon) \log(\delta))$ many queries to the label oracle, and With probability $1 - \delta$, it returns a hypothesis $\hat{h} : X \to \pm 1$ such that $\Pr_{(x_1,x_2)\sim D}\left(\hat{h}(x_1, x_2) \neq h^*(x_1, x_2)\right) \leq \epsilon$.

1. Let $T = cd\log(1/\epsilon)$ where c is a sufficient large positive constant. This will be the number of iterations we run the following steps of the algorithm.

2. Let S be a multiset of $m = c\log(T/\delta)d^4$ many examples of $(x_1, x_2) \sim D$ where c is a sufficiently large constant. Apply the Forster decomposition in Fact C.1 to the sets of x_1 and x_2 in S respectively. This gives two mappings $f_{A_1} : V_1 \to \mathbb{R}^{\dim(V_1)}$ and $f_{A_2} : V_2 \to \mathbb{R}^{\dim(V_2)}$ where $V_1, V_2 \neq 0$ are subspaces of $\mathbb{R}^d$.

3. Let $D_{f_{A_1},f_{A_2}}$ be the distribution of $(f_{A_1}(x_1), f_{A_2}(x_2))$ for $(x_1, x_2) \sim D$ conditioned on $x_1 \in V_1$ and $x_2 \in V_2$. Let E be a multiset $cd^2log(T/\delta)/(\alpha\lambda\epsilon^2/d)^2$ (c is a sufficiently large constant) many samples of $(x_1', x_2') \sim D_{f_{A_1},f_{A_2}}$. We then apply the subroutine Algorithm 6 on E and distribution $D_{f_{A_1},f_{A_2}}$, which with probability $1 - \delta/(2T)$ will return a unit vector $\hat{w}_1 \in \mathbb{R}^{\dim(V_1)}$ as a partial classifier on X_1, such that for its classifying region $R \stackrel{\text{def}}{=} \left\{x_1 \in X_1 : x_1 \in V_1 \wedge |\langle \hat{w}_1, f_{A_1}(x_1)\rangle| \geq 1/\left(2\sqrt{d}\right)\right\}$, it satisfies the following properties:

   (a) $\Pr_{(x_1,x_2)\sim D}(x_1 \in R) \geq 1/(5d)$; and

   (b) $\Pr_{(x_1,x_2)\sim D}(x_1 \in R \wedge h_{\hat{w}}(x_1) \neq h^*(x_1)) \leq \epsilon/(2T)$.

   Namely, this implies that this partial classifier classifies the region R which has at least $1/(5d)$ mass and makes at most $\epsilon/(2T)$ error in this region.

4. Let $\hat{w}_1^{(1)}, \cdots, \hat{w}_1^{(t)}$ be the partial classifiers we get from all previous t iterations and let $R_1, \cdots R_t$ be their corresponding classifying regions. If the unclassified mass $\Pr_{(x_1,x_2)\sim D}\left(x_1 \notin \bigcup_{i=1}^t R_i\right) > \epsilon/2$, then let D' be the distribution of $(x_1, x_2) \sim D$ conditioned on $x_1 \notin \bigcup_{i=1}^t R_i$ and repeat step 2 on the remaining mass D' (notice that we can always rejection sampling D' efficiently). Otherwise, we output the following decision list on x_1:
   If $x_1 \in R_1$, output $h_{\hat{w}_1^{(1)}}(x_1)$,
   elseif $x_1 \in R_2$, output $h_{\hat{w}_1^{(2)}}(x_1)$,
   elseif ...

Algorithm 6 Subroutine for Co-training Partial Classifier using Label Queries

Input: Let E be a multiset of $cd^2\log(T/\delta)/(\alpha\lambda\epsilon^2/d)^2$ many unlabeled examples $(x_1, x_2) \in \mathbb{R}^{d_1} \times \mathbb{R}^{d_2}$ and D be the distribution $D_{f_{A_1}, f_{A_2}}$ from Algorithm 5.

Output: The algorithm uses at most $O(d\log(d)\log(T/\delta))$ many queries to the oracle and output two unit vectors $\hat{w}_1 \in \mathbb{R}^{d_1}$ and $\hat{w}_2 \in \mathbb{R}^{d_2}$ such that at most $c'\alpha\lambda\epsilon^2/d$ fraction (for sufficiently small constant c' depending only on c) of the samples $(x_1, x_2) \in E$ does not satisfy the **margin consistency condition** w.r.t. $\hat{w}_1$ and $\hat{w}_1$ defined as:
If $|\langle \hat{w}_1, x_1\rangle| \geq 1/\left(2\sqrt{d}\right)$ and $|\langle \hat{w}_2, x_2\rangle| \geq 1/\left(2\sqrt{d}\right)$, then $h_{\hat{w}_1}(x_1) = h_{\hat{w}_2}(x_2)$.

1. We will initialize $w_1 \in \mathbb{R}^{d_1}$ and $w_2 \in \mathbb{R}^{d_2}$ as two random vectors drawn uniformly from the unit sphere.
2. If there is at most $c'\alpha\lambda\epsilon^2/d$ fraction of samples in E that does not satisfy the margin consistency condition w.r.t. w_1 and w_2, then, we output w_1 and w_2. Otherwise, we sample at most $O\left(\frac{\log(d)\log(T/\delta)d}{\alpha\lambda\epsilon^2}\right)$ many samples of (x_1, x_2) from D, and query the label when we see a sample (x_1, x_2) that does not satisfy the margin consistency condition. If there is no such sample, then output failure. Let (x_1, x_2) be this sample that we queried the label, then either $h_{w_1}(x_1)$ or $h_{w_2}(x_2)$ is not the correct label. We then use the margin perceptron in Fact C.3 to update either w_1 or w_2. We repeat this step $cd\log(d)$ many times where c is a sufficiently large constant.
3. If there is still any sample in E that does not satisfy the margin consistency, we go back to step 1 and try a different pair of random unit vectors. We repeat this at most $c\log(T/\delta)$ many times where c is a sufficiently large constant then output failure.

consistency condition. Furthermore, in each iteration, since there is at least $c'\alpha\lambda\epsilon^2/d$ (where c' is a sufficiently small constant depending on c) fraction of samples in E that does not satisfy the margin consistency. Using VC inequality, we know that each sample from $D_{f_{A_1}, f_{A_2}}$ does not satisfy the consistency condition with probability $\Omega(\alpha\lambda\epsilon^2/d)$. Therefore, accumulated over all iterations, the probability that we do not find an example that does not satisfy the consistency condition and output failure is at most $O(\delta/T)$ for each call of Algorithm 6. This proves the correctness of Algorithm 6.

Now we prove the correctness of Algorithm 5. We first prove the following observations for Step 3 in Algorithm 5.

Fact C.4. In Step 3 of Algorithm 5, let D be the original input distribution and D' be the distribution for the unclassified mass in this tth iteration. Let $R_i \stackrel{\text{def}}{=} \left\{x_1 \in X_1 : x_1 \in V_1 \wedge |\langle \hat{w}_1^{(i)}, f_{A_1}(x_1)\rangle| \geq 1/\left(2\sqrt{d}\right)\right\}$ where $\hat{w}_1^{(i)}$ is the $\hat{w}_1$ in ith iteration. Then with probability at least $\delta/(2T)$, the unit vector $\hat{w}_1^{(t)}$ will satisfy:

(a) $\Pr_{(x_1,x_2)\sim D'}(x_1 \in R_t) \geq 1/(5d)$; and

(b) Let the error region for partial classifier $h_{\hat{w}_1^{(t)}}$ be

$$R_{\text{error}} \stackrel{\text{def}}{=} \left\{x_1 \in X_1 : x_1 \notin \bigcup_{i=1}^{t-1} R_i \wedge x_1 \in V_1 \wedge h_{\hat{w}_1^{(t)}} \neq h^*(x_1)\right\},$$

then $\Pr_{(x_1,x_2)\sim D}(x_1 \in R_{\text{error}}) \leq \epsilon/(2T)$.

Proof. For Property (a), it is easy to see that any such region R_t can be expressed as a degree-2 polynomial threshold function and, therefore has VC-dimension at most $O(d^2)$. Given $\Pr_{(x_1,x_2)\sim_u S}(x_1 \in R_t) \geq 1/(4d)$ (this follows from Fact C.2) and $|S| = c\log(T/\delta)d^4$, we have that with probability at least $\delta/(10T)$,

$$\Pr_{(x_1,x_2)\sim D'}(x_1 \in R_t) \geq 1/(4d) - 1/(30d) \geq 1/(5d).$$

This proves Property (a).

For Property (b), we assume $\Pr_{(x_1,x_2)\sim D}(x_1 \in \mathbb{R}_{\text{error}}) \geq \epsilon/(2T)$ and prove by contradiction. We defined the classifying region from D on X_1 as $C_1 \stackrel{\text{def}}{=} \{x_1 : x_1 \notin \bigcup_{i=1}^{t-1} R_i \wedge x_1 \in R_t\}$, and on X_2 as $C_2 \stackrel{\text{def}}{=}$

$\left\{x_2 \in X_2 : \left|\left\langle \hat{w_2}^{(t)}, f_{A_2}(x_2)\right\rangle\right| \geq 1/\left(2\sqrt{d}\right)\right\}$ where $\hat{w_2}^{(t)}$ is the $\hat{w_2}$ in tth iteration. Furthermore, let $C_1^+ = \left\{x \in C_1 : h_{\hat{w_1}^{(t)}}\left(f_{A_1}(x_1)\right) = 1\right\}$ (similarly for C_1^-, C_2^+ and C_2^-). Same as what we have shown above, we have $\Pr_{D'}(C_2) \geq 1/(5d)$, therefore $\Pr_D(C_2) \geq \Omega(\epsilon/d)$. WLOG, this implies $\Pr_D\left(C_2^+\right) \geq \Omega(\epsilon/d)$. Then since $C_1^+ \cap R_{\text{error}}$ and $C_1^- \cap R_{\text{error}}$ are symmetric, we have $\Pr_D\left(C_1^- \cap R_{\text{error}}\right) \geq \alpha\epsilon$. Then from the λ-weak dependence condition,

$$\Pr_{(x_1,x_2\sim D)}\left(x_1 \in C_1^- \cap R_{\text{error}} \wedge x_2 \in C_2^+\right) \geq \lambda\alpha\epsilon^2/d\,.$$

This implies

$$\Pr_{(x_1,x_2)\sim D_{f_{A_1},f_{A_2}}}\left(h_{\hat{w_1}^{(t)}}(f_{A_1}(x_1)) = -1 \wedge h_{\hat{w_2}^{(t)}}(f_{A_2}(x_2)) = +1 \wedge x_1 \in C_1 \wedge x_2 \in C_2\right) \geq \lambda\alpha\epsilon^2/d\,.$$

However, from the correctness of subroutine Algorithm 6, we know that there must be a corresponding unit vector $\hat{w_2} \in \mathbb{R}^{\dim(V_2)}$ such that at most $c\alpha\lambda\epsilon^2/d$ fraction of samples in E does not satisfy the margin consistency w.r.t. $\hat{w_1}$ and $\hat{w_2}$. Combining this with the fact that this region has VC-dimension at most $O(d^2)$. We have such a region can have mass at most $c\alpha\lambda\epsilon^2/d$ for a sufficiently small constant c with probability $1 - \delta/(100T)$. This proves that there is a contradiction with probability $1 - \delta/(100T)$, therefore, property (b) holds with such probability. □

Then we show that the rejection sampling for $D_{f_{A_1},f_{A_2}}$ in Step 3 is efficient. Still, let D be the original input distribution and D' be the distribution for the unclassified mass in tth iteration. Notice that for any sample from D to pass the rejections sampling, we need $x_1 \notin \bigcup_{i=1}^{t-1} R_i$ and $x_1 \in V_1$ and $x_2 \in V_2$. According to Fact C.2, we have w.h.p

$$\Pr_{(x_1,x_2)\sim D'}(x_1 \in V_1) = \Pr_{(x_1,x_2)\sim D'}\left(x_1 \notin \bigcup_{i=1}^{t-1} R_i \wedge x_1 \in V_1\right) \geq 1/(5d)\,.$$

Since D' comes from at least $\Omega(\epsilon)$ mass of D, we have

$$\Pr_{(x_1,x_2)\sim D}\left(x_1 \notin \bigcup_{i=1}^{t-1} R_i \wedge x_1 \in V_1\right) = \Omega(\epsilon/d)\,.$$

Then similarly for x_2, we also have

$$\Pr_{(x_1,x_2)\sim D}(x_2 \in V_2) \geq \Pr_{(x_1,x_2)\sim D}\left(x_1 \notin \bigcup_{i=1}^{t-1} R_i \wedge x_2 \in V_2\right) = \Omega(\epsilon/d)\,.$$

According to λ-weak dependence property of D, we have the sampling efficiency from D is at least

$$\Pr_{(x_1,x_2)\sim D}\left(x_1 \notin \bigcup_{i=1}^{t-1} R_i \wedge x_1 \in V_1 \wedge x_2 \in V_1\right) = \Omega(\epsilon^2/d^2)\,.$$

This proves that the rejection sampling is efficient.

Given the above facts, since each iteration classifies $\Omega(1/d)$ fraction of the remaining mass, it only takes $T = O(d\log(1/\epsilon))$ many iterations to reduce the unclassified mass to at most $\epsilon/2$. Furthermore, it is easy to see the output decision list will incur error at most $T\epsilon/(2T) \leq \epsilon/2$ with failure probability accumulated from each iteration to be at most δ. Therefore, the output hypothesis has an error at most ϵ with failure probability at most δ. This completes the proof. □

D. The Label Complexity Obtained by (Wang & Zhou, 2008) Is Not Sufficient for Learning

The work in (Wang & Zhou, 2008) studied a different algorithm that combines co-training with pool-based active learning. They claim that if distribution D satisfies the λ-expansion assumption they defined, then every hypothesis class with VC dimension d can be learned by Algorithm 7 with $\tilde{O}(d\log(1/\epsilon)/\lambda)$ labeled examples. However, we explain here that the result obtained by Algorithm 7 does not make sense because the distributional assumption defined in their paper contradicts

Algorithm 7 Algorithm in (Wang & Zhou, 2008)

Input:

Unlabeled dataset $U = \{x^{(1)}, x^{(2)}, \cdots\}$ where each example $x^{(t)}$ is given as a pair $\left(x_1^{(t)}, x_2^{(t)}\right)$.

Process:

Ask the user to label m_0 unlabeled examples drawn randomly from D to compose the labeled dataset L.

for $i = 0, 1, \cdots s$ **do**

1. Train two classifier $h_1^{(i)}$ and $h_2^{(i)}$ consistent with L in each view, respectively;
2. Apply $h_1^{(i)}$ and $h_2^{(i)}$ to the unlabeled dataset U to find the contention points Q_i;
3. Ask the user to label m_{i+1} unlabeled examples drawn randomly from Q_i, then add them into L and delete from U.

Output:

$h_{\text{final}} = \text{combine}(h_1^{(s)}, h_2^{(s)})$

the assumption of co-training and can not be satisfied by any distribution in co-training. Furthermore, we will give very simple counter-examples, showing that the label complexity $\tilde{O}(d\log(1/\epsilon)/\lambda)$ is not enough for Algorithm 7 to learn a good hypothesis.

In their paper, they make the following assumption on the underlying distribution.

Definition D.1. A distribution D satisfies λ-expansion if for every $S_1 \subseteq X_1$ and for every $S_2 \subseteq X_2$

$$\Pr_{x\sim D}(S_1 \oplus S_2) \geq \lambda \min\{\Pr_{x\sim D}(S_1 \wedge S_2), \Pr_{x\sim D}(\bar{S_1} \wedge \bar{S_2})\},$$

where $(S_1 \oplus S_2)$ means for an example pair $x = (x_1, x_2)$ exactly one of $i \in [2]$ satisfies $x_i \in S_i$. A distribution D is λ-expanding with respect to hypothesis class $H_1 \times H_2$ if the above holds for all $S_1 \in H_1 \cap X_1$, $S_2 \in H_2 \cap X_2$, where for $j \in [2]$, $H_j \cap X_j = \{h \cap X_j \mid h \in H_j\}$.

They claim the following theorem based on their assumption.

Theorem D.2 (Theorem 1 in (Wang & Zhou, 2008))**.** *For data distribution D λ-expanding with respect to hypothesis class $H_1 \times H_2$, let ϵ, δ denote the final desired accuracy and confidence parameters. If $s = \lceil \frac{\log(\lambda/8\epsilon)}{\log 1/C} \rceil$ and $m_i = \frac{16}{\lambda}(4V\log(16/\lambda) + 2\log(8(s+1)/\delta))$, $(i = 0, 1, \ldots, s)$, Algorithm 7 will generate a classifier with error rate no more than ϵ with probability $1-\delta$. Here $V = \max\{VC(H_1), VC(H_2)\}$ and $C = \frac{\lambda/4+1/\lambda}{1+1/\lambda}$.*

We first point out that the above result is not meaningful, because the λ-expansion assumption they defined cannot be satisfied by any distribution in the co-training setting. Let h_1^*, h_2^* be the target hypothesis in the two views that are not constant. Let $S_1 := \{x_1 \in X_1 \mid h_1^*(x_1) = 1\}$ and $S_2 := \{x_2 \in X_2 \mid h_2^*(x_2) = 1\}$. Notice that for any distribution D and any hypothesis class H_1, H_2, if D is λ-expanding with respect to $H_1 \times H_2$, then S_1, S_2 according to definition should satisfy

$$\Pr_{x\sim D}(S_1 \oplus S_2) \geq \lambda \min\{\Pr_{x\sim D}(S_1 \wedge S_2), \Pr_{x\sim D}(\bar{S_1} \wedge \bar{S_2})\}.$$

However, according to the co-training assumption, every pair of examples (x_1, x_2) must have the same label, while $S_1 \oplus S_2$ implies that one of the x_i is labeled positive by h_i^* and the other one is labeled negative by the target hypothesis. Thus, according to the co-training assumption, $\Pr_{x\sim D}(S_1 \oplus S_2) = 0$. On the other hand, $\min\{\Pr_{x\sim D}(S_1 \wedge S_2), \Pr_{x\sim D}(\bar{S_1} \wedge \bar{S_2})\} > 0$ because the target hypothesis is not constant. In conclusion, the only choice for λ is 0, which implies that the algorithm for each round needs to query an infinite number of examples.

We remark that the notion of λ-expansion was originally defined in (Balcan et al., 2004). In the original definition, λ-expansion was only defined for D^+, which is the distribution of D conditioned on the label to be positive. A distribution D^+ has λ-expansion if for every $S_1 \subseteq X_1^+, S_2 \subseteq X_2^+$,

$$\Pr_{x\sim D^+}(S_1 \oplus S_2) \geq \lambda \min\{\Pr_{x\sim D^+}(S_1 \wedge S_2), \Pr_{x\sim D^+}((X_1^+ \setminus S_1) \wedge (X_2^+ \setminus S_2))\}.$$

The original definition of λ-expansion is significantly different from the definition used in (Wang & Zhou, 2008).

We now give counterexamples showing that such a label complexity is not sufficient to learn a good hypothesis. The simplest counterexample can be obtained when the true label is unbalanced. We can even assume the parameter $\lambda = 1$ in their bound. Then in the first round, $m_i = \Theta(d + \log(\log(1/\epsilon)/\delta))$. Assume the target hypothesis only has a probability of $1/(2m_i)$ to label an example to be negative. (We remark that in this case $\alpha := 1/(2m_i) \gg \epsilon$ when $\epsilon = o(1/d)$.) Then by Markov's inequality, with probability at least $1/2$, the first m_0 labeled examples are all positive. This implies that even the constant hypothesis is consistent with all labeled examples and has a very low error. However, after obtaining such a pair of hypotheses, the algorithm cannot continue, because no new examples will be queried. Even if we do not consider such an extreme case, where the initial hypotheses are constant, there are still other counterexamples. Assume the initial hypothesis h_i has error 1% over a random positive example but has error 100% over a random negative example. (Such cases can happen, because there are no negative examples in the initial round, so there is no guarantee of the prediction over a random negative example.) Under this assumption, every example queried by the algorithm will be a positive example, and the algorithm makes no progress on learning the negative examples.