Multitask Learning via Shared Features: Algorithms and Hardness

Konstantina Bairaktari BAIRAKTARI.K@NORTHEASTERN.EDU
Khoury College of Computer Sciences, Northeastern University

Guy Blanc GBLANC@STANFORD.EDU and **Li-Yang Tan** LYTAN@STANFORD.EDU
Department of Computer Science, Stanford University

Jonathan Ullman JULLMAN@CCS.NEU.EDU and **Lydia Zakynthinou**
ZAKYNTHINOU.L@NORTHEASTERN.EDU
Khoury College of Computer Sciences, Northeastern University

Editors: Gergely Neu and Lorenzo Rosasco

Abstract

We investigate the computational efficiency of multitask learning of Boolean functions over the d-dimensional hypercube, that are related by means of a feature representation of size $k \ll d$ shared across all tasks. We present a polynomial time multitask learning algorithm for the concept class of halfspaces with margin γ, which is based on a simultaneous boosting technique and requires only $\mathrm{poly}(k/\gamma)$ samples-per-task and $\mathrm{poly}(k \log(d)/\gamma)$ samples in total.

In addition, we prove a computational separation, showing that assuming there exists a concept class that cannot be learned in the attribute-efficient model, we can construct another concept class such that can be learned in the attribute-efficient model, but cannot be multitask learned efficiently—multitask learning this concept class either requires super-polynomial time complexity or a much larger total number of samples.

Keywords: multitask learning, shared feature representation, learning halfspaces, attribute-efficient learning, computational hardness

1. Introduction

A remarkable pattern in modern machine learning is that complex models often transfer surprisingly well to solve new tasks with very little additional data for that task—far less than one would need to solve that task from scratch. This sort of *multitask learning* (sometimes called *meta learning*, *transfer learning*, or *few-shot learning*) (Thrun, 1998; Thrun and Mitchell, 1995; Argyriou et al., 2008; Pontil and Maurer, 2013; Baxter, 1997; Thrun and Pratt, 2012; Sun et al., 2017) is possible because the tasks share some common structure that makes a model for one task relevant for solving the others. One useful structure is a *shared representation*. We assume that there is a low-dimensional representation of the data that is sufficient for solving every learning task—because the relevant features are shared across all tasks, we can pool the data for all tasks to find the representation, and because this is low-dimensional, each task can then be solved with relatively few samples.

The existence of low-dimensional representations often provably reduces the number of samples needed to solve each task (Maurer et al., 2016; Tripuraneni et al., 2020), and popular heuristics like MAML (Finn et al., 2017) or other gradient-based methods (e.g. (Nichol and Schulman, 2018; Raghu et al., 2019; Antoniou et al., 2019)) are often successful at multitask learning. However, much less is known about *computationally efficient algorithms* for exploiting these shared representations, or about complexity-theoretic barriers that are specific to multitask learning. While there is an elegant emerging body of research on provably efficient methods for multitask learning (Balcan et al., 2015;

Du et al., 2020; Tripuraneni et al., 2021; Thekumparampil et al., 2021; Collins et al., 2022), so far this work is currently limited to simple regression problems, or makes strong distribution assumptions, or both, so little is known about algorithms for classification tasks or for more general distributions.

In this work we focus on the particular setting of binary classification where the shared representation is simply a subset of the input features. In this setting we give the first computationally efficient algorithm for multitask learning of halfspaces, under a distribution-free margin assumption on the halfspaces but with no distributional assumptions. This result follows from a more general extension of *AdaBoost* (Freund and Schapire, 1997) to the multitask setting. We also prove a computational separation, showing that under a natural complexity assumption, there is a concept class such that: (1) The class can be learned in the *attribute-efficient model* (Littlestone, 1987)—any single task can be learned in polynomial time with sample complexity proportional to the number of relevant features. (2) The class cannot be multitask learned efficiently—the corresponding multitask problem where we want to learn multiple concepts over the same set of features cannot be solved efficiently unless the total data across all tasks is much larger.

1.1. Our Results

We now give a more detailed, but still high-level and informal statement of our two main results. To describe our results we need to introduce some notation, although we defer formal preliminaries to Section 2. We assume that there are n learning tasks. For each task there is a corresponding distribution $D^{(i)}$ over labeled examples $(x, y) \in \{\pm 1\}^d \times \{\pm 1\}$. We assume realizable tasks so that there is a known *concept class* $\mathcal{C}$ and for each task there is some $f^{(i)} \in \mathcal{C}$ such that $\Pr_{(x,y)\sim D^{(i)}}[f^{(i)}(x) = y] = 1$. Our learning algorithm is given m samples from each distribution $D^{(i)}$, for a total of mn samples, and must return $\hat{f}^{(1)}, \ldots, \hat{f}^{(n)}$ that label the data well on average over the tasks $\frac{1}{n}\sum_{i=1}^{n} \Pr_{(x,y)\sim D^{(i)}}[\hat{f}^{(i)}(x) = y] \approx 1$.

A naïve baseline solution is to solve each task separately, and our goal is to improve over this baseline. Since we can't hope to do so without some relationship between the functions $f^{(i)}$, in this work we consider cases where:

1. the functions in $\mathcal{C}$ have a small number of *relevant coordinates*, and
2. the total number of relevant coordinates among all of the functions $f^{(1)}, \ldots, f^{(n)}$ is small.

The set of up to k coordinates in total that are relevant for at least one of the tasks is what we call the *shared representation*—they are shared because they are the same across tasks, and they are a representation because we can think of these coordinates as a way of reducing examples to a lower-dimensional form that suffices for learning.

Efficient Multitask Learning of Halfspaces. First we consider the case where $\mathcal{C}$ consists of *k-sparse, γ-margin halfspaces*, meaning functions of the form $f(x) = \mathrm{sign}(\theta \cdot x)$ for some $\theta \in \mathbb{R}^d$ with at most k non-zero coordinates, which also satisfy the condition $|\theta \cdot x| \geq \gamma \cdot \|\theta\|_2$ for all inputs x. Note that our margin condition is weaker than the standard margin assumption. [1]

Since each halfspace is k-sparse, the naïve baseline of solving each task separately would require $m = O(k \log(d))$ samples for each task. In many applications, the number of total features d may

1. The typical margin condition used in the literature requires that $|\theta \cdot x| \geq \gamma' \|\theta\|_2 \|x\|_2$. For example this condition is used to analyze the classical Perceptron algorithm for learning halfspaces. A margin of γ in our definition corresponds to a margin of $\gamma'/\|x\|_2 = \gamma/\sqrt{d}$ using that definition.

be extremely large relative to the number of relevant features k, so we think of drawing $\log(d)$ samples-per-task as prohibitive. Our main theorem says that, assuming a shared representation of k features, we can indeed do better provided the number of tasks is at least $\log(d)$.

Theorem 1 (Informal) *Suppose we have n distributions $D^{(1)}, \dots, D^{(n)}$ each labeled by a k-sparse, γ-margin halfspace. There is a $\mathrm{poly}(d, k, n, m, 1/\gamma)$-time algorithm that draws m samples from each task and solves the multitask learning problem provided each task has $m \geq \mathrm{poly}(k/\gamma)$ samples and there are $nm \geq \mathrm{poly}(k \log(d)/\gamma)$ samples in total.*

The key feature of our result is that the number of samples *per task* can be entirely independent of d provided we have enough tasks to solve. Note that requirement of having at least $\mathrm{poly}(k \log(d))$ *total* samples is necessary up to polynomial factors—in the special case where all the tasks are identical, the problem is simply that of learning a single k-sparse γ-margin halfspace which requires $\Omega(k \log d)$ samples.

We achieve this result via a generalization of the AdaBoost algorithm for the multitask setting. In the single-task setting, a boosting algorithm takes a sequence of *weak learners* that predict the label slightly better than random, and combines them to obtain a *strong learner* that predicts the labels nearly perfectly. We give a variant for the multitask setting that takes a sequence of weak learners that have some advantage over random on average over tasks and combine them to obtain a new learner that predicts labels nearly perfectly.

Separating Multitask and Attribute-Efficient Learning. The strength of Theorem 1 is that its total sample complexity is close to the information-theoretically optimal number of samples needed to learn a single k-sparse halfspace, which is $O(k \log(d))$, and the overall running time is polynomial in d. Single-task learning algorithms with these properties—polynomial running time and sample-complexity growing polynomially in the number of relevant features, and only polylogarithmically in the number of irrelevant features—are called *attribute efficient* Littlestone (1987). For multi-task learning, the most interesting regime is where the total sample complexity also satisfies this property, and in this regime, multi-task learning strictly generalizes attribute-efficient learning by splitting the samples into multiple tasks, with only a few samples per task.

Attribute-efficient learning is a very challenging problem and large-margin halfspaces are one of the few classes with known attribute-efficient learning algorithms algorithm (Littlestone, 1987; Valiant, 1999). Perhaps any problem that has an attribute-efficient learner also has a polynomial-time multi-task learner under a shared-feature representation? We give strong evidence that this is not the case, by showing that, under a plausible assumption, there is some class $\mathcal{C}'$ consisting of functions of $k + \log\log(d)$ relevant variables, that can be learned in the attribute-efficient model but for which there is no efficient multitask learning algorithm.

Theorem 2 (Informal) *Assume there is some class of functions $\mathcal{C}$ with k relevant variables and a distribution D such that $\mathcal{C}$ has a sample-efficient learner over D, but every efficient learner for $\mathcal{C}$ requires many more samples. Then there is another concept class $\mathcal{C}'$ over $d' \lesssim d + \log\log(d)$ features with $k' \lesssim k + \log\log(d)$ relevant variables and a distribution D' such that:*

1. *$\mathcal{C}'$ can be attribute-efficiently learned over D'*
2. *$\mathcal{C}'$ can be multitask learned with few samples in exponential time, but*
3. *any polynomial-time algorithm that multitask learns $\mathcal{C}'$ over D' requires many more samples.*

An interesting feature of this result is that it gives *representation-independent* hardness, meaning it makes no assumptions about the form of the multitask learner's output.

We note that there are many concept classes that could be used to instantiate our assumption, an obvious example being k-sparse parities. For parities, $O(k\log(d))$ samples suffice for exponential-time algorithms, but the best known efficient algorithms require $\Omega(d^{1-1/k})$ samples (Klivans and Servedio, 2006). Our result is quantitative and can be instantiated with many choices of parameters, so we give a concrete example. If we assume that the current attribute-efficient learning algorithms for k-sparse parities are optimal, then we get another concept class that can be attribute-efficient learned with $s = \mathrm{poly}(k\log(d))$ samples, and can be multitask learned with $nm = O(k\log(d))$ total samples and $m = O(k\log\log(d))$ samples-per-task, but for which any polynomial-time multitask learner requires $nm = \Omega(d^{1-1/k})$ total samples whenever there are $n = \Omega(\log d)$ tasks.

We note that our separation only holds for attribute-efficient and multitask learning for the specific distribution E over the $d' - d$ new features of class $\mathcal{C}'$ of our example. In contrast, our positive result for learning halfspaces is distribution-free. It is an intriguing open problem to separate *distribution-free* attribute-efficient and multitask learning.

2. Preliminaries

For an integer n, we write $[n] = \{1, \ldots, n\}$. Each unlabeled sample is a vector of features $x \in \{\pm 1\}^d$. We use x_j to denote the j-th feature of sample x and x_{-j} to denote the vector x with the j-th coordinate removed. We consider Boolean classifiers of the form $f : \{\pm 1\}^d \to \{\pm 1\}$. We also write $v_{|v_j=b}$ to denote the vector v where the j-th coordinate is set to $b \in \{\pm 1\}$. We say a feature is *relevant* if changing its value has the potential to change the value of the function, and define $\mathrm{Rel}(f) = \{j \in [d] \mid \exists v_{-j} \in \{\pm 1\}^{d-1} \text{ s.t. } f(v_{|v_j=+1}) \neq f(v_{|v_j=-1})\}$.

We write $(x, y) \sim D$ when (x, y) is drawn from a distribution D. We denote the support of a distribution D by $\mathrm{Supp}(D)$.

In the multitask learning setting, we assume that there exist n tasks (or users) and each task $i \in [n]$ consists of a distribution $D^{(i)}$ over labeled samples in $\{\pm 1\}^d \times \{\pm 1\}$ and a classification function $f^{(i)}$ such that $y = f^{(i)}(x)$ for any $(x, y) \in \mathrm{Supp}(D)$. For each task, we receive a sample set of size m, denoted by $S^{(i)} = \left\{(x_1^{(i)}, y_1^{(i)}), \ldots, (x_m^{(i)}, y_m^{(i)})\right\}$ where $(x_j^{(i)}, y_j^{(i)})$ is the j-th sample drawn i.i.d. from $D^{(i)}$ and $y_j^{(i)} = f^{(i)}(x_j^{(i)})$ is its label. Our goal is to design a learning algorithm, which, given $S^{(1)}, \ldots, S^{(n)}$, returns *hypotheses* for each task, $h^{(1)}, \ldots, h^{(n)}$, with small average error. For each task $i \in [n]$ and hypothesis $h^{(i)}$, we define the population and training error by $\mathrm{error}^{(i)}(h^{(i)}) := \Pr_{(x,y)\sim D^{(i)}}[h^{(i)}(x) \neq y]$ and $\widehat{\mathrm{error}}^{(i)}(h^{(i)}) = \frac{1}{m}\sum_{(x,y)\in S^{(i)}} \mathbb{1}[h^{(i)}(x) \neq y]$, respectively and the population and training average error by $\text{avg-error}(h^{(1)}, \ldots, h^{(n)}) := \frac{1}{n}\sum_{i\in[n]} \mathrm{error}^{(i)}(h^{(i)})$ and $\widehat{\text{avg-error}}(h^{(1)}, \ldots, h^{(n)}) := \frac{1}{n}\sum_{i\in[n]} \widehat{\mathrm{error}}^{(i)}(h^{(i)})$, respectively. More formally, the single task setting, where $n = 1$, is equivalent to the PAC (*probably approximately correct*) learning model (in its realizable case), introduced by (Valiant, 1984), where our goal is to return a single hypothesis $h^{(1)}$ so that with probability at least $1 - \delta$ over the randomness of the dataset and the algorithm, $\mathrm{error}^{(1)}(h^{(1)}) \leq \varepsilon$, for given accuracy parameters $\varepsilon, \delta \in (0, 1)$. Similarly, in the multitask setting, our goal is to return hypotheses $h^{(1)}, \ldots, h^{(n)}$ so that with probability at least $1 - \delta$, $\text{avg-error}(h^{(1)}, \ldots, h^{(n)}) \leq \varepsilon$.

Multitask learning can be more sample-efficient than single-task learning when the tasks are related. Intuitively, if samples for one task are informative for learning a good hypothesis for another task, then pooling all samples to learn all tasks simultaneously may require less samples in total than learning each task separately. In order to formalize the relationship between tasks, we adopt one of the standard assumptions in the literature, that of a *shared-feature representation*. More specifically, this representation will take the form of a small subset of the features that contains the relevant variables for every task (although we can consider other constraints on the set of relevant features).

Definition 1 (Multitask Learning under Shared-Feature Representation) *Let $\mathcal{C}$ be a class of functions $f : \{\pm 1\}^d \to \{\pm 1\}$ and let $\mathcal{V} \subseteq 2^{[d]}$ be a collection of subsets of $[d]$. We say that $\mathcal{C}$ is $\mathcal{V}$-*multitask learnable for a class of distributions $\mathcal{D}$ with n tasks, m samples-per-task, and accuracy parameters $\varepsilon, \delta \in (0,1)$ *if there exists algorithm $\mathcal{A}$ such that $\forall f^{(1)}, \ldots, f^{(n)} \in \mathcal{C}$ that satisfy $\bigcup_{i\in[n]} \mathrm{Rel}(f^{(i)}) \in \mathcal{V}$, $\forall D^{(1)}, \ldots, D^{(n)} \in \mathcal{D}$, given m i.i.d. samples from each $D^{(i)}$ labeled by $f^{(i)}$, returns hypotheses $h^{(1)}, \ldots, h^{(n)}$ such that with probability at least $1-\delta$ over the samples and the algorithm,* $\text{avg-error}(h^{(1)}, \ldots, h^{(n)}) \le \varepsilon$*, i.e.,* $\frac{1}{n}\sum_{i\in[n]} \Pr_{(x,y)\sim D^{(i)}}[h^{(i)}(x) \neq y] \le \varepsilon$.

Note that, under this definition, it is not required that all features in $\mathcal{V}$ are relevant for all tasks but rather that for each task, the relevant features belong in $\mathcal{V}$.

Instead of minimizing the average error among all n tasks, another natural but stronger requirement would be to minimize the maximum error per task: $\max_{i\in[n]} \mathrm{error}^{(i)}$. This cannot be achieved under the shared-feature representation assumption in the general case. Suppose that the last $n-1$ tasks only depend on a single variable whereas the first task depends on the remaining $k-1$ relevant variables. In this case, only the samples of $S^{(1)}$ are informative for the first task and so returning a hypothesis $h^{(1)}$ with $\mathrm{error}^{(1)}(h^{(1)}) \le \varepsilon$ would require m to be as large as required for the single-task setting (for example, for $(k-1)$-sparse halfspaces, $m = O(k \log(d))$). This is in contrast to other settings (e.g. the collaborative learning setting (Blum et al., 2017)) which however make much stronger assumptions on the relationship between tasks.

Comparison with attribute-efficient learning We aim to study the computational efficiency of multitask learning under a shared-feature representation of size $k \ll d$ *together with* sample efficiency. If we are only concerned with sample complexity, we can solve the problem by brute-force in $\mathrm{poly}(d^k)$ time. If we are only concerned with computational complexity, then each user can learn their own task given enough samples-per-task. Thus the most interesting regime is when we require the number of samples in total and per task to be close to the information-theoretic minimum, and require an algorithm running in time $\mathrm{poly}(d,k)$. In this regime, if we were to assume all users have the same task, the problem would become exactly that of attribute efficient learning, and thus a natural question is whether multitask learning can be harder than attribute efficient learning or not.

We give a formal definition of attribute-efficient learning (Littlestone, 1987) here. In Section 4, we will construct a class of functions and distribution for which attribute-efficient learning is feasible in polynomial time but multitask learning is not, unless the total number of samples is much larger.

Definition 2 (Attribute-Efficient Learning) *Let $\mathcal{C}$ be a class of functions $f : \{\pm 1\}^d \to \{\pm 1\}$. Let $\mathrm{len}(\mathcal{C})$ denote the description length of the class.*[2] *We say that $\mathcal{C}$ is* attribute-efficient learnable for a class of distributions $\mathcal{D}$ with accuracy parameters $\varepsilon, \delta \in (0,1)$, *if there exists a $\mathrm{poly}(d)$-time algorithm $\mathcal{A}$ such that $\forall f \in \mathcal{C}$, $\forall D \in \mathcal{D}$, given $s = \mathrm{poly}(\mathrm{len}(\mathcal{C}))$ samples from D labeled by f, returns a hypothesis h such that with probability at most $1-\delta$ over the randomness of the sample and the algorithm,* $\Pr_{(x,y)\sim D}[h(x) \neq y] \le \varepsilon$.

2. We will use a binary encoding scheme.

For example, in the case where the concept class $\mathcal{C}$ is the class of parity functions on at most k features, the size of the class is $|C| = O\left(\binom{d}{k} \cdot 2^k\right)$ and its description length is $\mathrm{len}(\mathcal{C}) = \log|\mathcal{C}| = O(k\log(d))$. So in this case, attribute-efficient learning $\mathcal{C}$ would require a learning algorithm with sample complexity $s = \mathrm{poly}(k\log(d))$. As mentioned above, under the assumption that there exists a shared-feature representation of size $k \ll d$, multitask learning with n tasks and $m = s/n$ samples-per-task is strictly more general and is reduced to the attribute-efficient setting when all tasks have the same classifier and distribution over unlabeled examples. In this example, we would be interested in designing a multitask learning algorithm with n tasks, each having a (potentially different) parity function over a subset of the shared features $V \subseteq [d]$, $|V| \leq k$, and a distribution $D^{(i)}$ over $\{\pm1\}^d \times \{\pm1\}$, with $m = \mathrm{poly}(k\log(d))/n$ samples-per-task. What is more, since we consider the total number of features d to be too high, we would be interested in multitask learning algorithms which are accurate even in the regime where $m \ll \log(d) < s$.

3. Efficient multitask learning of halfspaces

In this section, we present a multitask learning algorithm for the case where each task is a large-margin halfspace classifier over some subset of a common set of features $V \subseteq [d]$ such that $|V| \leq k$. For each task $i \in [n]$, we have a function $f^{(i)} : \{\pm1\}^d \to \{\pm1\}$ of the form $f^{(i)}(x) = \mathrm{sign}(\theta^{(i)} \cdot x)$, where $\theta^{(i)} \in \mathbb{R}^d$. For simplicity, we only consider linear separators. Furthermore, we assume that each task's classifier $f^{(i)}$ is such that no example x falls too close to the boundary of the halfspace $\theta^{(i)}$. That is, we assume that all classifiers $f^{(i)}$ are *halfspaces with margin* γ, as defined below.

Definition 3 *For any $f : \{\pm1\}^d \to \{\pm1\}$ a classifier of the form $f(x) = \mathrm{sign}(\theta \cdot x)$, we say that such an f is* a halfspace with margin γ with respect to a distribution D *if it holds that $\frac{|\theta\cdot x|}{\|\theta\|_2} \geq \gamma$ for all $x \in \mathrm{supp}(D)$. We call γ the* margin.

Note that this is a weaker condition than the standard large-margin assumption that is used in single-task learning. The standard large-margin assumption requires that $\frac{|\theta\cdot x|}{\|\theta\|_2\|x\|_2} \geq \gamma$ for all $x \in \mathrm{Supp}(D)$. In our setting $\|x\|_2 = \sqrt{d}$ and thus our assumption is weaker. In fact, under the standard assumption, a single task can already be solved using a sample size of $\tilde{O}(1/\gamma^2\varepsilon)$ (see (Blum, 2006) and references therein), which is independent of d, yet if we executed the same analysis under our assumption the bound would translate to $\tilde{O}(d/\gamma^2\varepsilon)$

Remark 1 (Example margin: Sparse integer weighted halfspaces) *One simple class of halfspaces with large margin are k-sparse halfspaces with bounded integer weights. More formally, let $\mathcal{D}$ be any distribution over $\{\pm1\}^{d+1}$ where, for $x \sim \mathcal{D}$, $x_{i+1} = 1$ with probability 1 (the last index is used to encode a bias term). Consider any $w \geq 1$, $\theta \in \mathbb{R}^{d+1}$ where θ_i is an integer in $[-w, w]$ for all $i \leq d$, and $\theta_{d+1} = 1/2$. Then, for all $x \in \mathrm{supp}(D)$, $|\theta \cdot x| \geq 1/2$. In particular, if θ is k-sparse, then $f(x) = \mathrm{sign}(\theta \cdot x)$ has margin $\frac{1}{2w\sqrt{k}}$ which notably is independent of d.*

The main theorem of this section is the following.

Theorem 3 (Large-margin halfspaces are multitask learnable) *Let $\mathcal{V}_k = \{V \subseteq [d] \mid |V| \leq k\}$. Let $\mathcal{D}$ be a class of distributions and $\mathcal{C}_\gamma$ be the class of halfspaces with margin γ with respect to every $D \in \mathcal{D}$ (Definition 3). For any $\varepsilon, \delta \in (0,1)$, $m = \Omega\left(\frac{k^2\log^2(1/\varepsilon)}{\gamma^2\varepsilon}\right)$, $nm =$*

$\Omega\left(\frac{k^2\log(d)\log(1/\varepsilon)}{\gamma^2\varepsilon}+\frac{\log(1/\delta)}{\varepsilon}\right)$, *class $\mathcal{C}_\gamma$ is $\mathcal{V}_k$-multitask learnable for distributions $\mathcal{D}$ with n tasks, m samples-per-task, and accuracy parameters $\varepsilon, \delta \in (0,1)$, in time $O(\frac{nmdk^2\log(1/\varepsilon)}{\gamma^2})$.*

In particular, if $n = \Omega(\log d)$ and $\delta = d^{-O(1)}$, having $m = O(\frac{k^2\log^2(1/\varepsilon)}{\gamma^2\varepsilon})$ samples-per-task suffices. To prove Theorem 3, we need the next fact, ensuring that large-margin classifiers have a feature that is highly correlated with the label. This feature will serve as a weak learner in our analysis.

Fact 1 (Discriminator Lemma) *Let $f : \{\pm1\}^d \to \{\pm1\}$ be a halfspace classifier over features in V, such that $|V| \le k$. That is, $f(x) = \operatorname{sign}(\theta \cdot x)$ and $\theta_j = 0$ for all $j \notin V$. If f has margin γ w.r.t. D, there exists a feature $\ell \in V$ such that $\left|\mathbb{E}_{(x,y)\sim D}[f(x)\cdot x_\ell]\right| \ge \frac{\gamma}{\sqrt{k}}$.*

There exist several similar statements in the literature, see for example (Hajnal et al., 1993). We prove this version in Appendix A.1 for completeness. Our simultaneous boosting algorithm will use a generalization of this fact, stated in Lemma 1 below and proven in Appendix A.2.

```
BOOST($S^{(1)}, \dots, S^{(n)}, \mathcal{H}, t$) :
Input: Samples $S^{(1)}, \dots, S^{(n)}$ each with $m$ points, a concept class $\mathcal{H}$, and a step count $t$.
Output: A hypothesis for each of the $n$ tasks.

  Initialize the hypotheses $h^{(1)}, \dots, h^{(n)}$ each to the constant 0 functions.
  Repeat $t$ times:
    1. (Reweight points). For each $i \in [n]$ and $j \in [m]$, set $w_j^{(i)} = \exp(-y_j^{(i)} \cdot h^{(i)}(x_j^{(i)}))$
       and for each $i \in [n]$ set $W^{(i)} = \sum_{j\in[m]} w_j^{(i)}$.
    2. (Choose a weak learner). Choose $h^\star$ to maximize
          $$h^\star = \arg\max_{h\in\mathcal{H}} \sum_{i\in[n]} W^{(i)} \cdot \left(\sum_{j\in[m]} \frac{w_j^{(i)}}{W^{(i)}} \cdot y_j^{(i)} h(x_j^{(i)})\right)^2 \qquad (1)$$
    3. (Update hypotheses). For each $i \in [n]$, update $h^{(i)} \leftarrow h^{(i)} + \alpha^{(i)} \cdot h^\star$ where
          $$\alpha^{(i)} = \frac{1}{2}\ln\left(\frac{\sum_{j\in[m]} w_j^{(i)} \cdot \mathbb{1}[y_j^{(i)} = h^\star(x_j^{(i)})]}{\sum_{j\in[m]} w_j^{(i)} \cdot \mathbb{1}[y_j^{(i)} \neq h^\star(x_j^{(i)})]}\right).$$
  Output $h^{(1)}, \dots, h^{(n)}$.
```

Figure 1: Pseudocode for simultaneous boosting

Definition 4 (Simultaneous weak-learning assumption) *A class of weak learners, $\mathcal{H}$, satisfies the Γ-simultaneous weak-learning assumption for functions $f^{(1)}, \dots, f^{(n)}\{\pm1\}^d \to \{\pm1\}$ w.r.t. to a* *class of distributions $\mathcal{D}$, if for all input distributions $D^{(1)}, \ldots, D^{(n)} \in \mathcal{D}$ and nonnegative weights $w_1, \ldots, w_n$, there exists $h \in \mathcal{H}$ satisfying $\sum_{i\in[n]} w_i \cdot \mathbb{E}_{(x,y)\sim D^{(i)}}[f^{(i)}(x)h(x)]^2 \geq \Gamma \cdot \sum_{i\in[n]} w_i$.*

Lemma 1 (Simultaneous discriminator lemma) *Let $f^{(1)}, \ldots, f^{(n)} : \{\pm 1\}^d \to \{\pm 1\}$ be halfspaces over a set of features $V \subseteq [d]$, such that $|V| \leq k$, and that have margin γ w.r.t. every distribution in a class $\mathcal{D}$. Then, the class of single feature projection functions, $\mathcal{H}_{\text{proj}} := \{x \mapsto x_\ell \mid \ell \in [d]\}$ satisfies the $(\Gamma = \frac{\gamma^2}{k^2})$-simultaneous weak-learning assumption for functions $f^{(1)}, \ldots, f^{(n)}$ w.r.t $\mathcal{D}$.*

Lemma 2 (Simultaneous boosting fits a training set) *For any samples $S^{(1)}, \ldots, S^{(n)}$ of size m, weak-learning class $\mathcal{H}$, and number of steps t, let $\textsc{Boost}(S^{(1)}, \ldots, S^{(n)}, \mathcal{H}, t)$ return $h^{(1)}, \ldots, h^{(n)}$. For each $s \in [t]$, let $\Gamma_s \in [0,1]$ be unique value that satisfies the following expression when $h^\star$ is chosen in the s^{th} iteration:*

$$\sum_{i\in[n]} W^{(i)} \cdot \left(\sum_{j\in[m]} \frac{w_j^{(i)}}{W^{(i)}} \cdot y_j^{(i)} h^\star(x_j^{(i)}) \right)^2 = \Gamma_s \cdot \sum_{i\in[n]} W^{(i)}$$

Then,

$$\frac{1}{nm} \sum_{i\in[n]} \sum_{j\in[m]} \mathbb{1}[\text{sign}(h^{(i)}(x_j^{(i)})) \neq y_j^{(i)}] \leq \prod_{s\in[t]} \left(1 - \frac{\Gamma_s}{2}\right).$$

Note that Γ_s is defined in such a way that, if $\mathcal{H}$ satisfies the Γ-simultaneous weak-learning assumption, then $\Gamma_s \geq \Gamma$ for all iterations. The proof of Lemma 2 is in Appendix A.3

As an immediate corollary, if $\mathcal{H}$ satisfies the simultaneous weak-learning assumption, for an appropriate choice of t, simultaneous boosting will fit the training set with almost perfect accuracy.

Corollary 1 *Let any functions $f^{(1)}, \ldots, f^{(n)}$ and $\mathcal{H}$ be a class of weak-learners satisfying the Γ-simultaneous weak-learning assumption for $f^{(1)}, \ldots, f^{(n)}$. Then, for any samples $S^{(1)}, \ldots, S^{(n)}$ of size m labeled by $f^{(1)}, \ldots, f^{(n)}$ and $t = O\left(\frac{\log(1/\varepsilon)}{\Gamma}\right)$, $\textsc{Boost}(S^{(1)}, \ldots, S^{(n)}, \mathcal{H}, t)$ returns hypotheses $h^{(1)}, \ldots, h^{(n)}$ such that $\widehat{\text{avg-error}}(h^{(1)}, \ldots, h^{(n)}) \leq \varepsilon$.* [3]

We also bound the running time of simultaneous boosting. We include the proof in Appendix A.4.

Proposition 1 (Running time) *The running time of $\textsc{Boost}(S^{(1)}, \ldots, S^{(n)}, \mathcal{H}, t)$ when each $S^{(i)}$ has m samples is $O(nmt|\mathcal{H}|)$.*

The last step in proving Theorem 3 is bounding the generalization error. We first prove a general theorem on bounding generalization in the multitask setting using VC dimension in Section 3.1 and then apply it to learning large-margin halfspaces to complete our proof in Appendix A.9.

3. For conciseness, here we slightly abuse notation by writing $\widehat{\text{avg-error}}(h^{(1)}, \ldots, h^{(n)})$ to denote the average training error of the classification functions $\text{sign}(h^{(i)}(x))$.

3.1. Generalization based on VC dimension

The goal of this section is to prove the following theorem bounding the number of samples needed to generalize in the multitask setting. The formal version is given in Theorem 5.

Theorem 4 (Generalization in the multitask setting) *Let $\mathcal{C}$ be a class of functions $f : \{\pm 1\}^d \to \{\pm 1\}$ and $\mathcal{V} \subseteq 2^{[d]}$ be subsets of features. For $\mathrm{VC}(\mathcal{C} \mid \mathcal{V})$ as defined in Definition 8, any $\varepsilon, \delta \in (0,1)$, and $m = O\left(\mathrm{VC}(\mathcal{C} \mid \mathcal{V}) \cdot \frac{\log(1/\varepsilon)}{\varepsilon}\right)$, $nm = O\left(\frac{\log|\mathcal{V}| + \log(1/\delta)}{\varepsilon}\right)$, given random size-$m$ samples for each of n tasks, any $h^{(1)}, \dots, h^{(n)} \in \mathcal{C}$ with a shared-feature representation $V \in \mathcal{V}$ with $\widehat{\text{avg-error}}(h^{(1)}, \dots, h^{(n)}) \le \varepsilon$, will have $\text{avg-error}(h^{(1)}, \dots, h^{(n)}) \le 4\varepsilon$ with probability $1 - \delta$.*

$\mathrm{VC}(\mathcal{C} \mid \mathcal{V})$ will correspond to the VC dimension of the concept class once a representation is fixed. It can be substantially smaller than $\mathrm{VC}(\mathcal{C})$. For example, if $\mathcal{C}$ is the set of all k-sparse halfspaces, then $\mathrm{VC}(\mathcal{C}) = \Theta(k \log d)$. However, if all n tasks correspond to a halfspace over the same k features, then we take $\mathcal{V} := \{V \subseteq 2^{[d]} \mid |V| \le k\}$, and have $\mathrm{VC}(\mathcal{C} \mid \mathcal{V}) = k + 1$. Once a representation is fixed, $\mathcal{C}$ just corresponds to halfspaces over a set of k features, which has VC dimension $\Theta(k)$.

Theorem 4 roughly speaking, says that each task need only have enough samples to learn assuming the representation $V \in \mathcal{V}$ is already known, and the total number of samples should be enough to learn which representation V is used. We begin with some basic definitions.

Definition 5 (Generalization failure probability, single-task setting) *For a concept class $\mathcal{C}$ of functions $f : X \to \{\pm 1\}$, distribution D over $X \times \{\pm 1\}$, error parameter $\varepsilon > 0$, and sample size m, we define $\delta_{\mathrm{gen}}(\mathcal{C}, m, \varepsilon, D)$ to be the probability, over a random sample S of m points from D, that there exists some $f \in \mathcal{C}$ that has at most ε error on the sample S but for which $\Pr_{(x,y)\sim D}[f(x) \neq y] \ge 4\varepsilon$. We define the generalization failure probability of $\mathcal{C}$ with sample size m and error parameter ε to be $\delta_{\mathrm{gen}}(\mathcal{C}, m, \varepsilon) := \sup_{\textit{distribution } D} \delta_{\mathrm{gen}}(\mathcal{C}, m, \varepsilon, D)$.*

Any algorithm that returns a hypothesis $h \in \mathcal{C}$ with less than ε error on m random samples will learn to error $< 4\varepsilon$ with probability at least $1 - \delta_{\mathrm{gen}}(\mathcal{C}, m, \varepsilon)$. We extend this notion to multitask learning.

Definition 6 (Generalization failure probability, multitask setting) *For a concept class $\mathcal{C}$ of functions $f : \{\pm 1\}^d \to \{\pm 1\}$, $\mathcal{V} \subseteq 2^{[d]}$ a collection of subsets of features, distributions $D^{(1)}, \dots, D^{(n)}$ over $\{\pm 1\}^d \times \{\pm 1\}$, error parameter ε, number of tasks n, and samples-per-task m, we define $\delta_{\mathrm{gen}}(\mathcal{C}, \mathcal{V}, n, m, \varepsilon, D^{(1)}, \dots, D^{(n)})$ to be the probability over random samples $S^{(i)} \sim (D^{(i)})^m$, $i \in [n]$, that there are $h^{(1)}, \dots, h^{(n)} \in \mathcal{C}$ satisfying $\bigcup_{i \in [n]} \mathrm{Rel}(h^{(i)}) \in \mathcal{V}$ for which simultaneously $\widehat{\text{avg-error}}(h^{(1)}, \dots, h^{(n)}) \le \varepsilon$ but $\text{avg-error}(h^{(1)}, \dots, h^{(n)}) \ge 4\varepsilon$. Then, we define, $\delta_{\mathrm{gen}}(\mathcal{C}, \mathcal{V}, n, m, \varepsilon) := \sup_{\textit{distributions } D^{(1)}, \dots, D^{(n)}} \delta_{\mathrm{gen}}(\mathcal{C}, \mathcal{V}, n, m, \varepsilon, D^{(1)}, \dots, D^{(n)})$.*

Once again, any algorithm that returns hypotheses $h^{(1)}, \dots, h^{(n)}$ with average error at most ε on the training set, will, given m samples per task, with probability $1 - \delta_{\mathrm{gen}}(\mathcal{C}, \mathcal{V}, n, m, \varepsilon)$, learn hypotheses with $\text{avg-error}(h^{(1)}, \dots, h^{(n)}) < 4\varepsilon$. We extend classical results bounding $\delta_{\mathrm{gen}}(\mathcal{C}, m, \varepsilon)$ based on VC dimension to the multitask setting. We begin with the basic definitions of VC theory.

Definition 7 (VC dimension (Vapnik and Chervonenkis, 1971)) *For any concept class $\mathcal{C}$ and domain X, the shattering number of n points is $\Pi_{\mathcal{C}}(n) := \max_{x_1, \dots, x_n \in X} |\{(c(x_1), \dots, c(x_n)) : c \in \mathcal{C}\}|$, i.e., the maximum number of unique assignments functions in the concept class can take on a set of n points. The VC dimension of $\mathcal{C}$, denoted $\mathrm{VC}(\mathcal{C}) := \sup\{d \in \mathbb{N} : \Pi_{\mathcal{C}}(d) = 2^d\}$, is the cardinality of the largest data set for which every unique assignment to that set is satisfied by a function in $\mathcal{C}$.*

In the multitask setting, rather than scaling with the VC dimension of $\mathcal{C}$, the number of samples needed per task will scale with the VC dimension after the representation is already known.

Definition 8 (VC dimension given representation) *Given a concept class* $\mathcal{C} \subseteq \{f : \{\pm 1\}^d \to \{\pm 1\}\}$ *and* $\mathcal{V} \subseteq 2^{[d]}$ *a collection of subsets of features, for any* $V \subseteq [d]$*, let* $(\mathcal{C} \mid V) := \{f \in \mathcal{C} : \mathrm{Rel}(f) \subseteq V\}$*, be the concepts consistent with a shared representation* V*. We define the VC dimension of* $\mathcal{C}$ *given the representation* $\mathcal{V}$ *to be* $\mathrm{VC}(\mathcal{C} \mid \mathcal{V}) := \max_{V \in \mathcal{V}} \mathrm{VC}(\mathcal{C} \mid V)$.

We are most interested in settings where $\mathrm{VC}(\mathcal{C} \mid \mathcal{V}) \ll \mathrm{VC}(\mathcal{C})$. We can now formalize the main result of this section.

Theorem 5 (Generalization bound in the multitask setting, formal version of Theorem 4) *Given a concept class* $\mathcal{C}$ *of functions* $f : \{\pm 1\}^d \to \{\pm 1\}$, $\mathcal{V} \subseteq 2^{[d]}$ *a collection of subsets of features, and accuracy parameters* ε, δ*, set* $m = O\left(\mathrm{VC}(\mathcal{C} \mid \mathcal{V}) \cdot \frac{\log(1/\varepsilon)}{\varepsilon}\right)$*, and* n *s.t.* $nm = O\left(\frac{\log|\mathcal{V}| + \log(1/\delta)}{\varepsilon}\right)$*. Then* $\delta_{\mathrm{gen}}(\mathcal{C}, \mathcal{V}, n, m, \varepsilon) \le \delta$.

The proof of Theorem 5 follows Blumer, Ehrenfeucht, Haussler, and Warmuth's classical VC generalization bounds (Blumer et al., 1989), with appropriate modifications made for the multitask setting. To do so, we need to define an extension of shattering numbers to the multitask setting.

Definition 9 *For any function* $f : X \to Y$ *and sample* $S = (x_1, \ldots, x_m) \in X^m$*, we use* $f(S)$ *as shorthand for the vector* $(f(x_1), \ldots, f(x_m))$*. For any concept class* $\mathcal{C}$ *of functions* $f : \{\pm 1\}^d \to \{\pm 1\}$ *and* $\mathcal{V} \subseteq 2^{[d]}$ *a collection of subsets of features, we define the shattering number for* n *tasks and* m *samples-per-task, denoted* $\Pi_{\mathcal{C},\mathcal{V}}(n, m)$*, to be*

$$\max_{S^{(1)}, \ldots, S^{(n)} \in (\{\pm 1\}^d)^m} \left| \left\{ (f^{(1)}(S^{(1)}), \ldots, f^{(n)}(S^{(n)})) : f^{(1)}, \ldots, f^{(n)} \in \mathcal{C} \text{ and } \cup_{i \in [n]} \mathrm{Rel}(f^{(i)}) \in \mathcal{V} \right\} \right|$$

to be the maximum number of unique assignments for n *tasks each with* m *data points.*

We can now state the main technical lemma of this section.

Lemma 3 *For any concept class* $\mathcal{C}$ *of functions* $f : \{\pm 1\}^d \to \{\pm 1\}$, $\mathcal{V} \subseteq 2^{[d]}$ *a collection of subsets of features, error parameter* $\varepsilon > 0$*, number of tasks* n*, and samples-per-task* m*, if* $nm\varepsilon \ge 2$*, then* $\delta_{\mathrm{gen}}(\mathcal{C}, \mathcal{V}, n, m, \varepsilon) \le 2 \cdot \Pi_{\mathcal{C},\mathcal{V}}(n, 2m) \cdot \exp(-nm\varepsilon/10)$.

We'll collect two basic probability facts that will be used in the proof of Lemma 3.

Fact 2 (Application of Chebyshev's inequality) *Let* x *be a random variable in* $\mathbb{R}$ *with mean* μ *and for which* $\mathrm{Var}[x] \le \frac{\mu^2}{8}$*. Then,* $\Pr[x \ge \mu/2] \ge \frac{1}{2}$.

The second fact we need is a slight twist on standard Chernoff bounds. We prove it in Appendix A.5.

Proposition 2 *For any* $m, n, \ell \in \mathbb{N}$*, suppose there are* n *groups of* $2m$ *items, and of the* $2nm$ *items,* ℓ *are marked. If we uniformly select* m *items from each of the* n *groups* without replacement*, the probability at least* $\frac{2\ell}{3}$ *items are selected in total is at most* $\exp(-\ell/20)$).

The proof of Lemma 3 in Appendix A.6 mostly follows the exposition in (Kearns and Vazirani, 1994, §3.5.2) of (Blumer et al., 1989)'s classic result, with appropriate modifications for the multitask setting.

We are now nearly ready to prove Theorem 5. As in classical VC theory, we'll apply the Sauer-Shelah Lemma.

Fact 3 (Sauer-Shelah Lemma (Sauer, 1972; Shelah, 1972)) *For any concept class $\mathcal{C}$ and sample-size $m \geq \mathrm{VC}(\mathcal{C})$, $\Pi_C(m) \leq \left(\frac{em}{\mathrm{VC}(\mathcal{C})}\right)^{\mathrm{VC}(\mathcal{C})}$.*

We'll plug in Fact 3 as a blackbox into the following proposition, which we prove in Appendix A.7, to bound shattering numbers in the multitask setting.

Proposition 3 *For any concept class $\mathcal{C}$ of functions $f : \{\pm1\}^d \to \{\pm1\}$, $\mathcal{V} \subseteq 2^{[d]}$ a collection of subsets of features and sample size $m \geq \mathrm{VC}(\mathcal{C} \mid \mathcal{V})$,*

$$\Pi_{\mathcal{C},\mathcal{V}}(n, m) \leq \sum_{V \in \mathcal{V}} (\Pi_{(\mathcal{C}|V)}(m))^n \leq |\mathcal{V}| \cdot \left(\frac{em}{\mathrm{VC}(\mathcal{C} \mid \mathcal{V})}\right)^{n \cdot \mathrm{VC}(\mathcal{C}|\mathcal{V})}.$$

Theorem 5 is an easy consequence of Lemma 3 and Proposition 3 (proven in Appendix A.8). Then, in Appendix A.9, we apply the machinery from Section 3.1 to complete the proof of Theorem 3.

4. Lower Bound

We present a lower bound that separates attribute-efficient learning from multitask learning with near-optimal computational and sample efficiency. We start with a function class $\mathcal{C}_k$, which contains exactly one function g_V for each subset of features V of size at most k. Therefore, knowing the set of relevant features V suffices to learn the function g_V. Now, we assume that our dataset consists of samples whose label is either a bit from a t-secret sharing scheme with probability ε, where the secret is V, or the output of the original function g_V from $\mathcal{C}_k$ otherwise. We show that this new function class $\mathcal{C}_k^{(t)}$ is attribute-efficient learnable for this data distribution because, for number of samples that is linear in the description length of the function class $\mathrm{len}(\mathcal{C}_k^{(t)})$, we see all the bits of all the t shares with high probability. In this way, we can recover the secret V, which uniquely identifies the labeling function. In the multitask setting, every user has their own function from $\mathcal{C}_k^{(t)}$, corresponding to different subsets of features $V^{(1)}, \ldots, V^{(n)}$. We consider the case where every user has $t-1$ samples, so recovering the secret through the shares is impossible. Assuming that the total number of samples is less than the number of samples that are necessary to learn $\mathcal{C}_k$ in $\mathrm{poly}(d)$ time, we prove that, by ignoring the shares and using only the rest of the samples, we can multitask learn $\mathcal{C}_k^{(t)}$ for the same data distribution in time $d^k\mathrm{poly}(d)$, but we cannot multitask learn it in any $\mathrm{poly}(d)$ time.

Let us now describe our construction in more detail. For $k \in \mathbb{N}$, the function class $\mathcal{C}_k$ is a class of Boolean functions $g_V : \{\pm1\}^d \to \{\pm1\}$ that depend only on the variables in set $V \in \mathcal{V}$, where $\mathcal{V} = \{V \subseteq [d] \mid |V| \leq k\}$. Namely, $\mathcal{C}_k = \{g_V : \{\pm1\}^d \to \{\pm1\} : V \in \mathcal{V}\}$. For example, $\mathcal{C}_k$ could be the class of parity functions over at most k variables, that is, $g_V(x) = \prod_{i \in V} x_i$ when $|V| \leq k$.

We can describe the set V using a string $\tilde{V}$ in $\{\pm1\}^{k(\log(d)+1)}$. More specifically, we encode every coordinate in V into a $\{\pm1\}$ string of length $\log(d)$ and use the extra first (most significant) bit to denote that this is a valid coordinate by setting it to $+1$. If $|V| \leq k$, then we encode each of the remaining $k - |V|$ coordinates into a string of length $\log(d) + 1$ that consists only of -1. Since the binary string $\tilde{V}$ uniquely identifies the corresponding g_V, the description length of $\mathcal{C}_k$ is $\mathrm{len}(\mathcal{C}_k) = k(\log(d) + 1)$.

For our lower bound, we consider that set V is a secret that we want to share using a simple secret-sharing scheme. A t-secret sharing scheme "hides" the secret in t shares so that: (1) the secret can be reconstructed using the t shares, and (2) the secret cannot be reconstructed with the knowledge of any $t-1$ or fewer shares. In addition to the secret V, our scheme will receive as input a vector $r \in \{\pm 1\}^{tk(\log(d)+1)}$, which represents the randomness of the scheme. For every $p \in [t]$ we denote the p^{th} share of the secret-sharing scheme with input secret V, randomness vector r, and threshold t, by $\text{share}^t(V;r)_p$. It is generated as follows:

$$\text{share}^t(V;r)_p = \begin{cases} \left(r_{(p-1)k(\log(d)+1)+1}, \dots, r_{pk(\log(d)+1)}\right), & \text{if } p \in [t-1] \\ \left(\left(\prod_{p'=1}^{t-1} r_{(p-1)k(\log(d)+1)+1}\right) \tilde{V}_1, \dots, \left(\prod_{p'=1}^{t-1} r_{pk(\log(d)+1)}\right) \tilde{V}_{k(\log(d)+1)}\right), & \text{if } p = t. \end{cases}$$

To reconstruct the secret, we compute $(\prod_{p=1}^{t} \text{share}^t(V;r)_{p,1}, \dots, \prod_{p=1}^{t} \text{share}^t(V;r)_{p,k(\log(d)+1)})$ which must be equal to $\tilde{V}$. Then, we split the string into k substrings of length $\log(d)+1$ and interpret each separately. If a substring starts with -1, we ignore it. Otherwise, we ignore the first bit and consider that the element whose binary encoding we see is in V.

Using this secret sharing scheme, we define the class of functions $\mathcal{C}_k^{(t)}$ and distribution E_ε, for which we prove the separation of Theorem 7 as follows:

Definition 10 *Given function class $\mathcal{C}_k$, we define the class $\mathcal{C}_k^{(t)} = \{f_{V,r} \mid V \in \mathcal{V}$ and $r \in \{\pm 1\}^{tk(\log(d)+1)}\}$ of functions $f_{V,r} : \{\pm 1\}^{d+\log(tk(\log d+1))+1} \to \{\pm 1\}$ defined by*

$$f_{V,r}(x,p,q,b) = \begin{cases} \text{share}^t(V;r)_{p,q}, & \text{if } b = 1 \\ g_V(x), & \text{if } b = -1, \end{cases}$$

where $g_V \in \mathcal{C}_k$, $q \in \{\pm 1\}^{\log(k(\log d+1))}$ and $p \in \{\pm 1\}^{\log(t)}$ indicate the q^{th} bit of the p^{th} share.

Definition 11 *We let distribution $E_\varepsilon = \text{Ber}(1/2)^{\log(tk(\log(d)+1))} \times \text{Ber}(\varepsilon)$, where $\text{Ber}(\varepsilon)$ denotes the Bernoulli distribution with support $\{\pm 1\}$ and parameter ε.*

We draw (p,q,b) from E_ε. That is, p and q are chosen uniformly at random, whereas the bit $b=1$ with probability ε and -1 otherwise.We now state our separation in Theorem 6 and Theorem 7, which we prove in Appendix B.1 and Appendix B.2 respectively.

Theorem 6 *Function class $\mathcal{C}_k^{(t)}$ (Definition 10) is attribute-efficient learnable for the class of distributions D' over labeled examples, where each example's features are drawn from $D_x \times E_\varepsilon$ s.t. $D_x \in \Delta(\{\pm 1\}^d)$ and E_ε as in Definition 11, with accuracy parameters (ε, δ), sample complexity $N = \tilde{O}\left(tk\log(d)\frac{\log(1/\delta)}{\varepsilon}\right) = \tilde{O}(\text{len}(\mathcal{C}_k^{(t)}))$, and time complexity $O(N)$.*

Theorem 7 *Assume that at least $s \le a(d)$ examples are necessary to learn class $\mathcal{C}_k$ in $\text{poly}(d)$ time for distribution D over labeled examples, where $a(d)$ is a polynomial in d and the features are drawn from $D_x \in \Delta(\{\pm 1\}^d)$, for all accuracy parameters $\varepsilon \in (0,1/2)$, $\delta \in (0,1)$. Let set of subsets $\mathcal{V}_k^{(t)} = \{V = V_x \cup \{d+1, \dots, d+\log(tk(\log d+1))+1\} \mid V_x \subseteq [d], |V_x| \le k\}$. Then for distribution D', where the features are drawn from $D_x \times E_\varepsilon$, and n tasks with $m \ge O\left(\frac{k}{\varepsilon}\log(\frac{1}{\varepsilon})\log(\frac{n}{\delta})\right)$ samples-per-task, such that $O\left(\frac{1}{\varepsilon}(k\log(d)+\log(\frac{1}{\delta}))\right) \le nm < s$, class $\mathcal{C}_k^{(m+1)}$ is $\mathcal{V}_k^{(m+1)}$-multitask learnable in $(2d)^k a(d)$ time with accuracy parameters $(2\varepsilon, \delta)$, but $\mathcal{C}_k^{(m+1)}$ is not $\mathcal{V}_k^{(m+1)}$-multitask learnable in any $\text{poly}(d)$ time with accuracy parameters $(\frac{\varepsilon}{32}, \delta)$.*

For threshold $t = m + 1$, Theorem 6 says that for $N \geq \tilde{O}(\frac{mk}{\varepsilon}\log(d)\log(\frac{1}{\delta}))$, samples we can attribute-efficient learn $\mathcal{C}_k^{(m+1)}$ for distribution over the features of the examples $D_x \times E_\varepsilon$ with parameters (ε, δ). But by Theorem 7, for $nm = N \in [O(\frac{1}{\varepsilon}(k\log(d) + \log(\frac{1}{\delta}))), s)$ samples we cannot $\mathcal{V}_k^{(m+1)}$-multitask learn $\mathcal{C}_k^{(m+1)}$ for the same distribution in polynomial time with parameters $(\varepsilon/32, \delta)$. Combining the two theorems, we see that we can attribute-efficient learn $\mathcal{C}_k^{(m+1)}$ but not multitask learn it in polynomial time with respect to the size of the input, when each task has m samples for $m \in \left[O\left(\frac{k}{\varepsilon}\log(\frac{1}{\varepsilon})\log(\frac{n}{\delta})\right), \tilde{O}\left(\frac{\varepsilon\cdot s}{k\log d\log(\frac{1}{\delta})}\right)\right)$.

Learning k-sparse parities. To make the result more concrete we look at the case where $\mathcal{C}_k$ consists of k-sparse parities. The best known polynomial time algorithms to learn this $\mathcal{C}_k$ require $\Omega(d^{1-1/k})$ samples (Klivans and Servedio, 2006). If $d^{1-1/k}$ samples are required to learn this class in polynomial time, then for $N \in \left[\tilde{O}(\frac{k^2\log(d)}{\varepsilon^2}\log(\frac{1}{\delta})\log(\frac{n}{\delta})), d^{1-1/k}\right)$ samples we can multitask learn $\mathcal{C}_k^{(m+1)}$ in $(2d)^k a(d)$ time for $m = O(\frac{k}{\varepsilon}\log(\frac{1}{\varepsilon})\log(\frac{n}{\delta}))$ and attribute-efficient learn it in some $\text{poly}(d)$ time, but we need more samples in total to multitask learn it in any $\text{poly}(d)$ time.

Acknowledgments

We thank Adam Smith and Zhiwei Steven Wu for helpful discussions. GB and LT were supported by NSF grant CCF-1942123. KB, JU, LZ were supported by NSF grants CCF-1750640, CNS-1816028, CNS-212060. KB was also supported by NSF grants CCF-1909314 and CCF-1750716, and LZ was also supported by a Facebook Fellowship.

References

Antreas Antoniou, Harri Edwards, and Amos Storkey. How to train your MAML. In *International Conference on Learning Representations*, ICLR '19, 2019.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine learning*, 73(3):243–272, 2008.

Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. Efficient representations for lifelong learning and autoencoding. In *Conference on Learning Theory*, pages 191–210. PMLR, 2015.

Jonathan Baxter. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine learning*, 28(1):7–39, 1997.

Avrim Blum. Random Projection, Margins, Kernels, and Feature-Selection. In *Subspace, Latent Structure and Feature Selection*, pages 52–68. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-34138-3.

Avrim Blum, Nika Haghtalab, Ariel D Procaccia, and Mingda Qiao. Collaborative pac learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/186a157b2992e7daed3677ce8e9fe40f-Paper.pdf.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989.

Liam Collins, Aryan Mokhtari, Sewoong Oh, and Sanjay Shakkottai. Maml and anil provably learn representations. *arXiv preprint arXiv:2202.03483*, 2022.

Simon S Du, Wei Hu, Sham M Kakade, Jason D Lee, and Qi Lei. Few-shot learning via learning the representation, provably. *arXiv preprint arXiv:2002.09434*, 2020.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, ICML '17, pages 1126–1135. PMLR, 2017.

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

András Hajnal, Wolfgang Maass, Pavel Pudlák, György Turán, and Márió Szegedy. Threshold circuits of bounded depth. *J. Comput. Syst. Sci.*, 46(2):129–154, apr 1993. ISSN 0022-0000.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. In *The collected works of Wassily Hoeffding*, pages 409–426. Springer, 1994.

Michael J Kearns and Umesh Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

Adam R Klivans and Rocco A Servedio. Toward attribute efficient learning of decision lists and parities. *Journal of Machine Learning Research*, 7(4), 2006.

Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1987.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge University Press, 2nd edition, 2017. ISBN 110715488X, 9781107154889.

Alex Nichol and John Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2018.

Massimiliano Pontil and Andreas Maurer. Excess risk bounds for multitask learning with trace norm regularization. In *Conference on Learning Theory*, pages 55–76. PMLR, 2013.

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. In *International Conference on Learning Representations*, ICLR '19, 2019.

N. Sauer. On the density of families of sets. *Journal of Combinatorial Theory, Series A*, 13(1):145 – 147, 1972. ISSN 0097-3165. doi: https://doi.org/10.1016/0097-3165(72)90019-2. URL http://www.sciencedirect.com/science/article/pii/0097316572900192.

Saharon Shelah. A combinatorial problem; stability and order for models and theories in infinitary languages. *Pacific Journal of Mathematics*, 41(1):247–261, 1972.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

Kiran Koshy Thekumparampil, Prateek Jain, Praneeth Netrapalli, and Sewoong Oh. Sample efficient linear meta-learning by alternating minimization. *arXiv preprint arXiv:2105.08306*, 2021.

Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998.

Sebastian Thrun and Tom M Mitchell. Lifelong robot learning. *Robotics and autonomous systems*, 15(1-2):25–46, 1995.

Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

Nilesh Tripuraneni, Michael Jordan, and Chi Jin. On the theory of transfer learning: The importance of task diversity. *Advances in Neural Information Processing Systems*, 33:7852–7862, 2020.

Nilesh Tripuraneni, Chi Jin, and Michael Jordan. Provable meta-learning of linear representations. In *International Conference on Machine Learning*, ICML '21, pages 10434–10443. PMLR, 2021.

L. G. Valiant. A Theory of the Learnable. *Commun. ACM*, 27(11):1134–1142, November 1984. ISSN 0001-0782. doi: 10.1145/1968.1972.

Leslie G Valiant. Projection learning. *Machine Learning*, 37(2):115–130, 1999.

V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, January 1971. doi: 10.1137/1116025.

Appendix A. Proofs from Section 3

A.1. Proof of Fact 1

Proof By assumption, $\forall x \in \mathrm{supp}(D)$, $\frac{|\theta \cdot x|}{\|\theta\|_2} \geq \gamma$. Then,

$$\begin{aligned}
\|\theta\|_2 \gamma &\leq \mathop{\mathbb{E}}_{(x,y)\sim D}[|\theta \cdot x|] \\
&= \mathop{\mathbb{E}}_{(x,y)\sim D}[f(x) \cdot \theta \cdot x] \\
&= \sum_{j \in V} (\theta_j \mathop{\mathbb{E}}_{(x,y)\sim D}[f(x) \cdot x_j]) \\
&\leq \|\theta\|_1 \max_{j \in V} |\mathop{\mathbb{E}}_{(x,y)\sim D}[f(x) \cdot x_j]|
\end{aligned}$$

The proof is complete by observing that $\frac{\|\theta\|_2}{\|\theta\|_1} \geq \frac{1}{\sqrt{|V|}} \geq \frac{1}{\sqrt{k}}$. ■

A.2. Proof of Lemma 1

Proof Let $\mathcal{H}_{\text{relevant}} \subseteq \mathcal{H}_{\text{proj}}$ be the projection functions corresponding to the $|V| \le k$ relevant features. Then,

$$\begin{aligned}
\max_{h \in \mathcal{H}_{\text{relevant}}} \left(\sum_{i\in[n]} w_i \cdot \mathop{\mathbb{E}}_{x\sim D^{(i)}}[f^{(i)}(x)h(x)]^2 \right) &\ge \frac{1}{|\mathcal{H}_{\text{relevant}}|} \cdot \sum_{h\in\mathcal{H}_{\text{relevant}}} \sum_{i\in[n]} w_i \cdot \mathop{\mathbb{E}}_{x\sim D^{(i)}}[f^{(i)}(x)h(x)]^2 \\
&\ge \frac{1}{k} \cdot \sum_{i\in[n]} w_i \cdot \max_{h\in\mathcal{H}_{\text{relevant}}} \left(\mathop{\mathbb{E}}_{x\sim D^{(i)}}[f^{(i)}(x)h(x)]^2 \right) \\
&\ge \frac{1}{k} \cdot \sum_{i\in[n]} w_i \cdot \frac{\gamma^2}{k} && \text{(Fact 1)} \\
&\ge \frac{\gamma^2}{k^2} && (\textstyle\sum_{i\in[n]} w_i = 1)
\end{aligned}$$

■

A.3. Proof of Lemma 2

Proof For each $s \in [t]$, let $h_s^{(1)}, \dots, h_s^{(n)}$ be the hypothesis at the end of the $s^{(th)}$ iteration, with $s = 0$ use to denote the start of the algorithm. We will track the exponential loss,

$$L_s := \sum_{i\in[n]} \sum_{j\in[m]} \exp\left(-y_j^{(i)} \cdot h_s^{(i)}(x_j^{(i)})\right).$$

We will prove, by induction, that $L_s \le nm \cdot \prod_{s\in[t]} \left(1 - \frac{\Gamma_s}{2}\right)$. The base case of $s = 0$ holds with equality. For any $s \ge 1$, let $w_j^{(i)}$ and $W^{(i)}$ be the weights during the s^{th} iteration. Then,

$$\begin{aligned}
L_s &= \sum_{i\in[n]} \sum_{j\in[m]} \exp\left(-y_j^{(i)} \cdot h_s^{(i)}(x_j^{(i)})\right) \\
&= \sum_{i\in[n]} \sum_{j\in[m]} \exp\left(-y_j^{(i)} \cdot \left(h_{s-1}^{(i)}(x_j^{(i)}) + \alpha_s^{(i)} \cdot h_s^\star(x_j^{(i)})\right)\right) \\
&= \sum_{i\in[n]} \sum_{j\in[m]} w_j^{(i)} \cdot \exp\left(-y_j^{(i)} \cdot \alpha_s^{(i)} \cdot h_s^\star(x_j^{(i)})\right).
\end{aligned}$$

For each $i \in [n]$, we'll use the shorthand:

$$\begin{aligned}
W_=^{(i)} &:= \sum_{j\in[m]} w_j^{(i)} \cdot \mathbb{1}[y_j^{(i)} = h^\star(x_j^{(i)})], \\
W_{\neq}^{(i)} &:= \sum_{j\in[m]} w_j^{(i)} \cdot \mathbb{1}[y_j^{(i)} \neq h^\star(x_j^{(i)})].
\end{aligned}$$

Hence,

$$\begin{aligned}
L_s &= \sum_{i\in[n]} W_{=}^{(i)} \cdot \exp(-\alpha_s^{(i)}) + W_{\neq}^{(i)} \cdot \exp(\alpha_s^{(i)}) \\
&= \sum_{i\in[n]} W_{=}^{(i)} \cdot \sqrt{\frac{W_{\neq}^{(i)}}{W_{=}^{(i)}}} + W_{\neq}^{(i)} \cdot \sqrt{\frac{W_{=}^{(i)}}{W_{\neq}^{(i)}}} \\
&= \sum_{i\in[n]} 2\sqrt{W_{=}^{(i)} \cdot W_{\neq}^{(i)}}
\end{aligned}$$

Note that $W_{=}^{(i)} + W_{\neq}^{(i)} = W^{(i)}$. For each $i \in [n]$, we define

$$\gamma^{(i)} := \sum_{j\in[m]} \frac{w_j^{(i)}}{W^{(i)}} \cdot y_j^{(i)} h^\star(x_j^{(i)}).$$

and observe $W_{=}^{(i)} - W_{\neq}^{(i)} = W^{(i)} \cdot \gamma^{(i)}$. As a result, we have that $W_{=}^{(i)} = W^{(i)}/2 \cdot (1 + \gamma^{(i)})$ and $W_{\neq}^{(i)} = W^{(i)}/2 \cdot (1 - \gamma^{(i)})$. Continuing,

$$\begin{aligned}
L_s &= \sum_{i\in[n]} 2\sqrt{W_{=}^{(i)} \cdot W_{\neq}^{(i)}} \\
&= \sum_{i\in[n]} W^{(i)} \cdot \sqrt{(1+\gamma^{(i)}) \cdot (1-\gamma^{(i)})} \\
&\leq \sum_{i\in[n]} W^{(i)} \cdot (1 - \frac{(\gamma^{(i)})^2}{2}) \\
&= L_{s-1} - \sum_{i\in[n]} W^{(i)} \cdot \frac{(\gamma^{(i)})^2}{2} \\
&= L_{s-1} - \frac{1}{2} \cdot \sum_{i\in[n]} W^{(i)} \left(\sum_{j\in[m]} \frac{w_j^{(i)}}{W^{(i)}} \cdot y_j^{(i)} h^\star(x_j^{(i)}) \right)^2 \\
&= L_{s-1} - \frac{\Gamma_s}{2} \cdot \sum_{i\in[n]} W^{(i)} \\
&= L_{s-1} \cdot \left(1 - \frac{\Gamma_s}{2}\right) && (\textstyle\sum_{i\in[n]} W^{(i)} = L_{s-1})
\end{aligned}$$

Hence, we have that $L_t \leq nm \cdot \prod_{s\in[t]} \left(1 - \frac{\Gamma_s}{2}\right)$. The desired holds because classification error is upper bounded by $L_t/(nm)$. ■

A.4. Proof of Proposition 1

Proof The running time is dominated by finding which weak-learner maximizes Equation (1). To do so so, we can loop over all $|\mathcal{H}|$ weak-learners and compute their advantage, which takes time $O(nm)$. This must be done in each of t iterations, given a total runtime of $O(nmt|\mathcal{H}|)$. ■

A.5. Proof of Proposition 2

Proof The expected number of items selected is $\mu = \ell/2$. First, suppose that we selected the items *with replacement*. Then, whether each marked item is selected is independent. In this setting, by a standard Chernoff bound, we have that the probability at least $\frac{2\ell}{3} = (1 + \frac{1}{2})\mu$ items are selected is at most $\exp(-\mu/10) = \exp(-\ell/20)$.

At a high level, sampling *without replacement* can only improve this bound. In more detail, let $\boldsymbol{X}$ be the number of marked items selected when sampling with replacement, and $\boldsymbol{Y}$ be for the setting of sampling without replacement. The only information about $\boldsymbol{X}$ needed for the above Chernoff bound to hold is an upper bound on the moment generating function $\mathbb{E}[e^{\lambda \boldsymbol{X}}]$ for appropriately chosen λ. It is therefore sufficient to argue that $\mathbb{E}[e^{\lambda \boldsymbol{Y}}] \leq \mathbb{E}[e^{\lambda \boldsymbol{X}}]$ for every $\lambda \in \mathbb{R}$.

Let $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ and $\boldsymbol{y}_1, \dots, \boldsymbol{y}_n$ be random variables indicating the number of marked items selected from each of the n groups when sampling with replacement and without replacement respectively. As $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ are independent,

$$\mathbb{E}[e^{\lambda \boldsymbol{X}}] = \prod_{i \in [n]} \mathbb{E}[e^{\lambda \boldsymbol{x}_i}].$$

Similarly $\boldsymbol{y}_1, \dots, \boldsymbol{y}_n$ are independent, as the items selected in one group do not effect the items selected in other groups, so,

$$\mathbb{E}[e^{\lambda \boldsymbol{Y}}] = \prod_{i \in [n]} \mathbb{E}[e^{\lambda \boldsymbol{y}_i}].$$

It is therefore sufficient to prove that $\mathbb{E}[e^{\lambda \boldsymbol{y}_i}] \leq \mathbb{E}[e^{\lambda \boldsymbol{x}_i}]$ for each $i \in [n]$. This is proven in (Hoeffding, 1994, Theorem 4), which states that sampling without replacement can only decrease the moment generating function. ■

A.6. Proof of Lemma 3

Proof Fix distributions $D^{(1)}, \dots, D^{(n)}$ each over $\{\pm 1\}^d \times \{\pm 1\}$. For each $i \in [n]$, let $\boldsymbol{S}^{(i)} \sim (D^{(i)})^m$ be a size-m random sample, and let $\boldsymbol{A}$ be the event that there are $h^{(1)}, \dots, h^{(n)} \in \mathcal{C}$ satisfying $\bigcup_{i \in [n]} \mathrm{Rel}(h^{(i)}) \in \mathcal{V}$ for which

1. At most ε-fraction of points in S are misclassified:

$$\sum_{(x,y) \in \boldsymbol{S}^{(i)}} \mathbb{1}[h^{(i)}(x) \neq y] \leq nm\varepsilon.$$

2. The population error satisfies $\text{avg-error}(h^{(1)}, \dots, h^{(n)}) \geq 4\varepsilon$, i.e.,

$$\frac{1}{n} \sum_{i \in [n]} \varepsilon^{(i)} \geq 4\varepsilon \qquad \text{where} \quad \varepsilon^{(i)} := \Pr_{(x,y) \sim D^{(i)}}[h^{(i)}(x) \neq y].$$

Our goal is to upper bound $\Pr[\boldsymbol{A}]$. Suppose we draw fresh samples $\boldsymbol{T}^{(i)} \sim (D^{(i)})^m$ for each $i \in [n]$. Let $\boldsymbol{z}_j^{(i)}$ indicate whether $h^{(i)}$ misclassifies the jth point in $\boldsymbol{T}^{(i)}$. Then, $\boldsymbol{z}_j^{(i)} \sim \mathrm{Ber}(\varepsilon^{(i)})$, and the $z_j^{(i)}$ are independent across $i \in [n], j \in [m]$. Using $Z = \sum_{i\in[n],j\in[m]} z_j^{(i)}$ to indicate the total number of points misclassified on the fresh samples, we have

$$\mathbb{E}[Z] = m \sum_{i\in[n]} \varepsilon^{(i)} \geq 4nm\varepsilon.$$

$$\mathrm{Var}[Z] = m \sum_{i\in[n]} \varepsilon^{(i)}(1-\varepsilon^{(i)}) \leq \mathbb{E}[Z].$$

Applying Fact 2, as long as $4nm\varepsilon \geq 8$, $\Pr[Z \geq 2nm\varepsilon] \geq \frac{1}{2}$. Let $\boldsymbol{B}$ be the event, depending on both the original S samples and the fresh T samples, that there are $h^{(1)}, \ldots, h^{(n)} \in \mathcal{C}$ satisfying $\bigcup_{i\in[n]} \mathrm{Rel}(h^{(i)}) \in \mathcal{V}$ meeting the following three criteria.

1. At most ε-fraction of points in S are misclassified:

$$\sum_{i\in[n]} \sum_{(x,y)\in S^{(i)}} \mathbb{1}[h^{(i)}(x) \neq y] \leq nm\varepsilon.$$

2. The average test error is at least 4ε:

$$\frac{1}{n}\sum_{i\in[n]} \Pr_{(x,y)\sim D^{(i)}}[h^{(i)}(x) \neq y] \geq \varepsilon.$$

3. At least 2ε-fraction of points in T are misclassified:

$$\sum_{i\in[n]} \sum_{(x,y)\in T^{(i)}} \mathbb{1}[h^{(i)}(x) \neq y] \geq 2nm\varepsilon.$$

Due to the first two criteria, $\boldsymbol{B}$ can only occur if $\boldsymbol{A}$ occurs. Furthermore, $\Pr[\boldsymbol{B} \mid \boldsymbol{A}] = \Pr[Z \geq 2nm\varepsilon] \geq \frac{1}{2}$. As a result, we have that

$$2\Pr[\boldsymbol{B}] = 2(\Pr[\boldsymbol{B} \mid \boldsymbol{A}]\Pr[\boldsymbol{A}]) \geq 2(\frac{1}{2}\Pr[\boldsymbol{A}]) = \Pr[\boldsymbol{A}].$$

Therefore, in order to upper bound $\Pr[\boldsymbol{A}]$, it is sufficient to upper bound $\Pr[\boldsymbol{B}]$. Indeed we will show that just the first and third criteria of $\boldsymbol{B}$ are unlikely to occur together.

We consider an alternative and equivalent generation process for the samples. For each task $i \in [n]$, we draw $2m$ samples from $D^{(i)}$. Then, we partition half of those samples into $S^{(i)}$ and the other half into $T^{(i)}$.

Consider a single possible labeling for all $2nm$ points. Let ℓ be the total number of misclassified points, and ℓ_S and ℓ_T be the number of misclassified points that are partitioned into S and T respectively. Then, in order for $\boldsymbol{B}$ to occur, it must be the case that $\ell_S \leq nm\varepsilon$ and $\ell_T \geq 2nm\varepsilon$. In particular, this implies that $\ell \geq 2nm\varepsilon$ and $\frac{\ell_T}{\ell} \geq \frac{2}{3}$.

By Proposition 2, for a single labeling of the $2nm$ points, $\boldsymbol{B}$ occurs with probability at most $\exp(-\frac{nm\varepsilon}{10})$. There are only $\Pi_{\mathcal{C},\mathcal{V}}(n, 2m)$ possible labelings for the $2nm$ points. Therefore, we can upper bound,

$$\Pr[\boldsymbol{A}] \leq 2\Pr[\boldsymbol{B}] \leq 2 \cdot \Pi_{\mathcal{C},\mathcal{V}}(n, 2m) \cdot \exp\left(-nm\varepsilon/10\right).$$

■

A.7. Proof of Proposition 3

Proof First, we prove the first inequality. For any $V \in \mathcal{V}$, given that $\bigcup_{i\in[n]} \mathrm{Rel}(f^{(i)}) = V$, the number of unique ways that $f^{(i)}$ can classify a sample of size m is at most $\Pi_{(\mathcal{C}|V)}(m)$. Therefore, the total ways to classify all nm points is at most $(\Pi_{(\mathcal{C}|V)}(m))^n$. Summing over all $V \in \mathcal{V}$ gives us the desired upper bound.

To prove the second inequality, we just plug in Fact 3, giving $\Pi_{(\mathcal{C}|V)}(m) \le \left(\frac{em}{\mathrm{VC}(\mathcal{C}|\mathcal{V})}\right)^{\mathrm{VC}(\mathcal{C}|\mathcal{V})}$ ∎

A.8. Proof of Theorem 5

Proof For the m, n given, we'll have $m \ge \mathrm{VC}(\mathcal{C} \mid \mathcal{V})$, so Proposition 3 applies, and $nm\varepsilon \ge 2$, so Lemma 3 applies. Therefore,

$$\begin{aligned}\delta_{\mathrm{gen}}(\mathcal{C}, \mathcal{V}, n, m, \varepsilon) &\le 2 \cdot \Pi_{\mathcal{C},\mathcal{V}}(n, 2m) \cdot \exp\left(-nm\varepsilon/10\right) && \text{(Lemma 3)}\\ &\le 2|\mathcal{V}| \cdot \left(\frac{2em}{\mathrm{VC}(\mathcal{C} \mid \mathcal{V})}\right)^{n\cdot\mathrm{VC}(\mathcal{C}|\mathcal{V})} \cdot \exp\left(-nm\varepsilon/10\right) && \text{(Proposition 3)}\end{aligned}$$

By setting $m = O\left(\mathrm{VC}(\mathcal{C} \mid \mathcal{V}) \cdot \frac{\log(1/\varepsilon)}{\varepsilon}\right)$, we have that

$$\left(\frac{2em}{\mathrm{VC}(\mathcal{C} \mid \mathcal{V})}\right)^{n\cdot\mathrm{VC}(\mathcal{C}|\mathcal{V})} \le \exp\left(nm\varepsilon/20\right).$$

As a result, we can bound

$$\delta_{\mathrm{gen}}(\mathcal{C}, \mathcal{V}, n, m, \varepsilon) \le 2|\mathcal{V}| \cdot \exp\left(-nm\varepsilon/20\right)$$

which is at most δ when $nm = O\left(\frac{\log|\mathcal{V}| + \log(1/\delta)}{\varepsilon}\right)$, as desired. ∎

A.9. Proof of Theorem 3

Proof Pick any $f^{(1)}, \dots, f^{(n)} \in \mathcal{C}_\gamma$ with shared representation in $\mathcal{V}_k$. By Lemma 1, the class of projection functions $\mathcal{H}_{\mathrm{proj}} := \{x \mapsto x_\ell \mid \ell \in [d]\}$ satisfies the $(\Gamma = \frac{\gamma^2}{k^2})$-simultaneous weak-learning assumption for $f^{(1)}, \dots, f^{(n)}$.

Let $S^{(1)}, \dots, S^{(n)}$ be samples of m points for each of the n tasks. By Corollary 1, for,

$$t = O\left(\frac{\log(4/\varepsilon)}{\Gamma}\right) = O\left(\frac{k^2 \log(1/\varepsilon)}{\gamma^2}\right)$$

the output of $\textsc{Boost}(S^{(1)}, \dots, S^{(n)}, \mathcal{H}_{\mathrm{proj}}, t)$ will have average training error at most $\varepsilon/4$.

Next, we prove generalization. The hypotheses output by $\textsc{Boost}$ all depend on the same t projection functions, so have a shared representation in $\mathcal{V}_t = \{V \subseteq [d] \mid |V| \le t\}$ which satisfies $\log|\mathcal{V}_t| = O(t \log d)$. Once a representation $V \subseteq [d]$ is fixed the class $(\mathcal{C}_\gamma \mid V)$ consists of γ-margin halfspaces of the coordinates in V, which has VC dimension at most $|V|$. Therefore, $\mathrm{VC}(\mathcal{C}_\gamma \mid \mathcal{V}) \leq t$. Applying Theorem 5 gives that, if n, m are chosen so that

$$m = O\left(t \cdot \frac{\log(1/\varepsilon)}{\varepsilon}\right) = O\left(\frac{k^2 \log^2(1/\varepsilon)}{\gamma^2 \varepsilon}\right),$$
$$nm = O\left(\frac{t\log(d) + \log(1/\delta)}{\varepsilon}\right) = O\left(\frac{(k^2/\gamma^2)\log(1/\varepsilon)\log(d) + \log(1/\delta)}{\varepsilon}\right),$$

then with probability $1-\delta$, hypotheses $h^{(1)}, \ldots, h^{(n)}$ output by $\textsc{Boost}(S^{(1)}, \ldots, S^{(n)}, \mathcal{H}_{\text{proj}}, t)$ have $\text{avg-error}(h^{(1)}, \ldots, h^{(n)}) \leq \varepsilon$.

Finally, the runtime is bounded by Proposition 1. ■

Appendix B. Proofs from Section 4

B.1. Proof of Theorem 6

Proof Let S be the input sample of size $N = \frac{8}{\varepsilon} tk(\log(d)+1)\ln(\frac{2tk(\log(d)+1)}{\delta})$. Each labeled example from S is of the form $((x,p,q,b), f_{V,r}(x,p,q,b))$ where $x \sim D_x$, p and q are drawn uniformly at random, and $b \sim \mathrm{Ber}(\varepsilon)$. By the definition of the secret-sharing scheme, since V, r are fixed, given all the bits of all the shares, that is, if $\forall p, q$, there exists $x \in \{\pm 1\}^d$ such that example $(x,p,q,1) \in S$, then we can recover the secret V. We will describe the recovery algorithm at the end of this proof.

We denote the number of examples with $b=1$ by N_1. Let $Y_1, \ldots, Y_{N_1}$ be independent random variables such that $Y_i = \mathbb{1}\{b_i = 1\}$, where b_i is the value of b of the ith example and $\mathbb{1}\{A\}$ is 1 when A is true and 0 otherwise. Then $N_1 = \sum_{i=1}^N Y_i$ and $\mathbb{E}[N_1] = \varepsilon N$. Let $H_{p,q}$ be the event that $\nexists x : (x,p,q,1) \in S$, that is, the dataset does not include an example of the form $(x,p,q,1)$ for any $x \in \{\pm 1\}^d$.

$$\Pr\left[\exists p, q\ H_{p,q}\right] \leq \Pr\left[\exists p, q\ H_{p,q} | N_1 \geq \frac{\varepsilon}{8}N\right] + \Pr\left[N_1 < \frac{\varepsilon}{8}N\right] \tag{2}$$

We first bound the first term of Equation (2). Since p, q are drawn uniformly at random, the probability that a pair (p,q) is drawn is $\left(2^{\log(t)+\log(k(\log(d)+1))}\right)^{-1} = (tk(\log(d)+1))^{-1}$. By union bound we have that

$$\begin{aligned}
\Pr\left[\exists p, q\ H_{p,q} | N_1 \geq \frac{\varepsilon}{8}N\right] &\leq tk(\log(d)+1)\Pr\left[H_{p,q} | N_1 \geq \frac{\varepsilon}{8}N\right] \\
&\leq tk(\log(d)+1)\left(1 - \frac{1}{tk(\log(d)+1)}\right)^{\frac{\varepsilon}{8}N} \\
&= tk(\log(d)+1)\left(1 - \frac{1}{tk(\log(d)+1)}\right)^{tk(\log(d)+1)\ln(2tk(\log(d)+1)/\delta)} \\
&\qquad \text{(substituting for the value of } \tfrac{\varepsilon}{8}N\text{)} \\
&\leq tk(\log(d)+1)e^{-\ln(2tk(\log(d)+1)/\delta)} \qquad \text{(since } (1-1/x)^x \leq e^{-x}\text{)} \\
&\leq \frac{\delta}{2}
\end{aligned}$$

We now turn to the second term of Equation (2). Since $N > \frac{8}{\varepsilon} \ln \frac{2}{\delta}$, it is easy to verify that $\frac{\varepsilon}{2} N > \sqrt{2\varepsilon N \ln \frac{2}{\delta}}$. This implies that $\frac{\varepsilon}{8} N < \frac{\varepsilon}{2} N < \varepsilon N - \sqrt{2\varepsilon N \ln(2/\delta)}$.

Recall that $N_1 = \sum_{i=1}^{N} Y_i$ is a sum of independent random variables in $\{0,1\}$ and $\mathbb{E}[N_1] = \varepsilon N$. Let $\beta = \sqrt{\frac{2\ln(2/\delta)}{\varepsilon N}} \in (0,1)$, then

$$\begin{aligned}\Pr\left[N_1 < \frac{\varepsilon N}{8}\right] &\le \Pr\left[N_1 < \varepsilon N - \sqrt{2\varepsilon N \ln(2/\delta)}\right] \\ &= \Pr\left[N_1 < (1-\beta)\varepsilon N\right] \\ &\le e^{-\beta^2 \varepsilon N/2} \\ &\qquad \text{(by Chernoff bounds (Mitzenmacher and Upfal, 2017, Theorem 4.5))} \\ &= \frac{\delta}{2}.\end{aligned}$$

Overall, by Equation (2), we conclude that with probability at least $1-\delta$, the input sample includes all pairs p, q.

The process of reconstructing the secret set V computes the product of the labels of the samples with $b = 1$ per coordinate q for all p, ignoring possible duplicates, which takes time $O(N) = O\left(\frac{tk(\log(d)+1)}{\varepsilon} \ln\left(\frac{2tk(\log(d)+1)}{\delta}\right)\right)$. Then converts $\tilde{V}$ to V in $O(k(\log(d)+1)$ time. After recovering the secret V, our algorithm returns the unique function g_V. Overall, the algorithm has time complexity $O\left(\frac{tk\log(d)}{\varepsilon} \ln\left(\frac{tk\log(d)}{\delta}\right)\right)$. The description length of the class $\mathcal{C}_k^{(t)}$ is $\mathrm{len}(\mathcal{C}_k^{(t)}) = \log(|\mathcal{C}_k^{(t)}|) = \log(|\mathcal{V}| \cdot 2^{tk(\log(d)+1)}) = O(tk\log(d))$. Thus, given polynomial in the description length of class $\mathcal{C}_k^{(t)}$ examples, this algorithm runs in polynomial time in $\mathrm{len}(\mathcal{C}_k^{(t)})$, and with probability $1-\delta$, returns a hypothesis $h = g_V$ with error at most ε, that is,

$$\begin{aligned}&\Pr_{(x,p,q,b)\sim D_x \times E_\varepsilon}\left[h(x,p,q,b) \neq f_{V,r}(x,p,q,b)\right] \\ \le &\Pr_{(x,p,q)\sim D_x \times E_\varepsilon^{(1)}}\left[h(x,p,q,1) \neq f_{V,r}(x,p,q,1)\right] \cdot \varepsilon \\ &+ \Pr_{(x,p,q)\sim D_x \times E_\varepsilon^{(-1)}}\left[h(x,p,q,-1) \neq f_{V,r}(x,p,q,-1)\right] \\ \le &\ \varepsilon,\end{aligned}$$

where $E_\varepsilon^{(1)}$ and $E_\varepsilon^{(-1)}$ correspond to distribution E_ε conditioned on $b = 1$ and $b = -1$, respectively.

The parameter t is a free parameter that determines the description length of $\mathcal{C}_k^{(t)}$ and the time and sample complexity correspondingly. However, for any $t \in \mathbb{N}$, $\mathcal{C}_k^{(t)}$ is attribute-efficient learnable with sample complexity $\mathrm{poly}(\mathrm{len}(\mathcal{C}_k^{(t)}))$ and time complexity $\mathrm{poly}(\mathrm{len}(\mathcal{C}_k^{(t)}))$, that is, polynomial in the size of the input dataset. ■

B.2. Proof of Theorem 7

Proof We first prove that $\mathcal{C}_k^{(m+1)}$ is $\mathcal{V}_k^{(m+1)}$-multitask learnable in exponential time in $k\log(d)$. We consider that the given dataset is $S = \{(x_j^{(i)}, p_j^{(i)}, q_j^{(i)}, b_j^{(i)}, y_j^{(i)})_{i\in[n], j\in[m]}\}$, where $(x_j^{(i)}, p_j^{(i)}, q_j^{(i)}, b_j^{(i)}, y_j^{(i)})$

is drawn i.i.d. from $D^{(i)}$, that is $(x_j^{(i)}, p_j^{(i)}, q_j^{(i)}, b_j^{(i)}) \sim D_x \times E_\varepsilon$ and $y_j^{(i)} = f^{(i)}(x_j^{(i)}, p_j^{(i)}, q_j^{(i)}, b_j^{(i)})$, for $f^{(i)} \in \mathcal{C}_k^{(t)}$.

Every task i is associated with a set $V^{(i)} \subseteq V_x$ such that $f^{(i)}(x, p, q, -1) = g_{V^{(i)}}(x)$. The naive algorithm iterates over every possible $V_x \subseteq [d]$ of size k and for each task $i \in [n]$ finds a $\hat{V}^{(i)} \subseteq V_x$ that defines a function $g_{\hat{V}^{(i)}} \in \mathcal{C}_k$ that is consistent with the samples of this task with $b = -1$. In more detail, in time $O(\binom{d}{k} nm2^k) \leq O((2d)^k nm)$, this algorithm finds $\hat{h}^{(1)}, \ldots, \hat{h}^{(n)}$ in $\mathcal{C}_k$ such that

$$\hat{h}^{(i)}(x_j^{(i)}, p_j^{(i)}, q_j^{(i)}, b_j^{(i)}) = y_j^{(i)}, \forall i \in [n] \text{ and } j \in [m] \text{ s.t. } b_j^{(i)} = -1.$$

We will now show that for any $f^{(1)}, \ldots, f^{(n)} \in \mathcal{C}_k^{(t)}$ and $D^{(1)}, \ldots, D^{(n)}$ where the features are drawn from $D_x \times E_\varepsilon$, with probability at least $1 - \delta$ over the samples the average error of the functions $\hat{h}^{(1)}, \ldots, \hat{h}^{(n)}$ is

$$\frac{1}{n} \sum_{i \in [n]} \Pr_{(x,p,q,b,y) \sim D^{(i)}} [\hat{h}^{(i)}(x, p, q, b) \neq y] \leq 2\varepsilon.$$

There are two ways the result is influenced by our decision to ignore samples with $b = 1$. Firstly, when we draw a new sample to predict its label we have not learnt anything about the secret shares, thus

$$\begin{aligned}
&\frac{1}{n} \sum_{i \in [n]} \Pr_{(x,p,q,b,y) \sim D^{(i)}} [\hat{h}^{(i)}(x, p, q, b) \neq y] \leq \\
&\frac{1}{n} \sum_{i \in [n]} \Pr_{(x,p,q,b,y) \sim D^{(i)}} [\hat{h}^{(i)}(x, p, q, b) \neq y \mid b = -1] + \varepsilon. \qquad (\Pr[b = 1] = \varepsilon)
\end{aligned}$$

Secondly, the number of samples we actually use is smaller than the number of samples we have in total. Let $L^{(i)}(h) = \Pr_{(x,p,q,b,y) \sim D^{(i)}}[h(x, p, q, b) \neq y \mid b = -1]$ be the error of function h when it is used for task i . Then,

$$\begin{aligned}
&\Pr_S \left[\frac{1}{n} \sum_{i \in [n]} L^{(i)}(\hat{h}^{(i)}) \geq \varepsilon \right] \\
&\leq \Pr_S \left[\frac{1}{n} \sum_{i \in [n]} L^{(i)}(\hat{h}^{(i)}) \geq \varepsilon \mid \left(N_{-1} > O\left(\frac{k \log(d) + \log(\frac{3}{\delta})}{\varepsilon} \right) \right) \wedge \left(\forall i \in [n] \;\; N_{-1}^{(i)} > O\left(\frac{k}{\varepsilon} \log(\frac{1}{\varepsilon}) \right) \right) \right] \\
&+ \Pr_S \left[N_{-1} \leq O\left(\frac{k \log(d) + \log(\frac{3}{\delta})}{\varepsilon} \right) \right] + \Pr_S \left[\exists i \in [n] : N_{-1}^{(i)} \leq O\left(\frac{k}{\varepsilon} \log(\frac{1}{\varepsilon}) \right) \right], \qquad (3)
\end{aligned}$$

where N_{-1} is the total number of samples with $b = -1$ and $N_{-1}^{(i)}$ is the number of samples of task i with $b = -1$.

Our approach learns functions $g_{V^{(1)}}, \ldots, g_{V^{(n)}} \in \mathcal{C}_k$ using only the relevant samples. Let $\mathcal{V} = \{V_x : V_x \subseteq [d], |V_x| \leq k\}$ and $(\mathcal{C}_k \mid V_x) = \{g \in \mathcal{C}_k : \text{Rel}(g) \subseteq V_x\}$, then $\text{VC}(\mathcal{C}_k|\mathcal{V}) = \max_{V_x \in \mathcal{V}} \log(|(\mathcal{C}_k \mid V_x)|) = \max_{V_x \in \mathcal{V}} \log(|\{g_V \in \mathcal{C}_k : V \subseteq V_x\}|) = k$. Additionally, we have that $|\mathcal{V}| \leq \binom{d}{k}2^k \leq e^{k\ln(ed)}$. Applying Theorem 5, we have that for $N_{-1} \geq O(\frac{k\log(d)+\log(3/\delta)}{\varepsilon})$ and $\min_{i\in[n]} N_{-1}^{(i)} \geq O(\frac{k}{\varepsilon}\log(1/\varepsilon))$ with probability at least $1-\delta/3$ over the samples with $b=-1$ we output $\hat{h}^{(1)},\ldots,\hat{h}^{(n)}$ such that

$$\frac{1}{n}\sum_{i\in[n]} \Pr_{(x,p,q,b,y)\sim D^{(i)}}[\hat{h}^{(i)}(x,p,q,b) \neq y \mid b=-1] \leq \varepsilon.$$

This bounds the first term of Equation (3) by $\delta/3$.

We can write N_{-1} as the sum of independent random variables $Y_j^{(i)} = \mathbb{1}\{b_j^{(i)} = -1\}$, specifically $N_{-1} = \sum_{i\in[n]}\sum_{j\in[m]} Y_j^{(i)}$, with expectation $\mathbb{E}[N_{-1}] = (1-\varepsilon)nm$. For $nm > 8\cdot\frac{1}{(1-\varepsilon)\varepsilon}(k\log(d)+\log(\frac{3}{\delta}))$, we can see that $\frac{(1-\varepsilon)}{2}nm > \sqrt{2(1-\varepsilon)nm\ln(3/\delta)}$ and, hence, $\frac{(1-\varepsilon)}{8}nm < \frac{(1-\varepsilon)}{2}nm < (1-\varepsilon)nm - \sqrt{2(1-\varepsilon)nm\ln(3/\delta)}$. As a result, for $\beta = \sqrt{\frac{2\ln(3/\delta)}{(1-\varepsilon)nm}} \in (0,1)$

$$\begin{aligned}\Pr\left[N_{-1} \leq O\left(\frac{1}{\varepsilon}(k\ln(d)+\ln(\frac{3}{\delta}))\right)\right] &\leq \Pr\left[N_{-1} \leq \frac{(1-\varepsilon)}{8}nm\right]\\ &= \Pr[N_{-1} \leq (1-\beta)(1-\varepsilon)nm]\\ &\leq e^{-\beta^2(1-\varepsilon)nm/2} = \frac{\delta}{3}.\end{aligned}$$

(by Chernoff bound (Mitzenmacher and Upfal, 2017, Theorem 4.5))

This bounds the second term of Equation (3) by $\delta/3$, for $nm \geq O(\frac{1}{\varepsilon}(k\log(d)+\log(\frac{1}{\delta})))$, because $\frac{1}{1-\varepsilon} < 2$.

Similarly, we see that for all tasks $i \in [n]$ the number of samples we use can also be written as a sum of random variables, i.e. $N_{-1}^{(i)} = \sum_{j\in[m]} Y_j^{(i)}$, with expectation $\mathbb{E}[N_{-1}^{(i)}] = (1-\varepsilon)m$. Assuming that $m \geq 8\frac{k}{(1-\varepsilon)\varepsilon}\log(1/\varepsilon)\log(3n/\delta)$, for all tasks $\frac{(1-\varepsilon)}{2}m > \sqrt{2(1-\varepsilon)m\ln(3n/\delta)}$ and $\frac{(1-\varepsilon)}{8\ln(3n/\delta)}m < \frac{(1-\varepsilon)m}{2} < (1-\varepsilon)m - \sqrt{2(1-\varepsilon)m\ln(3n/\delta)}$.Therefore, for $\gamma = \sqrt{\frac{2\ln(3/\delta)}{(1-\varepsilon)m}} \in (0,1)$

$$\begin{aligned}\Pr\left[\exists i\in[n] : N_{-1}^{(i)} \leq O\left(\frac{k}{\varepsilon}\log(1/\varepsilon)\right)\right] &\leq n\Pr\left[N_{-1}^{(i)} \leq O\left(\frac{k}{\varepsilon}\log(1/\varepsilon)\right)\right]\\ &\leq n\Pr\left[N_{-1}^{(i)} \leq \frac{(1-\varepsilon)}{8\ln(3n/\delta)}m\right]\\ &\leq n\Pr\left[N_{-1}^{(i)} \leq (1-\varepsilon)m - \sqrt{2(1-\varepsilon)m\ln(3n/\delta)}\right]\\ &= n\Pr\left[N_{-1}^{(i)} \leq (1-\gamma)(1-\varepsilon)m\right]\\ &\leq e^{-\gamma^2(1-\varepsilon)m/2} = \frac{\delta}{3}.\end{aligned}$$

(by Chernoff bound (Mitzenmacher and Upfal, 2017, Theorem 4.5))

This bounds the last term of Equation (3) by $\delta/3$ for $m \geq O(\frac{k}{\varepsilon}\log(1\varepsilon)\log(n/\delta))$. All in all, we see that with probability at least $1-\delta$ over the samples

$$\frac{1}{n}\sum_{i\in[n]} \Pr_{(x,p,q,b,y)\sim D^{(i)}}[\hat{h}^{(i)}(x,p,q,b) \neq y \mid b=-1] \leq \varepsilon$$

and, hence,

$$\frac{1}{n}\sum_{i\in[n]} \Pr_{(x,p,q,b,y)\sim D^{(i)}}[\hat{h}^{(i)}(x,p,q,b) \neq y] \leq 2\varepsilon.$$

Therefore, we conclude that $\mathcal{C}_k^{(m+1)}$ is $\mathcal{V}_k^{(m+1)}$-multitask learnable with accuracy parameters $(2\varepsilon, \delta)$, at most $m = O\left(\frac{k}{\varepsilon}\log(\frac{1}{\varepsilon})\log\left(\frac{\log(d)}{\delta}\right)\right)$ samples-per-task and $nm = O\left(\frac{k}{\varepsilon}(\log(d) + \log(\frac{1}{\delta}))\right)$ samples overall, in time $(2d)^k nm$. Since $nm < s \leq a(d)$, the time is $(2d)^k a(d)$. This concludes the first part of the proof.

For the second, let $\mathcal{A}$ be a $\mathrm{poly}(d)$-time $\mathcal{V}_k^{(m+1)}$-multitask learning algorithm for class $\mathcal{C}_k^{(m+1)}$ with accuracy $(\varepsilon/32, \delta)$. For any $f^{(1)}, \dots, f^{(n)} \in \mathcal{C}_k^{(m+1)}$ and $D^{(1)}, \dots, D^{(n)}$ where the features of the examples are drawn from $D_x \times E_\varepsilon$, with probability at least $1-\delta$, it returns $h^{(1)}, \dots, h^{(n)}$ such that

$$\frac{1}{n}\sum_{i\in[n]} \Pr_{(x,p,q,b,y)\sim D^{(i)}}[h^{(i)}(x,p,q,b) \neq y] \leq \frac{\varepsilon}{32}. \tag{4}$$

We will use $\mathcal{A}$ to attribute-efficient learn the function class $\mathcal{C}_k$.

Let $S = ((x_1, y_1), \dots, (x_N, y_N))$ be the dataset, where $x_i \sim D_x$, and $y_i = g_V(x_i)$ for $g_V \in \mathcal{C}_k$. We construct dataset S' as follows. We first split the dataset into n tasks, with m examples each, denoting the j^{th} example of the i^{th} task by $(x_j^{(i)}, y_j^{(i)})$. We choose $\{r_i\}_{i\in[n]}$ from $\{\pm 1\}^{(m+1)k(\log(d)+1)}$, such that for all pairs $i \neq i'$, $r_i \neq r_{i'}$. Then for every example $j \in [m]$ of every task $i \in [n]$ we draw $(p_j^{(i)}, q_j^{(i)}, b_j^{(i)})$ from distribution E_ε. We set:

$$\tilde{y}_j^{(i)} = \begin{cases} \text{share}^{(m+1)}(V_{\text{aux}}; r_i)_{p,q} & \text{if } b = 1 \\ y_j^{(i)} & \text{if } b = -1 \end{cases}$$

where V_{aux} is such that $\tilde{V}_{\text{aux}} = \{-1\}^{k(\log(d)+1)}$. Creating the new dataset of size $nm \times (d + \log((m+1)k(\log d + 1)) + 2)$ requires time $O(nm(m + k\log(d))$, which is at most $s^2 + sd\log(d) \leq \mathrm{poly}(d)$.

For the new dataset $S' = \{(x_j^{(i)}, p_j^{(i)}, q_j^{(i)}, b_j^{(i)}, \tilde{y}_j^{(i)})_{j\in[m], i\in[n]}\}$ we have that $(x_j^{(i)}, p_j^{(i)}, q_j^{(i)}, b_j^{(i)}) \sim D_x \times E_\varepsilon$. Moreover, for every task $i \in [n]$, since the number of examples is m, which is smaller than the reconstruction threshold of the secret-sharing scheme which is $m+1$, there exists r_i' such that $\text{share}^{(m+1)}(V_{\text{aux}}; r_i)_p = \text{share}^{(m+1)}(V; r_i')_p$ for all p that appear in this task's dataset. Thus, there exist $f^{(i)} = f_{V, r_i'}$ in $\mathcal{C}_k^{(m+1)}$ for every $i \in [n]$, such that $\tilde{y}_j^{(i)} = f^{(i)}(x_j^{(i)}, p_j^{(i)}, q_j^{(i)}, b_j^{(i)})$.

If we give $\mathcal{A}$ the dataset S', then by assumption, in $\mathrm{poly}(d)$ time with probability $1-\delta$, it returns functions $h^{(1)}, \dots, h^{(n)}$ that satisfy the guarantee of Equation (4). Since $\Pr[b = -1] = 1 - \varepsilon$, we have that

$$\frac{1}{n}\sum_{i=1}^{n} \Pr_{(x,p,q)\sim D_x \times E_\varepsilon^{(-1)}}\left[h^{(i)}(x,p,q,-1) \neq g_V(x)\right] \leq \frac{\varepsilon}{32(1-\varepsilon)} \leq \frac{\varepsilon}{16}, \tag{5}$$

where $E_\varepsilon^{(-1)}$ is the distribution E_ε conditional on $b = -1$.

In order to get a prediction for a new x, we compute

$$h(x) = \text{majority}(\{h^{(i)}(x,p,q,-1)\}_{i\in[n], p\in[m+1], q\in[k(\log(d)+1)]}).$$

This process takes $O(n(m+1)k(\log(d)+1)d)$ time. We define the set of good parameters

$$\texttt{Good} = \left\{(i,p,q) \in [n] \times [m+1] \times [k(\log(d)+1)] : \Pr_{x\sim D_x}\left[h^{(i)}(x,p,q,-1) \neq g_V(x)\right] \leq \frac{\varepsilon}{4}\right\}. \tag{6}$$

The size of $\texttt{Good}$ is at least $\frac{3}{4}n(m+1)k(\log(d)+1)$ because otherwise, if

$$|\texttt{Bad}| = |[n] \times [m+1] \times [k(\log(d)+1)] \setminus \texttt{Good}| > \frac{1}{4}n(m+1)k(\log(d)+1),$$

by Equation (5),

$$\begin{aligned}
&\frac{1}{n(m+1)k(\log(d)+1)} \sum_{i=1}^{n}\sum_{p=1}^{m+1}\sum_{q=1}^{k(\log(d)+1)} \Pr_{x\sim D_x}\left[h^{(i)}(x,p,q,-1) \neq g_V(x)\right] \leq \frac{\varepsilon}{16} \\
\Rightarrow\ &\frac{1}{n(m+1)k(\log(d)+1)} \sum_{(i,p,q)\in\texttt{Bad}} \Pr_{x\sim D_x}\left[h^{(i)}(x,p,q,-1) \neq g_V(x)\right] \leq \frac{\varepsilon}{16} \\
\Rightarrow\ &\frac{|\texttt{Bad}|}{n(m+1)k(\log(d)+1)} \cdot \frac{\varepsilon}{4} < \frac{\varepsilon}{16} \\
\Rightarrow\ &\frac{1}{4}\cdot\frac{\varepsilon}{4} < \frac{\varepsilon}{16},
\end{aligned}$$

which is a contradiction.

For the majority $h(x)$ to make a mistake, at least half of $\{h^{(i)}(x,p,q,-1)\}_{i\in[n],p\in[m+1],q\in[k(\log(d)+1)]}$ must make a mistake on x. This requires that at least a $\frac{1}{4}$ fraction of $\{h^{(i)}(x,p,q,-1)\}_{(i,p,q)\in\texttt{Good}}$ makes a mistake. Thus, by the definition of the $\texttt{Good}$ set (Equation (6)),

$$\Pr_{x\sim D_x}[h(x) \neq g_V(x)] \leq 4 \cdot \frac{\varepsilon}{4} \leq \varepsilon.$$

Therefore, this process runs in $\mathrm{poly}(d)$ time and returns a hypothesis h such that with probability at least $1-\delta$, has error at most ε for every function of class $\mathcal{C}_k$. By our assumption, this would be a contradiction and as a result we conclude that no such algorithm $\mathcal{A}$ exists. ■