When is Memorization of Irrelevant Training Data Necessary for High-Accuracy Learning?

Gavin Brown* Mark Bun* Vitaly Feldman† Adam Smith*
Kunal Talwar†

December 18, 2020

Abstract

Modern machine learning models are complex and frequently encode surprising amounts of information about individual inputs. In extreme cases, complex models appear to memorize entire input examples, including seemingly irrelevant information (social security numbers from text, for example). In this paper, we aim to understand whether this sort of memorization is necessary for accurate learning. We describe natural prediction problems in which every sufficiently accurate training algorithm must encode, in the prediction model, essentially all the information about a large subset of its training examples. This remains true even when the examples are high-dimensional and have entropy much higher than the sample size, and even when most of that information is ultimately irrelevant to the task at hand. Further, our results do not depend on the training algorithm or the class of models used for learning.

Our problems are simple and fairly natural variants of the next-symbol prediction and the cluster labeling tasks. These tasks can be seen as abstractions of image- and text-related prediction problems. To establish our results, we reduce from a family of one-way communication problems for which we prove new information complexity lower bounds.

*Computer Science Department, Boston University. {grbrown,mbun,ads22}@bu.edu. GB and AS are supported in part by NSF award CCF-1763786 as well as a Sloan Foundation research award. MB is supported by NSF award CCF-1947889.

†Apple.

Contents

1 Introduction

Algorithms for supervised machine learning take in training data, attempt to extract the relevant information, and produce a prediction algorithm, also called a *model* or *hypothesis*. The model is used to predict a particular feature on future examples, ideally drawn from the same distribution as the training data. Training algorithms operate on a huge range of prediction tasks, from image classification to language translation, often involving highly sensitive data. To succeed, models must of course contain information about the data they were trained on. In fact, many well-known machine learning algorithms create models that explicitly encode their training data: the "model" for the k-Nearest Neighbor classification algorithm is a description of the dataset, and Support Vector Machines include points from the dataset as the "support vectors." These models can clearly be said to memorize at least part of their training data.

Commonly, however, memorization is an implicit, unintended side effect. Recent work [31, 42] has studied how deep neural networks, when trained using standard techniques on small amounts of data, may reproduce the data exactly. More strikingly, Carlini et al. [12] demonstrated that models for next word prediction may regurgitate short sequences of random digits. This occurred with standard training techniques, where the only modification to the dataset was to add several copies of the sequence. The learning algorithm, apparently unable to tell the difference between the random sequence and natural language phrases, memorized the sequence verbatim. The reasons this behavior appears are of interest to both the foundations of machine learning and privacy. For example, a model accidentally memorizing Social Security numbers from a text data set presents a glaring opportunity for identity theft.

In this paper, we aim to understand when this sort of memorization is unavoidable. We give natural prediction problems in which *every reasonably accurate training algorithm must encode, in the prediction model, information about a large subset of its training examples.* Importantly, this holds even when most of that information is ultimately irrelevant to the task at hand. We show this for two types of tasks: a next-symbol prediction task (intended to abstract language modeling tasks) and a multiclass classification problem in which each class distribution is a simple product distribution in $\{0,1\}^d$ (intended to abstract a range of tasks like image labeling). We prove these results by deriving new bounds on the information complexity of learning, building on the formalism of Bassily, Moran, Nachum, Shafer, and Yehudayoff [6].

We note that the word "memorization" is commonly used in the literature to refer to the phenomenon of *label memorization* in which a learning algorithm fits arbitrarily chosen (or noisy) labels of training data points. Such memorization is a well-documented property of modern deep learning and is related to interpolation (or perfect fitting of all the training labels) [41, 3, 26, 40]. Feldman [19] recently showed that label memorization is necessary for achieving near-optimal accuracy on test data when the data distribution is *long-tailed*. Further, Feldman and Zhang [21] empirically demonstrate the importance of label memorization for deep learning algorithms on standard image classification datasets. In contrast, we study settings in which most of the information about entire high-dimensional (and high-entropy) training examples must be encoded by near-optimal learning algorithms.

Problem setting We define a *problem instance* p as a distribution over labeled examples: $p \in \Delta(\mathcal{X})$ where $\mathcal{X} = \mathcal{F} \times \mathcal{Y}$ is a space of examples (in $\mathcal{F}$) paired with labels in $\mathcal{Y}$. A dataset

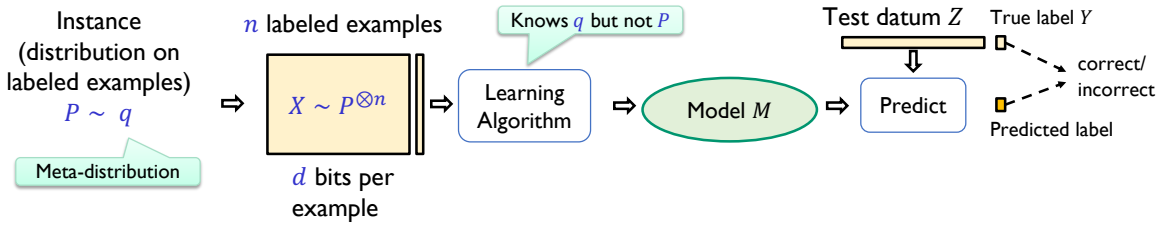


Figure 1: Problem setting. We aim to understand the information about the data X that is encoded in the model description M.

$X \in \mathcal{X}^n$ is generated by sampling i.i.d. from such a distribution. We use d to denote the dimension of the data, so X can be described in $\Theta(nd)$ bits. In contrast to the well-known PAC model of learning, we do not explicitly consider a concept class of functions. Rather, the instance p is itself drawn from a metadistribution q, dubbed the *learning task*. The learning task q is assumed to be known to the learner, but the specific problem instance is a priori unknown. We write P to denote a random instance (distributed according to q), and p to denote a particular realization. See Figure 1.

The learning algorithm A receives a sample $X \sim P^{\otimes n}$ and produces a model $M = A(X)$ that can be interpreted as a (possibly randomized) map $M : \mathcal{F} \to \mathcal{Y}$. The model errs on a test data point (z, y) if $M(z) \neq y$ (for simplicity, we only consider misclassification error). The learner A's overall error, denoted $\mathsf{err}_{q,n}(A)$ on the task q with sample size n is its expected error over P drawn from q, X drawn from $P^{\otimes n}$ and test point (Z, Y) drawn from P. That is,

$$\mathsf{err}_{q,n}(A) \stackrel{\text{def}}{=} \Pr_{\substack{P\sim q,\\ X\sim P^{\otimes n},(Z,Y)\sim P,\\ \text{coins of } A,M}} (M(Z) \neq Y \text{ where } M = A(X)) \tag{1}$$

For some probability calculations, we use the shorthand "A errs" to denote a misclassification by $A(X)$ (the event above), so that $\Pr(A \text{ errs}) = \mathsf{err}_{q,n}(A)$.

Example 1.1. One learning task we consider is labeling the components in a mixture of N product distributions on the hypercube. An interesting special case is a uniform mixture of *uniform distributions over subcubes*. Here each component $j \in [N]$ of the mixture is specified by a sparse set of fixed indices $\mathcal{I}_j \subseteq d$ with values $\{b_j(i)\}_{i\in\mathcal{I}_j}$. Each labeled example is generated by picking a subpopulation $j \in [N]$ uniformly at random (which also serves as the label), for each $i \in \mathcal{I}_j$ setting $z(i) = b_j(i)$, and picking the other entries uniformly at random to obtain a feature vector $z \in \{0,1\}^d$. The labeled example is then (z, j). A natural meta-distribution q generates each set $\mathcal{I}_j$ by adding indices i to $\mathcal{I}_j$ independently with some probability, and fixes the values $b_j(i)$ at those indices uniformly.

Given a set of n labeled examples and a test point z' (drawn from the same distribution, but missing its label), the learner's job is to infer the label of the mixture component which generated z'. ■

Given q, a particular meta-distribution, and n, the number of samples in the data set, there exists a learner A_{OPT} depending on q (called *Bayes-optimal*) that minimizes the overall error on the task q. For any given task, this minimal error will be our reference—for $\epsilon \geq 0$, a learner is ϵ-suboptimal (for q and n) if its error is within ϵ of that of A_{OPT} on samples of size

n, that is, $\mathsf{err}_{q,n}(A) \leq \mathsf{err}_{q,n}(A_{OPT}) + \epsilon$. In our problem of cluster labeling (Example 1.1), the optimal learner would work roughly as follows: for each component j of the mixture, produce a set $\hat{\mathcal{I}}_j$ of features that are fixed in the samples from that component (indices which take two values in different samples are irrelevant to classification). We have $\mathcal{I}_j \subseteq \hat{\mathcal{I}}_j$, but with few samples from cluster j, $\hat{\mathcal{I}}_j$ will contain many irrelevant indices. The optimal learner will balance the Hamming distance on $\hat{\mathcal{I}}_j$ against the probability of achieving a set of fixed indices of that size, plus the fact that we expect half of the non-fixed indices to match. In our analysis, it will suffice to analyze the much simpler, but still low-error, learner that simply compares Hamming distances to a single randomly-chosen sample from each component.

1.1 Our contributions

We present natural prediction problems q where any algorithm with near-optimal accuracy on q must memorize $\Omega(nd)$ bits about the sample. Furthermore, this memorized information is really about the specific sample X and not about the data distribution P.

Theorem 1.1 (Informal; see Corollaries 3.2 and 4.2)**.** *For all n and d, there exist natural tasks q for which any algorithm A that satisifes, for small constant ϵ,*

$$\mathsf{err}_{q,n}(A) \leq \mathsf{err}_{q,n}(A_{OPT}) + \epsilon$$

also satisfies

$$I(X; M \mid P) = \Omega(nd)\,,$$

where $P \sim q$ is the distribution on labeled examples, $X \sim P^{\otimes n}$ is a sample of size n from P, $M = A(X)$ is the model, and examples lie in $\{0,1\}^d$ (so $H(X) \leq nd$). The asymptotic expression holds for any sequence of n, d pairs; the constant depends only on ϵ.

To interpret this result, recall that the conditional mutual information is defined via two conditional entropy terms: $I(X; M \mid P) = H(X \mid P) - H(X \mid M, P)$. Consider an informed observer who knows the full data distribution P (but not X). The term $I(X; M|P)$ captures how the observer's uncertainty about X is reduced after observing the model M. Since P is a full description of the problem instance, the term $H(X \mid P)$ captures the uncertainty about what is "unique to the data set," such as noise or irrelevant features. So $I(X; M \mid P) = \Omega(nd)$ means that *not only must the learning algorithm encode a constant fraction of the information it receives, but also that a constant fraction of what it encodes is irrelevant to the task.* For one of the problems we consider, we even get that $I(X; M|P) = (1 - o(1))H(X'|P)$ where X' is a subset of X of expected size $\Omega(n)$. That is, a subset of examples is encoded nearly completely in the model.

The meta-distribution q captures the learner's initial uncertainty about the problem, and is essential to the result: if the exact distribution P were known to the learner A, it could simply ignore X and write down an optimal classifier for P as its model M. In that case, we would have $I(X; M|P) = 0$. That said, since conditional information is an average over realizations of P, our result also means that for every learner, $I(M; X)$ is large for some particular p in the support of q. Such worst-case bounds were considered in [6, 30, 29], with which we compare below.

We study two classes of learning tasks. The first is a next-symbol prediction problem (intended to abstract language modeling tasks). The second is the cluster labeling problem

where individual classes are mixtures of product distributions over the boolean hypercube (a generalization of Example 1.1 that allows more natural mixture weights). The exact problems are defined in Section 1.2.

In all the tasks we consider, data are drawn from a mixture of subpopulations. We consider settings where there are likely to be $\Omega(n)$ components of the mixture distribution from which the data set contains only one example. Leveraging new communication complexity bounds, we show that $\Omega(d)$ bits about most of these "singleton" examples must be encoded in M for the algorithm to perform well on average. (We discuss the mixture structure and our proof techniques in more detail below.)

Returning to the cluster labeling problem in Example 1.1, recall that the learner receives an nd-bit data set, which has entropy $\Theta(nd)$, even conditioned on P (when σ is bounded away from 1). This "remaining uncertainty" $H(X|P)$ is, ignoring lower-order terms, exactly the uncertainty about noise applied to the cluster centers. Showing $I(X; M \mid P) = \Omega(nd)$, then, establishes not only that the model must contain a large amount of information about X, but also that it must encode a large amount of information about the noise, information completely irrelevant to the task at hand.

On a technical level, our results are related to those of Bassily, Moran, Nachum, Shafer, and Yehudayoff [6] (and later [30, 29]), who study lower bounds on the mutual information $I(X; M)$ achievable by a PAC learner for a given class of Boolean functions $\mathcal{H}$. Specifically, for the class $\mathcal{H}_{thresh}$ of threshold functions on $[2^d]$, they give a learning task[1] for which every *proper and consistent* learning algorithm (i.e., one that is limited to outputting function in $\mathcal{H}_{thresh}$ that labels the training data perfectly) satisfies $I(X; M) = \Omega(\log d)$ [6, 30]. Furthermore, Nachum et al. [30] extend this result to provide a hypothesis class $\mathcal{H}$ with VC dimension n over the input space $[n] \times \{0,1\}^d$ such that learners receiving $\Omega(n)$ samples must leak at least $I(X; M) = \Omega(n \cdot \log(d - \log n))$ bits about the input via their message. The direct sum construction in [30] is similar to our construction: they construct a learning problem out of a product of simpler problems, in their case thresholds, and relate the difficulty of the overall problem to that of the components.

Our results on next-symbol prediction can be cast in terms of learning threshold functions. As such, our results provide an alternative to those of [5, 30]. First, they are quantitatively stronger: we give a lower bound of $(1-g(\epsilon))nd$ rather than $\Omega(n \log d)$. Second, the assumption on the learner is very different: our bounds apply to all sufficiently accurate learners, whereas those of [5, 30] require the learner to be proper and consistent (in the parameter regimes we work in, their assumptions does not imply high accuracy, so the assumptions are not comparable). The shift in assumptions does partly address an open question in [30].

Implications While the problems we describe are intentionally simplified to allow for clean theoretical analysis, they rely on properties of natural learning problems such as similarity to a cluster center. Thus our results suggest that memorization of irrelevant information that is often observed in practice, is a fundamental property of learning and not an artifact of the particular deep learning techniques.

These results have implications for learning algorithms that (implicitly) limit memorization. One class of such algorithms are aim to compress models (for example to reduce memory

[1]The results of [6, 30, 29] are formulated in terms of worst-case information leakage over a class of problems; by a minimax argument, they imply the existence of a single hard meta-distribution q.

usage) since description length upper bounds the mutual information. Another class of such algorithms is differentially private training procedures [16]. More formally, it is known that differential privacy implies a bound on the mutual information $I(X; M)$ [27, 17, 34, 10]. Our results imply that such algorithm might not be able to achieve the same accuracy as unrestricted algorithms. In itself, that is nothing new: there is a long line of work on differentially private learning (e.g., [8, 25]), including a number of striking separations from nonprivate algorithms [11, 5, 2]. There are also well established attacks on statistical and learning algorithms for high-dimensional problems (starting with [15]; see [18] for a survey of the theory, and a recent line of work on membership inference attacks [36] for empirical results). However, our results show a novel aspect of the limits of private learning: in the settings we consider, *successful learners must memorize exactly those parts of the data that are most likely to be sensitive*—outliers from small subpopulations, including their peculiar details (modeled here as noisy or irrelevant features).

Variations on the main result Different learning tasks exhibit variations and refinements of this central result. The mutual information lower bound implies that the model itself must be large, occupying at least $\Omega(nd)$ bits. But for some tasks we present, there exist ϵ-suboptimal models needing only $O(n \log(n/\epsilon) \log d)$ bits to write down (in the parameter regime we consider, where the problem scales with n). That is, with n samples the learning algorithm must output a model exponentially larger than what is required with sufficient data (or exact knowledge of P). In particular, for a given target accuracy level, there is a gradual drop in the size of the model, and the information specific to X, that is necessary (starting at $\Theta(n_0 d)$ where n_0 is the minimal sample size needed for that accuracy, and tending to $O(n_0 \log n_0 \log d)$ as the sample size n grows). For task-specific discussion, see Section 4 Remark 2, and Appendix D.2.

Another variation of our results gives a qualitatively stronger lower bound. For some tasks, we are able to demonstrate that *entire* samples must be memorized, in the following sense:

Theorem 1.2 (Informal; see Corollary 3.2)**.** *There exist natural tasks q for which every data set X has a subset of records X_S such that*

- $\mathbb{E}[|X_S|] = \Omega(n)$ *and* $H(X_S|P) = \Omega(nd)$*, and*
- *any algorithm A that satisifes* $\mathsf{err}_{q,n}(A) \leq \mathsf{err}_{q,n}(A_{OPT}) + \epsilon$ *also satisfies*

$$I(X_S; M \mid P) = (1 - o(1))H(X_S|P)\,.$$

This statement implies $I(X; M \mid P) = \Omega(nd)$, but is a qualitatively different statement: as the learning algorithm's accuracy approaches optimal, it must reveal *everything* about those examples in X_S, at least information-theoretically. For these tasks, there is no costless compression the learning algorithm can apply: any reduction will increase the achievable error. We conjecture that this "whole-sample memorization" type of lower bound applies to all the tasks we study, and in fact give a simple conjecture on one-way information complexity that would imply such strong memorization (see below and Section 4.2).

Finally, in addition to the memorization of noise or irrelevant features, in some settings we show how near-optimal models may be forced to memorize examples that are themselves

entirely "useless," i.e. could be ignored with only a negligible loss in accuracy. That is, not only must irrelevant details of useful examples be memorized, but one must also memorize examples that come from very low-probability parts of the distribution. This behavior relies on a particular type of mixture structure and the long-tailed distribution setup of [19]. We explain the concept and the statement more carefully in Section 2.

1.2 Techniques: Subpopulations, Singletons, and Information Complexity

The learning tasks q we consider share a basic structure: each distribution P consists of a mixture, with coefficients $D \in \Delta([N])$ over subpopulations $j \in [N]$, each with a different distribution C_j over labeled examples. The mixture coefficients D may be deterministic (e.g. uniform) or random; for now, the reader may keep in mind the uniform mixture setting, with $N = n$ (so the number of subpopulations is the same as the sample size). The C_j's are themselves sampled i.i.d. from a meta-distribution q_c.

As in [19], we look at how the learning algorithm behaves on the subset of examples that are *singletons*, that is, sole representatives (in X) of their subpopulation. For any data set X, let $X_S \subseteq X$ denote the subset of singletons. We consider mixture weights D where X_S has size $\Omega(n)$ with high probability. We show that for our tasks, a successful learner must roughly satisfy $I(X_S; M|P) = \Omega(d|X_S|)$.

One-Way Information Complexity of Singletons We prove the bound by showing that a learner implies a good strategy for a related communication game, dubbed Singletons(k, q_c). In this game, nature generates k distributions $C_1, \ldots, C_k$, i.i.d. from q_c, along with a uniformly random index $j^* \in [k]$. One player, Alice, receives a list $(x_1, \ldots, x_k)$ of labeled examples, where $x_j \sim C_j$. A second player Bob, receives only the feature vector z from a fresh draw $(z, y) \sim C_{j^*}$. Alice sends a single message M to Bob, who produces a predicted label $\hat{y}$. Alice and Bob win if $\hat{y} = y$.

Example 1.2 (Nearest neighbor)**.** For the hypercube task corresponding to Example 1.1, let q_{HC} be the distribution from which the $\{C_j\}$ are sampled. In Singletons(k, q_{HC}), there are k centers $c_1, \ldots, c_k$ uniform in $\{0,1\}^d$. Alice gets a list $X' = (x_1, \ldots, x_k) \in (\{0,1\}^d)^k$, where $\forall j, \forall i \in \mathcal{I}_j, x_j(i) = b_j(i)$ and $\forall i \notin \mathcal{I}_j$, $x_j(i) = \text{Bernoulli}(1/2)$ (the label, j, is implicit in the ordered list). Bob receives z for a random index j^* and must predict j^*. Equivalently, we may view the game as a version of the nearest neighbor problem, treating Alice's input list $(x_1, \ldots, x_k)$ as uniformly random in $(\{0,1\}^d)^k$, and Bob as receiving a corrupted version of the one of the x_j's. If each bit is chosen to be fixed independently with probability ρ, this corruption is equivalent to $z = BSC_{\frac{1-\rho}{2}}(x_{j^*})$. Again, Bob must guess j^* with the help of a single message from Alice. Now if Bob were to see Alice's entire input, his best strategy would be to guess the nearest neighbor in X' to z. He succeeds with high probability as long as $\rho \geq c\sqrt{\frac{\ln k}{d}}$ for $c > \sqrt{2}$. ■

The question is thus: can Bob succeed with high probability when Alice sends $o(nd)$ bits? One novel technical result bounds the information complexity of this nearest neighbor problem.

Lemma 1.3 (informal; see Lemma 4.9)**.** *For all $k, d \in \mathbb{N}$, $c > \sqrt{2}$, $\rho = c\sqrt{\frac{\log k}{d}}$, and*

ϵ_k sufficiently small, the information complexity of (one-way) ϵ_k-suboptimal protocols for Singletons(k, q_{HC}) is $I(X'; M) \geq \frac{1-2h(\epsilon_k)}{c^2 \ln 2} \cdot kd$, where h is the binary entropy function.

We prove this using the strong data processing inequality, analogous to its recent use for bounding the communication complexity of the Gap-Hamming problem [23].

The proof of this result is subtle, and does not proceed by separately bounding the information complexity of solving each of the k subproblems implicit in Singletons(k, q_{HC}). The parameter ρ is high enough that one can reliably detect proximity to any given one of Alice's inputs with a message of size $\frac{d}{\log k} = o(d)$. Our proof crucially uses the fact that Bob must select from among k possibilities. It shows that his optimal strategy is to detect proximity to each of Alice's inputs with failure probability $\approx 1/k$, with his total failure probability controlled by a union bound.

We conjecture in Section 4.2 that the constant factor of $\frac{1}{2 \ln 2}$ in the above lemma can be improved to 1, matching exactly the upper bound from the naive algorithm. In a related "one-shot" case of the Gap-Hamming problem, where the data points are independent and uniform, we are able to prove $I(X; M) \geq (1 - o(1))H(X)$. This result, Theorem G.1 in Appendix G, suggests that no more sophisticated algorithm exists. The information complexity of Gap-Hamming is known to be $\Omega(n)$ for both one- and two-way communication, but we are not aware of a result showing a leading constant of 1.

Next-bit Prediction and One-shot Learning Inspired by the empirical results of [12], we demonstrate a sequence prediction task which requires memorization. Each subpopulation j is associated with a fixed "reference string," and samples from the subpopulation are noisy prefixes of this string.

Example 1.3 (Next-Symbol Prediction)**.** In the Next-Symbol Prediction task the component distribution q_{NSP} draws a reference string $c_j \in \{0, 1\}^d$ uniformly at random. Samples from j are generated by randomly picking a length $\ell \in \{0, \ldots, d-1\}$, then generating $z \sim BSC_\delta(c_j(i : \ell))$ for some small noise parameter δ. The label is a noisy version of the next bit: $y \sim BSC_\delta(c_j(\ell + 1))$. ■

Unlike cluster problems, where the label is the subpopulation, each subpopulation can be treated independently. The core of our lower bound for this task, then, is to prove a "one-shot" lower bound on the setting where the learner receives exactly one sample.

Lemma 1.4 (informal; see Lemma 3.7)**.** *Any algorithm that is ϵ-suboptimal on (noiseless) Singletons$(1, q_{NSP})$ satisfies*

$$I(X; M) \geq H(X) \left(1 - h\left(2\epsilon\right)\right) - \log d. \tag{2}$$

The $\log d$ term arises from uncertainty about the length of the input, which need not be conveyed by the model. The proof proceeds by proving that Bob's correctness is determined by his ability to output Alice's relevant bit. For any fixed length of Alice's input, the problem is similar to a communication complexity problem called Augmented Indexing. We adapt the outline of a proof appearing in [4, 20].

1.3 Related Concepts

Our results are closely related to a number of other lines of work in machine learning. First, as mentioned earlier, one can view our results as a significant strengthening of recent results on label memorization [19], showing the memorization of nearly entire training examples. At a technical level, our proofs are significantly more involved (see "Proof Techniques" below).

Representation Complexity Another closely related concept is *probabilistic representation complexity* [7, 20]. For given error parameter ϵ, the representation complexity $\mathsf{PRep}_\epsilon(C)$ of a class C of concepts (functions from $\mathcal{F}$ to $\mathcal{Y}$) is roughly the smallest number of bits needed to represent a hypothesis that approximates (up to expected error ϵ) a concept $c \in C$ on an example distribution $P_f \in \Delta(\mathcal{F})$, in the worst case over pairs (c, P_f).[2] This complexity measure characterizes the sample complexity of "pure" differentially private learners for C [7].

Interpreted in our setting, representation complexity aims to understand the *length* of the message M, when the task q is a distribution over pairs (c, P_f) (that is, where the data distribution P consists of examples drawn from P_f and labeled with c). By a minimax argument, one can show that $\mathsf{PRep}_\epsilon(C)$ lower bounds not only M's length, but also the information it contains about P: one can find q such that $I(P; M)$ is at least $\mathsf{PRep}_\epsilon(C)$. This does imply that $I(X; M)$ must be large, but it says nothing about the information in M that is specific to a particular sample X: in fact, the bound is saturated by learners that get enough data to construct a hypothesis that is just a function of P, so that $I(X; M|P)$ is small.

The bounds we prove here are qualitatively stronger. We give settings where the analogue of representation complexity is small (namely, a learner that knows P can construct a model of size about $n \log(n/\epsilon) \log d$), but where a learner which only gets a training sample must write down a very large model ($\Omega(nd)$ bits) to do well.

Time-Space Tradeoffs for Learning Another related line of work (beginning with [37, 33]; see [22] for a more recent summary of results) establishes time-space tradeoffs for learning: problems where any learning algorithm requires either a large memory or a large number of samples. A prime example is parity learning over d bits, which is shown to require either $\Omega(d^2)$ bits of memory or exponentially many samples. The straightforward algorithm for parity learning requires $O(d)$ samples, so this result shows that any feasible algorithm must store, up to constant factors, as many bits as are required to store the dataset [33].

Our work sets a specific number of samples under which learning is feasible and, for that number of samples, establishes an information lower bound on the *output* of the algorithm. This implies not only a communication lower bound but also one on memory usage: the algorithm must store the model immediately prior to releasing it. Some of our tasks exhibit the property that, with additional data, an algorithm can output a substantially smaller model. These learning tasks might exhibit a time-space tradeoff, although not as dramatic as the requirement of exponentially many samples. Intuitiviely, the underlying concept in parity learning must be learned "all at once," since the learner's belief about the bits in the second half of the string depend heavily on its belief about the bits in the first half, for example. Our

[2] See [20] for an exact definition.

problem instances can be learned "piece-by-piece," where the algorithm can learn chunks of bits independently of the rest of the sample.

Information Bottlenecks Our work fits into the broad category of *information bottleneck* results in information theory [39]. An information bottleneck is a compression scheme for extracting from a random variable V all the information relevant for the estimation of another random variable W while discarding all irrelevant information in V. In our setting, one may take $V = X$ to be the data set, and W to be the true distribution P (where the loss of a model is its misclassification error). This form of information bottleneck was recently described in general terms [1]. Our results lower bound the extent to which nontrivial compression is possible, showing that the Markov chain $P - X - M$ must in particular satisfy $I(X; M) \gg I(M; P)$.

Information bottlenecks have been put forward as a theory of how implicit feature representations evolve during training [38]. That line of work studies how the prediction process transforms information from a test datum during prediction (i.e. as one moves through layers of a neural network), and is thus distinct from our study of how learning algorithms are able to extract information from training data sets.

1.4 Organization of this paper

In Section 2 we specify our general framework for learning tasks, detailing how mixture coefficients and subpopulations are sampled. We also introduce and prove our "central reduction," showing how an algorithm for a learning task provides an algorithm for solving the singletons-only task. In Sections 3 and 4 we formally define Next-Symbol Prediction and Hypercube Cluster Labeling, the latter of which is a modified version of the binary cluster labeling problem discussed in Example 1.1, and provide lower bounds on $I(X; M \mid P)$ for both tasks.

2 Subpopulations, Singletons, and Long Tails

Before analyzing the specific learning tasks, we present the key points of the framework upon which our tasks our built. We also outline the high-level structure of our task-specific bounds.

Recall that we define a problem instance P, which is a random variable drawn from a metadistribution q, to be a distribution over labeled data. In our work P will be a mixture over N *subpopulations*, each of which may have its own distribution, label, or classification rules. We decompose $P = (D, C)$, where D is a list of N mixture coefficients and C is a list of N distributions over labeled examples, one for each subpopulation. To sample a data point from a problem instance p, we first sample a subpopulation $j \sim D$ and then sample the labeled point $(z, y) \sim C_j$.

The metadistribution q is specified by two generative processes, one for generating D and the other for generating C. (Formally, we will take C to be parameters, not distributions, but ignore the distinction for now.) The first process, described below in Section 2.1, depends only on N and a list of frequencies π, which we refer to as a "prior." The details of the second process will be task-specific, but for each task there will be a "component distribution" $q_c \in \Delta(\mathcal{X})$ from which the entries in C are sampled i.i.d.

The learning task is thus completely determined by the sample size n, the number of subpopulations N, the (task-specific) component distribution q_c, and the prior π. We give this standard setting a name.

Definition 2.1. We call our standard learning task Learn(n, N, q_c, π). Problem instance $P = (D, C)$ is generated from q. A data set of n i.i.d. samples are drawn from p and given to the learning algorithm. One test sample (z, y) is drawn independently from p, and the model predicts a label. ■

When the other terms are clear from context, we will shorten this to Learn(q_c), since only the component distribution will change from task to task.

Our results rely on the analyzing how algorithms perform on subpopulations for which they receive exactly one data point. We call these points *singletons*. To clarify the analysis, we define a second type of task. This is also a learning task, but, unlike in Learn(n, N, q_c, π), the samples are no longer i.i.d.

Definition 2.2. We denote by Singletons(k, q_c) the singletons-only task on k subpopulations. In this task k subpopulation parameters C_j are sampled i.i.d. from q_c, and from each C_j is sampled exactly one labeled data point. These k samples form the data set given to the learner. There is an index $j^* \in [k]$ sampled uniformly at random; the test sample is drawn from C_{j^*}. ■

We return to Singletons(k, q_c), and its relationship to Learn(n, N, q_c, π), in Section 2.2.

2.1 Generating Mixtures over Subpopulations

We generate a mixture over N subpopulations using the process introduced in [19]. Although our central results will hold in the setting where the mixture is uniform (and thus chosen without randomness), this process sets up a *qualitatively different type* of "memorization of useless information," an example of which is crystallized in Example 2.2 and occurs naturally in long-tailed distributions. We encourage the reader to keep the uniform case in mind for simplicity but remember that the results apply to broad settings exhibiting varied behavior.

Starting with a list π of nonnegative values, for each subpopulation $j \in [N]$, we sample $\delta_j \sim \text{Uniform}(\pi)$. To create distribution D, we normalize:

$$D(j) = \frac{\delta_j}{\sum_{i \in [N]} \delta_i}.$$

This quasi-independent sampling facilitates certain computations. In particular, we will want to quantify the following: given that subpopulation j has exactly one representative in data set X, what is the probability the test sample (z, y) comes from the same subpopulation? The answer can be computed as a function of n, N, and π, independently of q_c and, crucially, the rest of the data set. We call this quantity

$$\tau_1 \stackrel{\text{def}}{=} \Pr[(z, y) \text{ comes from } j \mid X_S \text{ contains one sample from } j], \tag{3}$$

defining X_S to be the data set restricted to those singletons. By linearity of expectation we have $\Pr[(z, y) \text{ comes from a singleton subpopulation}] = \tau_1 \times |X_S|$.

We will also need to refer to the expected size of $|X_S|$. Like τ_1, this is a function only of n, N, and π. We have, defining the quantity as a fraction of the data set,

$$\mu_1 \stackrel{\text{def}}{=} \frac{\mathbb{E}[|X_S|]}{n}. \tag{4}$$

Our memorization results are most striking when $\tau_1 = \Omega(1/n)$ and $\mu_1 = \Omega(1)$. We provide two simple examples of when this is so.

Example 2.1 (Uniform)**.** If the list of frequencies is a single entry $\pi = (1/N)$, then the mixture over subpopulations will be uniform. Set $n = N$, so the number of samples is equal to the number of bins. For every subpopulation the probability a test sample comes from it is exactly $\frac{1}{n}$, so $\tau_1 = \frac{1}{n}$ as well. The expected fraction of singletons is also constant: we have by linearity of expectation that

$$\mu_1 = n \cdot \frac{\Pr[\text{subpopulation 1 receives 1 sample}]}{n} = \left(1 - \frac{1}{n}\right)^{n-1} \approx \frac{1}{e}.$$

■

Our central results apply cleanly to the uniform setting. In this setting, however, every example is "important," i.e. memorizing it will provide a significant gain in accuracy. In general mixtures this is not the case, and as the following example shows.

Example 2.2 (Bimodal)**.** We sketch the main points and include a full description in Appendix A. Suppose there are $N = 2^n$ subpopulations, and the prior is such that

$$\delta_j \sim \text{Uniform}(\pi) = \begin{cases} \frac{1}{2n} & \text{w.p. } n2^{-n} \\ \frac{1}{2 \cdot 2^n} & \text{w.p. } 1 - n2^{-n}. \end{cases}. \tag{5}$$

The exact probabilities will depend on the normalization constant $C = \sum_j \delta_j$ that, as a sum of independent random variables, will exhibit tight concentration about its mean. Since $\mathbb{E}[C] \approx 1$, the mixture coefficients won't change too much after normalization.

Call the bins with mass around $\frac{1}{2n}$ "heavy" and the others "light." Observe that about half the probability mass will lie in each group. Almost all the balls that go into light bins will be singletons; a constant fraction of the balls that go into heavy bins will be singletons. Thus $\mu_1 = \Omega(1)$ and, given that a subpopulation has a single representative, with constant probability it will be a heavy bin with mass $\Omega(1/n)$, so we have $\tau_1 = \Omega(1/n)$. But the light subpopulations which received points are unlikely to receive the test sample, and could be ignored with only an exponentially small increase in expected error, if only they were identified as light. ■

The bimodal example is stylized to show an extreme version of the phenomenon, but the concept of "useless" singletons arises in natural distributions. Informally, these "long-tailed distributions" have a significant portion of the distribution represented by many rare instances. A central motivation for [19], these distributions arise in practice and suggest that success on large-scale learning tasks may depend heavily on how the algorithm deals with these atypical instances. Like with the bimodal distribution, the learning algorithm will be unable to distinguish between examples that are very rare (and can be ignored) and those that represent an $\Omega(1/n)$ probability mass, which must be dealt with to perform near optimally on the whole task.

2.2 Central Reduction: Singletons Task to General Learning

We previously defined Learn(q_c) and Singletons(k, q_c), two distinct tasks. The former is the focus of our interest but the latter proves more amenable to analysis. Informally, an algorithm solving Learn(q_c) to near-optimal error will have to perform well on X_S. Let us quantify "near optimal error," which applies to both Learn(n, N, q_c, π) and Singletons(k, q_c).

Definition 2.3 (ϵ-suboptimality)**.** An algorithm A is *ϵ-suboptimal on task T*, where T is associated with metadistribution q, if

$$\mathsf{err}_{q,n}(A) \leq \inf_{A'} \mathsf{err}_{q,n}(A') + \epsilon.$$

In our shorthand of abbreviating the event "the model M output by A makes an error" as "A errs," this is

$$\Pr[A \text{ errs on } T] \leq \inf_{A'} \Pr[A' \text{ errs on } T] + \epsilon.$$

■

Let random variable $K = |X_S|$ be the number of singletons in the data set. An algorithm performing well on X_S in Learn(q_c) when $K = k$ can be modified to create an algorithm performing well on Singletons(k, q_c). This allows us to apply lower bounds proved for the singletons problem.

Lemma 2.1 (Central Reduction, Task-Agnostic)**.** *Suppose we have the following lower bound for every k: any algorithm $A^k(X')$ that is ϵ_k-suboptimal for* Singletons(k, q_c) *satisfies*

$$I(X'; A^k(X')) \geq f_k(\epsilon_k).$$

Then for any algorithm $A(X)$ that is ϵ-suboptimal on Learn(q_c) *there exists a sequence $\{\epsilon_k\}_{k=1}^n$ such that $\mathbb{E}_k[k\epsilon_k] \leq \frac{\epsilon+\phi_1(q_c)+\phi_2(q_c)}{\tau_1}$ and $I(X_S; M \mid K) \geq \mathbb{E}_k[f_k(\epsilon_k)]$.*

Furthermore, if $f_k(\epsilon_k) \geq k \cdot g(\epsilon_k)$ for some convex and nonincreasing $g(\cdot)$, then

$$I(X_S; M \mid K) \geq \mu_1 n \cdot g\left(\frac{1}{\mu_1 n} \cdot \frac{\epsilon + \phi_1(q_c) + \phi_2(q_c)}{\tau_1}\right).$$

Here $\phi_1(q_c)$ and $\phi_2(q_c)$ are task-specific terms. Letting E_1 be the event that the test sample comes from a subpopulation with exactly one representative, they are defined as

$$\phi_1(q_c) \stackrel{def}{=} \Pr[\bar{E}_1]\left(\Pr[A_{OPT} \text{ errs on } \text{Learn}(q_c) \mid \bar{E}_1] - \Pr[A \text{ errs on } \text{Learn}(q_c) \mid \bar{E}_1]\right),$$

$$\phi_2(q_c) \stackrel{def}{=} \sum_{k=1}^{N} \Pr[K = k \mid E_1]\Big(\Pr[A_{OPT} \text{ errs on } \text{Learn}(q_c) \mid E_1, K = k] - \inf_{A'} \Pr[A' \text{ errs on } \text{Singletons}(k, q_c)]\Big).$$

Proof. We first show how to use A to construct, for each k, a learning algorithm A^k solving Singletons(k, q_c). Given a data set X' of size k, A^k samples a data set $\tilde{X}$ of n entries, where $\tilde{X} \sim X \mid X_S = X'$, and X is a data set sampled from the process generating data sets for

Learn(n, N, q_c, π). A^k can perform this sampling, since $\tilde{X} \backslash X'$ will contain no samples from the same subpopulations as those that generated X' and the per-subpopulation distributions are sampled i.i.d. from q_c. A^k then runs A on $\tilde{X}$ and outputs model $\tilde{M} = A(\tilde{X})$. We have

$$\Pr[A^k \text{ errs on Singletons}(k, q_c)] = \Pr[A \text{ errs on Learn}(q_c) \mid E_1, K = k]. \tag{6}$$

Observe that, for all k, we have equality across the conditional distributions:

$$\Pr[X = x, M = m \mid K = k] = \Pr[\tilde{X} = x, \tilde{M} = m \mid K = k] \tag{7}$$

$$\Pr[X_S = x_S, M = m \mid K = k] = \Pr[X' = x_S, \tilde{M} = m \mid K = k]. \tag{8}$$

This implies that

$$I(X_S; M \mid K) = \mathop{\mathbb{E}}_{k}\left[I(X'; \tilde{M} \mid K = k)\right]. \tag{9}$$

Since A^k solves Singletons(k, q_c), if we can bound the error of A^k using Equation (6) we will be able to apply our lower bound $f_k(\cdot)$.

To that end, we first bound the error of A conditioned on the test sample coming from a subpopulation with a single representative in the data. We have

$$\begin{aligned}
\epsilon &= \Pr[A \text{ errs on Learn}(q_c)] - \Pr[A_{OPT} \text{ errs on Learn}(q_c)] \\
&= \Pr[\bar{E_1}]\big(\Pr[A \text{ errs on Learn}(q_c) \mid \bar{E_1}] - \Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid \bar{E_1}]\big) \\
&\quad + \Pr[E_1]\left(\Pr[A \text{ errs on Learn}(q_c) \mid E_1] - \Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid E_1]\right).
\end{aligned}$$

Rearranging and substituting in we get

$$\Pr[A \text{ errs on Learn}(q_c) \mid E_1] - \Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid E_1] \tag{10}$$

$$\leq \frac{1}{\Pr[E_1]}\Big(\epsilon + \Pr[\bar{E_1}]\big(\Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid \bar{E_1}] \tag{11}$$

$$- \Pr[A \text{ errs on Learn}(q_c) \mid \bar{E_1}]\big)\Big) \tag{12}$$

$$= \frac{\epsilon + \phi_1(q_c)}{\tau_1 \mu_1 n}. \tag{13}$$

Note that in general $\phi_1(q_c)$ may be positive.[3]

We now decompose the error over k.

$$\Pr[A \text{ errs on Learn}(q_c) \mid E_1] - \Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid E_1] \tag{14}$$

$$= \sum_{k=1}^{N} \Pr[K = k \mid E_1]\Big(\Pr[A \text{ errs on Learn}(q_c) \mid E_1, K = k]$$

$$- \Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid E_1, K = k]\Big) \tag{15}$$

$$= \sum_{k=1}^{N} \Pr[K = k \mid E_1]\Big(\Pr[A^k \text{ errs on Singletons}(k, q_c)] \tag{16}$$

$$- \Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid E_1, K = k]\Big) \tag{17}$$

[3]To see this, consider an algorithm that "assumes" $\bar{E_1}$. While such a behavior could introduce high error overall, it might beat optimal when that assumption is justified.

plugging in Equation (6). We wish to compare to the optimal error for Singletons(k, q_c), so we add and subtract a term with A^k_{OPT} denoting the optimal algorithm for Singletons(k, q_c). We have

$$\begin{aligned}
&\Pr[A \text{ errs on Learn}(q_c) \mid E_1] - \Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid E_1] \\
&= \sum_{k=1}^{N} \Pr[K = k \mid E_1] \Big(\Pr[A^k \text{ errs on Singletons}(k, q_c)] && (18) \\
&\qquad - \Pr[A^k_{OPT} \text{ errs on Singletons}(k, q_c)] && (19) \\
&\qquad + \Pr[A^k_{OPT} \text{ errs on Singletons}(k, q_c)] && (20) \\
&\qquad - \Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid E_1, K = k] \Big) && (21) \\
&= \sum_{k=1}^{N} \Pr[K = k \mid E_1] \Big(\epsilon_k + \Pr[A^k_{OPT} \text{ errs on Singletons}(k, q_c)] && (22) \\
&\qquad - \Pr[A_{OPT} \text{ errs on Learn}(q_c) \mid E_1, K = k] \Big) && (23) \\
&= \sum_{k=1}^{N} \Pr[K = k \mid E_1] \epsilon_k - \phi_2(q_c), && (24)
\end{aligned}$$

defining ϵ_k as the suboptimality of A^k on Singletons(k, q_c).

To apply convexity for $f_k(\epsilon_k) = kd \cdot g(\epsilon_k)$,, we combine with Equation (13) to get

$$\sum_{k=1}^{N} \Pr[K = k \mid E_1] \epsilon_k \le \frac{\epsilon + \phi_1(q_c)}{\tau_1 \mu_1 n} + \phi_2(q_c) \le \frac{\epsilon + \phi_1(q_c) + \phi_2(q_c)}{\tau_1 \mu_1 n}, \tag{25}$$

since $1 \ge \tau_1 \mu_1 n$. By Bayes rule, we have that

$$\begin{aligned}
\Pr[K = k \mid E_1] = \frac{\Pr[E_1 \mid K = k] \Pr[K = k]}{\Pr[E_1]} &= \frac{\tau_1 k \Pr[K = k]}{\tau_1 \mu_1 n} && (26) \\
&= \frac{k \Pr[K = k]}{\mu_1 n}, && (27)
\end{aligned}$$

so

$$\frac{\mathbb{E}[k\epsilon_k]}{\mu_1 n} \le \frac{\epsilon + \phi_1(q_c) + \phi_2(q_c)}{\tau_1 \mu_1 n}. \tag{28}$$

Use a convex version of the modification of Jensen's inequality in Lemma C.1, we have

$$\begin{aligned}
I(X; M) \ge \mathop{\mathbb{E}}_{k}[kd \cdot g(\epsilon_k)] &\ge \mathop{\mathbb{E}}_{k}[k] \cdot g\left(\frac{\mathbb{E}_k[k\epsilon_k]}{\mathbb{E}_k[k]} \right) && (29) \\
&= \mu_1 n \cdot g\left(\frac{\mathbb{E}_k[k\epsilon_k]}{\mu_1 n} \right) && (30) \\
&\ge \mu_1 n \cdot g\left(\frac{\epsilon + \phi_1(q_c) + \phi_2(q_c)}{\tau_1 \mu_1 n} \right). && (31)
\end{aligned}$$

□

2.3 Blueprint for Task-Specific Lower Bounds

For each task, we ultimately wish to lower bound $I(X; M \mid P)$. By the chain rule and nonnegativity of mutual information, it suffices to lower bound the mutual information with the singletons:

$$\begin{aligned} I(X; M \mid P) &= I(X_S, X \backslash X_S, K; M \mid P) \\ &= I(K; M \mid P) + I(X_S; M \mid K, P) + I(X \backslash X_S; M \mid K, X_S, P) \\ &\geq I(X_S; M \mid K, P). \end{aligned} \tag{32}$$

We remove P using the fact that conditioning never increases entropy, then add and substract a term to reach a cleaner form:

$$I(X_S; M \mid K, P) = H(X_S \mid K, P) - H(X_S \mid M, K, P) \tag{33}$$
$$\geq H(X_S \mid K, P) - H(X_S \mid M, K) \tag{34}$$
$$= H(X_S \mid K, P) - H(X_S \mid M, K) \tag{35}$$
$$+ I(X_S; M \mid K) - \big(H(X_S \mid K) - H(X_S \mid M, K)\big) \tag{36}$$
$$= I(X_S; M \mid K) - I(X_S; P \mid K). \tag{37}$$

To lower bound $I(X; M \mid P)$, then, we will lower bound $I(X_S; M \mid K)$ and upper bound $I(X_S; P \mid K)$.

The latter is easily done for our tasks, since the distributions are straightforward. Lower bounding $I(X_S; M \mid K)$ requires more effort. With our central reduction in Lemma 2.1 we have already proved a crucial step in that direction. To apply that lemma to a specific task, we need to calculate a number of quantities:

1. Upper bounds on the error of optimal algorithms and lower bounds on the error of any algorithm.
2. With those quantities in hand, we can show $\phi_1(q_c)$ and $\phi_2(q_c)$ are small.
3. Finally, at the core of the task-specific proofs, we need a lower bound on mutual information for algorithms solving Singletons(k, q_c) for any k.

Plugging these pieces into Lemma 2.1 will then finish the proof.

3 Next-Symbol Prediction

We present a simple sequence prediction task. Among other applications, sequence prediction is a standard problem in natural language processing [24].

3.1 Task Description and Main Result

In this task, the data samples are binary string of varyings with binary labels. With each sample we also associate a subpopulation identifier, so $\mathcal{X} \times \mathcal{Y} = \left([N] \times \bigcup_{\ell=0}^{d-1} \{0,1\}^\ell\right) \times \{0,1\}$. To instantiate the task in our framework we need only define the component distribution, from which the subpopulation distributions are drawn i.i.d.

Definition 3.1 (q_{NSP} Component Distribution)**.** For each subpopulation j, we define its distribution over labeled examples via a reference string $c_j \in \{0,1\}^d$. We draw $c_j \sim q_{NSP} = \text{Uniform}\left(\{0,1\}^d\right)$.

Samples from j are tuples: the subpopulation identifier j and noisy prefixes of c_j. To generate the prefix, we draw $\ell \in \{0,\ldots,d-1\}$ uniformly at random and produce $z = BSC_{\delta/2}(c_j(1:\ell))$, for a noise parameter $\delta < 1$, fixed and known to all parties. It will be convenient to think of the binary symmetric channel as rerandomizing each bit independently with probability δ. The label is a noisy version of the next bit: $y = \text{BSC}_{\delta/2}(c_j(\ell+1))$. ■

The technical detail of pairing each sample from subpopulation j with an identifier "j" both simplifies the analysis and has an analog practice of sequence prediction. In [9], the authors greatly improve the performance of the GPT-3 with access to a few examples of the test task. These examples can be regarded as a channel to inform the model which subpopulation the test instance arises from.

We state our main theorem for Next-Symbol Prediction along with an immediate corollary highlighting the key features of the asymptotic behavior.

Theorem 3.1. *Consider the problem* $\text{Learn}(n, N, q_{NSP}, \pi)$ *specified by* d, δ, π, N *and* n*.*

1. *Let* $S \subseteq [n]$ *be the indices of singleton data points.* $H(X_S|P) \geq \mu_1 \cdot n \cdot \frac{d+1}{2} \cdot h(\delta/2)$
2. *Any algorithm* A *that is* ϵ*-suboptimal on* $\text{Learn}(n, N, q_{NSP}, \pi)$ *satisfies*

$$I(X; M \mid P) \geq \mu_1 \cdot n \cdot \frac{d+1}{2}\left(h(\delta/2) - h\left(\frac{2\epsilon}{\tau_1 \cdot \mu_1 \cdot n \cdot (1-\delta)^2}\right)\right) - n \log N.$$

where μ_1, τ_1 *depend on* π *and* n *as defined in Equations* (3) *and* (4)*.*

Corollary 3.2. *Consider the setup of Theorem 3.1. Let* d, δ, π, N *be functions of* n*, and let* n *grow to* ∞*. Suppose* π *satisfies* $\mu_1 = \Omega(1)$ *and* $\tau_1 = \Omega(1/n)$*,* $\delta \in (0,1)$ *is constant, and* $d = \omega(\log N)$*. Then*

1. $H(X_S|P) = \Omega(nd)$
2. *Any algorithm* A *that is* ϵ*-suboptimal on* $\text{Learn}(n, N, q_{NSP}, \pi)$ *for* $\epsilon = o(1)$ *satisfies*

$$I(X; M \mid P) \geq (1 - o(1)) \cdot H(X_S \mid P). \tag{38}$$

Recall that this means A is forced to *memorize whole samples*; as the error of the algorithm approaches optimal, A must reveal almost all of the information about its singletons.

Remark 1. Unlike tasks based on clustering, one can analyze Next-Symbol Prediction by dealing with every subpopulation independently. This allows a similar but more-direct proof than the one we present here. We use the same proof structure across tasks for consistency of presentation.

In the rest of this section, we prove the main claims via the steps sketched in Section 2.3: we show that the data contains limited information about the problem instance P, analyze the optimal error, and provide the central lower bound on the $\text{Singletons}(k, q_{NSP})$ task.

3.2 Low Information about the Problem Instance

Lemma 3.3. *For* Learn(q_{NSP}),

$$I(X_S; P \mid K) \leq \mu_1 n \cdot \frac{d+1}{2} \cdot (1 - h(\delta/2)) + \mu_1 n \log N.$$

Proof. We can think of X_S as being generated by first selecting $|X_S| = k$ and then picking the subpopulations from which the singletons come. Write the subpopulation identifiers $\vec{J} \in [N]^k$, which needs at most $k \log N$ bits to describe. We have

$$\begin{aligned} I(X_S; P \mid K) &= I(X_S, \vec{J}; P \mid K) \\ &= I(X_S; P \mid \vec{J}, K) + I(\vec{J}; P \mid K) \\ &\leq \mathop{\mathbb{E}}_k[I(X_S; P \mid \vec{J} = \vec{j}, K = k)] + \mathop{\mathbb{E}}_k[k \log N], \end{aligned}$$

using the fact that $I(X_S; P \mid \vec{J} = \vec{j}, K = k)$ does not depend on $\vec{j}$.

With $\vec{j}$ and k fixed, all subpopulations are independent, so let X_1 be a singleton and write $I(X_S; P \mid \vec{J} = \vec{j}, K = k) = k \cdot I(X_1; C_1)$, using $C_1 \sim q_{NSP}$ to denote the the reference string. Note that L_1, the length of X_1, is fixed by X_1 and independent of C_1, so

$$I(X_1; C_1) = I(X_1, L_1; C_1) = I(L_1; C_1) + I(X_1; C_1 \mid L_1) = I(X_1; C_1 \mid L_1).$$

For each fixed $L_1 = \ell$, X_1 (with its label) is just $\ell + 1$ bits of C_1 run through a binary symmetric channel,

$$I(X_1; C_1 \mid, L_1 = \ell) = (\ell + 1) \cdot (1 - h(\delta/2)).$$

Putting together all the expectations and recalling that, by definition, $\mathbb{E}[k] = \mu_1 n$ finishes the proof. □

3.3 Error Analysis

Proposition 3.4 (Accuracy of Optimal Algorithm)**.** *Let E_0, E_1, and $E_{>1}$, respectively be the events that the test sample comes from a subpopulation with zero, one, or multiple representatives in the data set. Learning algorithm A_{OPT} for* Learn(n, N, q_{NSP}, π) *achieves*

1. $\Pr[A_{OPT} \text{ errs} \mid E_1] = \frac{1}{2} - \frac{(1-\delta)^2}{4} \leq \frac{1}{4} + \frac{\delta}{2}$.
2. $\Pr[A_{OPT} \text{ errs} \mid E_{>1}] \leq \Pr[A_{OPT} \text{ errs} \mid E_1]$.
3. $\Pr[A_{OPT} \text{ errs} \mid E_0] \leq \frac{1}{2}$.

Proof. Each subpopulation can be dealt with independently. We will analyze the error conditioned on E_1; the error of the optimal algorithm conditioned on $E_{>1}$ can be no greater (since the algorithm can ignore samples), establishing (2), and we have $\Pr[\text{error} \mid E_0] = \frac{1}{2}$ for all algorithms, establishing (3).

Condition on E_1, so Alice has one relevant string. Let ℓ_A denote the length of Alice's string and ℓ_B denote the length of Bob's string; Bob wants to output c_{ℓ_B+1}. If $\ell_A \geq \ell_B$, he should output the $\ell_B + 1$-th bit of her input, and otherwise answer randomly. Define a

"good event" G that occurs when Alice's input is longer than Bob's and neither her nor his ℓ_B+1-th bit was rerandomized. (Bob doesn't recieve his ℓ_B+1-th bit; it's the label his output is compared to.) We have

$$\Pr[G] = \left(\frac{1}{2} + \frac{1}{2d}\right)(1-\delta)(1-\delta) \geq \frac{1}{2}\cdot(1-\delta)^2.$$

Conditioned on G this algorithm has error 0, and conditioned on $\bar{G}$ *any* algorithm has accuracy $\frac{1}{2}$. So we have $\Pr[A_{OPT} \text{ errs} \mid E_1] = \frac{1}{2}(1-\Pr[G]) \leq \frac{1}{2} - \frac{(1-\delta)^2}{4}$. Furthermore, this fact implies no algorithm can do better when E_1 occurs. □

Proposition 3.5. *$\phi_1(q_{NSP}) \leq 0$ and $\phi_2(q_{NSP}) = 0$.*

Proof. We use the fact that the optimal strategy treats subpopulations independently, and thus is optimal for all k and no matter which subpopulation the test sample comes from. Therefore

$$\begin{aligned}\phi_1(q_{NSP}) = \Pr[\bar{E_1}]\Big(&\Pr[A_{OPT} \text{ errs on Learn}(q_{NSP}) \mid \bar{E_1}] \\ &- \Pr[A \text{ errs on Learn}(q_{NSP}) \mid \bar{E_1}]\Big) \\ \leq 1 \times 0.&\end{aligned}$$

For every k, the probability that A_{OPT} errs on Learn(q_{NSP}) conditioned on E_1 and $K=k$ is exactly the optimal error on Singletons(k, q_{NSP}), so

$$\begin{aligned}\phi_2(q_{NSP}) = \sum_{k=1}^{N} \Pr[K=k \mid E_1]\Big(&\Pr[A_{OPT} \text{ errs on Learn}(q_{NSP}) \mid E_1, K=k] \\ &- \inf_{A'} \Pr[A' \text{ errs on Singletons}(k, q_{NSP})]\Big) \\ = \sum_{k=1}^{N} \Pr[K=k \mid E_1] \times 0.&\end{aligned}$$

□

3.4 Lower Bound for Singletons Task

Lemma 3.6. *Any algorithm A that is ϵ_k-suboptimal on* Singletons(k, q_{NSP}) *satifies*

$$I(X'; M) \geq k \cdot \frac{d+1}{2}\left(1 - h\left(\frac{2\epsilon_k}{(1-\delta)^2}\right)\right).$$

Before proving this lemma, we prove a mutual information lower bound for algorithms solving just one instance of Next-Symbol Prediction. The proof extends one appearing in [4, 20] for a communication complexity task called Augmented Index.

Lemma 3.7 (One-Shot Lower Bound)**.** *Any algorithm A that is ϵ_1-suboptimal on "one-shot"* Singletons$(1, q_{NSP})$ *satisfies*

$$I(X; M) \geq \frac{d+1}{2}\left(1 - h\left(\frac{2\epsilon_1}{(1-\delta)^2}\right)\right).$$

Proof of Lemma 3.7. We first change the ϵ_1 assumption into a multiplicative error term. Recall from the proof of Proposition 3.4 that, when the learner receives one sample from a subpopulation, there is a "good event" G such that, conditioned on G, the optimal algorithm is correct and, conditioned on $\bar{G}$, all algorithms have accuracy $\frac{1}{2}$. Let $\gamma = \Pr[A \text{ errs} \mid G]$. So we can write

$$\begin{aligned}\Pr[A \text{ errs}] &= \Pr[A \text{ errs} \mid G]\Pr[G] + \Pr[A \text{ errs} \mid \bar{G}](1 - \Pr[G]) \\ &= \Pr[A \text{ errs} \mid G]\Pr[G] + \frac{1}{2}(1 - \Pr[G]) \\ &= \frac{1}{2} - \frac{\Pr[G]}{2}(1 - 2\gamma),\end{aligned}$$

and $\Pr[A_{OPT} \text{ errs}] = \frac{1}{2} - \frac{\Pr[G]}{2}$. Then we have $\epsilon_1 = \Pr[A \text{ errs}] - \Pr[A_{OPT} \text{ errs}] = \Pr[G] \cdot \gamma$. Since $\Pr[G] \geq \frac{1}{2} \cdot (1-\delta)^2$, we have $\gamma \leq \frac{2\epsilon_1}{(1-\delta)^2}$.

Let X and Z denote Alice and Bob's inputs, respectively, and let L_A and L_B be the lengths of their inputs. Since L_A is fixed given X, we can bound

$$\begin{aligned}H(X \mid M) = H(X, L_A \mid M) &= H(X \mid M, L_A) + H(L_A \mid M) \\ &\leq \mathop{\mathbb{E}}_{\ell_A}[H(X \mid M, L_A = \ell_A)] + \log d.\end{aligned}$$

We will fix ℓ and bound $H(X \mid M, L_A = \ell)$. Define $\Pr[A \text{ errs} \mid G, L_A = \ell] = \gamma_\ell$.

Recall that G means neither Alice nor Bob's L_B+1-th bit are rerandomized. Conditioning on G thus implies "correctness" and "outputting Alice's bit" are the same event. We now show that, if Bob can output Alice's bits, her message must contain a lot of information about her input. Crucially, conditioned on $L_A = \ell$, the good event G is independent of Alice's input X, since $L_B \perp L_A$ and Alice's string is a uniformly random binary string whether or not any bit is flipped. So

$$H(X \mid M, L_A = \ell) = H(X \mid M, G, L_A = \ell).$$

Below, in (39), we apply the chain rule for entropy over Alice's bits, including her label. In (40) we condition on $Z_1^{\ell_B}$; since this is a noisy version of $X_1^{\ell_B}$, which we already condition on, this does not change the entropy. In (41) we remove $X_1^{\ell_B}$, which can only increase the entropy of $X_1^{\ell_B+1}$.

$$H(X \mid M, G, L_A = \ell) = \sum_{\ell_B=0}^{\ell} H(X_{\ell_B+1} \mid X_1^{\ell_B}, M, G, L_A = \ell) \tag{39}$$

$$= \sum_{\ell_B=0}^{\ell} H(X_{\ell_B+1} \mid Z_1^{\ell_B}, X_1^{\ell_B}, M, G, L_A = \ell) \tag{40}$$

$$\leq \sum_{\ell_B=0}^{\ell} H(X_{\ell_B+1} \mid Z_1^{\ell_B}, M, G, L_A = \ell). \tag{41}$$

Now we relate the index ℓ_B to the random variable L_B, the length of Bob's input, and observe

that $\Pr[L_B = \ell_B \mid G, L_A = \ell] = \frac{1}{\ell+1}$, since G requires that $L_A \geq L_B$. So we have

$$\begin{aligned}
&H(X \mid M, L_A = \ell) \\
&\qquad \leq (\ell+1) \sum_{\ell_B=0}^{\ell-1} \Pr[L_B = \ell_B \mid G, L_A = \ell] \cdot H(X_{\ell_B+1} \mid Z_1^{\ell_B}, M, G, L_A = \ell) \\
&\qquad = (\ell+1) \cdot H(X_{L_B+1} \mid Z, M, G, L_A = \ell) \\
&\qquad \leq (\ell+1) \cdot h(\gamma_\ell),
\end{aligned}$$

applying Fano's inequality, since this is exactly Bob's task and, conditioned on G and $L_A = \ell$, he fails to guess the bit X_{L_B+1} with probability at most γ_ℓ.

We use the modification of Jensen's inequality in Lemma C.1 to push the expectation inside the binary entropy function and get

$$H(X \mid M, L_A) \leq \mathbb{E}[\ell+1] \cdot h\left(\frac{\mathbb{E}[(\ell+1)\gamma_\ell]}{\mathbb{E}[\ell+1]}\right) = \frac{d+1}{2} \cdot h\left(\frac{2 \cdot \mathbb{E}[(\ell+1)\gamma_\ell]}{d+1}\right) \tag{42}$$

We can evaluate the remaining expectation by using

$$\begin{aligned}
\gamma \cdot \Pr[G] = \Pr[A \text{ errs} \mid G] \Pr[G] &= \mathop{\mathbb{E}}_{\ell} [\Pr[A \text{ errs} \mid G, L_A = \ell] \Pr[G \mid L_A = \ell]] \\
&= \mathop{\mathbb{E}}_{\ell} [\gamma_\ell \Pr[G \mid L_A = \ell]]
\end{aligned}$$

and

$$\Pr[G] = \frac{d+1}{2d} \cdot (1-\delta)^2, \quad \Pr[G \mid L_A = \ell] = \frac{\ell+1}{d} \cdot (1-\delta)^2.$$

We have

$$\gamma = \frac{\mathbb{E}[\gamma_\ell \frac{\ell+1}{d} \cdot (1-\delta)^2]}{\frac{d+1}{2d} \cdot (1-\delta)^2} = \frac{2 \cdot \mathbb{E}[\gamma_\ell(\ell+1)]}{d+1}.$$

Plugging in $\gamma \leq \frac{2\epsilon_1}{(1-\delta)^2}$ finishes the proof. □

We can now prove the Next-Symbol Prediction lower bound for Singletons(k, q_{NSP}).

Proof of Lemma 3.6. Suppose algorithm A is ϵ_k-suboptimal on Singletons(k, q_{NSP}). Then, for each subpopulation $i \in [k]$, define its error above optimal ϵ_k^i so that $\mathbb{E}[\epsilon_k^i] = \epsilon_k$. By the same synthetic-dataset reduction used to prove Lemma 2.1, A can be turned into k algorithms $\{A_i\}$, each solving Singletons$(1, q_{NSP})$ to suboptimality $\{\epsilon_k^i\}$, and with mutual information

$$\begin{aligned}
I(X_S; A(X_S)) &\geq \sum_{i=1}^{k} I(X_i; A_i(X_i)) \\
&\geq \sum_{i=1}^{k} \frac{d+1}{2} \left(1 - h\left(\frac{2\epsilon_k^i}{(1-\delta)^2}\right)\right) \\
&\geq k \cdot \frac{d+1}{2} \left(1 - h\left(\frac{2\epsilon}{(1-\delta)^2}\right)\right),
\end{aligned}$$

writing the sum as an expectation over uniform $i \in [k]$ and applying Jensen's inequality. □

3.5 Completing the Proof

Proof of Theorem 3.1. By Lemma 3.3, we get the lower bound on $H(X_S \mid P)$ claimed in (1).

For (2) we assume an ϵ-suboptimal algorithm A for Learn(q_{NSP}). Our lower bound on Singletons(k, q_{NSP}) from Lemma 3.6 gives us a mutual information lower bound of

$$f_k(\epsilon_k) = k \cdot g(\epsilon_k) = k \cdot \frac{d+1}{2}\left(1 - h\left(\frac{2\epsilon}{(1-\delta)^2}\right)\right). \tag{43}$$

The function $1 - h(\cdot)$ is convex and strictly decreasing for arguments less than $\frac{1}{2}$ so we have, upper bounding via Proposition 3.5 both $\phi_1(q_{NSP})$ and $\phi_2(q_{NSP})$ with 0,

$$I(X_S; M \mid K) \geq \mu_1 n \cdot \frac{d+1}{2}\left(1 - h\left(\frac{2\epsilon}{\tau_1\mu_1 n(1-\delta)^2}\right)\right). \tag{44}$$

In Lemma 3.3 we proved $I(X_S; P \mid K) \leq \mu_1 n \cdot \frac{d+1}{2} \cdot (1 - h(\delta/2)) + \mu_1 n \log N$, so using the calculations in Section 2.3 we have

$$\begin{aligned} I(X; M \mid P) &\geq I(X_S; M \mid K) - I(X_S; P \mid K) && (45)\\ &\geq \mu_1 n \cdot \frac{d+1}{2}\left(h(\delta/2) - h\left(\frac{2\epsilon}{\tau_1\mu_1 n(1-\delta)^2}\right)\right) - \mu_1 n \log N. && (46) \end{aligned}$$

□

4 Hypercube Cluster Labeling

4.1 Task Description and Main Result

In this task, the data samples are binary strings labeled with their subpopulation index, so $\mathcal{X} \times \mathcal{Y} = \{0,1\}^d \times [N]$. Unlike Next-Symbol Prediction, this distribution is "noiseless:" there is a small set of relevant features which are deterministically set, and only the irrelevant features are picked at random.

Definition 4.1 (q_{HC} Component Distribution)**.** For each subpopulation j, we define its distribution over labeled examples via a small number of fixed positions. For each cluster $j \in [N]$, q_{HC} generates fixed indices as

- Independently for each index $i \in [d]$, flip a coin that comes up heads w.p. ρ.
- If heads:
  - Add i to $\mathcal{I}_j$, the indices of fixed features.
  - Flip a fair coin to set the value $b_j(i)$, the fixed feature value at that location.

Samples from the subpopulation are uniform over the hypercube defined by the *unfixed* indices: To generate a data point from subpopulation j, we let

$$z(i) = \begin{cases} b_j(i) & \text{if } i \in \mathcal{I}_j \\ \text{Bernoulli}(1/2) & \text{otherwise} \end{cases}.$$

This point's label is j. ■

Observe that, for any index $i \in [d]$ and strings z_1, z_2 from the same cluster we have $\Pr[z_1(i) = z_2(i)] = \frac{1+\rho}{2}$. This is identical to producing z_2 by running z_1 through a binary symmetric channel with parameter $\frac{1-\rho}{2}$, a fact which will be crucial in our proofs.

Theorem 4.1. *Consider the problem* $\text{Learn}(n, N, q_{HC}, \pi)$ *specified by* d, c, π, N, *and* n, *for* $c > \sqrt{2}$. *Set* $\rho = \frac{c\sqrt{\ln n}}{\sqrt{d}}$.

1. *Let* $S \subseteq [n]$ *be the indices of singleton data points.* $H(X_S \mid P) \geq \mu_1(1-\rho)nd$.

2. *Take any* ϵ*-suboptimal algorithm* A *solving* $\text{Learn}(n, N, q_{HC}, \pi)$. *It yields a sequence of algorithms* $\{A^k\}_{k=1}^n$, *each solving* $\text{Singletons}(k, q_{HC})$. *Define* $\{\epsilon_k\}_{k=1}^n$ *so that* A^k *is* ϵ_k*-suboptimal for* $\text{Singletons}(k, q_{HC})$. A *satisfies*

$$I(X; M \mid P) \geq \frac{d}{c^2 \ln 2 \cdot \log n} \cdot \mathop{\mathbb{E}}_{k} [k \log k \cdot (1 - 2h(\epsilon_k))] - n \log N.$$

3. *Suppose that, for some* $\beta > 0$, *with probability at least* $1 - \beta$ *we have* $K \geq \mu_1 n/2$. *Then, for some constant* $\alpha > 0$,

$$\begin{aligned} I(X; M \mid P) \geq \frac{(1-\beta)}{c^2 \ln 2} \cdot \mu_1 nd \left(1 - 2h\left(\frac{\epsilon + 6n^{-\alpha}}{\tau_1 \mu_1 n}\right)\right) \\ - nd\left(\frac{\log 2/\mu_1}{\log n} + \beta + \rho + \frac{\log N}{d}\right). \end{aligned}$$

   where μ_1, τ_1 *depend on* π *and* n *as defined in Equations* (3) *and* (4).

Our mutual information lower bound for $\text{Singletons}(k, q_{HC})$ doesn't quite match the $kd \cdot g(\epsilon_k)$ assumption in Lemma 2.1. If the number of singletons concentrates around its mean of $\mu_1 n$, we can provide the cleaner bound in part (3). Such an assumption is reasonable: for example, in the uniform case with $n = N$ it is the well-studied problem of tossing n balls into n bins, where tight concentration of the number of singletons can be shown via the Poisson approximation [28].

We have the following immediate corollary.

Corollary 4.2. *Consider the setup of Theorem 4.1. Let* d, π, c *and* N *be functions of* n *and let* n *grow to* ∞. *Suppose* π *is such that* $\mu_1 = \Omega(1)$, $\tau_1 = \Omega(1/n)$, *and* $\beta = \Pr[K < \mu_1 n/2] = o(1)$. *Suppose constant* $c > \sqrt{2}$ *and* $d = \omega(\max\{\log N, \log n\})$. *Then*

1. $H(X_S \mid P) = \Omega(nd)$.

2. *Any algorithm* A *that is* ϵ*-suboptimal on* $\text{Learn}(n, N, q_{HC}, \pi)$ *for* $\epsilon = o(1)$ *satisfies*

$$I(X; M \mid P) \geq \frac{1 - o(1)}{c^2 \ln 2} \cdot H(X_S \mid P).$$

Observe that, unlike the statement in Corollary 3.2, our lower bound does not achieve the (best possible) leading factor $1 - o(1)$ in front of $H(X_S \mid P)$. As we discuss in Section 4.2, we conjecture that this lower bound can be improved.

Remark 2. With exactly one sample from a subpopulation, the learning algorithm is forced to memorize. With additional samples, the learner can leak much less information by sending a smaller model. If the learner has $O(\log d)$ samples from subpopulation j, it can learn $(\mathcal{I}_j, b_j)$ exactly with high probability, since unfixed indices have feature values selected independently and uniformly. Given knowledge of $(\mathcal{I}_j, b_j)$, the learning algorithm can send a subset $T_j \subseteq \mathcal{I}_j$ of size $O(\log(n/\epsilon))$ and still achieve high accuracy, since samples from other clusters will match all the features in T_j with probability exponentially small in $|T_j|$.

We now briefly discuss a communication complexity conjecture and how its proof would improve the above results. In the rest of this section, we prove the main claims via the steps sketched in Section 2.3: we show that the data contains limited information about the problem instance P, analyze the optimal error, and provide the central lower bound on the Singletons(k, q_{HC}) task.

4.2 A Stronger Result under a Communication Complexity Conjecture

Our lower bound for this task relies on proving a one-way (external) information complexity lower bound for a task that, on its face, is not clearly related to the learning task at hand.

Definition 4.2 (Nearest of k Neighbors)**.** Alice receives k strings $x_1, \ldots, x_k \in \{0,1\}^d$, drawn i.i.d. from the uniform distribution. Bob receives a string $y \sim BSC_{\frac{1-\rho}{2}}(x_{j^*})$ for some index $j^* \in [k]$, also chosen uniformly at random. Alice and Bob succeed if Bob outputs j^*. ■

Our approach only yields a lower bound of $I(X; M) \geq \alpha \cdot kd$ for a constant $\alpha < 1$, even letting the error ϵ vanish. This result, in Lemma 4.8, does not admit the interpretation of "memorizing whole samples." We conjecture that α can be replaced by $1 - o(1)$. In Appendix G, we prove such a one-way information complexity bound for a similar communication task, the Gap-Hamming problem. The information complexity of Gap-Hamming is known to be $\Omega(d)$, but to the best of our knowledge no previous proofs provided the correct (at least for one-way communication) leading factor.

Conjecture 4.1. *Set $\rho = \frac{c\sqrt{\ln k}}{\sqrt{d}}$ for constant $c > \sqrt{2}$. Any one-way communication protocol for Nearest of k Neighbors with error at most ϵ satisfies*

$$I(X'; M) \geq (1 - f(\epsilon_k) - o(1)) \cdot kd,$$

where $f(\epsilon) \to 0$ as $\epsilon \to 0$.

A proof of this conjecture would immediately demonstrate the same "memorization of whole samples" behavior that we proved inherent in Next-Symbol Prediction.

Proposition 4.3. *Assume Conjecture 4.1 holds. Consider the setup of Theorem 4.1. Let d, π, c and N be functions of n and let n grow to ∞. Suppose π is such that $\mu_1 = \Omega(1)$, $\tau_1 = \Omega(1/n)$, and $\Pr[K \geq \mu_1 n/2] \geq 1 - o(1)$. Suppose constant $c > \sqrt{2}$ and $d = \omega(\max\{\log N, \log n\})$. Then*

- $H(X_S \mid P) = \Omega(nd)$.
- *Any algorithm A that is ϵ-suboptimal on* Learn(n, N, q_{HC}, π) *for $\epsilon = o(1)$ satisfies*

$$I(X; M \mid P) \geq (1 - o(1)) \cdot H(X_S \mid P).$$

4.3 Low Information about the Problem Instance

Lemma 4.4. *Set* $\rho = \frac{c\sqrt{\ln n}}{\sqrt{d}}$ *for constant* $c > \sqrt{2}$. *For* Learn(q_{HC}),

$$I(X_S; P \mid K) \leq \mu_1 n \cdot \rho d + \mu_1 n \log N.$$

The proof is similar to that of Lemma 3.3, the analogous bound for q_{NSP}.

Proof. We can think of X_S as being generated by first picking $|X_S| = k$ and then picking the subpopulations from which the singletons come. Write the labels as a vector $\vec{Y} \in [N]^k$, which needs at most $k \log N$ bits to describe. We have $I(X_S; P \mid K) \leq \mathbb{E}_k[k \cdot I(X_1; \mathcal{I}_1, b_1)] + \mathbb{E}_k[k \log N]$, using X_1 to denote a singleton sample. Since the unfixed bits are uniform and $|\mathcal{I}_1|$ is a random variable,

$$\begin{aligned} I(X_1; \mathcal{I}_1, b_1) = I(X_1; |\mathcal{I}_1|, \mathcal{I}_1, B_1) &= I(X_1; |\mathcal{I}_1|) + I(X_1; \mathcal{I}_1 B_1 \mid |\mathcal{I}_1|) \\ &= 0 + H(X_1 \mid |\mathcal{I}_1|) - H(X_1 \mid B_1, \mathcal{I}_1, |\mathcal{I}_y|) \\ &= d - (d - \mathbb{E}[|\mathcal{I}_1|]). \end{aligned}$$

Since $|\mathcal{I}_1| \sim \text{Bin}(d, \rho)$, we have $I(X_1; \mathcal{I}_1, b_1) = \rho d$.

Putting together the expectations and recalling that, by definition, $\mathbb{E}[k] = \mu_1 n$ finishes the proof. □

4.4 Error Analysis

We will show that the optimal error will be near zero when the test sample comes from a subpopulation represented in the data. To bound the error in the other case, we must control K_0, the number of unrepresented points in the data. Analogous to τ_1 and μ_1, we define the following terms

$$\tau_0 \stackrel{\text{def}}{=} \Pr[(z, y) \text{ comes from } j \mid X \text{ contains no samples from } j], \text{ and } \mu_0 \stackrel{\text{def}}{=} \frac{\mathbb{E}[K_0]}{n}. \tag{47}$$

Proposition 4.5 (Error Lower Bound)**.** *Let* E_0 *be the event that the test sample comes from a subpopulation with no representatives in the data set. Every algorithm* A *satisfies*

$$\Pr[A \textit{ errs on } \text{Learn}(q_c) \mid E_0] \geq 1 - \frac{1}{\mu_0 n}.$$

Proof. For any fixed $K_0 = k_0$, no algorithm can achieve error below $1 - \frac{1}{k_0}$, since all unrepresented subpopulations are equally likely. Decompose A's error over K_0:

$$\begin{aligned} \Pr[A \text{ errs on Learn}(q_c) \mid E_0] &= \sum_{k=0}^{n} \Pr[K_0 = k_0 \mid E_0] Pr[A \text{ errs on Learn}(q_c) \mid E_0, K_0 = k_0] \\ &\geq \sum_{k=0}^{n} \Pr[K_0 = k_0 \mid E_0] \left(1 - \frac{1}{k_0}\right). \end{aligned}$$

By Bayes' rule,

$$\Pr[K_0 = k_0 \mid E_0] = \frac{\Pr[E_0 \mid K_0 = k_0] \Pr[K_0 = k_0]}{\Pr[E_0]} = \frac{\tau_0 k_0 \Pr[K_0 = k_0]}{\tau_0 \mu_0 n}. \tag{48}$$

Thus, since the probabilities sum to 1,

$$\Pr[A \text{ errs on Learn}(q_c) \mid E_0] \geq \sum_{k=0}^{n} \Pr[K_0 = k_0 \mid E_0] - \sum_{k=0}^{n} \Pr[K_0 = k_0] \frac{k_0}{\mu_0 n k_0} \tag{49}$$

$$= 1 - \frac{1}{\mu_0 n}. \tag{50}$$

□

Proposition 4.6 (Accuracy of Naive Algorithm)**.** *Set* $\rho = \frac{c\sqrt{\ln n}}{\sqrt{d}}$ *for constant* $c > \sqrt{2}$*. Let* E_0, E_1*, and* $E_{>1}$*, respectively be the events that the test sample comes from subpopulation with zero, one, or multiple representatives in the data set. There is a single learning algorithm* A^* *solving both* $\text{Learn}(n, N, q_{HC}, \pi)$ *and* $\text{Singletons}(k, q_{HC})$ *for every* $k \leq n$ *that achieves, for some constant* $\alpha > 0$*,*

1. $\Pr[A^* \textit{ errs on } \text{Learn}(q_{HC}) \mid E_0] \leq \left(1 - \frac{1}{\mu_0 n}\right) + 3n^{-\alpha}$.
2. $\Pr[A^* \textit{ errs on } \text{Learn}(q_{HC}) \mid E_1] \leq 3n^{-\alpha}$.
3. $\Pr[A^* \textit{ errs on } \text{Learn}(q_{HC}) \mid E_{>1}] \leq \Pr[A^* \textit{ errs on } \text{Learn}(q_{HC}) \mid E_1]$.
4. $\Pr[A^* \textit{ errs on } \text{Singletons}(k, q_{HC})] \leq 3n^{-\alpha}$.

Proof. A^* has Alice send Bob a randomly-chosen data point from each subpopulation for which she has data, so (3) is established. Bob sets a threshold $\eta = \frac{d}{2} - \frac{c'}{2}\sqrt{d \ln n}$ for some constant $c' < c$ and outputs the label of the sample x nearest (in Hamming distance) to his test sample z. If there is no sample within distance η, Bob picks a random label that was not represented in Alice's dataset or, if there were none, an arbitrary label. Let $\{(X_j, j)\}$ be the labeled samples that Alice sends Bob. By symmetry it suffices for (2) to analyze the case when Z arises from the first subpopulation.

We show that, with high probability, $A_1 \triangleq d_H(Z, X_1) \leq \eta$ and, $\forall j \neq 1$, $A_j \triangleq d_H(Z, X_j) > \eta$. When this happens, Bob gets the correct answer.

We have $\mathbb{E}[A_1] = d\left(\frac{1-\rho}{2}\right) = \frac{d}{2} - \frac{c}{2}\sqrt{d \ln n}$. Since A_1 is a sum of d independent 0/1 random variables, apply the Hoeffding bound [28, pg. 77]:

$$\begin{aligned}
\Pr[A_1 \geq \eta] &= \Pr\left[A_1 \geq \left(\frac{d}{2} - \frac{c'}{2}\sqrt{d \ln n}\right) + \frac{c}{2}\sqrt{d \ln n} - \frac{c}{2}\sqrt{d \ln n}\right] \\
&\leq \Pr\left[\left|\frac{1}{d}A_1 - \frac{1}{d}\,\mathbb{E}[A_1]\right| \geq \left(\frac{c}{2} - \frac{c'}{2}\right)\frac{\sqrt{\ln n}}{\sqrt{d}}\right] \\
&\leq 2\exp\left\{-2d\left(\left(\frac{c}{2} - \frac{c'}{2}\right)\frac{\sqrt{\ln n}}{\sqrt{d}}\right)^2\right\} = 2n^{-\frac{(c-c')^2}{2}}.
\end{aligned}$$

Since $c' < c$, this will vanish.

We now bound $\Pr[A_2 \le \eta]$ and apply the union bound over j. Since Y and X_2 are independent, $\mathbb{E}[A_2] = \frac{d}{2}$. Apply a Chernoff bound [28, pg. 75]:

$$\begin{aligned}\Pr[A_2 \le \eta] &= \Pr\left[A_2 \le \frac{d}{2} - \frac{c'}{2}\sqrt{d \ln n}\right] \\ &= \Pr\left[A_2 \le \mathbb{E}[A_2] - \frac{c'}{2}\sqrt{d \ln n}\right] \\ &\le \exp\left\{-2\left(\frac{c'}{2}\sqrt{d \ln n}\right)^2 / d\right\} = n^{-(c')^2/2}.\end{aligned}$$

Alice sends Bob samples from at most $n-1$ other subpopulations with data, so

$$\Pr\left[\bigcup_{i>1}(A_i \le \eta)\right] \le \frac{n-1}{n^{(c')^2/2}} \le n^{1-(c')^2/2}.$$

With $c > c' > \sqrt{2}$, by the union bound the probability of $A_1 > \eta$ or $A_j \le \eta$ for some $j \ne 1$ is at most $3n^{-\alpha}$ for some constant $\alpha > 0$.

Observe that conditioned on E_0 and for all k_0, by the same arguments we have that with probability at most $3n^{-\alpha}$ the test sample will have distance less than η from any sample in Alice's dataset. If this fails to happen, Bob guesses a random unrepresented subpopulation, achieving error at most $\left(1 - \frac{1}{\mu_0 n}\right) + 3n^{-\alpha}$. This proves (1).

Finally, note that in Singletons(k, q_{HC}) the analysis of A_1 is identical and we need to take the union bound over $k \le n$ other singletons, so (4) holds. □

Proposition 4.7. *Set $\rho = \frac{c\sqrt{\ln n}}{\sqrt{d}}$ for constant $c > \sqrt{2}$. There is a constant $\alpha > 0$ such that $\phi_1(q_{HC}) \le 3n^{-\alpha}$ and $\phi_2(q_{HC}) = 3n^{-\alpha}$.*

Proof. Using a lower bound of 0 for the second terms and the upper bound of $3n^{-\alpha}$ for each k from Proposition 4.6.ii, we have

$$\begin{aligned}\phi_2(q_{HC}) = \sum_{k=1}^{N} \Pr[K = k \mid E_1]\Big(&\Pr[A_{OPT} \text{ errs on Learn}(q_{HC}) \mid E_1, K = k] && (51)\\ &- \inf_{A'} \Pr[A' \text{ errs on Singletons}(k, q_{HC})]\Big) && (52)\\ \le \sum_{k=1}^{N} \Pr[K = k \mid E_1] &\times 3n^{-\alpha} = 3n^{-\alpha}. && (53)\end{aligned}$$

For ϕ_1 we break up the terms across mutually exclusive events $\bar{E_1} = E_0 \cup E_{>1}$.

$$\begin{aligned}
\phi_1(q_{HC}) = \Pr[\bar{E_1}]\Big(&\Pr[A_{OPT} \text{ errs on Learn}(q_{HC}) \mid \bar{E_1}] \\
&- \Pr[A \text{ errs on Learn}(q_{HC}) \mid \bar{E_1}]\Big) \\
= \Pr[E_0]\Big(&\Pr[A_{OPT} \text{ errs on Learn}(q_{HC}) \mid E_0] && (54)\\
&- \Pr[A \text{ errs on Learn}(q_{HC}) \mid E_0]\Big) && (55)\\
+ \Pr[E_{>1}]\Big(&\Pr[A_{OPT} \text{ errs on Learn}(q_{HC}) \mid E_{>1}] && (56)\\
&- \Pr[A \text{ errs on Learn}(q_{HC}) \mid E_{>1}]\Big). && (57)
\end{aligned}$$

We have $\Pr[E_0]+\Pr[E_{>1}] \leq 1$. For the positive error terms in lines (54) and (56) we have upper bounds from Proposition 4.6. For line (57), use the lower bound $\Pr[A \text{ errs on Learn}(q_c) \mid E_0] \geq 1 - \frac{1}{\mu_0 n}$ from Proposition 4.5. Lower bound the final term in line (57) with 0.

$$\begin{aligned}
\phi_1(q_{HC}) &= \Pr[E_0]\left(\left(1 - \frac{1}{\mu_0 n}\right) + 3n^{-\alpha} - \left(1 - \frac{1}{\mu_0 n}\right)\right) \\
&\quad + \Pr[E_{>1}]\left(3n^{-\alpha} + 0\right) \\
&\leq 3n^{-\alpha}.
\end{aligned}$$

□

4.5 Lower Bound for Singletons Task

Our lower bound for Singletons(k, q_{HC}) is an immediate corollary of an identical lower bound on the external information complexity of Nearest of k Neighbors.

Lemma 4.8. *Set $\rho = \frac{c\sqrt{\ln n}}{\sqrt{d}}$ for constant $c > \sqrt{2}$. For every $k \leq n$, any one-way communication protocol for Nearest of k Neighbors with error at most ϵ satisfies*

$$I(X'; M) \geq \frac{kd}{c^2 \ln 2} \cdot \frac{\log k}{\log n} \cdot (1 - 2h(\epsilon_k)).$$

Corollary 4.9. *Set $\rho = \frac{c\sqrt{\ln n}}{\sqrt{d}}$ for constant $c > \sqrt{2}$. For every $k \leq n$, any algorithm A that is ϵ_k-suboptimal on* Singletons(k, q_{HC}) *is ϵ_k-suboptimal on Nearest of k Neighbors with the same information cost $I(X; M)$. Thus, A satisfies*

$$I(X'; M) \geq \frac{kd}{c^2 \ln 2} \cdot \frac{\log k}{\log n} \cdot (1 - 2h(\epsilon_k)).$$

Proof of Corollary 4.9. This follows immediately from Lemma 4.8 and two properties of Singletons(k, q_{HC}): (1) any two samples from different subpopulations are independent and uniform, and (2) any two samples z_1, z_2 from the same subpopulation have uniform z_1 and z_2 with the conditional distribution defined by $BSC_{\frac{1-\rho}{2}}(z_1)$. □

The proof of Lemma 4.8 is adapted from a one-shot proof appearing in [23] and relies on the strong data processing inequality for binary symmetric channels.

Strong Data Processing Inequality. *Suppose we have a Markov chain $M - X - Y$ where $X, Y \in \{0,1\}^d$, (for all $i \in d$, entrywise) $\Pr[X_i = 1] = p$, and $Y = \mathrm{BSC}_{\frac{1-\rho}{2}}(X)$. Then*

$$I(M;Y) \leq \rho^2 I(M;X).$$

Proof of Lemma 4.8. Since Bob, with access to M and test sample Z, can guess the index $J \in_R [k]$ with error ϵ_k, we have via Fano's inequality that

$$\begin{aligned} I(J;M,Z) &= H(J) - H(J \mid M,Z) && (58)\\ &\geq \log k - (\epsilon_k \log k - h(\epsilon_k)) && (59)\\ &= \left(1 - \epsilon_k - \frac{h(\epsilon_k)}{\log k}\right) \log k && (60)\\ &\geq (1 - 2h(\epsilon_k)) \log k, && (61) \end{aligned}$$

since for all $\epsilon_k \leq \frac{1}{2}$, $h(\epsilon_k) \geq \epsilon_k$. We will now upper bound $I(J;M,Z)$. Use P to denote the true distribution. We apply the "radius" property of mutual information and take Q to be the product of the marginals over M and Z: $Q_{M,Z} = P_M \times P_Z$:

$$\begin{aligned} I(J;M,Z) &= \inf_{Q_{M,Z} \in \Delta((M,Z))} \mathop{\mathbb{E}}_j \left[D_{KL}\left(P_{M,Z|J=j} \| Q_{M,Z}\right)\right] && (62)\\ &\leq \mathop{\mathbb{E}}_j \left[D_{KL}\left(P_{M,Z|J=j} \| P_M \times P_Z\right)\right]. && (63) \end{aligned}$$

Next, note that $M \perp J$ and $Z \perp J$, so we have

$$\begin{aligned} \mathop{\mathbb{E}}_j \left[D_{KL}\left(P_{M,Z|J=j} \| P_M \times P_Z\right)\right] &= \mathop{\mathbb{E}}_j \left[D_{KL}\left(P_{M,Z|J=j} \| P_{M|J=j} \times P_{Z|J=j}\right)\right] && (64)\\ &= \mathop{\mathbb{E}}_j \left[I(M; Z \mid J = j)\right]. && (65) \end{aligned}$$

Now we apply the SDPI for binary symmetric channels. When $J = j$, we have dependency graph drawn in Figure 2a, which is equivalent to the Markov chain in Figure 2b. Thus we can marginalize out $\{X_i\}_{i \neq j}$ and apply Lemma 4.5:

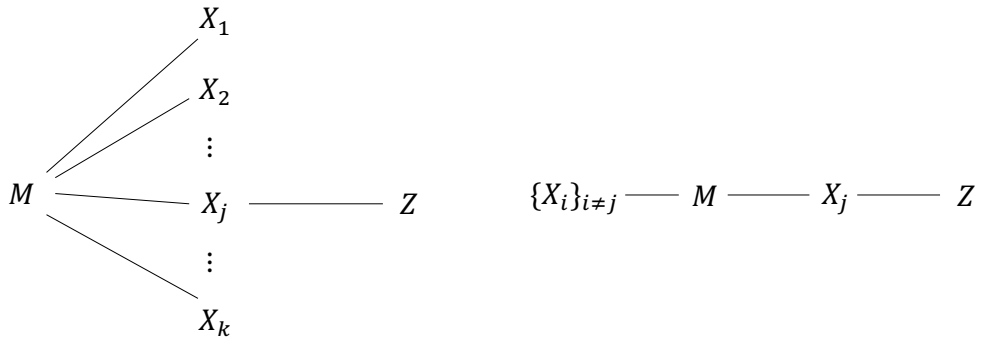
$$\mathop{\mathbb{E}}_j \left[I(M; Z \mid J = j)\right] \leq \mathop{\mathbb{E}}_j \left[\rho^2 I(M; X_j \mid J = j)\right]. \quad (66)$$

For any i, the mutual information between M and X_i is independent of J; it depends only on Alice's protocol. So

$$\mathop{\mathbb{E}}_j \left[\rho^2 I(M; X_j \mid J = j)\right] = \mathop{\mathbb{E}}_i \left[\rho^2 I(M; X_i)\right] \quad (67)$$

and, combining these steps and writing out the expectation, we have

$$I(J;M,Z) \leq \mathop{\mathbb{E}}_i \left[\rho^2 I(M; X_i)\right] = \frac{\rho^2}{k} \sum_{i=1}^{k} I(M; X_i). \quad (68)$$



(a) The dependency graph for Singletons(k, q_{HC}), conditioned on $J = j$.

(b) The same graph rearranged: Z depends on M only through X_j.

Figure 2

Using the chain rule for mutual information and the independence of the $\{X_i\}$, we have

$$\sum_j I(M; X_j) = \sum_j H(X_j) - H(X_j \mid M) \tag{69}$$

$$= \sum_j H(X_j \mid X_1^{j-1}) - H(X_j \mid M) \tag{70}$$

$$\leq \sum_j H(X_j \mid X_1^{j-1}) - H(X_j \mid M, X_1^{j-1}) \tag{71}$$

$$= I(M; X). \tag{72}$$

Therefore, combining Equations (61) and (68),

$$(1 - 2h(\epsilon_k)) \log k \leq I(J; M, Z) \leq \frac{\rho^2}{k} I(M; X). \tag{73}$$

Plug in $\rho = \frac{c\sqrt{\ln n}}{\sqrt{d}}$ and change the natural log to base 2

$$\frac{\rho^2}{k} = \frac{c^2 \ln 2 \cdot \log n}{kd}. \tag{74}$$

As observed in the upper bound, with constant $c > \sqrt{2}$ there is an algorithm whose accuracy vanishes as k increases. Rearranging we get a lower bound on $I(X; M)$. □

4.6 Completing the Proof

Proof of Theorem 4.1. Part (1) follows from Lemma 4.4's upper bound on $I(X; P \mid K)$.

For part (2), we assume an algorithm A for Learn(q_{HC}) with excess error ϵ_k. Our lower bound on Singletons(k, q_{HC}) from Corollary 4.9 gives us

$$f_k(\epsilon_k) = \frac{kd}{c^2 \ln 2} \cdot \frac{\log k}{\log n} \cdot (1 - 2h(\epsilon_k)) .$$

Using our central reduction of Lemma 2.1 and Lemma 4.4 to upper bound $I(X_S; P \mid K)$, we prove part (2):

$$\begin{aligned}
&I(X_S;M \mid P,K) \\
&\quad \geq I(X_S; M \mid K) - I(X_S; P \mid K) \\
&\quad \geq \frac{d}{c^2 \ln 2 \cdot \log n} \cdot \mathop{\mathbb{E}}_{k} \left[k \log k \cdot (1 - 2h(\epsilon_k))\right] - \mu_1 n \cdot \rho d + \mu_1 n \log N. \\
&\quad = \frac{d}{c^2 \ln 2 \cdot \log n} \cdot \mathop{\mathbb{E}}_{k} \left[k \log k \cdot (1 - 2h(\epsilon_k))\right] - nd\rho - n \log N.
\end{aligned}$$

Part (3) adds an assumption and allows us to get a cleaner bound. Let G be the good event that $K \geq \frac{\mu_1 n}{2}$. We assume $\Pr[G] \geq 1 - \beta$. With significant probability mass on the larger values, we can drop the smaller values of k. We have, via the nonnegativity of mutual information and Lemma 2.1,

$$\begin{aligned}
I(X_S; M \mid K) &= \Pr[G] \cdot I(X_S; M \mid K, G) + \Pr[\bar{G}] \cdot I(X_S; M \mid K, G) \\
&\geq (1 - \beta) \cdot I(X_S; M \mid K, G) \\
&\geq (1 - \beta) \cdot \mathop{\mathbb{E}}_{k \sim K|G} [f_k(\epsilon_k)] \\
&\geq (1 - \beta) \cdot \mathop{\mathbb{E}}_{k \sim K|G} \left[\frac{kd \log k}{c^2 \ln 2 \cdot \log n} \cdot (1 - 2h(\epsilon_k)) \right].
\end{aligned}$$

We have $\frac{\log k}{\log n} \geq \frac{\log \mu_1 n/2}{\log n}$, which allows us to remove this dependence on k. For simplicity write $c' = \frac{(1-\beta) d \log \mu_1 n/2}{c^2 \ln 2 \log n}$, which contains terms independent of k. The crucial step is to realize that the distribution of K conditioned on $K \geq \mu_1 n/2$ is larger (for all values $k \geq \mu_1 n/2$) than the unconditional distribution. Thus we can use the unconditional expectation as a lower bound and add/subtract the "other half" of the expectation:

$$\begin{aligned}
E_{k \sim K|G} \left[(1 - 2h(\epsilon_k))\, k \right] &= \sum_{k = \mu_1 n/2}^{n} \Pr[K = k \mid G] \cdot (1 - 2h(\epsilon_k)) \cdot k \\
&\geq \sum_{k = \mu_1 n/2}^{n} \Pr[K = k] \cdot (1 - 2h(\epsilon_k)) \cdot k \\
&\quad + \sum_{k=1}^{\mu_1 n/2 - 1} \Pr[K = k] \cdot (1 - 2h(\epsilon_k)) \cdot k \\
&\quad - \sum_{k=1}^{\mu_1 n/2 - 1} \Pr[K = k] \cdot (1 - 2h(\epsilon_k)) \cdot k \\
&\geq \mathbb{E}[(1 - 2h(\epsilon_k))k] - \beta n.
\end{aligned}$$

We use Lemma C.1 to push the expectation inside $h(\cdot)$, since $1 - 2h(\cdot)$ is convex and decreasing for sufficiently small arguments. This yields

$$\mathbb{E}[(1 - 2h(\epsilon_k))k] \geq \mu_1 n \left(1 - 2h\left(\frac{\mathbb{E}[k\epsilon_k]}{\mu_1 n}\right)\right).$$

From the central reduction, Lemma 2.1, we have $\mathbb{E}[k\epsilon_k] \leq \frac{\epsilon+\phi_1(q_{HC})+\phi_2(q_{HC})}{\tau_1}$ and by Proposition 4.6 both *phi* terms are upper bounded by $3n^{-\alpha}$ for some positive contant α. Combining this with the multiplicative and additive factors we haven't carried with us we get

$$\begin{aligned} I(X_S; M \mid K) &\geq c'\mu_1 n \left(1 - 2h\left(\frac{\epsilon + 6n^{-\alpha}}{\tau_1\mu_1 n}\right)\right) - c'\beta n \\ &\geq \frac{(1-\beta)}{c^2 \ln 2} \cdot \mu_1 nd \left(1 - 2h\left(\frac{\epsilon + 6n^{-\alpha}}{\tau_1\mu_1 n}\right)\right) - nd\left(\frac{\log 2/\mu_1}{\log n} + \beta\right), \end{aligned}$$

writing out c' and simplifying. Then, applying $I(X_S; M \mid P, K) \geq I(X_S; M \mid K) - I(X_S; P \mid K)$, we have the claimed

$$\begin{aligned} I(X; M \mid P) \geq \frac{(1-\beta)}{c^2 \ln 2} \cdot \mu_1 nd \left(1 - 2h\left(\frac{\epsilon + 6n^{-\alpha}}{\tau_1\mu_1 n}\right)\right) \\ - nd\left(\frac{\log 2/\mu_1}{\log n} + \beta + \rho + \frac{\log N}{d}\right). \end{aligned}$$

□

Appendix

A Generating Subpopulation Mixture Coefficients

A.1 Definitions

We generate a mixture over N subpopulations using the process introduced in [19]. We begin with a list π of nonnegative values. For each subpopulation j, we sample a value $\delta_j \sim \text{Uniform}(\pi)$. To create a probability distribution D, we normalize:

$$D(j) = \frac{\delta_j}{\sum_{i\in[N]} \delta_i}.$$

This process is identical for all j, so we define $\bar{\pi}^N$ as the resulting marginal distribution over the mixture coefficient for any single subpopulation.

The quantity τ_1 is defined in [19] as

$$\tau_1 = \frac{\mathbb{E}_{\alpha\sim\bar{\pi}^N}\left[\alpha^2(1-\alpha)^{n-1}\right]}{\mathbb{E}_{\alpha\sim\bar{\pi}^N}\left[\alpha(1-\alpha)^{n-1}\right]}.$$

Lemma 2.1 of [19] proves the equality

$$\mathop{\mathbb{E}}_{\substack{D\sim\mathcal{D}_\pi^N,\\ ID\sim D}}[D(j) \mid ID = id] = \tau_1.$$

Observe that, for any set of cluster identifiers $ID = id$,

$$\begin{aligned} \mathop{\mathbb{E}}_{\substack{D\sim\mathcal{D}_\pi^N,\\ ID\sim D}}[D(j) \mid ID = id] &= \sum_\alpha \alpha \cdot \Pr[D(j) = \alpha \mid ID = id] \\ &= \sum_\alpha \Pr[\iota(y) = j \mid D(j) = \alpha] \cdot \Pr[D(j) = \alpha \mid ID = id] \\ &= \Pr[\iota(y) = j \mid ID = id]. \end{aligned}$$

A.2 Details for Bimodal Prior

Here we provide the details necessary for Example 2.2. Recall that we set $N = 2^n$. To build π we add 1 copy of $\frac{1}{2n}$ and $n2^n - 1$ copies of $\frac{1}{2\cdot 2^n}$. This yields

$$\text{Uniform}(\pi) = \begin{cases} \frac{1}{2n} & \text{w.p. } n2^{-n} \\ \frac{1}{2\cdot 2^n} & \text{w.p. } 1 - n2^{-n}. \end{cases}. \tag{75}$$

We now show that the normalizing constant C will concentrate about its mean. Let H be the number of heavy bins that are drawn. We have, as a lower bound,

$$C = \frac{H}{2n} + \frac{1}{2\cdot 2^n}\left(2^n - H\right) = \frac{1}{2} + \frac{H}{2}\left(\frac{1}{n} - \frac{1}{2^n}\right) \geq \frac{1}{2}. \tag{76}$$

If $H \leq 2n$ then $C \leq \frac{3}{2}$, so by a Chernoff bound we have

$$\Pr[C \geq 3/2] \leq \Pr[H \geq 2\,\mathbb{E}[H]] \leq e^{-n/3}.$$

We can now lower bound μ_1 and τ_1 with results from [19]. Define the weight after normalization.

$$\texttt{weight}\left(\bar{\pi}^N, [\beta_1, \beta_2]\right) \stackrel{\text{def}}{=} N \cdot \mathop{\mathbb{E}}_{\alpha\sim\bar{\pi}^N}\left[\alpha \cdot \mathbf{1}_{\alpha\in[\beta_1,\beta_2]}\right]. \tag{77}$$

Equation 5 from [19] gives us that

$$\mu_1 n \geq \frac{n}{3}\texttt{weight}\left(\bar{\pi}^N, \left[0, \frac{1}{n}\right]\right) = \frac{n}{3}, \tag{78}$$

since $C \geq \frac{1}{2}$ implies $\max\alpha \leq \frac{1}{n}$. To bound τ_1 we use Lemma 2.5 from [19]:

$$\tau_1 \geq \frac{1}{5n}\texttt{weight}\left(\bar{\pi}^N, \left[\frac{1}{3n}, \frac{2}{n}\right]\right). \tag{79}$$

Observe that we always draw $\alpha \geq \frac{1}{3n}$ when (i) we draw a heavy bin and (ii) $C \leq \frac{3}{2}$, which always happens when $H \leq 2n$. By another Chernoff bound,

$$\Pr[H \geq 2n \text{ or } H \leq n/2] \leq \Pr[H \geq 2n] + \Pr[H \leq n/2] \leq e^{-n/3} + e^{-n/8}.$$

Since the probability of drawing a heavy bin is $n2^{-n}$, we have

$$\Pr_{\alpha\sim\bar{\pi}^N}\left[\alpha \geq \frac{1}{3n}\right] \geq n2^{-n}\left(1 - 2e^{-\Omega(n)}\right). \tag{80}$$

Thus, again applying $\max\alpha \leq \frac{1}{n}$,

$$\begin{aligned}
\tau_1 \geq \frac{1}{5n}\texttt{weight}\left(\bar{\pi}^N, \left[\frac{1}{3n}, \frac{2}{n}\right]\right) &= \frac{1}{5n}\cdot 2^n \cdot \mathop{\mathbb{E}}_{\alpha\sim\bar{\pi}^N}\left[\alpha \cdot \mathbf{1}_{\alpha\in[1/3n,2/n]}\right] && (81)\\
&\geq \frac{2^n}{5n}\cdot\frac{1}{3n}\cdot\Pr_{\alpha\sim\bar{\pi}^N}\left[\alpha \geq \frac{1}{3n}\right] && (82)\\
&\geq \frac{2^n}{5n}\cdot\frac{1}{3n}\cdot n2^{-n}\left(1 - e^{-\Omega(n)}\right). && (83)
\end{aligned}$$

Simplifying, we see that $\tau_1 = \Omega(1/n)$.

B Central Reduction via Non-Optimal Baseline

In some tasks the exactly-optimal algorithm A_{OPT} may be difficult to analyze, while a naive "baseline" algorithm A^* exists with error approaching optimal. To deal with this, we will use the following modification of Lemma 2.1, where the error terms ϕ_1 and ϕ_2 now depend on A^*. The proof is almost line-by-line identical to that of Lemma 2.1, observing in Equation (10) that

$$\epsilon = \Pr[A \text{ errs on Learn}(q_c)] - \Pr[A_{OPT} \text{ errs on Learn}(q_c)] \tag{84}$$
$$\geq \Pr[A \text{ errs on Learn}(q_c)] - \Pr[A^* \text{ errs on Learn}(q_c)] \tag{85}$$

and proceeding with A^* in place of A_{OPT}.

Lemma B.1 (Central Reduction)**.** *Suppose we have the following lower bound for every k: any algorithm $A^k(X')$ that is ϵ_k-suboptimal for* Singletons(k, q_c) *satisfies*

$$I(X'; A^k(X') \mid K = k) \geq f_k(\epsilon_k).$$

For any algorithm $A(X)$ that is ϵ-suboptimal on Learn(q_c)*, there exists a sequence $\{\epsilon_k\}_{k=1}^n$ such that $\mathbb{E}_k[k\epsilon_k] = \epsilon$ and $I(X_S; M \mid K) \geq \mathbb{E}_k[f_k(\epsilon_k)]$.*

Furthermore, if $f_k(\epsilon_k) \geq k \cdot g(\epsilon_k)$ for convex and nonincreasing $g(\cdot)$, then

$$I(X_S; M \mid K) \geq \mu_1 n \cdot g\left(\frac{\epsilon + \phi_1(q_c) + \phi_2(q_c)}{\tau_1 \mu_1 n}\right).$$

Here $\phi_1(q_c)$ and $\phi_2(q_c)$ are task-specific terms, defined by

$$\phi_1(q_c) = \Pr[\bar{E}_1]\left(\Pr[A^*t \textit{ errs on } \text{Learn}(q_c) \mid \bar{E}_1] - \Pr[A \textit{ errs on } \text{Learn}(q_c) \mid \bar{E}_1]\right), \tag{86}$$

$$\phi_2(q_c) = \sum_{k=1}^{N} \Pr[K = k \mid E_1]\Big(\Pr[A^* \textit{ errs on } \text{Learn}(q_c) \mid E_1, K = k] \tag{87}$$

$$- \inf_{A'} \Pr[A' \textit{ errs on } \text{Singletons}(k, q_c)]\Big). \tag{88}$$

E_1 is the event that the test sample comes from a subpopulation with exactly one representative.

C Expectation Trick for Jensen's Inequality

Jensen's Inequality. *Let X be a random variable and f a concave function. Then*

$$\mathbb{E}[f(X)] \leq f\left(\mathbb{E}[X]\right).$$

We will several times make use of the following application. Note that both inequalities applies in the other direction to convex functions.

Lemma C.1. *Suppose f is a concave function, g is a function, and X is a nonnegative random variable with distribution $p(x)$. Then*

$$\mathbb{E}\left[X \cdot f(g(X))\right] \leq \mathbb{E}[X] f\left(\frac{\mathbb{E}[X \cdot g(X)]}{\mathbb{E}[X]}\right).$$

Proof. We cannot directly apply Jensen's inequality, since $x \cdot f(x)$ will not in general be concave. Instead we introduce a distribution $q \propto x \cdot p(x)$. Note that $\sum_x x \cdot p(x) = \mathbb{E}_p[X]$, so we have

$$q(x) = \frac{x \cdot p(x)}{\mathbb{E}_p[X]}.$$

We now rewrite so that the expectation is over q:

$$\begin{aligned}
\mathop{\mathbb{E}}_{p}[X \cdot f(g(X))] &= \sum_x p(x) \cdot x \cdot f(g(x)) \\
&= \frac{\mathbb{E}_p[X]}{\mathbb{E}_p[X]} \sum_x p(x) \cdot x \cdot f(g(x)) \\
&= \mathop{\mathbb{E}}_{p}[X] \sum_x q(x) \cdot f(g(x)) \\
&= \mathop{\mathbb{E}}_{p}[X] \cdot \mathop{\mathbb{E}}_{q}[f(g(X))].
\end{aligned}$$

We now have the expectation of a concave function and can apply Jensen's inequality and finish the proof:

$$\begin{aligned}
\mathop{\mathbb{E}}_{p}[X \cdot f(X)] &\leq \mathop{\mathbb{E}}_{p}[X] \cdot f\left(\mathop{\mathbb{E}}_{q}[g(X)]\right) \\
&= \mathop{\mathbb{E}}_{p}[X] \cdot f\left(\sum_x \frac{p(x) \cdot c \cdot g(x)}{\mathbb{E}_p[X]}\right) \\
&= \mathop{\mathbb{E}}_{p}[X] \cdot f\left(\frac{\mathbb{E}_p[X \cdot g(X)]}{\mathbb{E}_p[X]}\right).
\end{aligned}$$

□

D Tasks Related to Next-Symbol Prediction

D.1 Lower Bound for Threshold Learning

Threshold learning is a simple and well-studied binary classification task. Data $z \in \{0,1\}^d$ receives a label according to threshold $c \in \{0,1\}^d$, so $y = 0$ if $c \geq z$ and $y = 1$ otherwise.

Via a reduction from Next-Symbol Prediction, we demonstrate memorization in this setting. This substantially strengthens results in [6, 30], which contained bounds of $\Omega(\log d)$ for any proper, consistent learner; we prove a lower bound of $\Omega(d)$ while allowing any ϵ-suboptimal learner. As in Next-Symbol Prediction, this one-shot lower bound can be extended to the n-sample, N-subpopulation setting.

Claim D.1. *There exists a metadistribution $q_{\mathcal{T}_d}$ over problem instances of (realizable) d-bit threshold learning such that any ϵ-suboptimal algorithm receiving exactly one sample X satisfies*

$$I(X; M \mid P) \geq (1 - f(\epsilon) - o(1)) \cdot H(X \mid P),$$

where $f(\epsilon) \xrightarrow{\epsilon \to 0} 0$, so the algorithm must memorize the whole sample as the error vanishes.

Definition D.1 ($q_{\mathcal{T}_d}$ Component Distribution)**.** Draw threshold $c \in \{0,1\}^d$ uniformly at random. To generate labeled data (z, y), first pick a prefix length $\ell \in \{0, \ldots, d-1\}$ uniformly at random. Set

$$\begin{aligned} z(1:\ell) &= \begin{cases} c(1:\ell) & \text{w.p. } 1/2 \\ \text{Uniform}(\{0,1\}^\ell) & \text{otherwise} \end{cases} \\ z(\ell+1) &= 1 \\ z(\ell+2:d) &= \vec{0}. \end{aligned}$$

Label according to the threshold: $y = 0$ if $c \geq z$ and $y = 1$ otherwise. ■

The crucial feature of this distribution is, for data points which contain a length-ℓ prefix of c, the label is 1 if and only if $c(\ell+1) = 1$. This is clear with a small example, where $d = 7$ and $\ell = 4$:

$$\begin{aligned} c &= \underline{1\ 0\ 0\ 1}\ 0\ 1\ 1 \\ z &= \underline{1\ 0\ 0\ 1}\ 1\ 0\ 0. \end{aligned}$$

We have $c < z$ and thus $c(z) = c(\ell+1) = 0$, capturing the "next-bit" behavior.

Lemma D.2. *For any algorithm A that is ϵ-suboptimal for $q_{\mathcal{T}_d}$, there is an algorithm A' for Singletons$(1, q_{NSP})$, under noise parameter $\delta = 0$, that is $(4\epsilon + o(1))$-suboptimal. Furthermore, $I(X; A(X)) = I(X; A'(X))$.*

Proof. Alice and Bob get two (noiseless) prefixes of c. Call them z_A and z_B and denote their lengths ℓ_A and ℓ_B. Under algorithm A', they create $\tilde{z}_A$ and $\tilde{z}_B$ by padding to length d with "$10\cdots0$." They then run A on these padded inputs and return A's output. The algorithms' (identical) outputs have the same mutual information with their inputs.

To bound the error of A', let G be the good event that both samples (in the threshold problem) contain prefixes of c (as opposed to randomly-drawn prefixes). Then, by construction,

$$\Pr[A' \text{ errs on Singletons}(1, q_{NSP})] = \Pr[A \text{ errs on Singletons}(1, q_{\mathcal{T}_d}) \mid G].$$

We expand over G and $\bar{G}$:

$$\begin{aligned} \epsilon &= \Pr[A \text{ errs on Singletons}(1, q_{\mathcal{T}_d})] - \Pr[A_{OPT} \text{ errs on Singletons}(1, q_{\mathcal{T}_d})] \\ &= \Pr[G]\left(\Pr[A \text{ errs on Singletons}(1, q_{\mathcal{T}_d}) \mid G] - \Pr[A_{OPT} \text{ errs on Singletons}(1, q_{\mathcal{T}_d}) \mid G]\right) \\ &\quad + \Pr[\bar{G}]\left(\Pr[A \text{ errs on Singletons}(1, q_{\mathcal{T}_d}) \mid \bar{G}] - \Pr[A_{OPT} \text{ errs on Singletons}(1, q_{\mathcal{T}_d}) \mid \bar{G}]\right) \\ &\geq \frac{1}{4}\left(\Pr[A \text{ errs on Singletons}(1, q_{\mathcal{T}_d}) \mid G] - \Pr[A_{OPT} \text{ errs on Singletons}(1, q_{\mathcal{T}_d}) \mid G]\right) - o(1) \\ &\geq \frac{1}{4}\left(\Pr[A' \text{ errs on Singletons}(1, q_{NSP})] - \inf_{A''} \Pr[A'' \text{ errs on Singletons}(1, q_{NSP})]\right) - o(1). \end{aligned}$$

To establish the inequalities, observe that A_{OPT}, with access to both inputs, may fail to learn if G occurred only when the strings match by accident. The probability of this is no more than the probability that (1) both strings have length less than $\log d$, or (2) at least $\log d$ random bits match, both of which have probability $o(1)$. Rearranging finishes the proof. □

Lemma D.3. *For Singletons$(1, q_{\mathcal{T}_d})$, we have $I(X;P) = \frac{d+1}{4} + O(1)$.*

Proof. Let L be the length of the data point and B the indicator for "X was drawn independently of c." We have

$$\begin{aligned} I(X;P) = I(X,L,B;P) + I(B;P \mid X) &= I(B;P) + I(L;P \mid B) + I(X;P \mid B,L) + O(1) \\ &= 0 + 0 + I(X;P \mid B,L) + O(1) \\ &= \frac{1}{2} I(X;P \mid B=1,L) + \frac{1}{2} I(X;P \mid B=0,L) + O(1) \\ &= 0 + \frac{1}{2} \cdot \frac{d+1}{2} + O(1), \end{aligned}$$

since, when $B = 1$, the bits of X are independent of the threshold and, when $B = 1$ and $L = \ell$ for any ℓ, the mutual information is exactly $\ell + 1$. □

Proof of Claim D.1. Via the one-sample lower bound for Next-Symbol Prediction in Lemma 3.7 and the reduction above, any ϵ-suboptimal algorithm A for Singletons$(1, q_{\mathcal{T}_d})$ has

$$I(X;M) \geq \frac{d+1}{2} \cdot (1 - h(8\epsilon + o(1))).$$

Reusing a calculation from Section 2.3 that uses the fact that $P \perp M \mid X$, we have

$$I(X;M \mid P) \geq I(X;M) - I(X;P) \geq \frac{d+1}{4} \cdot (1 - 2 \cdot h(8\epsilon + o(1))).$$

□

D.2 Two-Length Next-Symbol Prediction

We present a sequence prediction learning task which is less natural than Next-Symbol Prediction but, in contrast, allows Alice to send a significantly smaller message to Bob after she has received multiple samples. We focus on the one-shot case.

Definition D.2 (q_2 Component Distribution)**.** In Two-Length Next-Symbol Prediction, a problem instance is described by a reference string $c \in \{0,1\}^d$ and two indices $j, k \in \{0, \ldots, d-1\}$. The metadistribution q_2 samples x, j, and k uniformly and independently. Given problem instance $P = (x, j, k)$, labeled data is generated i.i.d. in the following way. Flip a fair coin:

- If heads, return $(x_{1:j}, x_{j+1})$.
- If tails, return $(x_{1:k}, x_{k+1})$.

■

Although the problem instance needs $d + 2\log d$ bits to describe, Bob only needs to know (j, k, x_{j+1}, x_{k+1}) to answer correctly. This requires only $2(\log d + 1)$ bits. If Alice recieves $n > 1$ samples, she learns (j, k, x_{j+1}, x_{k+1}) exactly as soon as she receives inputs of different lengths. She may fail to learn the instance if $j = k$ or if all of her inputs have the same length, so this failure probability is bounded above by

$$\Pr[\text{Alice fails to learn } (j, k, x_{j+1}, x_{k+1})] \leq \frac{1}{d} + \frac{1}{2^{n-1}}.$$

Nevertheless, in the $n = 1$ case, Alice will have to send almost all her input to compete with the optimal protocol.

Theorem D.4. *Any learning algorithm for one-shot Two-Length Next-Symbol Prediction with error ϵ above optimal succeeds on (standard) Next-Symbol Prediction with error at most 2ϵ above optimal.*

Proof. Let indicator random variables C_A and C_B denote, for Alice and Bob respectively, the result of the coin flip deciding if their inputs were to be of length j or k. (Note that this coin is flipped even when $j = k$). Let $C = \mathrm{XOR}(C_A, C_B)$, so $C = 1$ means one player received a length-j string and the other received a length-k string. The key observation is that conditioning on $C = 1$ results in a distribution identical to that of noiseless one-shot Next-Symbol Prediction: Alice and Bob's inputs are prefixes of the same string with lengths chosen uniformly at random and Bob needs to answer the next bit. This any protocol with $\Pr[A \text{ correct} \mid C = 1] = p$ also has accuracy p on Next-Symbol Prediction.

For protocol A for Modified Next-Symbol Prediction, define

$$\epsilon' = \Pr[A \text{ errs} \mid C = 1] - \Pr[A_{OPT} \text{ errs} \mid C = 1],$$

the error gap when applied to inputs from Next-Symbol Prediction. (Note that the optimal protocol for the modified task is also optimal for the standard task.) Then we have

$$\begin{aligned}
\epsilon &= \Pr[A \text{ errs}] - \Pr[A_{OPT} \text{ errs}] \\
&= \Pr[C = 1] \left(\Pr[A \text{ errs} \mid C = 1] - \Pr[A_{OPT} \text{ errs}] \mid C = 1\right) \\
&\quad + \Pr[C = 0] \left(\Pr[A \text{ errs} \mid C = 0] - \Pr[A_{OPT} \text{ errs}] \mid C = 0\right) \\
&\geq \frac{1}{2} \cdot \epsilon',
\end{aligned}$$

where the inequality follows because the error of A is at least that of A_{OPT}. This completes the proof. □

Corollary D.5. *Any learning algorithm for one-shot Modified Next-Symbol Prediction with error ϵ above optimal satisfies*

$$I(M; X) \geq \frac{d+1}{2} \left(1 - h(4\epsilon)\right).$$

Proof. Lemma 3.7 asks for γ such that

$$\Pr[A \text{ correct}] = \frac{1}{2} + OPT_{\text{sing}}(1 - \gamma).$$

This implies $H(X \mid M) \leq \frac{d+1}{2} h(\gamma/2)$. By the above theorem, on Next-Symbol Prediction A has, converting from accuracy to error,

$$\begin{aligned}
2\epsilon &\geq \Pr[A \text{ errs}] - \Pr[A_{OPT} \text{ errs}] \\
&= \left(1 - \left(\frac{1}{2} + OPT_{\text{sing}}(1 - \gamma)\right)\right) - \left(1 - \left(\frac{1}{2} + OPT_{\text{sing}}\right)\right) \\
&= OPT_{\text{sing}} \gamma \\
&\geq \frac{\gamma}{4}.
\end{aligned}$$

Letting L be the length of Alice's sample (counting the label), we can compute the entropy:

$$H(X) = H(X, L) = H(L) + H(X \mid L) \geq \mathop{\mathbb{E}}_{\ell}[H(X \mid L = \ell)] = \frac{d+1}{2}.$$

Combining the entropy lower bound and conditional entropy upper bound establishes the claim. □

E Differentially-Private Algorithms Have High Error

By definition, the output of a differentially private algorithm is not very sensitive to changes in the input data set. In our setting, with data sets drawn i.i.d. from a fixed distribution, existing results from the differential privacy literature allow us to formalize this with an upper bound on mutual information:

Claim E.1. *Fix distribution $P = p$ over examples in $\{0,1\}^d$ and suppose the data X is drawn from the product distribution $p^{\otimes n}$. Then, for any (α, β)-differentially private algorithm A,*

$$I(X; A(X) \mid P = p) \leq n \left(2\epsilon \cdot \frac{e^{2\epsilon} - 1}{e^{2\epsilon} + 1} + \delta d + h(\delta) \right).$$

Our lower bounds imply that, for small constant ϵ, and ϵ-suboptimal algorithm on the tasks we consider must have $I(X; A(X) \mid P) = \Omega(nd)$. This is inconsistent with the above bound, even if $\alpha = O(\log n)$ and β is a sufficiently small constant. Recall that an informal standard for "meaningful privacy" requires that $\alpha = O(1)$ and $\delta \ll 1/n$. Thus, even for unacceptably large values of the privacy parameters, differential privacy is incompatbile with low suboptimality.

Outline of Proof. The claim follows from well-known statements in the privacy literature, so we will only provide the high-level ideas. We emphasize that such a statement holds only in the case of data from product distributions.

The first step is to provide an upper bound on the mutual information of any (α, β)-DP algorithm accepting a single input, i.e. a noninteractive LDP algorithm. See e.g. Lemma 3.6 in Cheu and Ullman [14] for such a proof; one first shows that any one-sample (α, β)-DP algorithm can be converted into a one-sample $(2\alpha, 0)$-DP algorithm that is close in total variation distance.

The second step is to show that n multiplied by this bound is itself an upper bound on the mutual information of any n-sample (α, β)-DP algorithm. This follows from a simple argument applying linearity of expectation and the fact that fixing the other $n - 1$ inputs allows us to treat any n-sample algorithm as one-sample. □

F From Average-Case to Worst-Case via Minimax

We provide metadistributions (dependent on the sample size n and dimension d) upon which any near-optimal algorithm has high information cost. A metadistribution is over problem instances, each of which is itself a distribution over labeled examples. The "easy" direction of von Neumann's Minimax Theorem allows us to turn this into a worst-case guarantee.

Proposition F.1. *Suppose q^* is a metadistribution such that any algorithm that is ϵ-suboptimal for Learn(n, N, q^*, π) satisfies $I(X; M \mid P) = \Omega(nd)$. Then, for any A that is ϵ-suboptimal, there exists a problem instance $p^* \in \Delta(\mathcal{X})$ such that*

$$I(X; M \mid P = p^*) \geq I(X; M|P) = \Omega(nd).$$

Proof. Since $I(X; M|P) = \mathbb{E}_{p\sim q^*}[I(X; M \mid P = p)]$ is an expectation, there always exists a problem instance whose value is at least that of the expectation. □

The assumption of Propostion F.1 deserves some discussion. Specifically, recall that a learning algorithm A is ϵ-suboptimal for Learn$(n, N, q_{n,d}, \pi)$ if it competes with the best possible learner $A^{(n,d)}_{OPT}$ for $q_{n,d}$ when receiving a sample of size n. An equivalent requirement is that the error of A on p be close that that of $A^{(n,d)}_{OPT}$ on average over instances $p \sim q_{n,d}$. On one hand, this assumption is much weaker than assuming that A is near-optimal for each instance p (since we compare with the learner $A^{(n,d)}_{OPT}$ whose average error over $p \sim q_{n,d}$ is lowest, not a learner that is tailored to p). On the other hand, it is not obviously comparable to the requirements of properness and consistency made by Bassily et al. [6], Nachum et al. [30]. For one thing, not all of our learning tasks fit neatly into the framework of PAC learning. For those that do, the sample size n that we consider is generally less than (or comparable to) the VC-dimension of the underlying concept class—too low to guarantee that every proper and consistent learner has high accuracy. Understanding the full relationship between these different kinds of assumptions is a subject for future work.

G One-Way Information Complexity of Gap-Hamming

We provide a lower bound for the one-way information complexity of the Gap-Hamming problem. This bound achieves the "right" constant, establishing that Alice must send $(1 - o(1)) \cdot d$ bits of information (out of the d she receives) as her error vanishes. This offers evidence for Conjecture 4.1, since the Singletons(k, q_{HC}) task is a natural generalization of Gap-Hamming to multiple samples. The two-way communication complexity lower bound of $\Omega(n)$ was first established in [13], with later papers providing simplified proofs (see [32] for additional background.) Notably for this work, [23] offered an information-theoretic lower bound using a strong data-processing inequality. A modified version of their proof yielded our proof of Lemma 4.8, but it is not clear that this approach can yield avoid losing a constant factor. We introduce a simple and general technique for proving lower bounds on one-way communication over product distributions. This technique generalizes and extends a proof in [35], allowing it to be applied to information complexity of general one-way communication problems, and showing that it can be used to avoid losing constant factors. To the best of our knowledge, this is the first result achieving the $(1 - o(1))$ factor for Gap-Hamming.

In the one-way Gap-Hamming problem, Alice gets $x \in \{0, 1\}^d$ and Bob gets $y \in \{0, 1\}^d$. We will take these inputs to be uniform and independent distribution.[4] Bob's goal is to output,

[4]Since the inputs are independent and the communication is one-way, for any algorithm we have $I(M; X, Y) = I(Y; M) + I(X; M \mid Y) = I(X; M \mid Y)$, so the "internal" and "external" information complexities coincide.

with probability at least $1-\epsilon$, the following partial function (using Hamming distance)

$$GHP(x,y) = \begin{cases} 1 & \text{if } d(x,y) \le \frac{d}{2} - c\sqrt{d} \\ 0 & \text{if } d(x,y) \ge \frac{d}{2} + c\sqrt{d} \\ * & \text{otherwise} \end{cases}$$

This is a promise problem controlled by parameter c. For any fixed c, with constant probability the promise holds. We study what happens when Alice and Bob succeed with probability at least $1-\epsilon$ for a sufficiently small constant ϵ.

Theorem G.1. *Let ϵ denote the probability that Alice and Bob make an error. For any fixed $c > 0$, there exists a function $g_c(\epsilon, d)$ such that*

$$IC^{\rightarrow}_{\epsilon}(GH_c) \ge (1 - g_c(\epsilon, d)) \cdot d,$$

where $g_c(\epsilon, d) \to 0$ for any sequence of pairs $\{\epsilon_i, d_i\}_{i=1}^{\infty}$ such that $\epsilon_i \to 0$ and $d_i \to \infty$.

Theorem G.1 follows, with a bit of calculation, from the following lemma. To prove Lemma G.2, we prove a function-agnostic statement about one-way information complexity over product distributions, then apply it to Gap-Hamming.

Lemma G.2. *If Alice and Bob succeed with probability at least $1-\epsilon$ then, for sufficiently large d and all $p \in (0,1)$,*

$$I(M;X) \ge H(X) \cdot (1 - h(p)) \cdot (1 - 2\epsilon/\alpha) - 1,$$

where α can be bounded as follows:

$$\alpha \ge \left(1 - 2F_{\mathcal{N}(0,1)}\left(\frac{-c}{\sqrt{1-p}}\right) - \mathcal{O}\left(\frac{1}{\sqrt{d(1-p)}}\right)\right)\left(2F_{\mathcal{N}(0,1)}\left(\frac{-3c}{\sqrt{p}}\right) - \mathcal{O}\left(\frac{1}{\sqrt{dp}}\right)\right).$$

Here $F_{\mathcal{N}(0,1)}(\cdot)$ is the CDF of the standard normal distribution.

A General Approach to One-Way Information Complexity We will first prove the following lemma for lower bounding the one-way external information complexity of any communication task $f : \mathcal{X} \times \mathcal{X} \to \mathcal{Z}$ where Alice gets X, Bob gets Y, and $X \perp Y$. To "instantiate" the lemma we need a notion of *incompatibility* which applies to two of Alice' inputs. For any definition of incompatibility, say y *distinguishes* x, x' if $f(x,y) \neq f(x',y)$. We will need a lower bound on distinguishing

$$\Pr[y \text{ distinguishes } x, x' \mid x, x' \text{ incompatible}] \ge \alpha$$

and an upper bound on the number of compatible inputs

$$\max_x \left|\{x' : x, x' \text{ compatible}\}\right| \le N_{\max}$$

Lemma G.3. *Suppose $X \perp Y$ and Alice and Bob succeed in computing $f(x, y)$ with probability $1 - \epsilon$. Fix a definition of incompatibility (which can depend on ϵ). Then*

$$I(X; M) \geq H(X) - \log N_{max} - 1 - \frac{2\epsilon}{\alpha}(\log |\mathcal{X}| - \log N_{max}).$$

If Alice's input is uniform and we can bound $\alpha \geq \Omega(\sqrt{\epsilon})$, this simplifies to

$$I(M; X) \geq H(X) \cdot \left(1 - \frac{\log N_{\max}}{H(X)}\right) \cdot \left(1 - O\left(\sqrt{\epsilon}\right)\right) - 1.$$

Proof. We perform the following thought experiment. Given Alice's message m, samples $x, x' \sim X \mid M = m$ independently. Let D be the event that y *distinguishes* these two inputs: that $f(x, y) \neq f(x', y)$. Let $\Pi(m, y)$ denote Bob's portion of the protocol.[5] Observe that

$$\begin{aligned} \Pr[D \mid M = m] &\leq \Pr[\Pi(M, Y) \neq f(X, Y) \mid M = m] \\ &\quad + \Pr[\Pi(M, Y) \neq f(X', Y) \mid M = m] \\ &= 2 \Pr[\text{error} \mid M = m], \end{aligned} \tag{89}$$

since at least of one of x, x' correspond to the wrong answer. So the thought experiment allows us, with a lower bound on $\Pr[D \mid M = m]$, to show the protocol will have high error.

Letting C be the event that these two inputs are compatible, we have

$$\Pr[D \mid M = m] \geq \Pr[D \mid \bar{C}] \cdot \Pr[\bar{C} \mid M = m]. \tag{90}$$

The first term we bound using the problem itself and the definiton of incompatibility. Note that we've assumed $\Pr[D]$ is independent of M conditioned on C. The second term we bound via the entropy $H(X \mid M = m)$.

Fix Alice's message $M = m$ to Bob. Using the fact that $X \perp X' \mid M$, we have (conflating events and indicator random variables)

$$H(X \mid M = m) = H(X \mid X', M = m) = H(X, C \mid X', M = m).$$

Then, for any x', we have via the chain rule for entropy that

$$\begin{aligned} H(X, C \mid X' = x', M = m) &\\ &= H(X \mid C, X' = x', M = m) + H(C \mid X = x', M = m) \\ &= \left(1 - \Pr[\bar{C} \mid X' = x', M = m']\right) \cdot H(X \mid C, X' = x', M = m) \\ &\quad + \Pr[\bar{C} \mid X' = x', M = m'] \cdot H(X \mid C, X' = x', M = m) \\ &\quad + H(C \mid X = x', M = m) \\ &\leq \left(1 - \Pr[\bar{C} \mid X' = x', M = m']\right) \cdot \log N_{\max} \\ &\quad + \Pr[\bar{C} \mid X' = x', M = m'] \cdot \log |\mathcal{X}| \\ &\quad + 1. \end{aligned}$$

[5] We assume $\Pi(m, y)$ is deterministic. This is without loss of generality, since given any m, y pair there is a Bayes-optimal answer. Converting a randomized protocol to a Bayes-optimal one will not decrease the error and leaves $I(M; X)$ unaffected.

Take the expectation with respect to $X' \mid M = m$ and rearrange, getting

$$\Pr[\bar{C} \mid M = m] \geq \frac{H(X \mid M = m) - \log N_{\max} - 1}{\log |\mathcal{X}| - \log N_{\max}}.$$

Combining with (89) and (90), we get

$$2 \Pr[\text{error} \mid M = m] \geq \Pr[D \mid \bar{C}] \cdot \left(\frac{H(X \mid M = m) - \log N_{\max} - 1}{\log |\mathcal{X}| - \log N_{\max}} \right).$$

Then, since both sides are linear, we take the expectation over M and rearrange to bound the conditional entropy.

$$H(X \mid M) \leq \log N_{max} + 1 + \frac{2\epsilon}{\Pr[D \mid \bar{C}]} (\log |\mathcal{X}| - \log N_{max}).$$

With $I(M; X) = H(X) - H(X \mid M)$, this turns into a lower bound on mutual information. $\square$

Application to Gap Hamming We now return to Gap-Hamming. We define a notion of compatibility, upper bound $N_{\max}$, and lower bound α.

Proof of Lemma G.2. To apply Lemma G.3, we must define a notion of compatibility for Alice's inputs, compute $N_{\max}$ to bound the number of compatible inputs, and lower bound the probability that a given y distinguishes incompatible inputs. We will say

$$x \text{ and } x' \text{ are compatible if } d_H(x, x') \leq pd.$$

Here. $p \in (0, 1)$ is a parameter to be set later. For each x, then, the number of compatible x' is the volume of the Hamming ball of radius pd, so we can bound $N_{\max} \leq 2^{d \cdot h(p)}$, yielding

$$\frac{\log N_{\max}}{H(X)} \leq h(p).$$

We now lower-bound α, the probability that Y distinguishes X and X', assuming X, X' agree in at most $d(1-p)$ locations. We need Y to be close to one of X, X' and far from the other. These Hamming distances are the result of independent fair coin flips, so we define the following random variables,

$$A \sim \text{Bin}(d(1-p), 1/2), \quad B \sim \text{Bin}(pd, 1/2)$$

and note that

$$\begin{aligned} d(X, Y) &= A + B \\ d(X', Y) &= A + pd - B. \end{aligned}$$

Define the following independent good events, introducing parameter $c' > 0$:

$$G_A = \left\{ \left| A - \frac{(1-p)d}{2} \right| \leq c' \sqrt{d} \right\} \quad \text{and} \quad G_B = \left\{ \left| B - \frac{pd}{2} \right| \geq (c + c') \sqrt{d} \right\}.$$

When both occur, Y distinguishes X, X'. Assuming B takes a value in its lower tail,

$$d(X,Y) = A + B \leq \left(\frac{(1-p)d}{2} + c'\sqrt{d}\right) + \left(\frac{pd}{2} - (c+c')\sqrt{d}\right) \qquad = \frac{d}{2} - c\sqrt{d}$$
$$d(X',Y) = A + pd - B \geq \left(\frac{(1-p)d}{2} - c'\sqrt{d}\right) + pd - \left(\frac{pd}{2} - (c+c')\sqrt{d}\right) \quad = \frac{d}{2} + c\sqrt{d}$$

We lower bound $\Pr[G_A \cap G_B] = \Pr[G_A]\Pr[G_B]$ using the Berry-Esséen Theorem, stated below. Define scaled

$$Z_A = \frac{A - d(1-p)/2}{\sqrt{d(1-p)/4}} \quad \text{and} \quad Z_B = \frac{B - pd/2}{\sqrt{pd/4}}.$$

Then

$$\begin{aligned} G_A &\Leftrightarrow \left|A - \frac{d(1-p)}{2}\right| \leq c'\sqrt{d} \\ &\Leftrightarrow \left|Z_A\sqrt{d(1-p)/4} + \frac{d(1-p)}{2} - \frac{d(1-p)}{2}\right| \leq c'\sqrt{d} \\ &\Leftrightarrow |Z_A| \leq \frac{2c'}{\sqrt{1-p}}. \end{aligned}$$

So

$$Pr[G_A] \geq 1 - 2F_{\mathcal{N}(0,1)}\left(\frac{-2c'}{\sqrt{1-p}}\right) - \mathcal{O}\left(\frac{1}{\sqrt{d(1-p)}}\right)$$

and, similarly,

$$Pr[G_B] \geq 2F_{\mathcal{N}(0,1)}\left(\frac{-2(c+c')}{\sqrt{p}}\right) - \mathcal{O}\left(\frac{1}{\sqrt{dp}}\right).$$

Taking $c' = \frac{c}{2}$ yields the statement. □

Theorem G.4 (Berry-Esséen)**.** *If $X \sim \mathrm{Bin}(n, 1/2)$ and we have the scaled version $Z = \frac{X - n/2}{\sqrt{n/4}}$, then for all $a \in \mathbb{R}$*

$$\left|\Pr\left[Z \leq a\right] - F_{\mathcal{N}(0,1)}(a)\right| = \mathcal{O}\left(\frac{1}{\sqrt{n}}\right),$$

where $F_{\mathcal{N}(0,1)}(a)$ is the CDF of the unit Gaussian evaluated at a.

Completing the Proof Lemma G.2 holds for all values of p. We show how to set p (as a function of c, d, and ϵ) so that the information complexity is $(1 - o(1)) \cdot d$.

Proof of Lemma G.2. Recall that we have

$$I(M;X) \geq H(X) \cdot (1 - h(p)) \cdot (1 - 2\epsilon/\alpha) - 1,$$

with α:

$$\alpha \geq \left(1 - 2F_{\mathcal{N}(0,1)}\left(\frac{-c}{\sqrt{1-p}}\right) - \mathcal{O}\left(\frac{1}{\sqrt{d(1-p)}}\right)\right)\left(2F_{\mathcal{N}(0,1)}\left(\frac{-3c}{\sqrt{p}}\right) - \mathcal{O}\left(\frac{1}{\sqrt{dp}}\right)\right).$$

We wish to lower bound $I(X;M)$ as $\epsilon \to 0$ and $d \to \infty$. We will set $p = \frac{c_1}{\ln(c_2/\epsilon)}$ for some constants c_1, c_2 that will depend on c but not ϵ or d. Thus $p \xrightarrow{\epsilon \to 0} 0$, so the binary entropy function will go to zero. Furthermore, if $p \to 0$, then for any c the first term $(1 - 2F(\cdot) - O(\cdot))$ will be $\Omega(1)$. Thus it remains to deal with the right term. We will show

$$2F_{\mathcal{N}(0,1)}\left(\frac{-3c}{\sqrt{p}}\right) - \mathcal{O}\left(\frac{1}{\sqrt{dp}}\right) = \Omega(\sqrt{\epsilon}).$$

We can assume without loss of generality that $dp \to \infty$; if ϵ gets too small, we can set p larger until d "catches up," since any algorithm erring with small probability also satisfies a looser probability bound. To finish, then, we just need a simple lower bound on the CDF. For sufficiently small p, we can use the rectangle of width 2 whose left side sits at $\frac{-6c}{\sqrt{p}}$, so set

$$2F_{\mathcal{N}(0,1)}\left(\frac{-3c}{\sqrt{p}}\right) \geq 2\sqrt{2}\pi e^{-9c^2/p}.$$

There exist constants c_1, c_2 such that setting $p = \frac{c_1}{\ln(c_2/\epsilon)}$ will allow us to lower bound this term with $\alpha = \Omega(\sqrt{\epsilon})$, causing the $\frac{2\epsilon}{\alpha}$ term above above to vanish. □

Acknowledgments

We thank Ankit Garg for helpful conversations about the information complexity of the Gap-Hamming problem.

References

[1] Alexander A Alemi. Variational predictive information bottleneck. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–6. PMLR, 2020.

[2] Noga Alon, Roi Livni, Maryanthe Malliaris, and Shay Moran. Private PAC learning implies finite littlestone dimension. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 852–860, 2019.

[3] Devansh Arpit, Stanislaw Jastrzkebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 233–242. JMLR. org, 2017.

[4] Ziv Bar-Yossef, Thathachar S Jayram, Ravi Kumar, and D Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.

[5] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 464–473. IEEE, 2014.

[6] Raef Bassily, Shay Moran, Ido Nachum, Jonathan Shafer, and Amir Yehudayoff. Learners that use little information. In *Algorithmic Learning Theory*, pages 25–55. PMLR, 2018.

[7] Amos Beimel, Kobbi Nissim, and Uri Stemmer. Characterizing the sample complexity of pure private learners. *Journal of Machine Learning Research*, 20(146):1–33, 2019.

[8] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the sulq framework. In Chen Li, editor, *Proceedings of the Twenty-fourth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 13-15, 2005, Baltimore, Maryland, USA*, pages 128–138. ACM, 2005. doi: 10.1145/1065167.1065184. URL `https://doi.org/10.1145/1065167.1065184`.

[9] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[10] Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

[11] Mark Bun, Jonathan Ullman, and Salil Vadhan. Fingerprinting codes and the price of approximate differential privacy. *SIAM Journal on Computing*, 47(5):1888–1938, 2018.

[12] Nicholas Carlini, Chang Liu, Jernej Kos, Úlfar Erlingsson, and Dawn Song. The secret sharer: Measuring unintended neural network memorization & extracting secrets. 2018.

[13] Amit Chakrabarti and Oded Regev. An optimal lower bound on the communication complexity of gap-hamming-distance. *SIAM Journal on Computing*, 41(5):1299–1317, 2012.

[14] Albert Cheu and Jonathan Ullman. The limits of pan privacy and shuffle privacy for learning and estimation. *arXiv preprint arXiv:2009.08000*, 2020.

[15] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 202–210, 2003.

[16] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006. doi: 10.1007/11681878_14. URL `https://doi.org/10.1007/11681878_14`.

[17] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toni Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. In *Advances in Neural Information Processing Systems*, pages 2350–2358, 2015.

[18] Cynthia Dwork, Adam Smith, Thomas Steinke, and Jonathan Ullman. Exposed! a survey of attacks on private data. 2017.

[19] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–959, 2020.

[20] Vitaly Feldman and David Xiao. In *Conference on Learning Theory*, pages 1000–1019, 2014.

[21] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems*, 33, 2020.

[22] Sumegha Garg, Ran Raz, and Avishay Tal. Extractor-based time-space lower bounds for learning. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 990–1002, 2018.

[23] Uri Hadar, Jingbo Liu, Yury Polyanskiy, and Ofer Shayevitz. Communication complexity of estimating correlations. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 792–803, 2019.

[24] Dan Jurafsky and James H Martin. Speech and language processing. vol. 3, 2014.

[25] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[26] Siyuan Ma, Raef Bassily, and Mikhail Belkin. The power of interpolation: Understanding the effectiveness of sgd in modern over-parametrized learning. In *International Conference on Machine Learning*, pages 3325–3334. PMLR, 2018.

[27] Andrew McGregor, Ilya Mironov, Toniann Pitassi, Omer Reingold, Kunal Talwar, and Salil P. Vadhan. The limits of two-party differential privacy. In *FOCS*, pages 81–90, 2010.

[28] Michael Mitzenmacher and Eli Upfal. *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press, 2017.

[29] Ido Nachum and Amir Yehudayoff. Average-case information complexity of learning. In *Algorithmic Learning Theory*, pages 633–646. PMLR, 2019.

[30] Ido Nachum, Jonathan Shafer, and Amir Yehudayoff. A direct sum result for the information complexity of learning. *arXiv preprint arXiv:1804.05474*, 2018.

[31] Adityanarayanan Radhakrishnan, Mikhail Belkin, and Caroline Uhler. Overparameterized neural networks can implement associative memory. *arXiv preprint arXiv:1909.12362*, 2019.

[32] Anup Rao and Amir Yehudayoff. *Communication Complexity: and Applications*. Cambridge University Press, 2020.

[33] Ran Raz. Fast learning requires good memory: A time-space lower bound for parity learning. *Journal of the ACM (JACM)*, 66(1):1–18, 2018.

[34] Ryan Rogers, Aaron Roth, Adam Smith, and Om Thakkar. Max-information, differential privacy, and post-selection hypothesis testing. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 487–494. IEEE, 2016.

[35] Mert Saglam and Gábor Tardos. On the communication complexity of sparse set disjointness and exists-equal problems. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 678–687. IEEE, 2013.

[36] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017.

[37] Jacob Steinhardt, Gregory Valiant, and Stefan Wager. Memory, communication, and statistical queries. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, volume 49 of *JMLR Workshop and Conference Proceedings*, pages 1490–1516. JMLR.org, 2016. URL `http://proceedings.mlr.press/v49/steinhardt16.html`.

[38] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE, 2015.

[39] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

[40] Chulhee Yun, Suvrit Sra, and Ali Jadbabaie. Small relu networks are powerful memorizers: a tight analysis of memorization capacity. In *Advances in Neural Information Processing Systems*, pages 15558–15569, 2019.

[41] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[42] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Michael C Mozer, and Yoram Singer. Identity crisis: Memorization and generalization under extreme overparameterization. *arXiv preprint arXiv:1902.04698*, 2019.