Boosting with List-Decodable Codes

Addison Prairie APRAIRIE@STANFORD.EDU
Stanford

Li-Yang Tan LYTAN@STANFORD.EDU
Stanford

Editors: Steve Hanneke and Tor Lattimore

Abstract

Boosting is a fundamental technique for generically improving the accuracy of learning algorithms (Schapire 1989). Existing boosting algorithms construct a strong learner using $O(\log(\frac{1}{\epsilon})/\gamma^2)$ calls to a γ-advantage weak learner, and this round complexity is known to be optimal for generic boosters that succeed on all concept classes (Freund 1995).

We show that this lower bound can be circumvented for concept classes that satisfy a mild closure property. Specifically, we present a new boosting algorithm that, for any class $\mathcal{F}$ closed under $O(\log \frac{1}{\gamma})$-XOR, strong learns $\mathcal{F}$ using $O(\log \frac{1}{\epsilon})$ calls to a γ-advantage weak learner and a single batch of $\tilde{O}(\log(\frac{1}{\epsilon})/\gamma^2)$ additional samples.

Our algorithm arises from a new and simple connection between boosting and list-decodable codes. Viewing the target function as a message, we run the weak learner on its encoding and view the resulting weak hypothesis as a corrupted codeword. Feeding this corrupted codeword to a list decoder, we obtain a small list of candidate hypotheses, at least one of which is a strong hypothesis for the original function. Using additional samples, we identify and output this strong hypothesis.

Keywords: Boosting, Error Correcting Codes

1. Introduction

Boosting is a celebrated method for systematically improving the accuracy of learning algorithms. Given black-box access to a *weak learner*—an algorithm that produces hypotheses with accuracy slightly better than trivial—for a concept class, a boosting algorithm produces a *strong learner* that achieves arbitrarily high accuracy. Boosting algorithms typically work by sequentially running the weak learner multiple times and aggregating the weak hypotheses. Originally introduced by Schapire (1989), boosting is notable as a technique that has proven successful both in theory and in practice.

A key measure of a boosting algorithm's efficiency is the number of calls it makes to the weak learner, also known as its *round complexity*. To strong learn with accuracy $1-\epsilon$ given a weak learner with accuracy $\frac{1}{2}+\gamma$, existing boosting algorithms (e.g., AdaBoost, Freund and Schapire (1997)) make $O(\log(\frac{1}{\epsilon})/\gamma^2)$ calls to the weak learner. Early work of Freund showed that this is essentially optimal:

Theorem 1 (Freund (1995)) *For any boosting algorithm $\mathcal{B}$, there exists a concept class $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ and a $(\frac{1}{2}+\gamma)$-accuracy weak learner $\mathcal{W}$ such that $\mathcal{B}$ must call $\mathcal{W}$ at least $\Omega(\log(\frac{1}{\epsilon})/\gamma^2)$ times to learn $\mathcal{F}$ to $1-\epsilon$ accuracy.*

However, the hard concept class $\mathcal{F}$ used in this lower bound is highly artificial: for each $n \in \mathbb{N}$, the slice $\mathcal{F}_n$ consists of a single function $f : \{\pm 1\}^n \to \{\pm 1\}$ chosen uniformly at random. As a result, it does not rule out the possibility of more round-efficient boosting for the more structured concept classes that arise in theory and practice.

In this work, we evade Freund's lower bound by restricting the concept class being learned. Our approach applies to any class $\mathcal{F} = \{\mathcal{F}_n\}_{n \in \mathbb{N}}$ satisfying a mild closure assumption: we say that $\mathcal{F}$ is *closed under* k-XOR if, for every $n \in \mathbb{N}$ and $f \in \mathcal{F}_n$, the function $f^{\oplus k}(x_1, \ldots, x_k) = f(x_1) \oplus \cdots \oplus f(x_k)$ is in $\mathcal{F}_{n \cdot k}$. Under such an assumption on $\mathcal{F}$, we give a boosting algorithm that makes significantly fewer calls to $\mathcal{W}$:

Theorem 2 (see Theorem 10 for formal statement) *There exists an efficient boosting algorithm that, for any $\gamma, \epsilon > 0$, given a γ-weak learner $\mathcal{W}$ for a concept class $\mathcal{F}$ closed under $O(\log \frac{1}{\gamma})$-XOR, learns $\mathcal{F}$ to accuracy $1 - \epsilon$ with $O(\log \frac{1}{\epsilon})$ calls to $\mathcal{W}$ and $\tilde{O}(\log(\frac{1}{\epsilon})/\gamma^2)$ additional labeled samples.*

We make the following remarks about Theorem 2.

Additional Labeled Samples. Here, by additional labeled samples, we mean samples beyond those implicitly required for $O(\log \frac{1}{\epsilon})$ calls to the weak learner $\mathcal{W}$. Our $\tilde{O}(\log(\frac{1}{\epsilon})/\gamma^2)$ additional samples is modest compared to existing boosting algorithms that make $O(\log(\frac{1}{\epsilon})/\gamma^2)$ calls to $\mathcal{W}$, each of which would seemingly require at least one additional sample. Thus, the total number of labeled samples used by our algorithm is comparable to or less than that of existing boosting algorithms.

The Tradeoff we Incur. Our algorithm achieves a reduction in the number of calls to $\mathcal{W}$ at the cost of each call requiring more resources. When learning an unknown $f \in \mathcal{F}_n$, it calls $\mathcal{W}$ on $f^{\oplus k} \in \mathcal{F}_{n \cdot k}$ for $k := O(\log \frac{1}{\gamma})$. Thus, each call to $\mathcal{W}$ requires more samples and time than a call made by an existing boosting algorithm: to strong learn a function over an instance space $\mathcal{X}$, our booster calls the weak learner on a function on $\mathcal{X}^k$; for example, when learning functions over $\{\pm 1\}^n$ or $\mathbb{R}^n$, we call the weak learner on $\{\pm 1\}^{n \cdot k}$ or $\mathbb{R}^{n \cdot k}$, respectively. However, if $\mathcal{W}$ runs in polynomial time, this results in a rather modest $\mathrm{polylog}(\frac{1}{\gamma})$ increase in samples and time per call to $\mathcal{W}$, while significantly reducing the number of calls to $\mathcal{W}$.

Mildness of Our Closure Assumption. In the setting of $\gamma = n^{-O(1)}$ (the standard setting for weak learning), we only require closure under $O(\log n)$-XOR. Since the XOR of $O(\log n)$ bits can be computed by a $\mathrm{poly}(n)$-sized DNF/CNF, our boosting algorithm applies to a range of concept classes, even computationally weaker ones such as polynomial-size DNFs/CNFs and constant-depth threshold circuits.

1.1. Uniform-Distribution Boosting

One downside of most existing boosting algorithms is that, even to strong learn with respect to a single distribution (e.g., uniform), they require a weak learner that succeeds over arbitrary distributions. A strength of our approach is that it can be made distribution-specific: to achieve strong learning over the uniform distribution, it suffices to have a weak learner for the uniform distribution.

Theorem 3 (see Theorem 11 for formal statement) *There exists an efficient boosting algorithm that, for any $\gamma, \epsilon > 0$, given a uniform distribution γ-weak learner $\mathcal{W}$ for a concept class $\mathcal{F}$ closed*

under $O(\log(\frac{1}{\gamma})/\epsilon)$-XOR*, learns* $\mathcal{F}$ *to accuracy* $1-\epsilon$ *over the uniform distribution with only* 1 *call to* $\mathcal{W}$ *and* $O(\log(\frac{1}{\gamma})/\epsilon)$ *additional labeled samples.*

Theorems 2 and 3 are incomparable, as they apply to different settings of boosting. Quantitatively, Theorem 3 exhibits a worse dependence on ϵ but a better dependence on γ in the additional samples required. Moreover, it makes only a single call to $\mathcal{W}$, whereas Theorem 2 requires $O(\log \frac{1}{\epsilon})$ calls.

1.2. Other Related Work

Boosting Simple Learners. Recent work of Alon et al. (2021) gives a boosting algorithm that learns to 99% accuracy using only $O(\frac{1}{\gamma})$ calls to the weak learner $\mathcal{W}$, under the assumption that $\mathcal{W}$ produces hypotheses from a class of constant VC dimension. While this assumption and our closure assumption are incomparable, we remark that our boosting algorithms are efficient, running in time $\mathrm{poly}(n, \frac{1}{\gamma}, \frac{1}{\epsilon})$; in contrast, their algorithm runs in time exponential in $\frac{1}{\gamma}$.

Uniform-Distribution Boosting. Boneh and Lipton (1993) design a uniform-distribution boosting algorithm which similarly relies on the concept class being closed under XOR. However, their result uses $\tilde{O}(1/(\gamma^4\epsilon^2))$ additional samples and requires closure under $O(1/\gamma^2\epsilon)$-XOR. Comparing this to Theorem 3, our exponentially better dependence on γ in the closure assumption means that our result can be applied to a far broader range of parameters and concept classes. For example, when $\gamma = n^{-O(1)}$ and we aim for 99% accuracy, their algorithm requires closure under $\mathrm{poly}(n)$-XOR, ruling out computationally weaker classes such as AC^0. In contrast, our Theorem 3 only requires closure under $O(\log n)$-XOR, and therefore *does* apply to such classes.

Distribution-Specific Agnostic Boosting. Another approach to distribution-specific boosting is to place an assumption on the weak learner $\mathcal{W}$ rather than the concept class $\mathcal{F}$. In particular, Kalai and Kanade (2009), and later Feldman (2009), developed algorithms assuming that $\mathcal{W}$ is *agnostic*, a strong notion of noise tolerance. In this setting, the weak learner is not promised that the data it receives are labeled by a function in the target class $\mathcal{F}$. Instead, it must perform well even when the labels have been corrupted by arbitrary (potentially adversarial) noise; see Feldman (2009) for more details. While our assumption on $\mathcal{F}$ is incomparable to the assumption that $\mathcal{W}$ is agnostic, we remark that Theorem 3 achieves the same goal with only *one* call to $\mathcal{W}$, in contrast to the $O(\gamma^{-2})$ calls required by the distribution-specific agnostic booster above.

2. Technical Overview

The standard approach to boosting. To understand how our approach to boosting differs from existing techniques, we first explain why existing boosting algorithms require distribution-free weak learners and many calls to $\mathcal{W}$. Such algorithms work by calling the weak learner on a sequence of distributions $\mathcal{D}_1, \ldots, \mathcal{D}_T$ to obtain hypotheses $h_1, \ldots, h_T$, then combining these hypotheses via a weighted majority. Starting with $\mathcal{D}_1 := \mathcal{D}$ (the target distribution), subsequent distributions are chosen so that $\mathcal{D}_i$ places more weight on samples misclassified by $h_1, \ldots, h_{i-1}$. By carefully adjusting each $\mathcal{D}_i$ and weighting each hypothesis, AdaBoost iteratively reduces error with each call to $\mathcal{W}$.

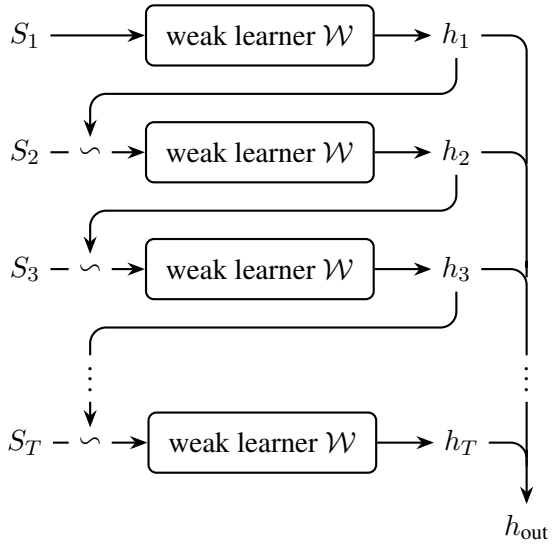

Figure 1: Existing boosting algorithms repeatedly call the weak learner, using previous hypotheses to reweight ($\smile$) the distribution of each dataset S_i fed to $\mathcal{W}$. The output h_{out} is then constructed by aggregating the weak hypotheses $h_1, \ldots, h_T$.

Our approach. Rather than modifying the distributions given to $\mathcal{W}$, we modify the function being learned. An initial dataset S_1 labeled by f is used to construct a dataset labeled by a new function f', which is then fed to $\mathcal{W}$. This outputs a hypothesis h' weakly computing f', which we use to construct a strong hypothesis h_{out} for f using additional data S_2. As described in the next section, the way we convert f to f' and recover h_{out} from h' is grounded in a new connection between boosting and error correcting codes.

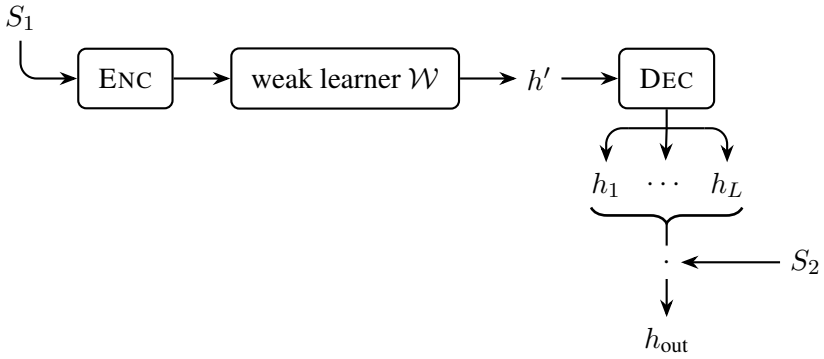

Figure 2: Our approach to boosting modifies the function being learned by $\mathcal{W}$, then feeds the resulting weak hypothesis into a decoder to produce a list $h_1, \ldots, h_L$, one of which is a strong hypothesis for the target function. Using additional samples (S_2), we find and output this strong hypothesis with high probability.

2.1. Boosting with List-Decodable Codes

To describe how to obtain a boosting algorithm from a list-decodable code, we first consider the simpler case when the instance space is $\{\pm 1\}^n$. Let $\mathcal{C}$ be a code represented by an encoding map $\text{ENC} : \{\pm 1\}^N \to \{\pm 1\}^M$, with $N := 2^n$. The property of these codes we are most interested in is approximate list-decodability: suppose a message sender begins with $w \in \{\pm 1\}^N$ and sends $c := \text{ENC}(w)$; if an adversary corrupts fewer than $\frac{1}{2} - \gamma$ of the bits of c, we want the message receiver to be able to recover a list of L candidate messages $w_1, \ldots, w_L \in \{\pm 1\}^N$, one of which is ϵ-close to the original message w. If an algorithm for doing so exists, we say that the code is *ϵ-approximately (γ, L) list-decodable* (see, for example, Impagliazzo et al. (2010), Definition 1.5). When $M = 2^m$ for $m \in \mathbb{N}$, we can view strings in $\{\pm 1\}^N$ and $\{\pm 1\}^M$ as truth tables for functions on n and m bits, respectively, and thus view ENC as a map between functions. Likewise, we can view circuits $C : \{\pm 1\}^n \to \{\pm 1\}$ as representing messages of length N.

At a high level, we obtain a boosting algorithm for a class $\mathcal{F}$ from a code $\mathcal{C}$ as follows. Given an unknown function $f \in \mathcal{F}_n$ that we want to learn, we consider a new function $f' : \{\pm 1\}^m \to \{\pm 1\}$ given by the encoding $f' := \text{ENC}(f)$. Assuming that $f' \in \mathcal{F}_m$ and we can generate random samples labeled by f' from random examples of f, we can feed our γ-weak learner $\mathcal{W}$ samples labeled by f' to obtain a circuit $C' : \{\pm 1\}^m \to \{\pm 1\}$ such that $\Pr_{\boldsymbol{y}}[f'(\boldsymbol{y}) = C'(\boldsymbol{y})] \geq \frac{1}{2} + \gamma$. Viewing the truth table of C' as a corrupted version of the truth table of f', this means the two have relative distance at most $\frac{1}{2} - \gamma$. Thus, if we run DEC on C', we obtain a list of circuits $C_1, \ldots, C_L$, among which at least one C_i computes our target function f with error at most ϵ. By testing this list against additional random samples labeled by f, we can identify and output this high-accuracy hypothesis with high probability.

In order for the framework above to work, we require two additional properties of $\mathcal{C}$ and $\mathcal{F}$. First, $\mathcal{C}$ should be *locally encodable*, meaning we can efficiently generate samples labeled by $\text{ENC}(f)$ given a small number of samples labeled by f. Second, we need $\mathcal{F}$ to be *closed under $\mathcal{C}$*, meaning $\text{ENC}(f) \in \mathcal{F}$ for every $f \in \mathcal{F}$. Then we have the following connection between codes and boosting.

Theorem 4 *Suppose that $\mathcal{C}$ is a code that is locally encodable and efficiently ϵ-approximately (γ, L) locally list-decodable. Then there exists an efficient boosting algorithm that, given a uniform-distribution γ-weak learner $\mathcal{W}$ for a concept class $\mathcal{F}$ closed under $\mathcal{C}$, learns $\mathcal{F}$ to accuracy $1 - O(\epsilon)$ over the uniform distribution with 1 call to $\mathcal{W}$ and $O(\log(L)/\epsilon)$ additional samples.*

The relationship between the properties of the code $\mathcal{C}$ and the resulting boosting algorithm can be summarized as follows.

Code $\mathcal{C}$		**Boosting Algorithm**
ϵ-approximately decodable	$\longleftrightarrow$	final error $O(\epsilon)$
list-decodable at radius $\frac{1}{2} - \gamma$	$\longleftrightarrow$	γ-weak learner
list size L	$\longleftrightarrow$	additional samples $O(\frac{\log L}{\epsilon})$
rate $N/M = 2^n/2^m$	$\longleftrightarrow$	input expansion m/n
locally list-decodable	$\longleftrightarrow$	efficient

Proof of Theorem 3. To prove our distribution-specific result, we instantiate the framework above with the XOR code. For a fixed $k \in \mathbb{N}$, the k-XOR code encodes $f : \mathcal{X} \to \{\pm 1\}$ as the function $f^{\oplus k} : \mathcal{X}^k \to \{\pm 1\}$. Then the proof of Theorem 3 is a straightforward combination of Theorem 4 with the list-decoding algorithm of Impagliazzo et al. (2010) for the XOR code.

2.2. Standard Boosting: Distributional Codes and Arbitrary Domains

Distributional error correcting codes. To extend this connection to the distributional setting, we generalize error correcting codes to measure corruption with respect to arbitrary distributions over the bits of the received word and message. Standard codes measure the error of a recovered message $\tilde{f}$ with respect to a message f as $\Pr_{\boldsymbol{x} \sim \mathcal{U}_n}[f(\boldsymbol{x}) \neq \tilde{f}(\boldsymbol{x})]$, and measure the corruption of a received word $\tilde{g}$ from a codeword g as $\Pr_{\boldsymbol{y} \sim \mathcal{U}_m}[g(\boldsymbol{y}) \neq \tilde{g}(\boldsymbol{y})]$. To generalize to the distributional setting, a *distributional code* additionally specifies a distribution $\mathcal{D}$ over $\{\pm 1\}^n$ and a distribution $\mathcal{D}'$ over $\{\pm 1\}^m$, which allow us to measure the error between f and $\tilde{f}$ as $\Pr_{\boldsymbol{x} \sim \mathcal{D}}[f(\boldsymbol{x}) \neq \tilde{f}(\boldsymbol{x})]$, and measure the corruption of $\tilde{g}$ relative to g as $\Pr_{\boldsymbol{y} \sim \mathcal{D}'}[g(\boldsymbol{y}) \neq \tilde{g}(\boldsymbol{y})]$.

Since the decoding algorithm for a distributional code must have *some* knowledge about the distribution $\mathcal{D}$ in order to properly decode a corrupted message, we model this information in the simplest way possible: the decoding algorithm receives, in addition to a corrupted codeword, a small number of independent samples $\boldsymbol{x} \backsim \mathcal{D}$. The definition of ϵ-approximate (γ, L) local list-decodability for distributional codes then extends naturally from the standard setting (see Definition 5).

Arbitrary domains. We'd also like for our boosting algorithms to work over arbitrary, potentially infinite domains. Previously, we viewed an error correcting code as implicitly mapping from one function $f : \{\pm 1\}^n \to \{\pm 1\}$ to another by encoding its truth table; however, when a concept class contains functions over an infinite domain $\mathcal{X}$, their truth tables are infinite strings for which standard notions of encoding and decoding no longer make sense. To capture this setting, we define codes as directly mapping a class of functions $f : \mathcal{X} \to \{\pm 1\}$ to $\textsc{Enc}(f) : \mathcal{X}' \to \{\pm 1\}$. With this change, we obtain a more general version of Theorem 4 that works over arbitrary domains and distributions, allowing us to obtain boosting algorithms in the standard setting.

A distributional decoder for the XOR Code. We combine this generic connection between boosting and distributional codes with a distributional decoder for the k-XOR code. As a mapping of functions, it encodes a function $f : \mathcal{X} \to \{\pm 1\}$ as $f^{\oplus k} : \mathcal{X}^k \to \{\pm 1\}$. We then describe and prove the correctness of an efficient distributional local list decoder for the XOR code. The algorithm itself is a straightforward generalization of the decoding algorithm of Impagliazzo et al. (2010), although the analysis of its correctness is fairly different. In particular, their proof crucially relies on the domain $\mathcal{X}$ being discrete (see Remark 15 for a more detailed comparison).

Proof of Theorem 2. Combining this decoder with a distributional generalization of Theorem 4, we obtain a distribution-free one-round boosting algorithm which has a better dependence on γ but a worse dependence on ϵ than existing boosting algorithms. To get the best of both, we use our one-round booster to obtain a .49-weak learner from a γ-weak learner, then apply an existing boosting algorithm to achieve accuracy $1 - \epsilon$. This concatenated boosting algorithm is what yields Theorem 2.

3. Preliminaries

Notation and naming conventions. We write $[n]$ to denote the set $\{1, 2, \ldots, n\}$. We use lowercase letters to denote bit strings, uppercase letters to denote vectors, and **boldface letters** (e.g., $\mathbf{x}$, $\mathbf{Y}$) to denote random variables. Next, fix $k \in \mathbb{N}$ and let $\mathcal{X}$ be a set. For $Z \in \mathcal{X}^k$ and $I \subseteq [k]$, we let

$Z_I \in \mathcal{X}^{|I|}$ denote the vector

$$Z_I \coloneqq (Z_{i_1}, \ldots, Z_{i_{|I|}}), \quad I = \{i_1, \ldots, i_{|I|}\}.$$

If $X, Y \in \mathcal{X}^{k/2}$ are two $\frac{k}{2}$-vectors and $S \subseteq [k]$ has size $|S| = \frac{k}{2}$, we denote by $\textsc{Il}_S(X, Y)$ the vector in $\mathcal{X}^k$ that interleaves X and Y according to S,

$$(\textsc{Il}_S(X, Y))_S = X, \quad (\textsc{Il}_S(X, Y))_{[k]\setminus S} = Y.$$

Finally, let $\mathcal{D}$ be a distribution over $\mathcal{X}$ and fix $k \in \mathbb{N}$, $x \in \mathcal{X}$. We denote by $\mathcal{D}^{k,x}$ the distribution generated by the following process: sample a random $\boldsymbol{i} \sim [k]$ and independent $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k \sim \mathcal{D}$, then output $\boldsymbol{X} \coloneqq (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{\boldsymbol{i}-1}, x, \boldsymbol{y}_{\boldsymbol{i}+1}, \ldots, \boldsymbol{y}_k) \in \mathcal{X}^k$. When we want to remember the underlying index $\boldsymbol{i}$ where x was planted, we may write $(\boldsymbol{i}, \boldsymbol{X}) \sim \mathcal{D}^{k,x}$.

Distributional error correcting codes. Here we provide the formal definition of distributional codes as described in Section 2.2. A *distributional code* is given by an encoder $\textsc{Enc} : \{\pm 1\}^{\mathcal{X}} \to \{\pm 1\}^{\mathcal{X}'}$, a message distribution $\mathcal{D}$ over $\mathcal{X}$, and a codeword distribution $\mathcal{D}'$ over $\mathcal{X}'$. With this, we can generalize the definition of local list decoding to the distributional setting:

Definition 5 (Approximately Local List-Decodable Distributional Codes) *We say that a code $\mathcal{C} = (\textsc{Enc}, \mathcal{D}, \mathcal{D}')$ is* ϵ-approximately, (γ, L, u) locally list-decodable *if there exists a decoding algorithm* $\textsc{Dec}$ *that, given as input $C' : \mathcal{X}' \to \{\pm 1\}$ and u independent samples $x_1, \ldots, x_u \backsim \mathcal{D}$, outputs a list $h_1, \ldots, h_L : \mathcal{X} \to \{\pm 1\}$ with the following guarantee: for any $f : \mathcal{X} \to \{\pm 1\}$ such that $\Pr_{\boldsymbol{y} \backsim \mathcal{D}'}[\textsc{Enc}(f)(\boldsymbol{y}) \neq C'(\boldsymbol{y})] < \frac{1}{2} - \gamma$, there exists some $i \in [L]$ such that $\Pr_{\boldsymbol{x} \backsim \mathcal{D}}[f(\boldsymbol{x}) \neq h_i(\boldsymbol{x})] < \epsilon$ with probability at least $\frac{3}{4}$ over the decoder's randomness and the random draw of samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_u \backsim \mathcal{D}$. We say that* $\textsc{Dec}$ *is* efficient *if it runs in time polynomial in $n, \frac{1}{\gamma}, \frac{1}{\epsilon}$ and the size of C'.*

Additionally, we will need our codes to be locally encodable in the following sense.

Definition 6 (local encodability) *We say that a code $\mathcal{C} = (\textsc{Enc}, \mathcal{D}, \mathcal{D}')$ is ℓ-locally encodable if there exists an efficient algorithm which, for any $f : \mathcal{X} \to \{\pm 1\}$, can produce a sample $\boldsymbol{x} \backsim \mathcal{D}'$ labeled by $\textsc{Enc}(f)$ given ℓ independent random samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell \backsim \mathcal{D}$ labeled by f.*

Standard learning definitions. We will use $\mathcal{F} = \{\mathcal{F}_n\}_{n \in \mathbb{N}}$ to denote a concept class, where $\mathcal{F}_n \subseteq \mathcal{F}$ is the subset of n-arity functions in $\mathcal{F}$ (e.g., those over $\{\pm 1\}^n$ or $\mathbb{R}^n$). We will call $\mathcal{F}_n$ the n-th slice. We will also need the following standard definitions for PAC learning and weak learning.

Definition 7 (Distribution-specific PAC learning) *For a concept class $\mathcal{F}_n$, we say an algorithm $\mathcal{A}$ learns $\mathcal{F}_n$ to accuracy $1 - \epsilon$ over a distribution $\mathcal{D}$ using m samples if the following is true: for any $f \in \mathcal{F}_n$, given m independent samples of the form $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x} \backsim \mathcal{D}$, $\mathcal{A}$ returns a hypothesis h that, with high probability, satisfies*

$$\Pr_{\boldsymbol{x} \backsim \mathcal{D}}[f(\boldsymbol{x}) = h(\boldsymbol{x})] \geq 1 - \epsilon.$$

We call $\mathcal{A}$ a γ-weak learner *for $\mathcal{F}_n$ over $\mathcal{D}$ using m samples if it learns $\mathcal{F}_n$ to accuracy $\frac{1}{2} + \gamma$ over $\mathcal{D}$ with m samples.*

If an algorithm $\mathcal{A}$ learns $\mathcal{F}_n$ to accuracy $1 - \epsilon$ using m samples for every distribution $\mathcal{D}$, we call it a *distribution-free* learning algorithm. When a learning algorithm is not defined with respect to a specific distribution, we will assume this means that it is distribution-free.

4. Proof of Theorem 4

First, we state and prove in full generality our generic connection between list-decodable distributional codes and boosting.

Theorem 8 (formal statement of Theorem 4) *Fix $\epsilon, \gamma > 0$, and have $\textsc{Enc} : \{\pm 1\}^{\mathcal{X}} \to \{\pm 1\}^{\mathcal{X}'}$. Suppose that $\mathcal{C} = (\textsc{Enc}, \mathcal{D}, \mathcal{D}')$ is a distributional code that is ℓ-locally encodable and ϵ-approximately (γ, L, u)-list decodable, and let $\mathcal{F}$ be a concept class closed under $\mathcal{C}$. There exists a boosting algorithm that, given γ-weak learner $\mathcal{W}$ for $\mathcal{F}$ on $\mathcal{D}'$ using s samples, strong learns $\mathcal{F}$ to accuracy $1 - 4\epsilon$ over $\mathcal{D}$ with one call to $\mathcal{W}$, u unlabelled samples from $\mathcal{D}$, and $\ell \cdot s + O(\log(L)/\epsilon)$ labeled samples from $\mathcal{D}$.*

Proof Suppose that $f : \mathcal{X} \to \{\pm 1\}$ is the unknown function we wish to learn. By the assumption that $\mathcal{F}$ is closed under $\mathcal{C}$, we know that $f' := \textsc{Enc}(f)$ is also in $\mathcal{F}$. Moreover, since $\mathcal{C}$ is locally encodable, we can draw $\ell \cdot s$ labeled samples from $\mathcal{D}$ to produce s samples from $\mathcal{D}'$ labeled by f'. Feeding these samples to $\mathcal{W}$, we obtain a hypothesis $C' : \mathcal{X}' \to \{\pm 1\}$ such that $\Pr_{\boldsymbol{y} \backsim \mathcal{D}'}[C'(\boldsymbol{y}) \neq f'(\boldsymbol{y})] < \frac{1}{2} - \gamma$. Running our approximate list decoder $\textsc{Dec}$ for $\mathcal{C}$ with input C' and unlabeled samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_u \backsim \mathcal{D}$, we get a list of hypothesis $h_1, \ldots, h_L : \mathcal{X} \to \{\pm 1\}$ such that, with probability at least $\frac{3}{4}$, there exists some $i \in [L]$ such that $\Pr_{\boldsymbol{x} \backsim \mathcal{D}}[h_i(\boldsymbol{x}) \neq f(\boldsymbol{x})] < \epsilon$. Given this list, we will draw a test set S of $t := 100 \cdot \log(L)/\epsilon$ labeled samples from $\mathcal{D}$ and do the following: for $i = 1, 2, \ldots, L$, we test the accuracy of h_i on this test set, outputting the first h_i with error at most 2ϵ. To analyze the correctness of this step, we define

$$\textsc{Err}(h_i) := \Pr_{\boldsymbol{x} \backsim \mathcal{D}}[f(\boldsymbol{x}) \neq h_i(\boldsymbol{x})], \quad \textsc{Err}_S(h_i) := \tfrac{1}{t} \cdot |\{x \in S : f(x) \neq h_i(x)\}|.$$

First, note that this algorithm only rejects a "good" h_i, one with error $\leq \epsilon$, if $\textsc{Err}_S(h_i) - \textsc{Err}(h_i) > \epsilon$. Viewing $\textsc{Err}_S(h_i)$ as the empirical mean of t-many i.i.d. Bernoulli random variables with mean $\textsc{Err}(h_i)$, we can apply Bernstein's inequality to see that

$$\Pr_{\boldsymbol{S}}\left[\textsc{Err}_{\boldsymbol{S}}(h_i) - \textsc{Err}(h_i) > \epsilon\right] \leq \exp\left(-\frac{t\epsilon^2}{2\epsilon + \frac{2}{3}\epsilon}\right) = \exp\left(-\tfrac{3}{8}t\epsilon\right) \leq \frac{1}{8L}.$$

Thus, the probability it rejects the good hypothesis is $\leq \frac{1}{8L}$. On the other hand, it accepts a "bad" h_i, one with error $\geq 4\epsilon$, only if $\textsc{Err}_S(h_i) \leq 2\epsilon \leq \frac{1}{2} \cdot \textsc{Err}(h_i)$. Again viewing $\textsc{Err}_S(h_i)$ as the empirical mean of t-many i.i.d. Bernoulli random variables with mean $\textsc{Err}(h_i)$, we can apply a multiplicative Chernoff bound to see that

$$\Pr_{\boldsymbol{S}}\left[\textsc{Err}_{\boldsymbol{S}}(h_i) < \tfrac{1}{2} \cdot \textsc{Err}(h_i)\right] \leq \exp\left(-\tfrac{1}{8}t\epsilon\right) \leq \frac{1}{8L}.$$

Thus, the probability that a "bad" h_i is output by our algorithm is $\leq \frac{1}{8L}$. Combining these two, conditioned on at least one hypothesis having error at most ϵ, the probability we do not output a circuit with error at most 4ϵ is at most $\frac{1}{4}$ by a union bound. Since the list contains a good hypothesis with probability $\geq \frac{3}{4}$, the total probability of outputting a good hypothesis is $\geq \frac{1}{2}$. Finally, note that the only labeled samples used are the $\ell \cdot s$ required for one call to the weak learner and the $O(\log(L)/\epsilon)$ required for testing the list output by the decoder, and the only unlabeled samples are the u used by $\textsc{Dec}$. ■

5. Proofs of Theorems 2 and 3

In this section, we utilize the previous section to prove our main results. We will do so by instantiating the previous section with the k-XOR code. For a function $f : \mathcal{X} \to \{\pm 1\}$, the encoder ENC_k of the k-XOR code maps f to $f^{\oplus k} : \mathcal{X}^k \to \{\pm 1\}$. For both of our boosting algorithms, we will use the following generalization of Impagliazzo et al. (2010) to the distributional (and continuous domain) setting.

Theorem 9 *There exists an efficient algorithm* DEC *such that for any* $\epsilon, \gamma > 0$ *and* $k \geq \Omega(\log(\frac{1}{\gamma})/\epsilon)$*, and every distribution* $\mathcal{D}$ *over* $\mathcal{X}$*,* DEC *is an* ϵ*-approximate* $(\gamma, O(k/\gamma^2), O(k \cdot \log(\frac{1}{\epsilon})/\gamma^2))$ *local list decoder for* $(\text{ENC}_k, \mathcal{D}, \mathcal{D}^k)$.

We will defer the proof of Theorem 9 to the next section, first using it to prove Theorems 2 and 3. The formal statement of Theorem 2 is the following:

Theorem 10 (formal statement of Theorem 2) *There exists a boosting algorithm that, for any* $\gamma, \epsilon > 0$ *and for* $k := O(\log \frac{1}{\gamma})$*, given a* γ*-weak learner* $\mathcal{W}$ *for a concept class* $\mathcal{F}$ *closed under* k-XOR*, strong learns* $\mathcal{F}_n$ *to accuracy* $1 - \epsilon$ *with* $O(\log \frac{1}{\epsilon})$ *calls to* $\mathcal{W}$ *on the* $(k \cdot n)$*-th slice and* $O(k \cdot \log(\frac{1}{\epsilon})/\gamma^2)$ *additional samples.*

Proof Suppose that $\mathcal{W}$ is our γ-weak learner for $\mathcal{F}$ requiring $s(\cdot)$ samples, and set $k := O(\log \frac{1}{\gamma})$ so that Theorem 9 yields a .1-approximate $(\gamma, O(k/\gamma^2), O(k \cdot \log(\frac{1}{\epsilon})/\gamma^2))$ local list decoder for the XOR code $(\text{ENC}_k, \mathcal{D}, \mathcal{D}')$ for any distribution $\mathcal{D}$. Combining this with Theorem 8, we obtain a distribution-free .49-weak learner for $\mathcal{F}$ making 1 call to $\mathcal{W}$ on the $(n \cdot k)$-th slice and, simulating unlabeled samples by simply drawing labeled examples, using at most S additional samples, where

$$S = O\left(\log(k/\gamma^2) + k/\gamma^2\right) = O\left(k/\gamma^2\right).$$

Call this .49-weak learner $\mathcal{W}'$. Giving an existing boosting algorithm such as AdaBoost (Freund and Schapire (1997)) (viewed as a boosting-by-filtering algorithm) access to $\mathcal{W}'$, we obtain a boosting algorithm which makes $O(\log \frac{1}{\epsilon})$ calls to $\mathcal{W}'$ and learns to accuracy $1 - \epsilon$. Since each call to $\mathcal{W}'$ requires 1 call to $\mathcal{W}$ and $O(k/\gamma^2)$ additional samples, this composed algorithm makes $O(\log \frac{1}{\epsilon})$ calls to $\mathcal{W}$ and uses $O(k \cdot \log(\frac{1}{\epsilon})/\gamma^2)$ additional samples. ■

Next, we state in full generality Theorem 3; the proof is a straightforward instantiation of Theorem 8 with Theorem 9, noting that since the distribution $\mathcal{D}$ is uniform over $\mathcal{X}$ (and thus $\mathcal{D}^k$ is uniform over $\mathcal{X}^k$), it yields a uniform-to-uniform boosting algorithm which can generate samples from $\mathcal{D}$ itself using random bits.

Theorem 11 (formal statement of Theorem 3) *There exists a boosting algorithm that, for any* $\gamma, \epsilon > 0$ *and for* $k := O(\log(\frac{1}{\gamma})/\epsilon)$*, given a uniform-distribution* γ*-weak learner* $\mathcal{W}$ *for a concept class* $\mathcal{F}$ *closed under* k-XOR*, strong learns* $\mathcal{F}_n$ *to accuracy* $1 - \epsilon$ *over the uniform distribution with* 1 *call to* $\mathcal{W}$ *on the* $(k \cdot n)$*-th slice and* $O(\log(\frac{1}{\gamma})/\epsilon)$ *additional samples.*

6. A List Decoder for the Distributional XOR Code

In this section, we describe and prove the correctness of our list-decoding algorithm for the k-XOR code. The key technical ingredient is the following list decoding algorithm for the direct product code, which maps a function $f : \mathcal{X} \to \{\pm 1\}$ to $f^k : \mathcal{X}^k \to \{\pm 1\}^k$, where $f^k(x_1, \ldots, x_k) = (f(x_1), \ldots, f(x_k))$.

Theorem 12 *There exists an efficient algorithm* DEC *that, for any $\epsilon, \gamma > 0$ and $k \geq 128 \cdot \log(\frac{1}{\gamma})/\epsilon$, and any distribution $\mathcal{D}$ over $\mathcal{X}$, given as input $C : \mathcal{X}^k \to \{\pm 1\}$ and $128 \cdot \log(\frac{1}{\epsilon})/\gamma$ unlabeled samples from $\mathcal{D}$, outputs a list of $L := \frac{64}{\gamma}$ hypotheses $h_1, \ldots, h_L : \mathcal{X} \to \{\pm 1\}$ with the following guarantee: for any $f : \mathcal{X} \to \{\pm 1\}$ such that $\Pr_{\boldsymbol{X} \sim \mathcal{D}^k}[C(\boldsymbol{X}) = f(\boldsymbol{X})] \geq \gamma$, there exists with probability $\geq \frac{3}{4}$ some $i \in [L]$ such that $\Pr_{\boldsymbol{x} \sim \mathcal{D}}[h_i(\boldsymbol{x}) = f(\boldsymbol{x})] \geq 1 - \epsilon$, where this probability is over the unlabeled samples from $\mathcal{D}$ and* DEC*'s internal randomness.*

Deferring the proof of Theorem 12 for a moment, we first remark on how one obtains a distributional list-decoder for the XOR code from the above. The strategy is the same as that of Impagliazzo et al. (2010) and thus we will not repeat the analysis here; given $C : \mathcal{X}^{2k} \to \{\pm 1\}$ computing $f^{\oplus k}$ to accuracy $\frac{1}{2} + \gamma$, they use the list decoder of Goldreich and Levin (1989) for the Hadamard code (along with $O(k/\gamma^2)$ samples from $\mathcal{D}$) to recover a $C' : \mathcal{X}^{2k} \to \{\pm 1\}^{2k}$ computing f^k to accuracy $\Omega(\gamma^2/k)$. One can then feed this into Theorem 12 to list decode the XOR code (see Impagliazzo et al. (2010), Section 5 for the exact details). Thus, establishing Theorem 12 proves Theorem 9. We thus turn our attention to proving Theorem 12.

6.1. A Distributional Decoder for the Direct Product Code

Here, we describe and prove the accuracy of the algorithm realizing Theorem 12. The decoding algorithm itself is a straightforward generalization of the decoding algorithm of Impagliazzo et al. (2010) to the distributional setting. Let $f : \mathcal{X} \to \{\pm 1\}$ be a function and $C : \mathcal{X}^k \to \{\pm 1\}^k$ be a hypothesis computing f^k to accuracy $\geq \gamma$,

$$\Pr_{\boldsymbol{X} \sim \mathcal{D}^k}\left[C(\boldsymbol{X}) = f^k(\boldsymbol{X})\right] \geq \gamma.$$

For $Y \in \mathcal{X}^k$ and $S \subseteq [k]$ of size $|S| = \frac{k}{2}$, we will call (Y, S) a *pair of advice*. For any pair of advice A and vector $\vec{Q} = ((Q^{(1)}, i^{(1)}), \ldots, (Q^{(t)}, i^{(t)}))$, where each $Q^{(j)} \in \mathcal{X}^{k/2}$ and $i^{(j)} \in [\frac{k}{2}]$, we denote by $C_{A,\vec{Q}} : \mathcal{X} \to \{\pm 1\}$ the circuit that, on input x, does the following:

1. For $(Q, i) \in \vec{Q}$:
   1.a) Write $Z := (Q_1, \ldots, Q_{i-1}, x, Q_{i+1}, \ldots, Q_{\frac{k}{2}})$, and $X := \text{IL}_S(Y_S, Z)$, and let $i^\star \in [k]$ be the index that x is placed at in X. If $C(X)_i = C(Y)_i$ for all $i \in S$, output $C(X)_{i^\star}$.
2. Output 0.

DEC runs by repeatedly drawing random advice then constructing the above circuit; more concretely, let $\mathcal{A}$ be the distribution on advice that is sampled by drawing a random $\boldsymbol{Y} \sim \mathcal{D}^k$ and

uniform $\boldsymbol{S} \subseteq [k]$ of size $|\boldsymbol{S}| = \frac{k}{2}$, and let $\mathcal{Q}$ represent the distribution $\mathcal{D}^{k/2} \times [\frac{k}{2}]$. Then DEC, given as input C, γ, ϵ, and k, runs as follows:

1. Set $t \coloneqq 64 \cdot \log(\frac{1}{\epsilon})/\gamma$. Draw 3 independent $\vec{\boldsymbol{Q}}_1, \vec{\boldsymbol{Q}}_2, \vec{\boldsymbol{Q}}_3 \sim \mathcal{Q}^t$.
2. For $j \in [\frac{64}{\gamma}]$, sample a random $\boldsymbol{A}_j \sim \mathcal{A}$ and add $C_{\boldsymbol{A}_j, \vec{\boldsymbol{Q}}_1}, C_{\boldsymbol{A}, \vec{\boldsymbol{Q}}_2}, C_{\boldsymbol{A}, \vec{\boldsymbol{Q}}_3}$ to the list.

Then the proof of Theorem 12 can be broken up into the following lemmas; first, we show that if a piece of advice A is "excellent" (see the appendix, Definition 16), then an output with that advice is likely to compute f to high accuracy:

Lemma 13 *Suppose that a piece of advice A is excellent. Then with probability at least $\frac{1}{2}$ over a random $\vec{\boldsymbol{Q}} \sim \mathcal{Q}^t$,*

$$\Pr_{\boldsymbol{x} \sim \mathcal{X}}[C_{A, \vec{\boldsymbol{Q}}}(\boldsymbol{x}) = f(\boldsymbol{x})] \geq 1 - \epsilon.$$

Next, we show that a randomly chosen piece of advice is likely to be excellent:

Lemma 14 *A random piece of advice $\boldsymbol{A} \sim \mathcal{A}$ is excellent with probability at least $\frac{\gamma}{4}$.*

Given these, the proof of Theorem 12 is simple: with probability at least $\frac{7}{8}$, DEC will draw a good piece of advice, and with probability at least $\frac{7}{8}$, one of the $\vec{\boldsymbol{Q}}_j$ will combine with this good advice to yield a high accuracy circuit in the outputted list. Each circuit can be constructed efficiently given C, $\boldsymbol{A}_j$ and $\vec{\boldsymbol{Q}}_i$, while each $\boldsymbol{A}_j$ and $\vec{\boldsymbol{Q}}_i$ can be sampled with k and $k \cdot t$ unlabeled samples, respectively. Thus, DEC is efficient and its overall unlabeled sample complexity is

$$(t+1) \cdot k \leq 128 \cdot k \cdot \frac{\log \frac{1}{\epsilon}}{\gamma}.$$

The proofs of Lemmas 13 and 14 can be found in Section A; to finish off this section, we discuss the similarities and differences between our proof strategy and that of Impagliazzo et al. (2010).

Remark 15 (Comparison with Impagliazzo et al. (2010)) Our decoding algorithm is a natural generalization of Impagliazzo et al. (2010) into the distributional setting, and the decision to break down the proof of Theorem 12 into Lemmas 13 and 14, as well as the definition of "excellent" (see Definition 16), are very similar to analysis in Impagliazzo et al. (2010). The most significant difference between our analysis and that in Impagliazzo et al. (2010) is in the proof of Lemma 13. Both proofs use as a key claim the fact that we can approximately swap the order of conditioning when proving that good advice yields a good circuit (see Claim 17); however, their proof uses properties of underlying "sampler" graphs over collections of subsets of $\mathcal{X}$ which crucially rely on $\mathcal{X}$ being finite (and don't generalize to the distributional setting). On the other hand, our proof takes a more direct route, largely making use of a concentration inequality (see Claims 18 and 19) to approximately exchange the order of conditioning.

Acknowledgments

We thank the COLT reviewers for helpful feedback and suggestions. The authors are supported by NSF awards 1942123, 2211237, 2224246, a Sloan Research Fellowship, and Omer Reingold's Simons Foundation investigators award.

References

Noga Alon, Alon Gonen, Elad Hazan, and Shay Moran. Boosting simple learners. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 481–489. Association for Computing Machinery, 2021. ISBN 9781450380539. doi: 10.1145/3406325.3451030. URL https://doi.org/10.1145/3406325.3451030.

Dan Boneh and Richard J. Lipton. Amplification of weak learning under the uniform distribution. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, page 347–351. Association for Computing Machinery, 1993. ISBN 0897916115. doi: 10.1145/168304.168372. URL https://doi.org/10.1145/168304.168372.

Vitaly Feldman. Distribution-specific agnostic boosting, 2009. URL https://arxiv.org/abs/0909.2927.

Yoav Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995. ISSN 0890-5401. doi: https://doi.org/10.1006/inco.1995.1136. URL https://www.sciencedirect.com/science/article/pii/S0890540185711364.

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. ISSN 0022-0000. doi: https://doi.org/10.1006/jcss.1997.1504. URL https://www.sciencedirect.com/science/article/pii/S002200009791504X.

Oded Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, page 25–32, New York, NY, USA, 1989. Association for Computing Machinery. ISBN 0897913078. doi: 10.1145/73007.73010. URL https://doi.org/10.1145/73007.73010.

Russell Impagliazzo, Ragesh Jaiswal, Valentine Kabanets, and Avi Wigderson. Uniform direct product theorems: Simplified, optimized, and derandomized. *SIAM Journal on Computing*, 39(4):1637–1665, 2010. doi: 10.1137/080734030. URL https://doi.org/10.1137/080734030.

Adam Tauman Kalai and Varun Kanade. Potential-based agnostic boosting. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems*, NIPS'09, page 880–888. Curran Associates Inc., 2009. ISBN 9781615679119.

Robert E. Schapire. The strength of weak learnability. In *30th Annual Symposium on Foundations of Computer Science*, pages 28–33, 1989. doi: 10.1109/SFCS.1989.63451.

Appendix A. Proofs of Lemmas 13 and 14

First, we will introduce some notation and definitions to help in our analysis. For advice $A = (Y, S)$, with $Y \in \mathcal{X}^k$ and $S \subseteq [k]$ with $|S| = \frac{k}{2}$, we define $N_A \subseteq \mathcal{X}^k$ to be the set of k-vectors that equal Y at every index in S,

$$N_A \coloneqq \left\{ X \in \mathcal{X}^k : X_i = Y_i \text{ for all } i \in S \right\}.$$

Likewise, we will have $\mathcal{N}_A$ represent the distribution induced by $\mathcal{D}^k$ on the set N_A. Next, we define $\textsc{Cons}(A) \subseteq N_A$ to be the subset of "consistent" vectors on which C correctly computes f at every index in S,

$$\textsc{Cons}(A) \coloneqq \{X \in N_A : C(X)_i = f(X_i) \text{ for all } i \in S\}.$$

We will denote by $\overline{\textsc{Cons}}(A) \in \mathbb{R}$ the density $\overline{\textsc{Cons}}(A) \coloneqq \mathcal{N}_A(\textsc{Cons}(A))$. For any $X \in \mathcal{X}^k$, we define $\textsc{Err}(X) \subseteq [k]$ to be the subset of indices on which C computes f incorrectly,

$$\textsc{Err}(X) = \{i \in [k] : C(X)_i \neq f(X_i)\}.$$

Finally, we denote by $\overline{\textsc{Err}}(X) \in \mathbb{R}$ the fractional error $\overline{\textsc{Err}}(X) \coloneqq \overline{\textsc{Err}}(X)/k$.

Definition 16 *We say that an advice pair $A = (Y, S)$ is* correct *if $C(Y) = f^k(Y)$. We say that A is* good *if it is correct and $\overline{\textsc{Cons}}(A) > \frac{\gamma}{2}$. Finally, we say that A is* excellent *if it is good and*

$$\mathop{\mathbb{E}}_{\boldsymbol{X} \sim \mathcal{N}_A} \left[\overline{\textsc{Err}}(\boldsymbol{X}) \mid \boldsymbol{X} \in \textsc{Cons}(A)\right] \leq \frac{\epsilon}{32}.$$

A.1. Proof of Lemma 13

In this section, we prove that give an excellent piece of advice $A = (Y, S)$, the output of our decoder computes f to high accuracy. For the sake of notational simplicity, we will assume without loss of generality that $S = \{\frac{k}{2} + 1, \ldots, k\}$. We will also think of $\textsc{Cons}(A)$ as a subset of $\frac{k}{2}$-vectors over $\mathcal{X}$,

$$\textsc{Cons}(A) \coloneqq \{X \in \mathcal{X}^{k/2} : C(X, Y_S)_i = y_i \text{ for all } i \in S\}.$$

Our proof of Lemma 13 will use the following claim, whose proof we defer to later in this section.

Claim 17 *Suppose that A is excellent. Then*

$$\mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{D}} \left[\Pr_{(\boldsymbol{i}, \boldsymbol{X}) \sim \mathcal{D}^{k/2, \boldsymbol{x}}} \left[C(\boldsymbol{X}, Y_S)_{\boldsymbol{i}} \neq f(\boldsymbol{x}) \mid \boldsymbol{X} \in \textsc{Cons}(A)\right]\right] \leq \frac{\epsilon}{4}. \tag{1}$$

In turn, The proofs of both Claim 17 and Lemma 13 will use the following. Write $\mu \coloneqq \mathcal{D}^{k/2}(\textsc{Cons}(A))$ and define $B \subseteq \mathcal{X}$ to be the subset of inputs x such that $\mathcal{D}^{k/2, x}(\textsc{Cons}(A))$ is too small,

$$B \coloneqq \left\{x \in \mathcal{X} : \mathcal{D}^{k/2, x}(\textsc{Cons}(A)) < \frac{\mu}{4}\right\}.$$

Then we have the following:

Claim 18 *If A is good, so that $\mu \geq \frac{\gamma}{2}$, then $\mathcal{D}(B) < \frac{\epsilon}{8}$.*

The proof of the above claim is a straightforward application of Claim 19, whose proof can be found at the end of this section. We will first prove Lemma 13 given Claim 17, then prove Claim 17.

Proof We begin by bounding the following expectation:

$$\mathop{\mathbb{E}}_{\vec{\boldsymbol{Q}} \sim \mathcal{Q}^t} \left[\Pr_{\boldsymbol{x} \sim \mathcal{D}} \left[C_{A, \vec{\boldsymbol{Q}}}(\boldsymbol{x}) \neq f(\boldsymbol{x})\right]\right] = \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{D}} \left[\Pr_{\vec{\boldsymbol{Q}} \sim \mathcal{Q}^t} \left[C_{A, \vec{\boldsymbol{Q}}}(\boldsymbol{x}) \neq f(\boldsymbol{x})\right]\right]. \tag{2}$$

By switching the order of the expectation, we can now consider the behavior of the randomized hypothesis $C_{A,\vec{\boldsymbol{Q}}}$ on a fixed input $x \in \mathcal{X}$. First, we will use Claim 18 to focus our analysis only on those x for which $\mathcal{D}^{k/2,x}(\textsc{Cons}(A))$ is relatively large,

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\Pr_{\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t}\left[C_{A,\vec{\boldsymbol{Q}}}(\boldsymbol{x}) \neq f(\boldsymbol{x})\right]\right] \leq \mathcal{X}(B) + \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\Pr_{\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t}\left[C_{A,\vec{\boldsymbol{Q}}}(\boldsymbol{x}) \neq f(\boldsymbol{x})\right] \cdot \mathbb{1}[\boldsymbol{x} \notin B]\right], \quad (3)$$

$$\leq \frac{\epsilon}{8} + \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\mathop{\mathbb{E}}_{\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t}\left[\mathbb{1}\left[C_{A,\vec{\boldsymbol{Q}}}(\boldsymbol{x}) \neq f(\boldsymbol{x})\right]\right] \cdot \mathbb{1}[\boldsymbol{x} \notin B]\right]. \quad (4)$$

Now, fix some $x \notin B$. We want to analyze the behavior of $C_{A,\vec{\boldsymbol{Q}}}$ on this x. To do so, let $E(\vec{\boldsymbol{Q}}, x)$ be the event that at least one of the randomly drawn $(\boldsymbol{Q}, \boldsymbol{i}) \in \vec{\boldsymbol{Q}}$ has $\boldsymbol{Z}_{(\boldsymbol{Q},\boldsymbol{i}),x} \in \textsc{Cons}(A)$, where we define

$$\boldsymbol{Z}_{(\boldsymbol{Q},\boldsymbol{i}),x} \coloneqq (\boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_{\boldsymbol{i}-1}, x, \boldsymbol{Q}_{\boldsymbol{i}+1}, \ldots, \boldsymbol{Q}_{k/2}).$$

First, note that $C_{A,\vec{\boldsymbol{Q}}}$ outputs some value in step 1.a) if and only if it finds some $(\boldsymbol{Q}, \boldsymbol{i}) \in \vec{\boldsymbol{Q}}$ such that $\boldsymbol{Z}_{(\boldsymbol{Q},\boldsymbol{i}),x} \in \textsc{Cons}(A)$. We will show that this occurs with high probability over $\vec{\boldsymbol{Q}}$. Note that drawing a random $(\boldsymbol{Q}, \boldsymbol{i}) \sim \mathcal{D}^{k/2} \times [\frac{k}{2}]$ and taking $\boldsymbol{Z}_{(\boldsymbol{Q},\boldsymbol{i}),x}$ is the same as drawing a random vector from $\mathcal{D}^{k/2,x}$, so that the probability a random $\boldsymbol{Z}_{(\boldsymbol{Q},\boldsymbol{i}),x}$ is in $\textsc{Cons}(A)$ is $\mathcal{D}^{k/2,x}(\textsc{Cons}(A))$. Since $x \notin B$, this is at least $\frac{\mu}{4} \geq \frac{\gamma}{8}$. Moreover, given that $\vec{\boldsymbol{Q}}$ is drawn by taking t-many independent $(\boldsymbol{Q}, \boldsymbol{i}) \sim \mathcal{Q}$, the probability that $E(\vec{\boldsymbol{Q}}, x)$ is false over a randomly drawn $\vec{\boldsymbol{Q}} \sim \mathcal{Q}^t$ is at most

$$\Pr_{\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t}\left[E(\vec{\boldsymbol{Q}}, x) \text{ is false}\right] = \left(1 - \mathcal{D}^{k/2,x}(\textsc{Cons}(A))\right)^t \leq \left(1 - \frac{\gamma}{8}\right)^t \leq \frac{\epsilon}{8},$$

since we had $t = 64 \cdot \log(\frac{1}{\epsilon})/\gamma$. Thus, for any good $x \notin B$,

$$\mathop{\mathbb{E}}_{\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t}\left[\mathbb{1}\left[C_{A,\vec{\boldsymbol{Q}}}(x) \neq f(x)\right]\right] \leq \frac{\epsilon}{8} + \mathop{\mathbb{E}}_{\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t}\left[\mathbb{1}\left[C_{A,\vec{\boldsymbol{Q}}}(x) \neq f(x)\right] \mid E(\vec{\boldsymbol{Q}}, x)\right].$$

Finally, we will compute this rightmost conditional expectation. For $\vec{\boldsymbol{Q}}$, let $(\boldsymbol{Q}, \boldsymbol{i})$ be the first pair such that $\boldsymbol{Z}_{(\boldsymbol{Q},\boldsymbol{i}),x} \in \textsc{Cons}(A)$. Conditioned on $E(\vec{\boldsymbol{Q}}, x)$, such a pair exists. In this case, the probability that $C_{A,\vec{\boldsymbol{Q}}}(x) \neq f(x)$ is the probability that $\boldsymbol{i} \in \textsc{Err}(\boldsymbol{Z}_{(\boldsymbol{Q},\boldsymbol{i}),x})$ conditioned on $\boldsymbol{Z}_{(\boldsymbol{Q},\boldsymbol{i}),x} \in \textsc{Cons}(A)$; but this probability is exactly

$$\Pr_{(\boldsymbol{i},\boldsymbol{X})\sim\mathcal{D}^{k/2,x}}\left[C(\boldsymbol{X}, Y_S)_{\boldsymbol{i}} \neq f(\boldsymbol{x}) \mid \boldsymbol{X} \in \textsc{Cons}(A)\right].$$

Having ξ denote the term

$$\xi \coloneqq \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\mathop{\mathbb{E}}_{\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t}\left[\mathbb{1}\left[C_{A,\vec{\boldsymbol{Q}}}(\boldsymbol{x}) \neq f(\boldsymbol{x})\right]\right] \cdot \mathbb{1}[\boldsymbol{x} \notin B]\right],$$

we have that

$$\xi \leq \frac{\epsilon}{8} + \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\Pr_{(\boldsymbol{i},\boldsymbol{X})\sim\mathcal{D}^{k/2,x}}\left[C(\boldsymbol{X}, Y_S)_{\boldsymbol{i}} \neq f(\boldsymbol{x}) \mid \boldsymbol{X} \in \textsc{Cons}(A)\right] \cdot \mathbb{1}[\boldsymbol{x} \notin B]\right],$$

$$\leq \frac{\epsilon}{8} + \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\Pr_{(\boldsymbol{i},\boldsymbol{X})\sim\mathcal{D}^{k/2,x}}\left[C(\boldsymbol{X}, Y_S)_{\boldsymbol{i}} \neq f(\boldsymbol{x}) \mid \boldsymbol{X} \in \textsc{Cons}(A)\right]\right] \leq \frac{\epsilon}{8} + \frac{\epsilon}{4},$$

where the last inequality comes from applying Claim 17. Substituting this into Equations (2) and (4), we have that

$$\mathop{\mathbb{E}}_{\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t}\left[\Pr_{\boldsymbol{x}\sim\mathcal{D}}\left[C_{A,\vec{\boldsymbol{Q}}}(\boldsymbol{x})\neq f(\boldsymbol{x})\right]\right]=\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\Pr_{\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t}\left[C_{A,\vec{\boldsymbol{Q}}}(\boldsymbol{x})\neq f(\boldsymbol{x})\right]\right]\leq\frac{\epsilon}{8}+\frac{\epsilon}{8}+\frac{\epsilon}{4}\leq\frac{\epsilon}{2}.$$

Applying Markov's inequality, we see that with probability at least $\frac{1}{2}$ over $\vec{\boldsymbol{Q}}\sim\mathcal{Q}^t$,

$$\Pr_{\boldsymbol{x}\sim\mathcal{D}}\left[C_{A,\vec{\boldsymbol{Q}}}(\boldsymbol{x})\neq f(\boldsymbol{x})\right]\leq\epsilon,$$

which completes the proof. ■

Next, we prove Claim 17.

Proof For any $x\in\mathcal{X}$, we will define $h,h':\mathcal{X}\to\mathbb{R}$ as follows,

$$\begin{aligned}\Phi(x)&\coloneqq\Pr_{(\boldsymbol{i},\boldsymbol{X})\sim\mathcal{D}^{k/2,x}}\left[C(\boldsymbol{X},A)_{\boldsymbol{i}}\neq f(\boldsymbol{x})\mid\boldsymbol{X}\in\textsc{Cons}(A)\right],\\ \hat{\Phi}(x)&\coloneqq\Pr_{(\boldsymbol{i},\boldsymbol{X})\sim\mathcal{D}^{k/2,x}}\left[C(\boldsymbol{X},A)_{\boldsymbol{i}}\neq f(\boldsymbol{x})\text{ and }\boldsymbol{X}\in\textsc{Cons}(A)\right].\end{aligned}$$

Then we can rewrite the lefthand side of Equation (1) using Claim 18,

$$\begin{aligned}\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\Phi(\boldsymbol{x})\right]&=\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\Phi(\boldsymbol{x})\cdot\mathbb{1}[\boldsymbol{x}\in B]\right]+\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\Phi(\boldsymbol{x})\cdot\mathbb{1}[\boldsymbol{x}\notin B]\right],&(5)\\ &\leq\frac{\epsilon}{8}+\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}[\Phi(\boldsymbol{x})\cdot\mathbb{1}[\boldsymbol{x}\notin B]],&(6)\\ &=\frac{\epsilon}{8}+\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\frac{\hat{\Phi}(\boldsymbol{x})}{\mathcal{D}^{k/2,\boldsymbol{x}}(\textsc{Cons}(A))}\cdot\mathbb{1}[\boldsymbol{x}\notin B]\right],&(7)\\ &\leq\frac{\epsilon}{8}+\frac{4}{\mu}\cdot\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}[\hat{\Phi}(\boldsymbol{x})].&(8)\end{aligned}$$

Now, we can expand and rearrange this expectation as follows,

$$\begin{aligned}
\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\hat{\Phi}(x)\right] &= \int_{\mathcal{X}} \hat{\Phi}(x)\, d\mathcal{D}(x), \\
&= \int_{\mathcal{X}} \left(\frac{2}{k} \cdot \sum_{i=1}^{\frac{k}{2}} \int_{\mathcal{X}^{k/2-1}} \mathbb{1}\left[i \in \textsc{Err}(Z_{(X,i),x}) \wedge Z_{(X,i),x} \in \textsc{Cons}(A) \right] d\mathcal{D}^{k/2-1}(X) \right) d\mathcal{D}(x), \\
&= \frac{2}{k} \cdot \sum_{i=1}^{\frac{k}{2}} \int_{\mathcal{X}} \int_{\mathcal{X}^{k/2-1}} \mathbb{1}\left[i \in \textsc{Err}(Z_{(X,i),x}) \wedge Z_{(X,i),x} \in \textsc{Cons}(A) \right] d\mathcal{D}^{k/2-1}(X)\, d\mathcal{D}(x), \\
&= \frac{2}{k} \cdot \sum_{i=1}^{\frac{k}{2}} \int_{\mathcal{X}^{k/2}} \mathbb{1}\left[i \in \textsc{Err}(X) \wedge X \in \textsc{Cons}(A) \right] d\mathcal{D}^{k/2}(X), \\
&= \int_{\mathcal{X}^{k/2}} \left(\frac{2}{k} \cdot \sum_{i=1}^{\frac{k}{2}} \mathbb{1}\left[i \in \textsc{Err}(X) \wedge X \in \textsc{Cons}(A) \right] \right) d\mathcal{D}^{k/2}(X), \\
&= \int_{\mathcal{X}^{k/2}} \left(\mathbb{1}[X \in \textsc{Cons}(A)] \cdot \overline{\textsc{Err}}(X) \right) d\mathcal{D}^{k/2}(X), \\
&= \mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{D}^{k/2}} \left[\mathbb{1}[\boldsymbol{X} \in \textsc{Cons}(A)] \cdot \overline{\textsc{Err}}(\boldsymbol{X}) \right] = \mu \cdot \mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{D}^{k/2}} \left[\overline{\textsc{Err}}(\boldsymbol{X}) \mid \boldsymbol{X} \in \textsc{Cons}(A) \right].
\end{aligned}$$

Substituting this into Equation (8) and applying the fact that A is excellent, we see that

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{D}} \left[\Phi(\boldsymbol{x}) \right] \leq \frac{\epsilon}{8} + 4 \cdot \mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{D}^{k/2}} \left[\overline{\textsc{Err}}(\boldsymbol{X}) \mid \boldsymbol{X} \in \textsc{Cons}(A) \right] \leq \frac{\epsilon}{4},$$

completing the proof. ■

Finally, we prove Claim 18. We do so by noting that it is a nearly direct consequence of the following, more general concentration inequality, which we prove below.

Claim 19 *Suppose that Ω is a set, $d \in \mathbb{N}$, and let μ be a distribution over Ω. Let $G \subseteq \Omega^d$ have density $\beta \coloneqq \mu^d(G)$. For any $x \in \Omega$ and $i \in [d]$, we define $\rho_i : \Omega \to [0,1]$ by*

$$\rho_i(x) \coloneqq \Pr_{\boldsymbol{X}\sim\mu^{(i,x)}} \left[\boldsymbol{X} \in G \right], \quad \mu^{(i,x)} \coloneqq \mu \times \cdots \mu \times \underbrace{\delta_x}_{i\text{-th pos}} \times \mu \cdots \times \mu,$$

where δ_x is the point mass distribution at x. Additionally, define $\rho : \Omega \to [0,1]$ to be the average $\rho(x) \coloneqq \frac{1}{d} \cdot \sum_{i=1}^{d} \rho_i(x)$. Then

$$\mu\left(\left\{ x \in \Omega : \rho(x) < \frac{\beta}{4} \right\} \right) < 16 \cdot \frac{\log \frac{1}{\beta}}{d}.$$

Proof First, we can rewrite the lefthand side of our target inequality as

$$\mu\left(\left\{ x \in \Omega : \rho(x) < \frac{\beta}{4} \right\} \right) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu} \left[\mathbb{1}\left[\frac{1}{d} \cdot \sum_{i=1}^{d} \rho_i(\boldsymbol{x}) < \frac{\beta}{4} \right] \right].$$

Note that for any $x \in \Omega$, if $\rho(x) < \frac{\beta}{4}$, then $\rho_i(x) < \frac{\beta}{2}$ for at least half of $i \in [d]$; thus, we have that

$$\mathbb{1}\left[\frac{1}{d} \cdot \sum_{i=1}^{d} \rho_i(x) < \frac{\beta}{4}\right] \leq \frac{2}{d} \cdot \sum_{i=1}^{d} \mathbb{1}\left[\rho_i(x) < \frac{\beta}{2}\right].$$

We can then combine the two equations above as follows,

$$\mu\left(\left\{x \in \Omega : \rho(x) < \frac{\beta}{4}\right\}\right) \leq \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mu}\left[\frac{2}{d} \cdot \sum_{i=1}^{d} \mathbb{1}\left[\rho_i(\boldsymbol{x}) < \frac{\beta}{2}\right]\right], \tag{9}$$

$$= \frac{2}{d} \cdot \sum_{i=1}^{d} \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mu}\left[\mathbb{1}\left[\rho_i(\boldsymbol{x}) < \frac{\beta}{2}\right]\right], \tag{10}$$

$$= \frac{2}{d} \cdot \sum_{i=1}^{d} \mu\left(\left\{x \in \Omega : \rho_i(x) < \frac{\beta}{2}\right\}\right). \tag{11}$$

For the sake of brevity, for each $i \in [d]$, we will define $B_i \subseteq \Omega$ by

$$B_i \coloneqq \left\{x \in \Omega : \rho_i(x) < \frac{\beta}{2}\right\},$$

so that the righthand side of Equation (11) is equal to $\frac{2}{d} \cdot \sum_{i=1}^{d} \mu(B_i)$. Our goal is now to bound this sum; to do so, we will use the following auxillary distributions in our analysis. Let $\mathcal{G}$ be the distribution induced by μ^d on G. Viewing this as a distribution over Ω^d, we can also consider its marginal distributions $\mathcal{G}_1, \ldots, \mathcal{G}_d$ over Ω. We proceed by bounding the KL-divergence of $\mathcal{G}$ and μ^d on both sides; on one hand, we see that

$$D_{\mathrm{KL}}(\mathcal{G} \,\|\, \mu^d) = \mathop{\mathbb{E}}_{\boldsymbol{X} \sim \mathcal{G}}\left[\log\left(\frac{d\mathcal{G}}{d\mu^d}(\boldsymbol{X})\right)\right] = \mathop{\mathbb{E}}_{\boldsymbol{X} \sim \mathcal{G}}\left[\log\left(\tfrac{1}{\beta}\right)\right] = \log \tfrac{1}{\beta}.$$

On the other hand, by the sub-additivity of KL-divergence from a product distribution, we have

$$D_{\mathrm{KL}}(\mathcal{G} \,\|\, \mu^d) \geq \sum_{i=1}^{d} D_{\mathrm{KL}}(\mathcal{G}_i \,\|\, \mu).$$

Combining the two and dividing by a factor of d, we have that

$$\frac{1}{d} \sum_{i=1}^{d} D_{\mathrm{KL}}(\mathcal{G}_i \,\|\, \mu) \leq \frac{\log \frac{1}{\beta}}{d}. \tag{12}$$

Next, we will bound $\mu(B_i)$ by $D_{\mathrm{KL}}(\mathcal{G}_i \,\|\, \mu)$. To do so, we can expand $D_{\mathrm{KL}}(\mathcal{G}_i \,\|\, \mu)$ as follows,

$$\begin{aligned} D_{\mathrm{KL}}(\mathcal{G}_i \,\|\, \mu) &= \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{G}_i}\left[\log\left(\frac{d\mathcal{G}_i}{d\mu}(\boldsymbol{x})\right)\right], \\ &= \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{G}_i}\left[\log\left(\frac{d\mathcal{G}_i}{d\mu}(\boldsymbol{x})\right) \cdot \mathbb{1}[\boldsymbol{x} \in B_i]\right] + \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{G}_i}\left[\log\left(\frac{d\mathcal{G}_i}{d\mu}(\boldsymbol{x})\right) \cdot \mathbb{1}[\boldsymbol{x} \notin B_i]\right], \\ &\geq \mathcal{G}_i(B_i) \cdot \log\left(\frac{\mathcal{G}_i(B_i)}{\mu(B_i)}\right) + (1 - \mathcal{G}_i(B_i)) \cdot \log\left(\frac{1 - \mathcal{G}_i(B_i)}{1 - \mu(B_i)}\right), \end{aligned}$$

where the last inequality comes from applying the log sum inequality to each expectation. We will denote $\delta_i \coloneqq \mu(B_i)$ and define $g_{\delta_i} : [0,1] \to \mathbb{R}$ by

$$g(x) \coloneqq x \cdot \log\left(\frac{x}{\delta_i}\right) + (1-x) \cdot \log\left(\frac{1-x}{1-\delta_i}\right).$$

With this, we can rewrite the preceding inequality as

$$D_{\mathrm{KL}}(\mathcal{G}_i \,\|\, \mu) \geq g_{\delta_i}(\mathcal{G}_i(B_i)). \tag{13}$$

To further bound this, we will use the following bound on $\mathcal{G}_i(B_i)$ in terms of $\mu(B_i)$. To do so, we first note that, using Bayes' theorem,[1]

$$\frac{d\mathcal{G}_i}{d\mu}(x) = \frac{\Pr_{\boldsymbol{X}\sim\mu^d}[\boldsymbol{X}_i = x \mid \boldsymbol{X} \in \mathcal{G}]}{\mu(x)} = \frac{\Pr_{\boldsymbol{X}\sim\mu^{(i,d)}}[\boldsymbol{X} \in G] \cdot \mu(x)}{\mu(x) \cdot \mu^d(G)} = \frac{\rho_i(x)}{\beta}.$$

Thus, we see that

$$\mathcal{G}_i(B_i) = \int_\Omega \mathbb{1}[x \in B_i]\, d\mathcal{G}_i(x) = \int_\Omega \mathbb{1}[x \in B_i] \cdot \frac{\rho_i(x)}{\beta}\, d\mu(x) \leq \frac{1}{2} \cdot \int_\Omega \mathbb{1}[x \in B_i]\, d\mu(x) = \frac{\mu(B_i)}{2}.$$

Thus, $\mathcal{G}_i(B_i) \in [0, \frac{\delta_i}{2}]$, and so we can bound Equation (13) below by

$$D_{\mathrm{KL}}(\mathcal{G}_i \,\|\, \mu) \geq g_{\delta_i}(\mathcal{G}_i(B_i)) \geq \min_{x\in[0,\frac{\delta_i}{2}]} g_{\delta_i}(x).$$

Now, note that the derivative of $g_{\delta_i}(x)$ with respect to x is

$$\frac{\partial}{\partial x} g_{\delta_i}(x) = \log\left(\frac{x}{1-x} \cdot \frac{1-\delta_i}{\delta_i}\right),$$

which, since $\frac{x}{1-x} \leq \frac{\delta_i}{1-\delta_i}$ for $x \leq \frac{\delta_i}{2}$, is always negative on $[0, \frac{\delta_i}{2}]$. Thus g_{δ_i} is minimized at the boundary of this interval, and so we know that

$$\begin{aligned}
D_{\mathrm{KL}}(\mathcal{G}_i \,\|\, \mu) \geq \min_{x\in[0,\frac{\delta_i}{2}]} g_{\delta_i}(x) &\geq g_{\delta_i}\left(\frac{\delta_i}{2}\right), \\
&= \frac{\delta_i}{2} \cdot \log\left(\frac{1}{2}\right) + \left(1 - \frac{\delta_i}{2}\right) \cdot \log\left(\frac{1-\frac{\delta_i}{2}}{1-\delta_i}\right), \\
&= \frac{\delta_i}{2} \cdot \log\left(\frac{1}{2}\right) + \left(1 - \frac{\delta_i}{2}\right) \cdot \log\left(1 + \frac{\delta_i}{2\cdot(1-\delta_i)}\right), \\
&\geq \frac{\delta_i}{2} \cdot \log\left(\frac{1}{2}\right) + \left(1 - \frac{\delta_i}{2}\right) \cdot \left(\frac{\delta_i}{2\cdot(1-\delta_i)} \cdot \frac{1}{1+\frac{\delta_i}{2(1-\delta_i)}} \cdot \frac{1}{\ln(2)}\right), \\
&= \frac{\delta_i}{2} \cdot \log\left(\frac{1}{2}\right) + \left(1 - \frac{\delta_i}{2}\right) \cdot \left(\frac{\delta_i}{2} \cdot \frac{1}{1-\frac{\delta_i}{2}} \cdot \frac{1}{\ln(2)}\right), \\
&= \frac{\delta_i}{2} \cdot \left(\frac{1}{\ln(2)} - 1\right) \geq \frac{\delta_i}{8}.
\end{aligned}$$

1. Although the equation is stated using conditional probabilities which may be 0 for continuous distributions, we note that the same end-to-end equality holds by replacing those conditional probabilities with conditional densities.

Combining this with Equations (11) and (12) and recalling that $\delta_i = \mu(B_i)$, we have that

$$\mu\left(\left\{x \in \Omega : \rho(x) < \frac{\beta}{4}\right\}\right) \leq \frac{2}{d} \cdot \sum_{i=1}^{d} \delta_i \leq \frac{16}{d} \cdot \sum_{i=1}^{d} D_{\mathrm{KL}}(\mathcal{G}_i \,\|\, \mu) \leq 16 \cdot \frac{\log \frac{1}{\beta}}{d},$$

which completes the proof. ■

A.2. Proof of Lemma 14

In this section, we prove that a random piece of advice is excellent with probability at least $\frac{\gamma}{4}$.

Proof First, we will bound the probability that a random piece of advice $\boldsymbol{A} \sim \mathcal{A}$ is good. Since C computes f^k to accuracy $\geq \gamma$, we know that a random $\boldsymbol{A} \sim \mathcal{A}$ is correct with probability at least γ. On the other hand, the probability that a random $\boldsymbol{A} \sim \mathcal{A}$ is correct but not good is

$$\Pr_{\boldsymbol{A}\sim\mathcal{A}}[\boldsymbol{A} \text{ is correct but not good}] = \Pr_{\boldsymbol{A}\sim\mathcal{A}}\left[\boldsymbol{Y} \text{ is correct } \wedge\ \overline{\textsc{Cons}}(\boldsymbol{A}) < \tfrac{\gamma}{2}\right].$$

While we had $\mathcal{A}$ denote the distribution on advice which first samples $\boldsymbol{Y} \sim \mathcal{D}^k$ then $\boldsymbol{S} \sim \binom{k}{k/2}$ independently, it's equivalent to first sample $\boldsymbol{S} \sim \binom{k}{k/2}$, then draw $\boldsymbol{Y}_1, \boldsymbol{Y}_2 \sim \mathcal{D}^{k/2}$ independently, and finally set $\boldsymbol{Y} \coloneqq \textsc{Il}_{\boldsymbol{S}}(\boldsymbol{Y}_1, \boldsymbol{Y}_2)$. With this view, we have

$$\begin{aligned}\Pr_{\boldsymbol{A}\sim\mathcal{A}}\left[\boldsymbol{Y} \text{ correct } \wedge\ \mathcal{N}_{\boldsymbol{A}}(\textsc{Cons}(\boldsymbol{A})) < \tfrac{\gamma}{2}\right] &= \Pr_{\boldsymbol{S},\boldsymbol{Y}_1,\boldsymbol{Y}_2}\left[\boldsymbol{Y} \text{ correct } \wedge \overline{\textsc{Cons}}(\boldsymbol{Y}_1, \boldsymbol{S}) < \tfrac{\gamma}{2}\right] \\ &\leq \Pr_{\boldsymbol{S},\boldsymbol{Y}_1,\boldsymbol{Y}_2}\left[\boldsymbol{Y} \text{ correct} \mid \overline{\textsc{Cons}}(\boldsymbol{Y}_1, \boldsymbol{S}) < \tfrac{\gamma}{2}\right].\end{aligned}$$

Conditioned on $\boldsymbol{S}$ and $\boldsymbol{Y}_1$, $\boldsymbol{Y} \coloneqq \textsc{Il}_{\boldsymbol{S}}(\boldsymbol{Y}_1, \boldsymbol{Y}_2)$ is distributed according to $\mathcal{N}_{(\boldsymbol{Y}_1,\boldsymbol{S})}$. Moreover, $\boldsymbol{Y}$ is correct only if $\boldsymbol{Y} \in \textsc{Cons}(\boldsymbol{Y}_1, \boldsymbol{S})$. Conditioned on $\overline{\textsc{Cons}}(\boldsymbol{Y}_1, \boldsymbol{S}) < \frac{\gamma}{2}$, this occurs with probability $\leq \frac{\gamma}{2}$. Thus, the righthand side of the above equation is at most $\frac{\gamma}{2}$, so that

$$\Pr_{\boldsymbol{A}\sim\mathcal{A}}[\boldsymbol{A} \text{ is good}] \geq \Pr_{\boldsymbol{A}\sim\mathcal{A}}[\boldsymbol{A} \text{ is correct}] - \Pr_{\boldsymbol{A}\sim\mathcal{A}}[\boldsymbol{A} \text{ is correct but not good}] \geq \gamma - \tfrac{\gamma}{2} \geq \tfrac{\gamma}{2}. \tag{14}$$

To bound the probability that a random $\boldsymbol{A} \sim \mathcal{A}$ is excellent, we will bound the probability that $\boldsymbol{A}$ is good but not excellent. Let $E_0(A)$ be the event that A is good, and have $E_1(A)$ represent the event that A is good but not excellent. We will write $\alpha \coloneqq \frac{\epsilon}{32}$. Note that if A is good but not excellent, then

$$\mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{N}_A}\left[\overline{\textsc{Err}}(\boldsymbol{X}) \mid \boldsymbol{X} \in \textsc{Cons}(\boldsymbol{A})\right] \geq \alpha,$$

which, in particular, implies that

$$\mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{N}_A}\left[\overline{\textsc{Err}}(\boldsymbol{X}) \cdot \mathbb{1}\left[\overline{\textsc{Err}}(\boldsymbol{X}) \geq \tfrac{\alpha}{2}\right] \mid \boldsymbol{X} \in \textsc{Cons}(\boldsymbol{A})\right] \geq \tfrac{\alpha}{2}.$$

We can use this to bound the probability that an edge is good but not excellent as follows,

$$\begin{aligned}
\Pr_{\boldsymbol{A}\sim\mathcal{A}}[E_1(\boldsymbol{A})] &= \mathop{\mathbb{E}}_{\boldsymbol{A}\sim\mathcal{A}}\left[\mathbb{1}\left[E_0(\boldsymbol{A}) \text{ but } \mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{N}_{\boldsymbol{A}}}\left[\overline{\textsc{Err}}(\boldsymbol{X}) \mid \boldsymbol{X}\in\textsc{Cons}(\boldsymbol{A})\right] \geq \alpha\right]\right], \\
&\leq \mathop{\mathbb{E}}_{\boldsymbol{A}\sim\mathcal{A}}\left[\mathbb{1}\left[E_0(\boldsymbol{A}) \text{ but } \mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{N}_{\boldsymbol{A}}}\left[\overline{\textsc{Err}}(\boldsymbol{X})\cdot\mathbb{1}\left[\overline{\textsc{Err}}(\boldsymbol{X})\geq\tfrac{\alpha}{2}\right] \mid \boldsymbol{X}\in\textsc{Cons}(\boldsymbol{A})\right] \geq \tfrac{\alpha}{2}\right]\right], \\
&\leq \frac{2}{\alpha}\cdot\mathop{\mathbb{E}}_{\boldsymbol{A}\sim\mathcal{A}}\left[\mathbb{1}[E_0(\boldsymbol{A})]\cdot\mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{N}_{\boldsymbol{A}}}\left[\overline{\textsc{Err}}(\boldsymbol{X})\cdot\mathbb{1}\left[\overline{\textsc{Err}}(\boldsymbol{X})\geq\tfrac{\alpha}{2}\right] \mid \textsc{Cons}(\boldsymbol{A})\right]\right], \\
&= \frac{2}{\alpha}\cdot\mathop{\mathbb{E}}_{\boldsymbol{A}\sim\mathcal{A}}\left[\frac{\mathbb{1}[E_0(\boldsymbol{A})]}{\overline{\textsc{Cons}}(\boldsymbol{A})}\cdot\mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{N}_{\boldsymbol{A}}}\left[\overline{\textsc{Err}}(\boldsymbol{X})\cdot\mathbb{1}\left[\boldsymbol{X}\in\textsc{Cons}(\boldsymbol{A})\wedge\overline{\textsc{Err}}(\boldsymbol{X})\geq\tfrac{\alpha}{2}\right]\right]\right], \\
&\leq \frac{4}{\alpha\cdot\gamma}\cdot\mathop{\mathbb{E}}_{\boldsymbol{A}\sim\mathcal{A}}\left[\mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{N}_{\boldsymbol{A}}}\left[\mathbb{1}\left[E_0(\boldsymbol{A})\wedge\boldsymbol{X}\in\textsc{Cons}(\boldsymbol{A})\wedge\overline{\textsc{Err}}(\boldsymbol{X})\geq\tfrac{\alpha}{2}\right]\cdot\overline{\textsc{Err}}(\boldsymbol{X})\right]\right], \\
&\leq \frac{4}{\alpha\cdot\gamma}\cdot\mathop{\mathbb{E}}_{\boldsymbol{A}\sim\mathcal{A}}\left[\mathop{\mathbb{E}}_{\boldsymbol{X}\sim\mathcal{N}_{\boldsymbol{A}}}\left[\mathbb{1}\left[E_0(\boldsymbol{A})\wedge\boldsymbol{X}\in\textsc{Cons}(\boldsymbol{A})\wedge\overline{\textsc{Err}}(\boldsymbol{X})\geq\tfrac{\alpha}{2}\right]\right]\right], \\
&\leq \frac{4}{\alpha\cdot\gamma}\cdot\Pr_{\boldsymbol{A}\sim\mathcal{A},\,\boldsymbol{X}\sim\mathcal{N}_{\boldsymbol{A}}}\left[E_0(\boldsymbol{A})\wedge\boldsymbol{X}\in\textsc{Cons}(\boldsymbol{A})\wedge\overline{\textsc{Err}}(\boldsymbol{X})\geq\tfrac{\alpha}{2}\right], \\
&\leq \frac{4}{\alpha\cdot\gamma}\cdot\underbrace{\Pr_{\boldsymbol{A}\sim\mathcal{A},\,\boldsymbol{X}\sim\mathcal{N}_{\boldsymbol{A}}}\left[\boldsymbol{A}\text{ is correct}\wedge\boldsymbol{X}\in\textsc{Cons}(\boldsymbol{A})\wedge\overline{\textsc{Err}}(\boldsymbol{X})\geq\tfrac{\alpha}{2}\right]}_{\xi}.
\end{aligned}$$

We will now reinterpret this last probability as follows. Rather than drawing $\boldsymbol{A}\sim\mathcal{A}$ then $\boldsymbol{X}\sim\mathcal{N}_{\boldsymbol{A}}$, we could instead do the following: draw $\boldsymbol{Y}\sim\mathcal{D}^k$, a random subset $\boldsymbol{S}\sim\binom{k}{k/2}$, and a random $\boldsymbol{Z}\sim\mathcal{D}^{k/2}$, then set $\boldsymbol{X}=\boldsymbol{Y}$, and $\boldsymbol{A}=(\textsc{Il}_{\boldsymbol{S}}(\boldsymbol{Y}_{\boldsymbol{S}},\boldsymbol{Z}),\boldsymbol{S})$. In this case, if $\boldsymbol{A}$ is correct and $\boldsymbol{X}\in\textsc{Cons}(\boldsymbol{A})$, then $\boldsymbol{S}$ must be disjoint from $\textsc{Err}(\boldsymbol{X})$; thus, we can bound this probability as

$$\begin{aligned}
\xi &\leq \Pr_{\boldsymbol{Y}\sim\mathcal{D}^k,\,\boldsymbol{S}\sim\binom{k}{k/2},\,\boldsymbol{Z}\sim\mathcal{D}^{k/2}}\left[\boldsymbol{S}\cap\textsc{Err}(\boldsymbol{Y})=\emptyset\ \wedge\ \overline{\textsc{Err}}(\boldsymbol{Y})\geq\tfrac{\alpha}{2}\right], \\
&\leq \Pr_{\boldsymbol{Y}\sim\mathcal{D}^k,\,\boldsymbol{S}\sim\binom{k}{k/2}}\left[\boldsymbol{S}\cap\textsc{Err}(\boldsymbol{Y})=\emptyset \mid \overline{\textsc{Err}}(\boldsymbol{Y})\geq\tfrac{\alpha}{2}\right], \\
&\leq \prod_{i=0}^{\frac{k}{2}-1}\left(1-\frac{\alpha}{2}-\frac{i}{k}\right)\leq\exp\left(-\frac{\alpha}{2}\cdot k-\frac{k}{8}\right)\leq\frac{\alpha\cdot\gamma^2}{16}.
\end{aligned}$$

Thus, we have that

$$\Pr_{\boldsymbol{A}\sim\mathcal{A}}[\boldsymbol{A}\text{ is good but not excellent}]\leq\frac{4}{\alpha\cdot\gamma}\cdot\frac{\alpha\gamma^2}{16}\leq\frac{\gamma}{4}.$$

Combining this with Equation (14), we can complete the proof,

$$\Pr_{\boldsymbol{A}\sim\mathcal{A}}[\boldsymbol{A}\text{ is excellent}]\geq\Pr_{\boldsymbol{A}\sim\mathcal{A}}[\boldsymbol{A}\text{ is good}]-\Pr_{\boldsymbol{A}\sim\mathcal{A}}[\boldsymbol{A}\text{ is good but not excellent}]\geq\tfrac{\gamma}{2}-\tfrac{\gamma}{4}=\tfrac{\gamma}{4},$$

which completes the proof. ∎