Generalization in Adaptive Data Analysis and Holdout Reuse

Cynthia Dwork[*]   Vitaly Feldman[†]   Moritz Hardt[‡]   Toniann Pitassi[§]

Omer Reingold[¶]   Aaron Roth[‖]

January 29, 2018

Abstract

Overfitting is the bane of data analysts, even when data are plentiful. Formal approaches to understanding this problem focus on statistical inference and generalization of individual analysis procedures. Yet the practice of data analysis is an inherently interactive and adaptive process: new analyses and hypotheses are proposed after seeing the results of previous ones, parameters are tuned on the basis of obtained results, and datasets are shared and reused. An investigation of this gap has recently been initiated by the authors in [DFH+14], where we focused on the problem of estimating expectations of adaptively chosen functions.

In this paper, we give a simple and practical method for reusing a holdout (or testing) set to validate the accuracy of hypotheses produced by a learning algorithm operating on a training set. Reusing a holdout set adaptively multiple times can easily lead to overfitting to the holdout set itself. We give an algorithm that enables the validation of a large number of adaptively chosen hypotheses, while provably avoiding overfitting. We illustrate the advantages of our algorithm over the standard use of the holdout set via a simple synthetic experiment.

We also formalize and address the general problem of data reuse in adaptive data analysis. We show how the differential-privacy based approach given in [DFH+14] is applicable much more broadly to adaptive data analysis. We then show that a simple approach based on description length can also be used to give guarantees of statistical validity in adaptive settings. Finally, we demonstrate that these incomparable approaches can be unified via the notion of approximate max-information that we introduce. This, in particular, allows the preservation of statistical validity guarantees even when an analyst adaptively composes algorithms which have guarantees based on either of the two approaches.

1 Introduction

The goal of machine learning is to produce hypotheses or models that generalize well to the unseen instances of the problem. More generally, statistical data analysis is concerned with estimating properties of the underlying data distribution, rather than properties that are specific to the finite data set at hand. Indeed, a large body of theoretical and empirical research was developed for ensuring generalization in a variety of settings. In this work, it is commonly assumed that each analysis procedure (such as a learning algorithm) operates on a freshly sampled dataset – or if not, is validated on a freshly sampled *holdout* (or testing) set.

Unfortunately, learning and inference can be more difficult in practice, where data samples are often reused. For example, a common practice is to perform feature selection on a dataset, and then use the features for some supervised learning task. When these two steps are performed on the same dataset, it is no

---

[*]Microsoft Research

[†]IBM Almaden Research Center. Part of this work done while visiting the Simons Institute, UC Berkeley

[‡]Google Research

[§]University of Toronto

[¶]Samsung Research America

[‖]Department of Computer and Information Science, University of Pennsylvania

longer clear that the results obtained from the combined algorithm will generalize. Although not usually understood in these terms, "Freedman's paradox" is an elegant demonstration of the powerful (negative) effect of adaptive analysis on the same data [Fre83]. In Freedman's simulation, variables with significant t-statistic are selected and linear regression is performed on this adaptively chosen subset of variables, with famously misleading results: when the relationship between the dependent and explanatory variables is non-existent, the procedure overfits, erroneously declaring significant relationships.

Most of machine learning practice does not rely on formal guarantees of generalization for learning algorithms. Instead a dataset is split randomly into two (or sometimes more) parts: the training set and the testing, or holdout, set. The training set is used for learning a predictor, and then the holdout set is used to estimate the accuracy of the predictor on the true distribution (Additional averaging over different partitions is used in cross-validation.). Because the predictor is independent of the holdout dataset, such an estimate is a valid estimate of the true prediction accuracy (formally, this allows one to construct a confidence interval for the prediction accuracy on the data distribution). However, in practice the holdout dataset is rarely used only once, and as a result the predictor may not be independent of the holdout set, resulting in overfitting to the holdout set [Reu03, RF08, CT10]. One well-known reason for such dependence is that the holdout data is used to test a large number of predictors and only the best one is reported. If the set of all tested hypotheses is known and independent of the holdout set, then it is easy to account for such multiple testing or use the more sophisticated approach of Ng [Ng97].

However such static approaches do not apply if the estimates or hypotheses tested on the holdout are chosen adaptively: that is, if the choice of hypotheses depends on previous analyses performed on the dataset. One prominent example in which a holdout set is often adaptively reused is hyperparameter tuning (*e.g.*[DFN07]). Similarly, the holdout set in a machine learning competition, such as the famous ImageNet competition, is typically reused many times adaptively. Other examples include using the holdout set for feature selection, generation of base learners (in aggregation techniques such as boosting and bagging), checking a stopping condition, and analyst-in-the-loop decisions. See [Lan05] for a discussion of several subtle causes of overfitting.

The concrete practical problem we address is how to ensure that the holdout set can be reused to perform validation in the adaptive setting. Towards addressing this problem we also ask the more general question of how one can ensure that the final output of adaptive data analysis generalizes to the underlying data distribution. This line of research was recently initiated by the authors in [DFH+14], where we focused on the case of estimating expectations of functions from i.i.d. samples (these are also referred to as statistical queries). Using a connection to differential privacy we gave an algorithm for answering a large number of adaptively chosen statistical queries (see Sec. 1.3 and Sec. 2.2 for more details).

1.1 Our Results

We propose a simple and general formulation of the problem of preserving statistical validity in adaptive data analysis. We show that the connection between differentially private algorithms and generalization from [DFH+14] can be extended to this more general setting, and show that similar (but sometimes incomparable) guarantees can be obtained from algorithms whose outputs can be described by short strings. We then define a new notion, *approximate max-information*, that unifies these two basic techniques and gives a new perspective on the problem. In particular, we give an adaptive composition theorem for max-information, which gives a simple way to obtain generalization guarantees for analyses in which some of the procedures are differentially private and some have short description length outputs. We apply our techniques to the problem of reusing the holdout set for validation in the adaptive setting.

1.1.1 A Reusable Holdout

We describe a simple and general method, together with two specific instantiations, for *reusing* a holdout set for validating results while provably avoiding overfitting to the holdout set. The analyst can perform any analysis on the training dataset, but can only access the holdout set via an algorithm that allows the analyst to validate her hypotheses against the holdout set. Crucially, our algorithm prevents overfitting to the

holdout set even when the analyst's hypotheses are chosen adaptively on the basis of the previous responses of our algorithm.

Our first algorithm, referred to as Thresholdout, derives its guarantees from differential privacy and the results in [DFH$^+$14, NS15]. For any function $\phi : \mathcal{X} \to [0, 1]$ given by the analyst, Thresholdout uses the holdout set to validate that ϕ does not overfit to the training set, that is, it checks that the mean value of ϕ evaluated on the training set is close to the mean value of ϕ evaluated on the distribution $\mathcal{P}$ from which the data was sampled. The standard approach to such validation would be to compute the mean value of ϕ on the holdout set. The use of the holdout set in Thresholdout differs from the standard use in that it exposes very little information about the mean of ϕ on the holdout set: if ϕ does not overfit to the training set, then the analyst receives only the confirmation of closeness, that is, just a single bit. On the other hand, if ϕ overfits then Thresholdout returns the mean value of ϕ on the training set perturbed by carefully calibrated noise.

Using results from [DFH$^+$14, NS15] we show that for datasets consisting of i.i.d. samples these modifications provably prevent the analyst from constructing functions that overfit to the holdout set. This ensures correctness of Thresholdout's responses. Naturally, the specific guarantees depend on the number of samples n in the holdout set. The number of queries that Thresholdout can answer is exponential in n as long as the number of times that the analyst overfits is at most quadratic in n.

Our second algorithm SparseValidate is based on the idea that if most of the time the analysts procedures generate results that do not overfit, then validating them against the holdout set does not reveal much information about the holdout set. Specifically, the generalization guarantees of this method follow from the observation that the transcript of the interaction between a data analyst and the holdout set can be described concisely. More formally, this method allows the analyst to pick any Boolean function of a dataset ψ (described by an algorithm) and receive back its value on the holdout set. A simple example of such a function would be whether the accuracy of a predictor on the holdout set is at least a certain value α. (Unlike in the case of Thresholdout, here there is no need to assume that the function that measures the accuracy has a bounded range or even Lipschitz, making it qualitatively different from the kinds of results achievable subject to differential privacy). A more involved example of validation would be to run an algorithm on the holdout dataset to select an hypothesis and check if the hypothesis is similar to that obtained on the training set (for any desired notion of similarity). Such validation can be applied to other results of analysis; for example one could check if the variables selected on the holdout set have large overlap with those selected on the training set. An instantiation of the SparseValidate algorithm has already been applied to the problem of answering statistical (and more general) queries in the adaptive setting [BSSU15]. We describe the formal guarantees for SparseValidate in Section 4.2.

In Section 5 we describe a simple experiment on synthetic data that illustrates the danger of reusing a standard holdout set, and how this issue can be resolved by our reusable holdout. The design of this experiment is inspired by Freedman's classical experiment, which demonstrated the dangers of performing variable selection and regression on the same data [Fre83].

1.2 Generalization in Adaptive Data Analysis

We view adaptive analysis on the same dataset as an execution of a sequence of steps $\mathcal{A}_1 \to \mathcal{A}_2 \to \cdots \to \mathcal{A}_m$. Each step is described by an algorithm $\mathcal{A}_i$ that takes as input a fixed dataset $S = (x_1, \ldots, x_n)$ drawn from some distribution $\mathcal{D}$ over $\mathcal{X}^n$, which remains unchanged over the course of the analysis. Each algorithm A_i also takes as input the outputs of the previously run algorithms $\mathcal{A}_1$ through $\mathcal{A}_{i-1}$ and produces a value in some range $\mathcal{Y}_i$. The dependence on previous outputs represents all the adaptive choices that are made at step i of data analysis. For example, depending on the previous outputs, $\mathcal{A}_i$ can run different types of analysis on S. We note that at this level of generality, the algorithms can represent the choices of the data analyst, and need not be explicitly specified. We assume that the analyst uses algorithms which *individually* are known to generalize when executed on a fresh dataset sampled independently from a distribution $\mathcal{D}$. We formalize this by assuming that for every fixed value $y_1, \ldots, y_{i-1} \in \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_{i-1}$, with probability at least $1 - \beta_i$ over the choice of S according to distribution $\mathcal{D}$, the output of $\mathcal{A}_i$ on inputs $y_1, \ldots, y_{i-1}$ and S has a desired property relative to the data distribution $\mathcal{D}$ (for example has low generalization error). Note that in this assumption $y_1, \ldots, y_{i-1}$ are fixed and independent of the choice of S, whereas the analyst will execute $\mathcal{A}_i$ on

values $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_{i-1}$, where $\boldsymbol{Y}_j = \mathcal{A}_j(S, \boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_{j-1})$. In other words, in the adaptive setup, the algorithm $\mathcal{A}_i$ can depend on the previous outputs, which depend on S, and thus the set S given to $\mathcal{A}_i$ is no longer an independently sampled dataset. Such dependence invalidates the generalization guarantees of individual procedures, potentially leading to overfitting.

Differential privacy: First, we spell out how the differential privacy based approach from [DFH$^+$14] can be applied to this more general setting. Specifically, a simple corollary of results in [DFH$^+$14] is that for a dataset consisting of i.i.d. samples any output of a differentially-private algorithm can be used in subsequent analysis while controlling the risk of overfitting, even beyond the setting of statistical queries studied in [DFH$^+$14]. A key property of differential privacy in this context is that it composes adaptively: namely if each of the algorithms used by the analyst is differentially private, then the whole procedure will be differentially private (albeit with worse privacy parameters). Therefore, one way to avoid overfitting in the adaptive setting is to use algorithms that satisfy (sufficiently strong) guarantees of differential-privacy. In Section 2.2 we describe this result formally.

Description length: We then show how description length bounds can be applied in the context of guaranteeing generalization in the presence of adaptivity. If the total length of the outputs of algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_{i-1}$ can be described with k bits then there are at most 2^k possible values of the input $y_1, \ldots, y_{i-1}$ to $\mathcal{A}_i$. For each of these individual inputs $\mathcal{A}_i$ generalizes with probability $1 - \beta_i$. Taking a union bound over failure probabilities implies generalization with probability at least $1 - 2^k \beta_i$. Occam's Razor famously implies that shorter hypotheses have lower generalization error. Our observation is that shorter hypotheses (and the results of analysis more generally) are also better in the adaptive setting since they reveal less about the dataset and lead to better generalization of *subsequent* analyses. Note that this result makes no assumptions about the data distribution $\mathcal{D}$. We provide the formal details in Section 2.3. In Section B we also show that description length-based analysis suffices for obtaining an algorithm (albeit not an efficient one) that can answer an exponentially large number of adaptively chosen statistical queries. This provides an alternative proof for one of the results in [DFH$^+$14].

Approximate max-information: Our main technical contribution is the introduction and analysis of a new information-theoretic measure, which unifies the generalization arguments that come from both differential privacy and description length, and that quantifies how much information has been learned about the data by the analyst. Formally, for jointly distributed random variables $(\boldsymbol{S}, \boldsymbol{Y})$, the max-information is the maximum of the logarithm of the factor by which uncertainty about $\boldsymbol{S}$ is reduced given the value of $\boldsymbol{Y}$, namely $I_\infty(\boldsymbol{S}, \boldsymbol{Y}) \doteq \log \max \frac{\mathbb{P}[\boldsymbol{S}=S \mid \boldsymbol{Y}=y]}{\mathbb{P}[\boldsymbol{S}=S]}$, where the maximum is taken over all S in the support of $\boldsymbol{S}$ and y in the support $\boldsymbol{Y}$. Informally, β-approximate max-information requires that the logarithm above be bounded with probability at least $1 - \beta$ over the choice of $(\boldsymbol{S}, \boldsymbol{Y})$ (the actual definition is slightly weaker, see Definition 10 for details). In our use, $\boldsymbol{S}$ denotes a dataset drawn randomly from the distribution $\mathcal{D}$ and $\boldsymbol{Y}$ denotes the output of a (possibly randomized) algorithm on $\boldsymbol{S}$. We prove that approximate max-information has the following properties

- An upper bound on (approximate) max-information gives generalization guarantees.

- Differentially private algorithms have low max-information for any distribution $\mathcal{D}$ over datasets. A stronger bound holds for approximate max-information on i.i.d. datasets. These bounds apply only to so-called pure differential privacy (the $\delta = 0$ case).

- Bounds on the description length of the output of an algorithm give bounds on the approximate max-information of the algorithm for any $\mathcal{D}$.

- Approximate max-information composes adaptively.

- Approximate max-information is preserved under post-processing.

4

Composition properties of approximate max-information imply that one can easily obtain generalization guarantees for adaptive sequences of algorithms, some of which are differentially private, and others of which have outputs with short description length. These properties also imply that differential privacy can be used to control generalization for any distribution $\mathcal{D}$ over datasets, which extends its generalization guarantees beyond the restriction to datasets drawn i.i.d. from a fixed distribution, as in [DFH$^+$14].

We remark that (pure) differential privacy and description length are otherwise incomparable – low description length is *not* a sufficient condition for differential privacy, since differential privacy precludes revealing even a small number of bits of information about any single individual in the data set. At the same time differential privacy does not constrain the description length of the output. Bounds on max-information or differential privacy of an algorithm can, however, be translated to bounds on *randomized description length* for a different algorithm with statistically indistinguishable output. Here we say that a randomized algorithm has randomized description length of k if for every fixing of the algorithm's random bits, it has description length of k. Details of these results and additional discussion appear in Sections 3 and A.

1.3 Related Work

This work builds on [DFH$^+$14] where we initiated the formal study of adaptivity in data analysis. The primary focus of [DFH$^+$14] is the problem of answering adaptively chosen statistical queries. The main technique is a strong connection between differential privacy and generalization: differential privacy guarantees that the distribution of outputs does not depend too much on any one of the data samples, and thus, differential privacy gives a strong stability guarantee that behaves well under adaptive data analysis. The link between generalization and approximate differential privacy made in [DFH$^+$14] has been subsequently strengthened, both qualitatively — by [BSSU15], who make the connection for a broader range of queries — and quantitatively, by [NS15] and [BSSU15], who give tighter quantitative bounds. These papers, among other results, give methods for accurately answering exponentially (in the dataset size) many adaptively chosen queries, but the algorithms for this task are not efficient. It turns out this is for fundamental reasons – Hardt and Ullman [HU14] and Steinke and Ullman [SU14] prove that, under cryptographic assumptions, no efficient algorithm can answer more than quadratically many statistical queries chosen adaptively by an adversary who knows the true data distribution.

Differential privacy emerged from a line of work [DN03, DN04, BDMN05], culminating in the definition given by [DMNS06]. There is a very large body of work designing differentially private algorithms for various data analysis tasks, some of which we leverage in our applications. See [Dwo11] for a short survey and [DR14] for a textbook introduction to differential privacy.

The classical approach in theoretical machine learning to ensure that empirical estimates generalize to the underlying distribution is based on the various notions of complexity of the set of functions output by the algorithm, most notably the VC dimension (see e.g. [SSBD14] for a textbook introduction). If one has a sample of data large enough to guarantee generalization for all functions in some class of bounded complexity, then it does not matter whether the data analyst chooses functions in this class adaptively or non-adaptively. Our goal, in contrast, is to prove generalization bounds *without* making any assumptions about the class from which the analyst can output functions. In this case the adaptive setting is very different from the non-adaptive setting.

An important line of work [BE02, PRMN04, SSSSS10] establishes connections between the *stability* of a learning algorithm and its ability to generalize. Stability is a measure of how much the output of a learning algorithm is perturbed by changes to its input. It is known that certain stability notions are necessary and sufficient for generalization. Unfortunately, the stability notions considered in these prior works do not compose in the sense that running multiple stable algorithms sequentially and adaptively may result in a procedure that is not stable. The measure we introduce in this work (max information), like differential privacy, has the strength that it enjoys adaptive composition guarantees. This makes it amenable to reasoning about the generalization properties of adaptively applied sequences of algorithms, while having to analyze only the individual components of these algorithms. Connections between stability, empirical risk minimization and differential privacy in the context of learnability have been recently explored in [WLF15].

Numerous techniques have been developed by statisticians to address common special cases of adaptive data analysis. Most of them address a single round of adaptivity such as variable selection followed by regression on selected variables or model selection followed by testing and are optimized for specific inference procedures (the literature is too vast to adequately cover here, see Ch. 7 in [HTF09] for a textbook introduction and [TT15] for a survey of some recent work). In contrast, our framework addresses multiple stages of adaptive decisions, possible lack of a predetermined analysis protocol and is not restricted to any specific procedures.

Freund gives an approach to obtaining data-dependent generalization bounds that takes into account the set of statistical queries that a given learning algorithm can produce for the distribution from which the data was sampled [Fre98]. A related approach of Langford and Blum also allows to obtain data-dependent generalization bounds based on the description length of functions that can be output for a data distribution [LB03]. Unlike our work, these approaches require the knowledge of the structure of the learning algorithm to derive a generalization bound. More importantly, the focus of our framework is on the design of *new* algorithms with better generalization properties in the adaptive setting.

Finally, inspired by our work, Blum and Hardt [BH15] showed how to reuse the holdout set to maintain an accurate leaderboard in a machine learning competition that allows the participants to submit adaptively chosen models in the process of the competition (such as those organized by Kaggle Inc.). Their analysis also relies on the description length-based technique we used to analyze SparseValidate.

2 Preliminaries and Basic Techniques

In the discussion below log refers to binary logarithm and ln refers to the natural logarithm. For simplicity we restrict our random variables to finite domains (extension of the claims to continuous domains is straightforward using the standard formalism). For two random variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ over the same domain $\mathcal{X}$ the max-divergence of $\boldsymbol{X}$ from $\boldsymbol{Y}$ is defined as

$$D_\infty(\boldsymbol{X}\|\boldsymbol{Y}) = \log \max_{x \in \mathcal{X}} \frac{\mathbb{P}[\boldsymbol{X} = x]}{\mathbb{P}[\boldsymbol{Y} = x]}.$$

δ-approximate max-divergence is defined as

$$D_\infty^\delta(\boldsymbol{X}\|\boldsymbol{Y}) = \log \max_{\mathcal{O} \subseteq \mathcal{X},\ \mathbb{P}[\boldsymbol{X} \in \mathcal{O}] > \delta} \frac{\mathbb{P}[\boldsymbol{X} \in \mathcal{O}] - \delta}{\mathbb{P}[\boldsymbol{Y} \in \mathcal{O}]}.$$

We say that a real-valued function over datasets $f : \mathcal{X}^n \to \mathbb{R}$ has sensitivity c for all $i \in [n]$ and $x_1, x_2, \ldots, x_n, x_i' \in \mathcal{X}$, $f(x_1, \ldots, x_i, \ldots, x_n) - f(x_1, \ldots, x_i', \ldots, x_n) \le c$. We review McDiarmid's concentration inequality for functions of low-sensitivity.

Lemma 1 (McDiarmid's inequality). *Let $\boldsymbol{X}_1, X_2, \ldots, \boldsymbol{X}_n$ be independent random variables taking values in the set $\mathcal{X}$. Further let $f : \mathcal{X}^n \to \mathbb{R}$ be a function of sensitivity $c > 0$. Then for all $\alpha > 0$, and $\mu = \mathbb{E}\left[f(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n)\right]$,*

$$\mathbb{P}\left[f(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n) - \mu \ge \alpha\right] \le \exp\left(\frac{-2\alpha^2}{n \cdot c^2}\right).$$

For a function $\phi : \mathcal{X} \to \mathbb{R}$ and a dataset $S = (x_1, \ldots, x_n)$, let $\mathcal{E}_S[\phi] \doteq \frac{1}{n} \sum_{i=1}^n \phi(x_i)$. Note that if the range of ϕ is in some interval of length α then $f(S) = \mathcal{E}_S[\phi]$ has sensitivity α/n. For a distribution $\mathcal{P}$ over $\mathcal{X}$ and a function $\phi : \mathcal{X} \to \mathbb{R}$, let $\mathcal{P}[\phi] \doteq \mathbb{E}_{x \sim \mathcal{P}}[\phi(x)]$.

2.1 Differential Privacy

On an intuitive level, differential privacy hides the data of any single individual. We are thus interested in pairs of datasets S, S' that differ in a single element, in which case we say S and S' are *adjacent*.

Definition 2. *[DMNS06, DKM⁺06] A randomized algorithm $\mathcal{A}$ with domain $\mathcal{X}^n$ for $n > 0$ is (ε, δ)-differentially private if for all pairs of datasets that differ in a single element $S, S' \in \mathcal{X}^n$: $D_\infty^\delta(\mathcal{A}(S) \| \mathcal{A}(S')) \leq \log(e^\varepsilon)$. The case when $\delta = 0$ is sometimes referred to as pure differential privacy, and in this case we may say simply that $\mathcal{A}$ is ε-differentially private.*

Differential privacy is preserved under adaptive composition. Adaptive composition of algorithms is a sequential execution of algorithms on the same dataset in which an algorithm at step i can depend on the outputs of previous algorithms. More formally, let $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m$ be a sequence of algorithms. Each algorithm $\mathcal{A}_i$ outputs a value in some range $\mathcal{Y}_i$ and takes as an input dataset in $\mathcal{X}^n$ as well as a value in $\bar{\mathcal{Y}}_{i-1} \doteq \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_{i-1}$. Adaptive composition of these algorithm is the algorithm that takes as an input a dataset $S \in \mathcal{X}^n$ and executes $\mathcal{A}_1 \to \mathcal{A}_2 \to \cdots \to \mathcal{A}_m$ sequentially with the input to $\mathcal{A}_i$ being S and the outputs $y_1, \ldots, y_{i-1}$ of $\mathcal{A}_1, \ldots, \mathcal{A}_{i-1}$. Such composition captures the common practice in data analysis of using the outcomes of previous analyses (that is $y_1, \ldots, y_{i-1}$) to select an algorithm that is executed on S.

For an algorithm that in addition to a dataset has other input we say that it is (ε, δ)-differentially private if it is (ε, δ)-differentially private for every setting of additional parameters. The basic property of adaptive composition of differentially private algorithms is the following result (*e.g.*[DL09]):

Theorem 3. *Let $\mathcal{A}_i : \mathcal{X}^n \times \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_{i-1} \to \mathcal{Y}_i$ be an $(\varepsilon_i, \delta_i)$-differentially private algorithm for $i \in [m]$. Then the algorithm $\mathcal{B} : \mathcal{X}^n \to \mathcal{Y}_m$ obtained by composing $\mathcal{A}_i$'s adaptively is $(\sum_{i=1}^m \varepsilon_i, \sum_{i=1}^m \delta_i)$-differentially private.*

A more sophisticated argument yields significant improvement when $\varepsilon < 1$ (*e.g.*[DR14]).

Theorem 4. *For all $\varepsilon, \delta, \delta' \geq 0$, the adaptive composition of m arbitrary (ε, δ)-differentially private algorithms is $(\varepsilon', m\delta + \delta')$-differentially private, where*

$$\varepsilon' = \sqrt{2m \ln(1/\delta')} \cdot \varepsilon + m\varepsilon(e^\varepsilon - 1).$$

Another property of differential privacy important for our applications is preservation of its guarantee under post-processing (*e.g.*[DR14, Prop. 2.1]):

Lemma 5. *If $\mathcal{A}$ is an (ϵ, δ)-differentially private algorithm with domain $\mathcal{X}^n$ and range $\mathcal{Y}$, and $\mathcal{B}$ is any, possibly randomized, algorithm with domain $\mathcal{Y}$ and range $\mathcal{Y}''$, then the algorithm $\mathcal{B} \circ \mathcal{A}$ with domain $\mathcal{X}^n$ and range $\mathcal{Y}'$ is also (ϵ, δ)-differentially private.*

2.2 Generalization via Differential Privacy

Generalization in special cases of our general adaptive analysis setting can be obtained directly from results in [DFH⁺14] and composition properties of differentially private algorithms. For the case of pure differentially private algorithms with general outputs over i.i.d. datasets, in [DFH⁺14] we prove the following result.

Theorem 6. *Let $\mathcal{A}$ be an ε-differentially private algorithm with range $\mathcal{Y}$ and let $\boldsymbol{S}$ be a random variable drawn from a distribution $\mathcal{P}^n$ over $\mathcal{X}^n$. Let $\boldsymbol{Y} = \mathcal{A}(\boldsymbol{S})$ be the corresponding output distribution. Assume that for each element $y \in \mathcal{Y}$ there is a subset $R(y) \subseteq \mathcal{X}^n$ so that $\max_{y \in \mathcal{Y}} \mathbb{P}[\boldsymbol{S} \in R(y)] \leq \beta$. Then, for $\varepsilon \leq \sqrt{\frac{\ln(1/\beta)}{2n}}$ we have $\mathbb{P}[\boldsymbol{S} \in R(\boldsymbol{Y})] \leq 3\sqrt{\beta}$.*

An immediate corollary of Thm. 6 together with Lemma 1 is that differentially private algorithms that output low-sensitivity functions generalize.

Corollary 7. *Let $\mathcal{A}$ be an algorithm that outputs a c-sensitive function $f : \mathcal{X}^n \to \mathbb{R}$. Let $\boldsymbol{S}$ be a random dataset chosen according to distribution $\mathcal{P}^n$ over $\mathcal{X}^n$ and let $\boldsymbol{f} = \mathcal{A}(\boldsymbol{S})$. If $\mathcal{A}$ is $\tau/(cn)$-differentially private then $\mathbb{P}[\boldsymbol{f}(\boldsymbol{S}) - \mathcal{P}^n[\boldsymbol{f}] \geq \tau] \leq 3 \exp\left(-\tau^2/(c^2 n)\right)$.*

By Theorem 3, pure differential privacy composes adaptively. Therefore, if in a sequence of algorithms $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m$ algorithm $\mathcal{A}_i$ is ε_i-differentially private for all $i \leq m-1$ then composition of the first $i-1$ algorithms is ε'_{i-1}-differentially private for $\varepsilon'_{i-1} = \left(\sum_{j=1}^{i-1} \varepsilon_j \right)$. Theorem 6 can be applied to preserve the generalization guarantees of the last algorithm $\mathcal{A}_m$ (that does not need to be differentially private). For example, assume that for every fixed setting of $\bar{y}_{m-1}$, $\mathcal{A}_m$ has the property that it outputs a hypothesis function h such that, $\mathbb{P}[\mathcal{E}_S[L(h)] - \mathcal{P}[L(h)] \geq \tau] \leq e^{-n\tau^2/d}$, for some notion of dimension d and a real-valued loss function L. Generalization bounds of this type follow from uniform convergence arguments based on various notions of complexity of hypotheses classes such as VC dimension, covering numbers, fat-shattering dimension and Rademacher complexity (see [SSBD14] for examples). Note that, for different settings of $\bar{y}_{m-1}$, different sets of hypotheses and generalization techniques might be used. We define $R(\bar{y}_{m-1})$ be all datasets S for which $\mathcal{A}_m(S, \bar{y}_{m-1})$ outputs h such that $\mathcal{E}_S[L(h)] - \mathcal{P}[L(h)] \geq \tau$. Now if $\varepsilon'_{m-1} \leq \sqrt{\tau^2/(2d)}$, then even for the hypothesis output in the adaptive execution of $\mathcal{A}_m$ on a random i.i.d. dataset $\boldsymbol{S}$ (denoted by $\boldsymbol{h}$) we have $\mathbb{P}[\mathcal{E}_{\boldsymbol{S}}[L(\boldsymbol{h})] - \mathcal{P}[L(\boldsymbol{h})] \geq \tau] \leq 3e^{-\tau^2 n/(2d)}$.

For approximate (ε, δ)-differential privacy, strong preservation of generalization results are currently known only for algorithms that output a function over $\mathcal{X}$ of bounded range (for simplicity we use range $[0,1]$) [DFH+14, NS15]. The following result was proved by Nissim and Stemmer [NS15] (a weaker statement is also given in [DFH+14, Thm. 10]).

Theorem 8. *Let $\mathcal{A}$ be an (ε, δ)-differentially private algorithm that outputs a function from $\mathcal{X}$ to $[0,1]$. For a random variable $\boldsymbol{S}$ distributed according to $\mathcal{P}^n$ we let $\boldsymbol{\phi} = \mathcal{A}(\boldsymbol{S})$. Then for $n \geq 2\ln(8/\delta)/\varepsilon^2$,*

$$\mathbb{P}\left[|\mathcal{P}[\boldsymbol{\phi}] - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}]| \geq 13\varepsilon \right] \leq \frac{2\delta}{\varepsilon} \ln\left(\frac{2}{\varepsilon} \right).$$

Many learning algorithms output a hypothesis function that aims to minimize some bounded loss function L as the final output. If algorithms used in all steps of the adaptive data analysis are differentially private and the last step (that is, $\mathcal{A}_m$) outputs a hypothesis h, then generalization bounds for the loss of h are implied directly by Theorem 8. We remark that this application is different from the example for pure differential privacy above since there we showed preservation of generalization guarantees of arbitrarily complex learning algorithm $\mathcal{A}_m$ which need not be differentially private. In Section 4 we give an application of Theorem 8 to the reusable holdout problem.

2.3 Generalization via Description Length

Let $\mathcal{A} : \mathcal{X}^n \to \mathcal{Y}$ and $\mathcal{B} : \mathcal{X}^n \times \mathcal{Y} \to \mathcal{Y}'$ be two algorithms. We now give a simple application of bounds on the size of $\mathcal{Y}$ (or, equivalently, the description length of $\mathcal{A}$'s output) to preserving generalization of $\mathcal{B}$. Here generalization can actually refer to any valid or desirable output of $\mathcal{B}$ for a given given dataset S and input $y \in \mathcal{Y}$. Specifically we will use a set $R(y) \subseteq \mathcal{X}^n$ to denote all datasets for which the output of $\mathcal{B}$ on y and S is "bad" (*e.g.* overfits). Using a simple union bound we show that the probability (over a random choice of a dataset) of such bad outcome can be bounded.

Theorem 9. *Let $\mathcal{A} : \mathcal{X}^n \to \mathcal{Y}$ be an algorithm and let $\boldsymbol{S}$ be a random dataset over $\mathcal{X}^n$. Assume that $R : \mathcal{Y} \to 2^{\mathcal{X}^n}$ is such that for every $y \in \mathcal{Y}$, $\mathbb{P}[\boldsymbol{S} \in R(y)] \leq \beta$. Then $\mathbb{P}[\boldsymbol{S} \in R(\mathcal{A}(\boldsymbol{S}))] \leq |\mathcal{Y}| \cdot \beta$.*

Proof.
$$\mathbb{P}[\boldsymbol{S} \in R(\mathcal{A}(\boldsymbol{S}))] \leq \sum_{y \in \mathcal{Y}} \mathbb{P}[\boldsymbol{S} \in R(y)] \leq |\mathcal{Y}| \cdot \beta.$$

$\square$

The case of two algorithms implies the general case since description length composes (adaptively). Namely, let $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m$ be a sequence of algorithms such that each algorithm $\mathcal{A}_i$ outputs a value in some range $\mathcal{Y}_i$ and takes as an input dataset in $\mathcal{X}^n$ as well as a value in $\bar{\mathcal{Y}}_{i-1}$. Then for every i, we can view the execution of $\mathcal{A}_1$ through $\mathcal{A}_{i-1}$ as the first algorithm $\bar{\mathcal{A}}_{i-1}$ with an output in $\bar{\mathcal{Y}}_{i-1}$ and $\mathcal{A}_i$ as the second

algorithm. Theorem 9 implies that if for every setting of $\bar{y}_{i-1} = y_1, \ldots, y_{i-1} \in \bar{\mathcal{Y}}_{i-1}$, $R(\bar{y}_{i-1}) \subseteq \mathcal{X}^n$ satisfies that $\mathbb{P}[\boldsymbol{S} \in R(\bar{y}_{i-1})] \leq \beta_i$ then

$$\mathbb{P}[\boldsymbol{S} \in R(\bar{\mathcal{A}}_{i-1}(\boldsymbol{S}))] \leq |\bar{\mathcal{Y}}_{i-1}| \cdot \beta_i = \prod_{j=1}^{i-1} |\mathcal{Y}_j| \cdot \beta_i.$$

In Section A we describe a generalization of description length bounds to randomized algorithms and show that it possesses the same properties.

3 Max-Information

Consider two algorithms $\mathcal{A} : \mathcal{X}^n \to \mathcal{Y}$ and $\mathcal{B} : \mathcal{X}^n \times \mathcal{Y} \to \mathcal{Y}'$ that are composed adaptively and assume that for every fixed input $y \in \mathcal{Y}$, $\mathcal{B}$ generalizes for all but fraction β of datasets. Here we are speaking of generalization informally: our definitions will support any property of input $y \in \mathcal{Y}$ and dataset S. Intuitively, to preserve generalization of $\mathcal{B}$ we want to make sure that the output of $\mathcal{A}$ does not reveal too much information about the dataset S. We demonstrate that this intuition can be captured via a notion of *max-information* and its relaxation *approximate max-information*.

For two random variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ we use $\boldsymbol{X} \times \boldsymbol{Y}$ to denote the random variable obtained by drawing $\boldsymbol{X}$ and $\boldsymbol{Y}$ independently from their probability distributions.

Definition 10. *Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be jointly distributed random variables. The max-information between $\boldsymbol{X}$ and $\boldsymbol{Y}$, denoted $I_\infty(\boldsymbol{X}; \boldsymbol{Y})$, is the minimal value of k such that for every x in the support of $\boldsymbol{X}$ and y in the support of $\boldsymbol{Y}$ we have $\mathbb{P}[\boldsymbol{X} = x \mid \boldsymbol{Y} = y] \leq 2^k \mathbb{P}[\boldsymbol{X} = x]$. Alternatively, $I_\infty(\boldsymbol{X}; \boldsymbol{Y}) = D_\infty((\boldsymbol{X}, \boldsymbol{Y}) \| \boldsymbol{X} \times \boldsymbol{Y})$. The β-approximate max-information is defined as $I_\infty^\beta(\boldsymbol{X}; \boldsymbol{Y}) = D_\infty^\beta((\boldsymbol{X}, \boldsymbol{Y}) \| \boldsymbol{X} \times \boldsymbol{Y})$.*

It follows immediately from Bayes' rule that for all $\beta \geq 0$, $I_\infty^\beta(\boldsymbol{X}; \boldsymbol{Y}) = I_\infty^\beta(\boldsymbol{Y}; \boldsymbol{X})$. Further, $I_\infty(\boldsymbol{X}; \boldsymbol{Y}) \leq k$ if and only if for all x in the support of $\boldsymbol{X}$, $D_\infty(\boldsymbol{Y} \mid \boldsymbol{X} = x \| \boldsymbol{Y}) \leq k$. Clearly, max-information upper bounds the classical notion of mutual information: $I_\infty(\boldsymbol{X}; \boldsymbol{Y}) \geq I(\boldsymbol{X}; \boldsymbol{Y})$.

In our use $(\boldsymbol{X}, \boldsymbol{Y})$ is going to be a joint distribution $(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S}))$, where $\boldsymbol{S}$ is a random n-element dataset and $\mathcal{A}$ is a (possibly randomized) algorithm taking a dataset as an input. If the output of an algorithm on any distribution $\boldsymbol{S}$ has low approximate max-information then we say that the algorithm has low max-information. More formally:

Definition 11. *We say that an algorithm $\mathcal{A}$ has β-approximate max-information of k if for every distribution $\mathcal{S}$ over n-element datasets, $I_\infty^\beta(\boldsymbol{S}; \mathcal{A}(\boldsymbol{S})) \leq k$, where $\boldsymbol{S}$ is a dataset chosen randomly according to $\mathcal{S}$. We denote this by $I_\infty^\beta(\mathcal{A}, n) \leq k$.*

An alternative way to define the (pure) max-information of an algorithm is using the maximum of the infinity divergence between distributions on two different inputs.

Lemma 12. *Let $\mathcal{A}$ be an algorithm with domain $\mathcal{X}^n$ and range $\mathcal{Y}$. Then $I_\infty(\mathcal{A}, n) = \max_{S, S' \in \mathcal{X}^n} D_\infty(\mathcal{A}(S) \| \mathcal{A}(S'))$.*

Proof. For the first direction let $k = \max_{S, S' \in \mathcal{X}^n} D_\infty(\mathcal{A}(S) \| \mathcal{A}(S'))$. Let $\boldsymbol{S}$ be any random variable over n-element input datasets for $\mathcal{A}$ and let $\boldsymbol{Y}$ be the corresponding output distribution $\boldsymbol{Y} = \mathcal{A}(\boldsymbol{S})$. We will argue that $I_\infty(\boldsymbol{Y}; \boldsymbol{S}) \leq k$, that $I_\infty(\boldsymbol{S}; \boldsymbol{Y}) \leq k$ follows immediately from the Bayes' rule. For every $y \in \mathcal{Y}$, there must exist a dataset S_y such that $\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S_y] \leq \mathbb{P}[\boldsymbol{Y} = y]$. Now, by our assumption, for every S, $\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S] \leq 2^k \cdot \mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S_y]$. We can conclude that for every S and every y, it holds that $\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S] \leq 2^k \mathbb{P}[\boldsymbol{Y} = y]$. This yields $I_\infty(\boldsymbol{Y}; \boldsymbol{S}) \leq k$.

For the other direction let $k = I_\infty(\mathcal{A}, n)$, let $S, S' \in \mathcal{X}^n$ and $y \in \mathcal{Y}$. For $\alpha \in (0, 1)$, let $\boldsymbol{S}$ be the random variable equal to S with probability α and to S' with probability $1 - \alpha$ and let $\boldsymbol{Y} = \mathcal{A}(\boldsymbol{S})$. By our assumption, $I_\infty(\boldsymbol{Y}; \boldsymbol{S}) = I_\infty(\boldsymbol{S}; \boldsymbol{Y}) \leq k$. This gives

$$\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S] \leq 2^k \mathbb{P}[\boldsymbol{Y} = y] \leq 2^k \left(\alpha \mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S] + (1 - \alpha) \mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S'] \right)$$

9

and implies

$$\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S] \leq \frac{2^k(1-\alpha)}{1 - 2^k\alpha} \cdot \mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S'].$$

This holds for every $\alpha > 0$ and therefore

$$\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S] \leq 2^k \cdot \mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S'].$$

Using this inequality for every $y \in \mathcal{Y}$ we obtain $D_\infty(\mathcal{A}(S)\|\mathcal{A}(S')) \leq k$. $\qquad\square$

Generalization via max-information: An immediate corollary of our definition of approximate max-information is that it controls the probability of "bad events" that can happen as a result of the dependence of $\mathcal{A}(S)$ on S.

Theorem 13. *Let $\boldsymbol{S}$ be a random dataset in $\mathcal{X}^n$ and $\mathcal{A}$ be an algorithm with range $\mathcal{Y}$ such that for some $\beta \geq 0$, $I_\infty^\beta(\boldsymbol{S}; \mathcal{A}(\boldsymbol{S})) = k$. Then for any event $\mathcal{O} \subseteq \mathcal{X}^n \times \mathcal{Y}$,*

$$\mathbb{P}[(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S})) \in \mathcal{O}] \leq 2^k \cdot \mathbb{P}[\boldsymbol{S} \times \mathcal{A}(\boldsymbol{S}) \in \mathcal{O}] + \beta.$$

In particular, $\mathbb{P}[(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S})) \in \mathcal{O}] \leq 2^k \cdot \max_{y \in \mathcal{Y}} \mathbb{P}[(\boldsymbol{S}, y) \in \mathcal{O}] + \beta$.

We remark that mutual information between $\boldsymbol{S}$ and $\mathcal{A}(\boldsymbol{S})$ would not suffice for ensuring that bad events happen with tiny probability. For example mutual information of k allows $\mathbb{P}[(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S})) \in \mathcal{O}]$ to be as high as $k/(2\log(1/\delta))$, where $\delta = \mathbb{P}[\boldsymbol{S} \times \mathcal{A}(\boldsymbol{S}) \in \mathcal{O}]$.

Composition of max-information: Approximate max-information satisfies the following adaptive composition property:

Lemma 14. *Let $\mathcal{A} : \mathcal{X}^n \to \mathcal{Y}$ be an algorithm such that $I_\infty^{\beta_1}(\mathcal{A}, n) \leq k_1$, and let $\mathcal{B} : \mathcal{X}^n \times \mathcal{Y} \to \mathcal{Z}$ be an algorithm such that for every $y \in \mathcal{Y}$, $\mathcal{B}(\cdot, y)$ has β_2-approximate max-information k_2. Let $\mathcal{C} : \mathcal{X}^n \to \mathcal{Z}$ be defined such that $\mathcal{C}(S) = \mathcal{B}(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S}))$. Then $I_\infty^{\beta_1+\beta_2}(\mathcal{C}, n) \leq k_1 + k_2$.*

Proof. Let $\mathcal{D}$ be a distribution over $\mathcal{X}^n$ and $\boldsymbol{S}$ be a random dataset sampled from $\mathcal{D}$. By hypothesis, $I_\infty^{\beta_1}(\boldsymbol{S}; \mathcal{A}(\boldsymbol{S})) \leq k_1$. Expanding out the definition for all $\mathcal{O} \subseteq \mathcal{X}^n \times \mathcal{Y}$:

$$\mathbb{P}[(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S})) \in \mathcal{O}] \leq 2^{k_1} \cdot \mathbb{P}[\boldsymbol{S} \times \mathcal{A}(\boldsymbol{S}) \in \mathcal{O}] + \beta_1 \ .$$

We also have for all $\mathcal{Q} \subseteq \mathcal{X}^n \times \mathcal{Z}$ and for all $y \in \mathcal{Y}$:

$$\mathbb{P}[(\boldsymbol{S}, \mathcal{B}(\boldsymbol{S}, y)) \in \mathcal{Q}] \leq 2^{k_2} \cdot \mathbb{P}[\boldsymbol{S} \times \mathcal{B}(S, y) \in \mathcal{Q}] + \beta_2 \ .$$

For every $\mathcal{O} \subseteq \mathcal{X}^n \times \mathcal{Y}$, define

$$\mu(\mathcal{O}) = \left(\mathbb{P}[(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S})) \in \mathcal{O}] - 2^{k_1} \cdot \mathbb{P}[\boldsymbol{S} \times \mathcal{A}(\boldsymbol{S}) \in \mathcal{O}]\right)_+ \ .$$

Observe that $\mu(\mathcal{O}) \leq \beta_1$ for all $\mathcal{O} \subseteq \mathcal{X}^n \times \mathcal{Y}$. For any event $\mathcal{Q} \subseteq \mathcal{X}^n \times \mathcal{Z}$, we have:

$$
\begin{aligned}
&\mathbb{P}[(\boldsymbol{S}, \mathcal{C}(\boldsymbol{S})) \in \mathcal{Q}] \\
={}& \mathbb{P}[(\boldsymbol{S}, \mathcal{B}(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S}))) \in \mathcal{Q}] \\
={}& \sum_{S \in \mathcal{X}^n, y \in \mathcal{Y}} \mathbb{P}[(S, \mathcal{B}(S, y)) \in \mathcal{Q}] \cdot \mathbb{P}[\boldsymbol{S} = S, \mathcal{A}(\boldsymbol{S}) = y] \\
\leq{}& \sum_{S \in \mathcal{X}^n, y \in \mathcal{Y}} \min\left(\left(2^{k_2} \cdot \mathbb{P}[S \times \mathcal{B}(S, y) \in \mathcal{Q}] + \beta_2\right), 1\right) \cdot \mathbb{P}[\boldsymbol{S} = S, \mathcal{A}(\boldsymbol{S}) = y] \\
\leq{}& \sum_{S \in \mathcal{X}^n, y \in \mathcal{Y}} \left(\min\left(2^{k_2} \cdot \mathbb{P}[S \times \mathcal{B}(S, y) \in \mathcal{Q}], 1\right) + \beta_2\right) \cdot \mathbb{P}[\boldsymbol{S} = S, \mathcal{A}(\boldsymbol{S}) = y] \\
\leq{}& \sum_{S \in \mathcal{X}^n, y \in \mathcal{Y}} \min\left(2^{k_2} \cdot \mathbb{P}[S \times \mathcal{B}(S, y) \in \mathcal{Q}], 1\right) \cdot \mathbb{P}[\boldsymbol{S} = S, \mathcal{A}(\boldsymbol{S}) = y] + \beta_2 \\
\leq{}& \sum_{S \in \mathcal{X}^n, y \in \mathcal{Y}} \min\left(2^{k_2} \cdot \mathbb{P}[S \times \mathcal{B}(S, y) \in \mathcal{Q}], 1\right) \cdot \left(2^{k_1} \cdot \mathbb{P}[\boldsymbol{S} = S] \cdot \mathbb{P}[\mathcal{A}(\boldsymbol{S}) = y] + \mu(S, y)\right) + \beta_2 \\
\leq{}& \sum_{S \in \mathcal{X}^n, y \in \mathcal{Y}} \min\left(2^{k_2} \cdot \mathbb{P}[S \times \mathcal{B}(S, y) \in \mathcal{Q}], 1\right) \cdot 2^{k_1} \cdot \mathbb{P}[\boldsymbol{S} = S] \cdot \mathbb{P}[\mathcal{A}(\boldsymbol{S}) = y] + \sum_{S \in \mathcal{X}^n, y \in \mathcal{Y}} \mu(S, y) + \beta_2 \\
\leq{}& \sum_{S \in \mathcal{X}^n, y \in \mathcal{Y}} \min\left(2^{k_2} \cdot \mathbb{P}[S \times \mathcal{B}(S, y) \in \mathcal{Q}], 1\right) \cdot 2^{k_1} \cdot \mathbb{P}[\boldsymbol{S} = S] \cdot \mathbb{P}[\mathcal{A}(\boldsymbol{S}) = y] + \beta_1 + \beta_2 \\
\leq{}& 2^{k_1 + k_2} \cdot \left(\sum_{S \in \mathcal{X}^n, y \in \mathcal{Y}} \mathbb{P}[S \times \mathcal{B}(S, y) \in \mathcal{Q}] \cdot \mathbb{P}[\boldsymbol{S} = S] \cdot \mathbb{P}[\mathcal{A}(\boldsymbol{S}) = y]\right) + \beta_1 + \beta_2 \\
={}& 2^{k_1 + k_2} \cdot \mathbb{P}[\boldsymbol{S} \times \mathcal{B}(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S})) \in \mathcal{Q}] + (\beta_1 + \beta_2) .
\end{aligned}
$$

Applying the definition of max-information, we see that equivalently, $I_\infty^{\beta_1 + \beta_2}(\boldsymbol{S}; \mathcal{C}(\boldsymbol{S})) \leq k_1 + k_2$, which is what we wanted. □

This lemma can be iteratively applied, which immediately yields the following adaptive composition theorem for max-information:

Theorem 15. *Consider an arbitrary sequence of algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_k$ with ranges $\mathcal{Y}_1, \ldots, \mathcal{Y}_k$ such that for all i, $\mathcal{A}_i : \mathcal{X}^n \times \mathcal{Y}_1 \times \ldots \times \mathcal{Y}_{i-1} \to \mathcal{Y}_i$ is such that $\mathcal{A}_i(\cdot, y_1, \ldots, y_{i-1})$ has β_i-approximate max-information k_i for all choices of $y_1, \ldots, y_{i-1} \in \mathcal{Y}_1 \times \ldots \times \mathcal{Y}_{i-1}$. Let the algorithm $\mathcal{B} : \mathcal{X}^n \to \mathcal{Y}_k$ be defined as follows:*
$\mathcal{B}(S)$:

1. ***Let*** $y_1 = \mathcal{A}_1(S)$.

2. ***For*** $i = 2$ *to* k: ***Let*** $y_i = \mathcal{A}_i(S, y_1, \ldots, y_{i-1})$

3. ***Output*** y_k

Then $\mathcal{B}$ has $(\sum_i \beta_i)$-approximate max-information $(\sum_i k_i)$.

Post-processing of Max-information: Another useful property that (approximate) max-information shares with differential privacy is preservation under post-processing. The simple proof of this lemma is identical to that for differential privacy (Lemma 5) and hence is omitted.

Lemma 16. *If $\mathcal{A}$ is an algorithm with domain $\mathcal{X}^n$ and range $\mathcal{Y}$, and $\mathcal{B}$ is any, possibly randomized, algorithm with domain $\mathcal{Y}$ and range $\mathcal{Y}''$, then the algorithm $\mathcal{B} \circ \mathcal{A}$ with domain $\mathcal{X}^n$ and range $\mathcal{Y}'$ satisfies: for every random variable $\boldsymbol{S}$ over $\mathcal{X}^n$ and every $\beta \geq 0$, $I_\infty^\beta(\boldsymbol{S}; \mathcal{B} \circ \mathcal{A}(\boldsymbol{S})) \leq I_\infty^\beta(\boldsymbol{S}; \mathcal{A}(\boldsymbol{S}))$.*

3.1 Bounds on Max-information

We now show that the basic approaches based on description length and (pure) differential privacy are captured by approximate max-information.

3.1.1 Description Length

Description length k gives the following bound on max-information.

Theorem 17. *Let $\mathcal{A}$ be a randomized algorithm taking as an input an n-element dataset and outputting a value in a finite set $\mathcal{Y}$. Then for every $\beta > 0$, $I_\infty^\beta(\mathcal{A}, n) \leq \log(|\mathcal{Y}|/\beta)$.*

We will use the following simple property of approximate divergence (*e.g.*[DR14]) in the proof. For a random variable $\boldsymbol{X}$ over $\mathcal{X}$ we denote by $p(\boldsymbol{X})$ the probability distribution associated with $\boldsymbol{X}$.

Lemma 18. *Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be two random variables over the same domain $\mathcal{X}$. If*

$$\mathop{\mathbb{P}}_{x \sim p(\boldsymbol{X})} \left[\frac{\mathbb{P}[\boldsymbol{X} = x]}{\mathbb{P}[\boldsymbol{Y} = x]} \geq 2^k \right] \leq \beta$$

then $D_\infty^\beta(\boldsymbol{X} \| \boldsymbol{Y}) \leq k$.

Proof Thm. 17. Let $\boldsymbol{S}$ be any random variable over n-element input datasets and let $\boldsymbol{Y}$ be the corresponding output distribution $\boldsymbol{Y} = \mathcal{A}(\boldsymbol{S})$. We prove that for every $\beta > 0$, $I_\infty^\beta(\boldsymbol{S}; \boldsymbol{Y}) \leq \log(|\mathcal{Y}|/\beta)$.

For $y \in \mathcal{Y}$ we say that y is "bad" if exists S in the support of $\boldsymbol{S}$ such that

$$\frac{\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S]}{\mathbb{P}[\boldsymbol{Y} = y]} \geq |\mathcal{Y}|/\beta.$$

Let B denote the set of all "bad" y's. From this definition we obtain that for a "bad" y, $\mathbb{P}[\boldsymbol{Y} = y] \leq \beta/|\mathcal{Y}|$ and therefore $\mathbb{P}[\boldsymbol{Y} \in B] \leq \beta$. Let $\mathcal{B} = \mathcal{X}^n \times B$. Then

$$\mathbb{P}[(\boldsymbol{S}, \boldsymbol{Y}) \in \mathcal{B}] = \mathbb{P}[\boldsymbol{Y} \in B] \leq \beta.$$

For every $(S, y) \notin \mathcal{B}$ we have that

$$\mathbb{P}[\boldsymbol{S} = S, \boldsymbol{Y} = y] = \mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S] \cdot \mathbb{P}[\boldsymbol{S} = S] \leq \frac{|\mathcal{Y}|}{\beta} \cdot \mathbb{P}[\boldsymbol{Y} = y] \cdot \mathbb{P}[\boldsymbol{S} = S],$$

and hence

$$\mathop{\mathbb{P}}_{(S,y) \sim p(\boldsymbol{S}, \boldsymbol{Y})} \left[\frac{\mathbb{P}[\boldsymbol{S} = S, \boldsymbol{Y} = y]}{\mathbb{P}[\boldsymbol{S} = S] \cdot \mathbb{P}[\boldsymbol{Y} = y]} \geq \frac{|\mathcal{Y}|}{\beta} \right] \leq \beta.$$

This, by Lemma 18, gives that $I_\infty^\beta(\boldsymbol{S}; \boldsymbol{Y}) \leq \log(|\mathcal{Y}|/\beta)$. $\qquad\square$

We note that Thms. 13 and 17 give a slightly weaker form of Thm. 9. Defining event $\mathcal{O} \doteq \{(S, y) \mid S \in R(y)\}$, the assumptions of Thm. 9 imply that $\mathbb{P}[\boldsymbol{S} \times \mathcal{A}(\boldsymbol{S}) \in \mathcal{O}] \leq \beta$. For $\beta' = \sqrt{|\mathcal{Y}|\beta}$, by Thm. 17, we have that $I_\infty^{\beta'}(\boldsymbol{S}; \mathcal{A}(\boldsymbol{S})) \leq \log(|\mathcal{Y}|/\beta')$. Now applying, Thm. 13 gives that $\mathbb{P}[\boldsymbol{S} \in R(\mathcal{A}(\boldsymbol{S}))] \leq |\mathcal{Y}|/\beta' \cdot \beta + \beta' = 2\sqrt{|\mathcal{Y}|\beta}$.

In Section A we introduce a closely related notion of *randomized description length* and show that it also provides an upper bound on approximate max-information. More interestingly, for this notion a form of the reverse bound can be proved: A bound on (approximate) max-information of an algorithm $\mathcal{A}$ implies a bound on the *randomized description length* of the output of a different algorithm with statistically indistinguishable from $\mathcal{A}$ output.

3.1.2   Differential Privacy

We now show that pure differential privacy implies a bound on max information. We start with a simple bound on max-information of differentially private algorithms that applies to all distributions over datasets. In particular, it implies that the differential privacy-based approach can be used beyond the i.i.d. setting in [DFH$^+$14].

Theorem 19. *Let $\mathcal{A}$ be an ϵ-differentially private algorithm. Then $I_\infty(\mathcal{A}, n) \leq \log e \cdot \epsilon n$.*

Proof. Clearly, any two datasets S and S' differ in at most n elements. Therefore, for every y we have $\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S] \leq e^{\epsilon n} \mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S']$ (this is a direct implication of Definition 2 referred to as group privacy [DR14]), or equivalently, $D_\infty(\mathcal{A}(S) \| \mathcal{A}(S')) \leq \log e \cdot \epsilon n$. By Lemma 3 we obtain the claim.   □

Finally, we prove a stronger bound on approximate max-information for datasets consisting of i.i.d. samples using the technique from [DFH$^+$14]. This bound, together with Thm. 13, generalizes Thm. 6.

Theorem 20. *Let $\mathcal{A}$ be an ε-differentially private algorithm with range $\mathcal{Y}$. For a distribution $\mathcal{P}$ over $\mathcal{X}$, let $\boldsymbol{S}$ be a random variable drawn from $\mathcal{P}^n$. Let $\boldsymbol{Y} = \mathcal{A}(\boldsymbol{S})$ denote the random variable output by $\mathcal{A}$ on input $\boldsymbol{S}$. Then for any $\beta > 0$, $I_\infty^\beta(\boldsymbol{S}; \mathcal{A}(\boldsymbol{S})) \leq \log e(\varepsilon^2 n/2 + \varepsilon\sqrt{n \ln(2/\beta)/2})$.*

Proof. Fix $y \in \mathcal{Y}$. We first observe that by Jensen's inequality,

$$\mathop{\mathbb{E}}_{S \sim \mathcal{P}^n}[\ln(\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S])] \leq \ln\left(\mathop{\mathbb{E}}_{S \sim \mathcal{P}^n}[\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S]]\right) = \ln(\mathbb{P}[\boldsymbol{Y} = y]).$$

Further, by definition of differential privacy, for two databases S, S' that differ in a single element,

$$\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S] \leq e^\varepsilon \cdot \mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S'].$$

Now consider the function $g(S) = \ln\left(\frac{\mathbb{P}[\boldsymbol{Y}=y \mid \boldsymbol{S}=S]}{\mathbb{P}[\boldsymbol{Y}=y]}\right)$. By the properties above we have that $\mathbb{E}[g(\boldsymbol{S})] \leq \ln(\mathbb{P}[\boldsymbol{Y} = y]) - \ln(\mathbb{P}[\boldsymbol{Y} = y]) = 0$ and $|g(S) - g(S')| \leq \varepsilon$. This, by McDiarmid's inequality (Lemma 1), implies that for any $t > 0$,

$$\mathbb{P}[g(\boldsymbol{S}) \geq t] \leq e^{-2t^2/(n\varepsilon^2)}. \tag{1}$$

For an integer $i \geq 1$, let $t_i \doteq \varepsilon^2 n/2 + \varepsilon\sqrt{n \ln(2^i/\beta)/2}$ and let

$$B_i \doteq \{S \mid t_i < g(S) \leq t_{i+1}\}.$$

Let

$$B_y \doteq \{S \mid g(S) > t_1\} = \bigcup_{i \geq 1} B_i.$$

By inequality (1), we have that for $i \geq 1$,

$$\mathbb{P}[g(\boldsymbol{S}) > t_i] \leq \exp\left(-2\left(\varepsilon\sqrt{n}/2 + \sqrt{\ln(2^i/\beta)/2}\right)^2\right).$$

By Bayes' rule, for every $S \in B_i$,

$$\frac{\mathbb{P}[\boldsymbol{S} = S \mid \boldsymbol{Y} = y]}{\mathbb{P}[\boldsymbol{S} = S]} = \frac{\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = S]}{\mathbb{P}[\boldsymbol{Y} = y]} = \exp(g(S)) \leq \exp(t_{i+1}).$$

Therefore,

$$\mathbb{P}[\boldsymbol{S} \in B_i \mid \boldsymbol{Y} = y] = \sum_{S \in B_i} \mathbb{P}[\boldsymbol{S} = S \mid \boldsymbol{Y} = y]$$

$$\leq \exp(t_{i+i}) \cdot \sum_{S \in B_i} \mathbb{P}[\boldsymbol{S} = S]$$

$$\leq \exp(t_{i+i}) \cdot \mathbb{P}[g(\boldsymbol{S}) \geq t_i]$$

$$= \exp\left(\varepsilon^2 n/2 + \varepsilon\sqrt{n\ln(2^{i+1}/\beta)/2} - 2\left(\varepsilon\sqrt{n}/2 + \sqrt{\ln(2^i/\beta)/2}\right)^2\right)$$

$$\leq \exp\left(\varepsilon\sqrt{n/2}\left(\sqrt{\ln(2^{i+1}/\beta)} - 2\sqrt{\ln(2^i/\beta)}\right) - \ln(2^i/\beta)\right)$$

$$< \exp(-\ln(2^i/\beta)) = \beta/2^i.$$

An immediate implication of this is that

$$\mathbb{P}[\boldsymbol{S} \in B_y \mid \boldsymbol{Y} = y] = \sum_i \mathbb{P}[\boldsymbol{S} \in B_i \mid \boldsymbol{Y} = y] \leq \sum_{i \geq 1} \beta/2^i \leq \beta.$$

Let $\mathcal{B} = \{(S, y) \mid y \in \mathcal{Y}, S \in B_y\}$. Then

$$\mathbb{P}[(\boldsymbol{S}, \boldsymbol{Y}) \in \mathcal{B}] = \mathbb{P}[(\boldsymbol{S}, \boldsymbol{Y}) \in B_{\boldsymbol{Y}}] \leq \beta. \tag{2}$$

For every $(S, y) \notin \mathcal{B}$ we have that

$$\mathbb{P}[\boldsymbol{S} = S, \boldsymbol{Y} = y] = \mathbb{P}[\boldsymbol{S} = S \mid \boldsymbol{Y} = y] \cdot \mathbb{P}[\boldsymbol{Y} = y] \leq \exp(t_1) \cdot \mathbb{P}[\boldsymbol{S} = S] \cdot \mathbb{P}[\boldsymbol{Y} = y],$$

and hence by eq.(2) we get that

$$\mathbb{P}_{(S,y)\sim p(\boldsymbol{S},\boldsymbol{Y})}\left[\frac{\mathbb{P}[\boldsymbol{S} = S, \boldsymbol{Y} = y]}{\mathbb{P}[\boldsymbol{S} = S] \cdot \mathbb{P}[\boldsymbol{Y} = y]} \geq \exp(t_1)\right] \leq \beta.$$

This, by Lemma 18, gives that

$$I_\infty^\beta(\boldsymbol{S}; \boldsymbol{Y}) \leq \log(\exp(t_1)) = \log e(\varepsilon^2 n/2 + \varepsilon\sqrt{n\ln(2/\beta)/2}).$$

$\square$

Applications: We give two simple examples of using the bounds on max-information obtained from differential privacy to preserve bounds on generalization error that follow from concentration of measure inequalities. Strong concentration of measure results are at the core of most generalization guarantees in machine learning. Let $\mathcal{A}$ be an algorithm that outputs a function $f : \mathcal{X}^n \to \mathbb{R}$ of sensitivity c and define the "bad event" $\mathcal{O}_\tau$ is when the empirical estimate of f is more than τ away from the expectation of $f(\boldsymbol{S})$ for $\boldsymbol{S}$ distributed according to some distribution $\mathcal{D}$ over $\mathcal{X}^n$. Namely,

$$\mathcal{O}_\tau = \{(S, f)\colon f(S) - \mathcal{D}[f] \geq \tau\}, \tag{3}$$

where $\mathcal{D}[f]$ denotes $\mathbb{E}_{S\sim\mathcal{D}}[f(S)]$.

By McDiarmid's inequality (Lem. 1) we know that, if $\boldsymbol{S}$ is distributed according to $\mathcal{P}^n$ then $\sup_{f:\mathcal{X}^n\to\mathbb{R}} \mathbb{P}[(\boldsymbol{S}, f) \in \mathcal{O}_\tau] \leq \exp(-2\tau^2/(c^2 n))$. The simpler bound in Thm. 19 implies following corollary.

Corollary 21. *Let $\mathcal{A}$ be an algorithm that outputs a c-sensitive function $f : \mathcal{X}^n \to \mathbb{R}$. Let $\boldsymbol{S}$ be a random dataset chosen according to distribution $\mathcal{P}^n$ over $\mathcal{X}^n$ and let $\boldsymbol{f} = \mathcal{A}(\boldsymbol{S})$. If for $\beta \geq 0$ and $\tau > 0$, $I_\infty^\beta(\boldsymbol{S}; \boldsymbol{f}) \leq \log e \cdot \tau^2/c^2$, then $\mathbb{P}[\boldsymbol{f}(\boldsymbol{S}) - \mathcal{P}^n[\boldsymbol{f}] \geq \tau] \leq \exp(-\tau^2/(c^2 n)) + \beta$. In particular, if $\mathcal{A}$ is $\tau^2/(c^2 n^2)$-differentially private then $\mathbb{P}[\boldsymbol{f}(\boldsymbol{S}) - \mathcal{P}^n[\boldsymbol{f}] \geq \tau] \leq \exp(-\tau^2/(c^2 n))$.*

Note that for $f(S) = \mathcal{E}_S[\phi]$, where $\phi : \mathcal{X} \to [0, 1]$ this result requires $\varepsilon = \tau^2$. The stronger bound allows to preserve concentration of measure even when $\varepsilon = \tau/(cn)$ which corresponds to $\tau = \varepsilon$ when $f(S) = \mathcal{E}_S[\phi]$.

Corollary 22. *Let $\mathcal{A}$ be an algorithm that outputs a c-sensitive function $f : \mathcal{X}^n \to \mathbb{R}$. Let $\boldsymbol{S}$ be a random dataset chosen according to distribution $\mathcal{P}^n$ over $\mathcal{X}^n$ and let $\boldsymbol{f} = \mathcal{A}(\boldsymbol{S})$. If $\mathcal{A}$ is $\tau/(cn)$-differentially private then $\mathbb{P}[\boldsymbol{f}(\boldsymbol{S}) - \mathcal{P}^n[\boldsymbol{f}] \geq \tau] \leq \exp\left(-3\tau^2/(4c^2 n)\right)$.*

Proof. We apply Theorem 20 with $\beta = 2\exp\left(-\tau^2/(c^2 n)\right)$ to obtain that

$$I_\infty^\beta(\boldsymbol{S}; \boldsymbol{f}) \leq \log e \cdot (\varepsilon^2 n/2 + \varepsilon \sqrt{n \ln(2/\beta)/2})) \leq \log e \cdot (\tau^2/(c^2 n)/2 + \tau^2/(c^2 n)/\sqrt{2}).$$

Applying Thm. 13 to McDiarmid's inequality we obtain that

$$\mathbb{P}[\boldsymbol{f}(\boldsymbol{S}) - \mathcal{P}^n[\boldsymbol{f}] \geq \tau] \leq \exp\left((1/2 + 1/\sqrt{2})\tau^2/(c^2 n)\right) \cdot \exp\left(-2\tau^2/(c^2 n)\right) + 2\exp\left(-\tau^2/(c^2 n)\right)$$
$$\leq \exp\left(-3\tau^2/(4c^2 n)\right),$$

where the last inequality holds when $\tau^2/(c^2 n)$ is larger than a fixed constant. $\qquad\square$

4 Reusable Holdout

We describe two simple algorithms that enable validation of analyst's queries in the adaptive setting.

4.1 Thresholdout

Our first algorithm Thresholdout follows the approach in [DFH+14] where differentially private algorithms are used to answer adaptively chosen statistical queries. This approach can also be applied to any low-sensitivity functions [1] of the dataset but for simplicity we present the results for statistical queries. Here we address an easier problem in which the analyst's queries only need to be answered when they overfit. Also, unlike in [DFH+14], the analyst has full access to the training set and the holdout algorithm only prevents overfitting to holdout dataset. As a result, unlike in the general query answering setting, our algorithm can efficiently validate an exponential in n number of queries as long as a relatively small number of them overfit.

Thresholdout is given access to the training dataset S_t and holdout dataset S_h and a budget limit B. It allows any query of the form $\phi : \mathcal{X} \to [0, 1]$ and its goal is to provide an estimate of $\mathcal{P}[\phi]$. To achieve this the algorithm gives an estimate of $\mathcal{E}_{S_h}[\phi]$ in a way that prevents overfitting of functions generated by the analyst to the holdout set. In other words, responses of Thresholdout are designed to ensure that, with high probability, $\mathcal{E}_{S_h}[\phi]$ is close to $\mathcal{P}[\phi]$ and hence an estimate of $\mathcal{E}_{S_h}[\phi]$ gives an estimate of the true expectation $\mathcal{P}[\phi]$. Given a function ϕ, Thresholdout first checks if the difference between the average value of ϕ on the training set S_t (or $\mathcal{E}_{S_t}[\phi]$) and the average value of ϕ on the holdout set S_h (or $\mathcal{E}_{S_h}[\phi]$) is below a certain threshold $T + \eta$. Here, T is a fixed number such as 0.01 and η is a Laplace noise variable whose standard deviation needs to be chosen depending on the desired guarantees (The Laplace distribution is a symmetric exponential distribution.) If the difference is below the threshold, then the algorithm returns $\mathcal{E}_{S_t}[\phi]$. If the difference is above the threshold, then the algorithm returns $\mathcal{E}_{S_h}[\phi] + \xi$ for another Laplacian noise variable ξ. Each time the difference is above threshold the "overfitting" budget B is reduced by one. Once it is exhausted, Thresholdout stops answering queries. In Fig. 1 we provide the pseudocode of Thresholdout.

We now establish the formal generalization guarantees that Thresholdout enjoys. As the first step we state what privacy parameters are achieved by Thresholdout.

Lemma 23. Thresholdout *satisfies $(2B/(\sigma n), 0)$-differential privacy.* Thresholdout *also satisfies $(\sqrt{32B \ln(2/\delta)}/(\sigma n), \delta)$-differential privacy for any $\delta > 0$.*

---

[1] Guarantees based on pure differential privacy follow from the same analysis. Proving generalization guarantees for low-sensitivity queries based on approximate differential privacy requires a modification of Thresholdout using techniques in [BSSU15].

Algorithm Thresholdout **Input:** Training set S_t, holdout set S_h, threshold T, noise rate σ, budget B

1. sample $\gamma \sim \text{Lap}(2 \cdot \sigma)$; $\hat{T} \leftarrow T + \gamma$

2. **For** each query ϕ **do**

   (a) **if** $B < 1$ output "$\perp$"

   (b) **else**

       i. sample $\eta \sim \text{Lap}(4 \cdot \sigma)$

       ii. **if** $|\mathcal{E}_{S_h}[\phi] - \mathcal{E}_{S_t}[\phi]| > \hat{T} + \eta$

           A. sample $\xi \sim \text{Lap}(\sigma)$, $\gamma \sim \text{Lap}(2 \cdot \sigma)$

           B. $B \leftarrow B - 1$ and $\hat{T} \leftarrow T + \gamma$

           C. output $\mathcal{E}_{S_h}[\phi] + \xi$

       iii. **else** output $\mathcal{E}_{S_t}[\phi]$.

Figure 1: The details of Thresholdout algorithm

Proof. Thresholdout is an instantiation of a basic tool from differential privacy, the "Sparse Vector Algorithm" ([DR14, Algorithm 2]), together with the Laplace mechanism ([DR14, Defn. 3.3]). The sparse vector algorithm takes as input a sequence of c sensitivity $1/n$ queries[2] (here $c = B$, the budget), and for each query, attempts to determine whether the value of the query, evaluated on the private dataset, is above a fixed threshold T or below it. In our instantiation, the holdout set S_h is the private data set, and each function ϕ corresponds to the following query evaluated on S_h: $f_\phi(S_h) := |\mathcal{E}_{S_h}[\phi] - \mathcal{E}_{S_t}[\phi]|$. (Note that the training set S_t is viewed as part of the definition of the query). Thresholdout then is equivalent to the following procedure: we run the sparse vector algorithm [DR14, Algorithm 2] with $c = B$, queries f_ϕ for each function ϕ, and noise rate 2σ. Whenever an above-threshold query is reported by the sparse vector algorithm, we release its value using the Laplace mechanism [DR14, Defn. 3.3] with noise rate σ (this is what occurs every time Thresholdout answers by outputting $\mathcal{E}_{S_h}[\phi] + \xi$). By the privacy guarantee of the sparse vector algorithm ([DR14, Thm. 3.25]), the sparse vector portion of Thresholdout satisfies $(B/(\sigma n), 0)$-differential privacy, and simultaneously satisfies $(\frac{\sqrt{8B \ln(2/\delta)}}{\sigma n}, \delta/2)$-differential privacy. The Laplace mechanism portion of Thresholdout satisfies $(B/(\sigma n), 0)$-differential privacy by [DR14, Thm. 3.6], and simultaneously satisfies $(\frac{\sqrt{8B \ln(2/\delta)}}{\sigma n}, \delta/2)$-differential privacy by [DR14, Thm. 3.6] and [DR14, Cor. 3.21]. Finally, the composition of two mechanisms, the first of which is (ϵ_1, δ_1)-differentially private, and the second of which is (ϵ_2, δ_2)-differentially private is itself $(\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-differentially private (Thm. 3). Adding the privacy parameters of the Sparse Vector portion of Thresholdout and the Laplace mechanism portion of Thresholdout yield the parameters of our theorem. $\square$

We note that tighter privacy parameters are possible (e.g. by invoking the parameters and guarantees of the algorithm "NumericSparse" ([DR14, Algorithm 3]), which already combines the Laplace addition step) – we chose simpler parameters for clarity.

Note the seeming discrepancy between the guarantee provided by Thresholdout and generalization guarantees in Theorem 8 and Corollary 7: while Theorem 8 promises generalization bounds for functions that are generated by a differentially private algorithm, here we allow an arbitrary data analyst to generate query functions in any way she chooses, with access to the training set and differentially private estimates of the means of her functions on the holdout set. The connection comes from preservation of differential privacy guarantee under post-processing (Lem. 5).

We can now quantify the number of samples necessary to achieve generalization error τ with probability

---

[2]In fact, the theorems for the Sparse Vector algorithm in Dwork and Roth are stated for sensitivity 1 queries – we use them for sensitivity $1/n$ queries of the form $\mathcal{E}_{S_h}[\phi]$, which results in all of the bounds being scaled down by a factor of n.

at least $1 - \beta$.

Lemma 24. *Let $\tau, \beta, T, B > 0$. Let $\boldsymbol{S}$ denote the holdout dataset of size n drawn i.i.d. from a distribution $\mathcal{P}$. Consider an algorithm that is given access to S_t and adaptively chooses functions $\phi_1, \ldots, \phi_m : \mathcal{X} \to [0, 1]$ while interacting with $\mathsf{Thresholdout}$ which is given datasets $\boldsymbol{S}$, S_t and parameters σ, B, T. If*

$$n \geq n_0(B, \sigma, \tau, \beta) \doteq \max\{2B/(\sigma\tau), \ln(6/\beta)/\tau^2\}$$

or

$$n \geq n_1(B, \sigma, \tau, \beta) \doteq \frac{80 \cdot \sqrt{B \ln(1/(\tau\beta))}}{\tau\sigma}$$

then for every $i \in [m]$, $\mathbb{P}\left[|\mathcal{P}[\phi_i] - \mathcal{E}_{\boldsymbol{S}}[\phi_i]| \geq \tau\right] \leq \beta$.

Proof. Consider the first guarantee of Lemma 23. In order to achieve generalization error τ via Corollary 7 (i.e. in order to guarantee that for every function ϕ we have: $\mathbb{P}\left[|\mathcal{P}[\phi_i] - \mathcal{E}_{\boldsymbol{S}}[\phi_i]| \geq \tau\right] \leq 6e^{-\tau^2 n}$) we need to have n large enough to achieve $(\varepsilon, 0)$-differential privacy for $\varepsilon = \tau$. To achieve this it suffices to have $n \geq 2B/(\sigma\tau)$. By ensuring that $n \geq \ln(6/\beta)/\tau^2$ we also have that $6e^{-\tau^2 n} \leq \beta$.

We can also make use of the second guarantee in Lemma 23 together with the results of Nissim and Stemmer [NS15] (Thm. 8). In order to achieve generalization error τ with probability $1 - \beta$ (i.e. in order to guarantee for every function ϕ we have: $\mathbb{P}\left[|\mathcal{P}[\phi_i] - \mathcal{E}_{\boldsymbol{S}}[\phi_i]| \geq \tau\right] \leq \beta$), we can apply Thm. 8 by setting $\epsilon = \sqrt{32B \ln(2/\delta)}/(\sigma n) = \tau/13$ and $\delta = \frac{\beta\tau}{26 \ln(26/\tau)}$. We can obtain these privacy parameters from Lemma 23 by choosing any $n \geq \frac{80 \cdot \sqrt{B \ln(1/(\tau\beta))}}{\tau\sigma}$ (for sufficiently small β and τ). We remark that a somewhat worse bound of $n_1(B, \sigma, \tau, \beta) = \frac{2\sqrt{32B \ln(8/\beta)}}{\tau^{3/2}\sigma}$ follows by setting $\epsilon = \tau/4$ and $\delta = (\beta/8)^{4/\tau}$ in [DFH$^+$14, Thm. 10]. □

Both settings lead to small generalization error and so we can pick whichever gives the larger bound. The first bound has grows linearly with B but is simpler and can be easily extended to other distributions over datasets and to low-sensitivity functions. The second bound has a quadratically better dependence on B at the expense of a slightly worse dependence on τ. We can now apply our main results to get a generalization bound for the entire execution of $\mathsf{Thresholdout}$.

Theorem 25. *Let $\beta, \tau > 0$ and $m \geq B > 0$. We set $T = 3\tau/4$ and $\sigma = \tau/(96 \ln(4m/\beta))$. Let $\boldsymbol{S}$ denote a holdout dataset of size n drawn i.i.d. from a distribution $\mathcal{P}$ and S_t be any additional dataset over $\mathcal{X}$. Consider an algorithm that is given access to S_t and adaptively chooses functions $\phi_1, \ldots, \phi_m$ while interacting with $\mathsf{Thresholdout}$ which is given datasets $\boldsymbol{S}$, S_t and values σ, B, T. For every $i \in [m]$, let $\boldsymbol{a}_i$ denote the answer of $\mathsf{Thresholdout}$ on function $\phi_i : \mathcal{X} \to [0, 1]$. Further, for every $i \in [m]$, we define the counter of overfitting*

$$\boldsymbol{Z}_i \doteq |\{j \leq i : |\mathcal{P}[\phi_j] - \mathcal{E}_{S_t}[\phi_j]| > \tau/2\}|.$$

Then

$$\mathbb{P}\left[\exists i \in [m], \boldsymbol{Z}_i < B \ \& \ |\boldsymbol{a}_i - \mathcal{P}[\phi_i]| \geq \tau\right] \leq \beta$$

whenever $n \geq \min\{n_0(B, \sigma, \tau/8, \beta/(2m)), \ n_1(B, \sigma, \tau/8, \beta/(2m))\} = O\left(\frac{\ln(m/\beta)}{\tau^2}\right) \cdot \min\{B, \sqrt{B \ln(m/(\beta\tau))}\}$.

Proof. There are two types of error we need to control: the deviation between $\boldsymbol{a}_i$ and the average value of ϕ_i on the holdout set $\mathcal{E}_{\boldsymbol{S}}[\phi_i]$, and the deviation between the average value of ϕ_i on the holdout set and the expectation of ϕ_i on the underlying distribution, $\mathcal{P}[\phi_i]$. Specifically, we decompose the error as

$$\mathbb{P}[\boldsymbol{a}_i \neq \perp \ \& \ |\boldsymbol{a}_i - \mathcal{P}[\phi_i]| \geq \tau] \leq \mathbb{P}\left[\boldsymbol{a}_i \neq \perp \ \& \ |\boldsymbol{a}_i - \mathcal{E}_{\boldsymbol{S}}[\phi_i]| \geq 7\tau/8\right] + \mathbb{P}\left[|\mathcal{P}[\phi_i] - \mathcal{E}_{\boldsymbol{S}}[\phi_i]| \geq \tau/8\right]. \quad (4)$$

To control the first term we need to bound the values of noise variables used by $\mathsf{Thresholdout}$. For the second term we will use the generalization properties of $\mathsf{Thresholdout}$ given in Lemma 24.

We now deal with the errors introduced by the noise variables. For $i \in [m]$, let $\boldsymbol{\eta}_i$, $\boldsymbol{\xi}_i$ and $\boldsymbol{\gamma}_i$ denote the random variables η, ξ and γ, respectively, at step i of the execution of $\mathsf{Thresholdout}$. We first note that each of

these variables is chosen from Laplace distribution at most m times. By properties of the Laplace distribution with parameter 4σ, we know that for every $t > 0$, $\mathbb{P}[|\boldsymbol{\eta}_i| \geq t \cdot 4\sigma] = e^{-t/2}$. Therefore for $t = 2\ln(4m/\beta)$ we obtain

$$\mathbb{P}[|\boldsymbol{\eta}_i| \geq 2\ln(4m/\beta) \cdot 4\sigma] \leq e^{-t/2} = \frac{\beta}{4m}.$$

By the definition of σ, $8\ln(4m/\beta) \cdot \sigma = \tau/12$. Applying the union bound we obtain that

$$\mathbb{P}[\exists i, \ |\boldsymbol{\eta}_i| \geq \tau/12] \leq \beta/4,$$

where by $\exists i$ we refer to $\exists i \in [m]$ for brevity. Similarly, $\boldsymbol{\xi}_i$ and $\boldsymbol{\gamma}_i$ are obtained by sampling from the Laplace distribution and each is re-randomized at most B times. Therefore

$$\mathbb{P}[\exists i, \ |\boldsymbol{\gamma}_i| \geq \tau/24] \leq B \cdot \beta/4m \leq \beta/4$$

and

$$\mathbb{P}[\exists i, \ |\boldsymbol{\xi}_i| \geq \tau/48] \leq B \cdot \beta/4m \leq \beta/4.$$

For answers that are different from $\perp$, we can now bound the first term of Equation (4) by considering two cases, depending on whether Thresholdout answers query $\boldsymbol{\phi}_i$ by returning $\boldsymbol{a}_i = \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i] + \boldsymbol{\xi}_i$ or by returning $\boldsymbol{a}_i = \mathcal{E}_{S_t}[\boldsymbol{\phi}_i]$. First, consider queries whose answers are returned using the former condition. Under this condition, $|\boldsymbol{a}_i - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i]| = |\boldsymbol{\xi}_i|$. Next, we consider the second case, those queries whose answers are returned using $\mathcal{E}_{S_t}[\boldsymbol{\phi}_i]$. By definition of the algorithm, we have

$$|\boldsymbol{a}_i - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i]| = |\mathcal{E}_{S_t}[\boldsymbol{\phi}_i] - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i]| \leq T + \boldsymbol{\gamma}_i + \boldsymbol{\eta}_i \leq 3\tau/4 + |\boldsymbol{\gamma}_i| + |\boldsymbol{\eta}_i|.$$

Combining these two cases implies that

$$\mathbb{P}\left[\exists i, \ \boldsymbol{a}_i \neq \perp \ \& \ |\boldsymbol{a}_i - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i]| \geq 7\tau/8\right] \leq \max\{\mathbb{P}\left[\exists i, \ |\boldsymbol{\xi}_i| \geq 7\tau/8\right], \ \mathbb{P}\left[\exists i, \ |\boldsymbol{\gamma}_i| + |\boldsymbol{\eta}_i| \geq \tau/8\right]\}.$$

Noting that $\tau/24 + \tau/12 = \tau/8$ and applying our bound on variables $\boldsymbol{\eta}_i$, $\boldsymbol{\xi}_i$ and $\boldsymbol{\gamma}_i$ we get

$$\mathbb{P}\left[\exists i, \ \boldsymbol{a}_i \neq \perp \ \& \ |\boldsymbol{a}_i - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i]| \geq 7\tau/8\right] \leq \beta/2. \tag{5}$$

By Lemma 24, for $n \geq \min\{n_0(B, \sigma, \tau/8, \beta/2m), n_1(B, \sigma, \tau/8, \beta/2m)\}$,

$$\mathbb{P}\left[|\mathcal{P}[\boldsymbol{\phi}_i] - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i]| \geq \tau/8\right] \leq \beta/2m.$$

Applying the union bound we obtain

$$\mathbb{P}\left[\exists i, \ |\mathcal{P}[\boldsymbol{\phi}_i] - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i]| \geq \tau/8\right] \leq \beta/2.$$

Combining this with Equation (5) and using in Equation (4) we get that

$$\mathbb{P}[\exists i, \ \boldsymbol{a}_i \neq \perp \ \& \ |\boldsymbol{a}_i - \mathcal{P}[\boldsymbol{\phi}_i]| \geq \tau] \leq \beta.$$

To finish the proof we show that under the conditions on the noise variables and generalization error used above, we have that if $\boldsymbol{Z}_i < B$ then $\boldsymbol{a}_i \neq \perp$. To see this, observe that for every $j \leq i$ that reduces Thresholdout's budget, we have

$$
\begin{aligned}
|\mathcal{P}[\boldsymbol{\phi}_j] - \mathcal{E}_{S_t}[\boldsymbol{\phi}_j]| &\geq |\mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_j] - \mathcal{E}_{S_t}[\boldsymbol{\phi}_j]| - |\mathcal{P}[\boldsymbol{\phi}_j] - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_j]| \\
&\geq |T + \boldsymbol{\gamma}_j + \boldsymbol{\eta}_j| - |\mathcal{P}[\boldsymbol{\phi}_j] - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_j]| \\
&\geq T - |\boldsymbol{\gamma}_j| - |\boldsymbol{\eta}_j| - |\mathcal{P}[\boldsymbol{\phi}_j] - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_j]|.
\end{aligned}
$$

This means that for every $j \leq i$ that reduces the budget we have $|\mathcal{P}[\boldsymbol{\phi}_j] - \mathcal{E}_{S_t}[\boldsymbol{\phi}_j]| \geq 3\tau/4 - \tau/24 - \tau/12 - \tau/8 = \tau/2$ and hence (when the conditions on the noise variables and generalization error are satisfied) for every i, if $\boldsymbol{Z}_i < B$ then Thresholdout's budget is not yet exhausted and $\boldsymbol{a}_i \neq \perp$. We can therefore conclude that

$$\mathbb{P}[\exists i, \boldsymbol{Z}_i < B \ \& \ |\boldsymbol{a}_i - \mathcal{P}[\boldsymbol{\phi}_i]| \geq \tau] \leq \beta.$$

$\square$

Note that in the final bound on n, the term $O\left(\frac{\ln(m/\beta)}{\tau^2}\right)$ is equal (up to a constant factor) to the number of samples that are necessary to answer m *non-adaptively* chosen queries with tolerance τ and confidence $1 - \beta$. In particular, as in the non-adaptive setting, achievable tolerance τ scales as $1/\sqrt{n}$ (up to the logarithmic factor). Further, this bound allows m to be exponentially large in n as long as B grows sub-quadratically in n (that is, $B \leq n^{2-c}$ for a constant $c > 0$).

Remark 26. *In Thm. 25 S_h is used solely to provide a candidate estimate of the expectation of each query function. The theorem holds for any other way to provide such estimates. In addition, the one-sided version of the algorithm can be used when catching only the one-sided error is necessary. For example, in many cases overfitting is problematic only if the training error estimate is larger than the true error. This is achieved by using the condition $\mathcal{E}_{S_h}[\phi] - \mathcal{E}_{S_t}[\phi] > \hat{T} + \eta$ to detect overfitting. In this case only one-sided errors will be caught by* Thresholdout *and only one-sided overfitting will decrease the budget.*

4.2 SparseValidate

We now present a general algorithm for validation on the holdout set that can validate many arbitrary queries as long as few of them fail the validation. The algorithm which we refer to as SparseValidate only reveals information about the holdout set when validation fails and therefore we use bounds based on description length to analyze its generalization guarantees.

More formally, our algorithm allows the analyst to pick any Boolean function of a dataset ψ (or even any algorithm that outputs a single bit) and provides back the value of ψ on the holdout set $\psi(S_h)$. SparseValidate has a budget m for the total number of queries that can be asked and budget B for the number of queries that returned 1. Once either of the budgets is exhausted, no additional answers are given. We now give a general description of the guarantees of SparseValidate.

Theorem 27. *Let $\boldsymbol{S}$ denote a randomly chosen holdout set of size n. Let $\mathcal{A}$ be an algorithm that is given access to* SparseValidate(m, B) *and outputs queries $\psi_1, \ldots, \psi_m$ such that each ψ_i is in some set Ψ_i of functions from $\mathcal{X}^n$ to $\{0, 1\}$. Assume that for every $i \in [m]$ and $\psi_i \in \Psi_i$, $\mathbb{P}[\psi_i(\boldsymbol{S}) = 1] \leq \beta_i$. Let $\boldsymbol{\psi}_i$ be the random variable equal to the i'th query of $\mathcal{A}$ on $\boldsymbol{S}$. Then $\mathbb{P}[\boldsymbol{\psi}_i(\boldsymbol{S}) = 1] \leq \ell_i \cdot \beta_i$, where $\ell_i = \sum_{j=0}^{\min\{i-1, B\}} \binom{i}{j} \leq m^B$.*

Proof. Let $\mathcal{B}$ denote the algorithm that represents view the interaction of $\mathcal{A}$ with SparseValidate(m, B) up until query i and outputs the all the $i - 1$ responses of SparseValidate(m, B) in this interaction. If there are B responses with value 1 in the interaction then all the responses after the last one are meaningless and can be assumed to be equal to 0. The number of binary strings of length $i - 1$ that contain at most B ones is exactly $\ell_i = \sum_{j=0}^{\min\{i-1, B\}} \binom{i}{j}$. Therefore we can assume that the output domain of $\mathcal{B}$ has size ℓ_i and we denote it by $\mathcal{Y}$. Now, for $y \in \mathcal{Y}$ let $R(y)$ be the set of datasets S such that $\psi_i(S) = 1$, where ψ_i is the function that $\mathcal{A}$ generates when the responses of SparseValidate(m, B) are y and the input holdout dataset is S (for now assume that $\mathcal{A}$ is deterministic). By the conditions of the theorem we have that for every y, $\mathbb{P}[\boldsymbol{S} \in R(y)] \leq \beta_i$. Applying Thm. 9 to $\mathcal{B}$, we get that $\mathbb{P}[\boldsymbol{S} \in R(\mathcal{B}(\boldsymbol{S}))] \leq \ell_i \beta_i$, which is exactly the claim. We note that to address the case when $\mathcal{A}$ is randomized (including dependent on the random choice of the training set) we can use the argument above for every fixing of all the random bits of $\mathcal{A}$. From there we obtain that the claim holds when the probability is taken also over the randomness of $\mathcal{A}$.

We remark that the proof can also be obtained via a more direct application of the union bound over all strings in $\mathcal{Y}$. But the proof via Thm. 9 demonstrates the conceptual role that short description length plays in this application. $\square$

In this general formulation it is the analyst's responsibility to use the budgets economically and pick query functions that do not fail validation often. At the same time, SparseValidate ensures that (for the appropriate values of the parameters) the analyst can think of the holdout set as a fresh sample for the purposes of validation. Hence the analyst can pick queries in such a way that failing the validation reliably indicates overfitting. To relate this algorithm to Thresholdout, consider the validation query function that is the indicator of the condition $|\mathcal{E}_{S_h}[\phi] - \mathcal{E}_{S_t}[\phi]| > \hat{T} + \eta$ (note that this condition can be evaluated using

an algorithm with access to S_h). This is precisely the condition that consumes the overfitting budget of Thresholdout. Now, as in Thresholdout, for every fixed ϕ, $\mathbb{P}[|\mathcal{E}_{S_h}[\phi] - \mathcal{P}[\phi]| \geq \tau] \leq 2e^{-2\tau^2 n}$. If $B \leq \tau^2 n / \ln m$, then we obtain that for every query ϕ generated by the analyst, we still have strong concentration of the mean on the holdout set around the expectation: $\mathbb{P}[|\mathcal{E}_{S_h}[\phi] - \mathcal{P}[\phi]| \geq \tau] \leq 2e^{-\tau^2 n}$. This implies that if the condition $|\mathcal{E}_{S_h}[\phi] - \mathcal{E}_{S_t}[\phi]| > \hat{T} + \eta$ holds, then with high probability also the condition $|\mathcal{P}[\phi] - \mathcal{E}_{S_t}[\phi]| > \hat{T} + \eta - \tau$ holds, indicating overfitting.

One notable distinction of Thresholdout from SparseValidate is that SparseValidate does not provide corrections in the case of overfitting. One way to remedy that is simply to use a version of SparseValidate that allows functions with values in $\{0, 1, \ldots, L\}$. It is easy to see that for such functions we would obtain the bound of the form $\ell_i \cdot L^{\min\{B, i-1\}} \cdot \beta_i$. To output a value in $[0, 1]$ with precision τ, $L = \lfloor 1/\tau \rfloor$ would suffice. However, in many cases a more economical solution would be to have a separate dataset which is used just for obtaining the correct estimates.

An example of the application of SparseValidate for answering statistical and low-sensitivity queries that is based on our analysis can be found in [BSSU15]. The analysis of generalization on the holdout set in [BH15] and the analysis of the Median Mechanism we give in Section B also rely on this sparsity-based technique.

An alternative view of this algorithm is as a general template for designing algorithms for answering some specific type of adaptively chosen queries. Generalization guarantees specific to the type of query can then be obtained from our general analysis. For example, an algorithm that fits a mixture of Gaussians model to the data could define the validation query to be an algorithm that fits the mixture model to the holdout and obtains a vector of parameters Θ_h. The validation query then compares it with the vector of parameters Θ_t obtained on the training set and outputs 1 if the parameter vectors are "not close" (indicating overfitting). Given guarantees of statistical validity of the parameter estimation method in the static setting one could then derive guarantees for adaptive validation via Thm. 27.

5 Experiments

We describe a simple experiment on synthetic data that illustrates the danger of reusing a standard holdout set and how this issue can be resolved by our reusable holdout. In our experiment the analyst wants to build a classifier via the following common strategy. First the analyst finds a set of single attributes that are correlated with the class label. Then the analyst aggregates the correlated variables into a single model of higher accuracy (for example using boosting or bagging methods). More formally, the analyst is given a d-dimensional labeled data set S of size $2n$ and splits it randomly into a training set S_t and a holdout set S_h of equal size. We denote an element of S by a tuple (x, y) where x is a d-dimensional vector and $y \in \{-1, 1\}$ is the corresponding class label. The analyst wishes to select variables to be included in her classifier. For various values of the number of variables to select k, she picks k variables with the largest absolute correlations with the label. However, she verifies the correlations (with the label) on the holdout set and uses only those variables whose correlation agrees in sign with the correlation on the training set and both correlations are larger than some threshold in absolute value. She then creates a simple linear threshold classifier on the selected variables using only the signs of the correlations of the selected variables. A final test evaluates the classification accuracy of the classifier on both the training set and the holdout set.

Formally, the algorithm is used to build a linear threshold classifier:

1. For each attribute $i \in [d]$ compute the correlation with the label on the training and holdout sets: $w_i^t = \sum_{(x,y) \in S_t} x_i y$ and $w_i^h = \sum_{(x,y) \in S_h} x_i y$. Let

$$W = \left\{ i \mid w_i^t \cdot w_i^h > 0; \ |w_i^t| \geq 1/\sqrt{n}; \ |w_i^h| \geq 1/\sqrt{n}| \right\}$$

   that is the set of variables for which w_i^t and w_i^h have the same sign and both are at least $1/\sqrt{n}$ in absolute value (this is the standard deviation of the correlation in our setting). Let V_k be the subset of variables in W with k largest values of $|w_i^t|$.

2. Construct the classifier $f(x) = \text{sgn}\left(\sum_{i \in V_k} \text{sgn}(w_i^t) \cdot x_i \right)$.

In the experiments we used an implementation of Thresholdout that differs somewhat from the algorithm we analyzed theoretically (given in Figure 1). Specifically, we set the parameters to be $T = 0.04$ and $\tau = 0.01$. This is lower than the values necessary for the proof (and which are not intended for direct application) but suffices to prevent overfitting in our experiment. Second, we use Gaussian noise instead of Laplacian noise as it has stronger concentration properties (in many differential privacy applications similar theoretical guarantees hold mechanisms based on Gaussian noise).

No correlation between labels and data: In our first experiment, each attribute is drawn independently from the normal distribution $N(0,1)$ and we choose the class label $y \in \{-1, 1\}$ uniformly at random so that there is no correlation between the data point and its label. We chose $n = 10,000$, $d = 10,000$ and varied the number of selected variables k. In this scenario no classifier can achieve true accuracy better than 50%. Nevertheless, reusing a standard holdout results in reported accuracy of over 63% for $k = 500$ on both the training set and the holdout set (the standard deviation of the error is less than 0.5%). The average and standard deviation of results obtained from 100 independent executions of the experiment are plotted in Figure 2 which also includes the accuracy of the classifier on another fresh data set of size n drawn from the same distribution. We then executed the same algorithm with our reusable holdout. The algorithm Thresholdout was invoked with $T = 0.04$ and $\sigma = 0.01$ explaining why the accuracy of the classifier reported by Thresholdout is off by up to 0.04 whenever the accuracy on the holdout set is within 0.04 of the accuracy on the training set. Thresholdout prevents the algorithm from overfitting to the holdout set and gives a valid estimate of classifier accuracy.
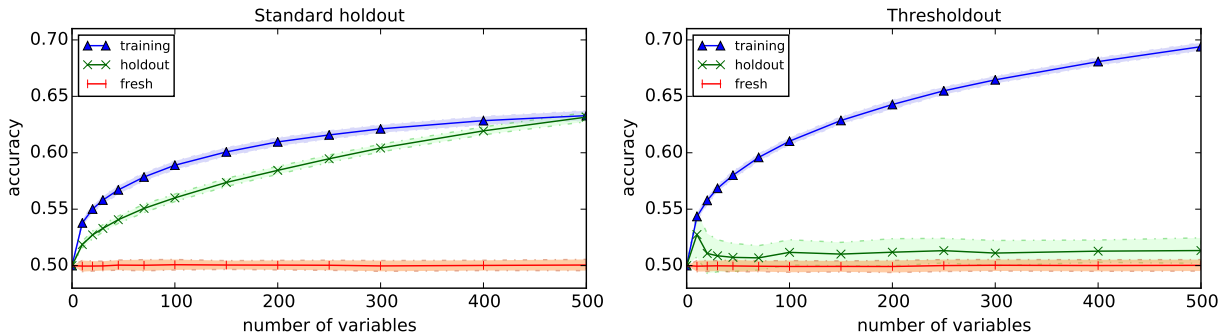

Figure 2: No correlation between class labels and data points. The plot shows the classification accuracy of the classifier on training, holdout and fresh sets. Margins indicate the standard deviation.

High correlation between labels and some of the variables: In our second experiment, the class labels are correlated with some of the variables. As before the label is randomly chosen from $\{-1, 1\}$ and each of the attributes is drawn from $N(0,1)$ aside from 20 attributes which are drawn from $N(y \cdot 0.06, 1)$ where y is the class label. We execute the same algorithm on this data with both the standard holdout and Thresholdout and plot the results in Figure 3. Our experiment shows that when using the reusable holdout, the algorithm still finds a good classifier while preventing overfitting. This illustrates that the reusable holdout simultaneously prevents overfitting and allows for the discovery of true statistical patterns.

In Figures 2 and 3, simulations that used Thresholdout for selecting the variables also show the accuracy on the holdout set as reported by Thresholdout. For comparison purposes, in Figure 4 we plot the actual accuracy of the generated classifier on the holdout set (the parameters of the simulation are identical to those used in Figures 2 and 3). It demonstrates that there is essentially no overfitting to the holdout set. Note that the advantage of the accuracy reported by Thresholdout is that it can be used to make further data dependent decisions while mitigating the risk of overfitting.

Discussion of the results: Overfitting to the standard holdout set arises in our experiment because the analyst reuses the holdout after using it to measure the correlation of single attributes. We first note that neither cross-validation nor bootstrap resolve this issue. If we used either of these methods to validate the
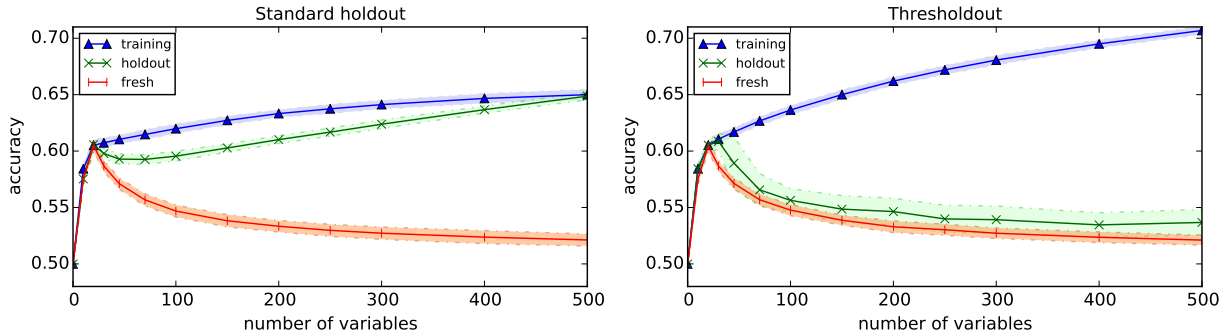
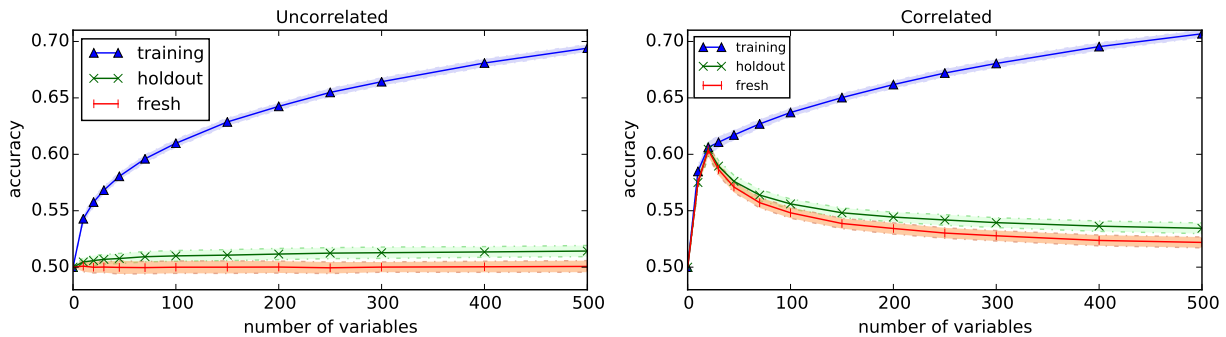Figure 3: Some variables are correlated with the label.

Figure 4: Accuracy of the classifier produced with Thresholdout on the holdout set.

correlations, overfitting would still arise due to using the same data for training and validation (of the final classifier). It is tempting to recommend other solutions to the specific problem we used in our experiment. Indeed, a significant number of methods in the statistics and machine learning literature deal with inference for fixed two-step procedures where the first step is variable selection (see [HTF09] for examples). Our experiment demonstrates that even in such simple and standard settings our method avoids overfitting without the need to use a specialized procedure – and, of course, extends more broadly. More importantly, the reusable holdout gives the analyst a general and principled method to perform multiple validation steps where previously the only known safe approach was to collect a fresh holdout set each time a function depends on the outcomes of previous validations.

6 Conclusions

In this work, we give a unifying view of two techniques (differential privacy and description length bounds) which preserve the generalization guarantees of subsequent algorithms in adaptively chosen sequences of data analyses. Although these two techniques both imply low max-information – and hence can be composed together while preserving their guarantees – the kinds of guarantees that can be achieved by either alone are incomparable. This suggests that the problem of generalization guarantees under adaptivity is ripe for future study on two fronts. First, the existing theory is likely already strong enough to develop practical algorithms with rigorous generalization guarantees, of which Thresholdout is an example. However additional empirical work is needed to better understand when and how the theory should be applied in specific application

scenarios. At the same time, new theory is also needed. As an example of a basic question we still do not know the answer to: even in the simple setting of adaptively reusing a holdout set for computing the expectations of boolean-valued predicates, is it possible to obtain stronger generalization guarantees (via any means) than those that are known to be achievable via differential privacy?

References

[BDMN05] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the SuLQ framework. In *PODS*, pages 128–138, 2005.

[BE02] Olivier Bousquet and André Elisseeff. Stability and generalization. *JMLR*, 2:499–526, 2002.

[BH15] Avrim Blum and Moritz Hardt. The ladder: A reliable leaderboard for machine learning competitions. In *ICML*, pages 1006–1014, 2015.

[BNS13] Amos Beimel, Kobbi Nissim, and Uri Stemmer. Characterizing the sample complexity of private learners. In *ITCS*, pages 97–110, 2013.

[BSSU15] Raef Bassily, Adam Smith, Thomas Steinke, and Jonathan Ullman. More general queries and less generalization error in adaptive data analysis. *CoRR*, abs/1503.04843, 2015.

[CT10] Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11:2079–2107, 2010.

[DFH+14] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. Preserving statistical validity in adaptive data analysis. *CoRR*, abs/1411.2664, 2014. Extended abstract in STOC 2015.

[DFN07] Chuong B. Do, Chuan-Sheng Foo, and Andrew Y. Ng. Efficient multiple hyperparameter learning for log-linear models. In *NIPS*, pages 377–384, 2007.

[DKM+06] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pages 486–503, 2006.

[DL09] C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proceedings of the 2009 International ACM Symposium on Theory of Computing (STOC)*, 2009.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC '06*, pages 265–284, 2006.

[DN03] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *PODS*, pages 202–210. ACM, 2003.

[DN04] Cynthia Dwork and Kobbi Nissim. Privacy-preserving datamining on vertically partitioned databases. In *CRYPTO*, pages 528–544, 2004.

[DR14] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(34):211–407, 2014.

[Dwo11] Cynthia Dwork. A firm foundation for private data analysis. *CACM*, 54(1):86–95, 2011.

[Fre83] David A. Freedman. A note on screening regression equations. *The American Statistician*, 37(2):152–155, 1983.

[Fre98] Yoav Freund. Self bounding learning algorithms. In *COLT*, pages 247–258, 1998.

[HR10]   M. Hardt and G. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *FOCS*, pages 61–70, 2010.

[HTF09]   Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer series in statistics. Springer, 2009.

[HU14]   Moritz Hardt and Jonathan Ullman. Preventing false discovery in interactive data analysis is hard. In *FOCS*, pages 454–463, 2014.

[Lan05]   John Langford. Clever methods of overfitting. `http://hunch.net/?p=22`, 2005.

[LB03]   John Langford and Avrim Blum. Microchoice bounds and self bounding learning algorithms. *Machine Learning*, 51(2):165–179, 2003.

[MMP⁺10]   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. IEEE Computer Society, 2010.

[Ng97]   Andrew Y. Ng. Preventing "overfitting" of cross-validation data. In *ICML*, pages 245–253, 1997.

[NS15]   Kobbi Nissim and Uri Stemmer. On the generalization properties of differential privacy. *CoRR*, abs/1504.05800, 2015.

[PRMN04]   Tomaso Poggio, Ryan Rifkin, Sayan Mukherjee, and Partha Niyogi. General conditions for predictivity in learning theory. *Nature*, 428(6981):419–422, 2004.

[Reu03]   Juha Reunanen. Overfitting in making comparisons between variable selection methods. *Journal of Machine Learning Research*, 3:1371–1382, 2003.

[RF08]   R. Bharat Rao and Glenn Fung. On the dangers of cross-validation. an experimental evaluation. In *International Conference on Data Mining*, pages 588–596. SIAM, 2008.

[RR10]   Aaron Roth and Tim Roughgarden. Interactive privacy via the median mechanism. In *42nd ACM STOC*, pages 765–774. ACM, 2010.

[SSBD14]   Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms.* Cambridge University Press, 2014.

[SSSSS10] Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Learnability, stability and uniform convergence. *The Journal of Machine Learning Research*, 11:2635–2670, 2010.

[SU14]   Thomas Steinke and Jonathan Ullman. Interactive fingerprinting codes and the hardness of preventing false discovery. *arXiv preprint arXiv:1410.1228*, 2014.

[TT15]   Jonathan Taylor and Robert J. Tibshirani. Statistical learning and selective inference. *Proceedings of the National Academy of Sciences*, 112(25):7629–7634, 2015.

[WLF15]   Yu-Xiang Wang, Jing Lei, and Stephen E. Fienberg. Learning with differential privacy: Stability, learnability and the sufficiency and necessity of ERM principle. *CoRR*, abs/1502.06309, 2015.

A   From Max-information to Randomized Description Length

In this section we demonstrate additional connections between max-information, differential privacy and description length. These connections are based on a generalization of description length to randomized algorithms that we refer to as *randomized description length*.

Definition 28. *For a universe $\mathcal{Y}$ let $\mathcal{A}$ be a randomized algorithm with input in $\mathcal{X}$ and output in $\mathcal{Y}$. We say that the output of $\mathcal{A}$ has randomized description length k if for every fixed setting of random coin flips of $\mathcal{A}$ the set of possible outputs of $\mathcal{A}$ has size at most 2^k.*

We first note that just as the (deterministic) description length, randomized description length implies generalization and gives a bound on max-information.

Theorem 29. *Let $\mathcal{A} : \mathcal{X}^n \to \mathcal{Y}$ be an algorithm with randomized description length k and let $\boldsymbol{S}$ be a random dataset over $\mathcal{X}^n$. Assume that $R : \mathcal{Y} \to 2^{\mathcal{X}^n}$ is such that for every $y \in \mathcal{Y}$, $\mathbb{P}[\boldsymbol{S} \in R(y)] \leq \beta$. Then $\mathbb{P}[\boldsymbol{S} \in R(\mathcal{A}(\boldsymbol{S}))] \leq 2^k \cdot \beta$.*

Theorem 30. *Let $\mathcal{A}$ be an algorithm with randomized description length k taking as an input an n-element dataset and outputting a value in $\mathcal{Y}$. Then for every $\beta > 0$, $I_\infty^\beta(\mathcal{A}, n) \leq \log(|\mathcal{Y}|/\beta)$.*

Proof. Let $\boldsymbol{S}$ be any random variable over n-element input datasets and let $\boldsymbol{Y}$ be the corresponding output distribution $\boldsymbol{Y} = \mathcal{A}(\boldsymbol{S})$. It suffices to prove that for every $\beta > 0$, $I_\infty^\beta(\boldsymbol{S}; \boldsymbol{Y}) \leq k + \log(1/\beta)$.

Let R be the set of all possible values of the random bits of $\mathcal{A}$ and let $\mathcal{R}$ denote the uniform distribution over a choice of $r \in R$. For $r \in R$, let $\mathcal{A}_r$ denote $\mathcal{A}$ with the random bits set to r and let $\boldsymbol{Y}_r = \mathcal{A}_r(\boldsymbol{S})$. Observe that by the definition of randomized description length, the range of $\mathcal{A}_r$ has size at most 2^k. Therefore, by Theorem 17, we obtain that $I_\infty^\beta(\boldsymbol{S}; \boldsymbol{Y}_r) \leq \log(2^k/\beta)$.

For any event $\mathcal{O} \subseteq \mathcal{X}^n \times \mathcal{Y}$ we have that

$$\mathbb{P}[(\boldsymbol{S}, \boldsymbol{Y}) \in \mathcal{O}] = \mathop{\mathbb{E}}_{r \sim \mathcal{R}}[\mathbb{P}[(\boldsymbol{S}, \boldsymbol{Y}_r) \in \mathcal{O}]]$$

$$\leq \mathop{\mathbb{E}}_{r \sim \mathcal{R}} \left[\frac{2^k}{\beta} \cdot \mathbb{P}[\boldsymbol{S} \times \boldsymbol{Y}_r \in \mathcal{O}] + \beta \right]$$

$$= \frac{2^k}{\beta} \cdot \mathop{\mathbb{E}}_{r \sim \mathcal{R}}[\mathbb{P}[\boldsymbol{S} \times \boldsymbol{Y}_r \in \mathcal{O}]] + \beta$$

$$= \frac{2^k}{\beta} \cdot \mathbb{P}[\boldsymbol{S} \times \boldsymbol{Y} \in \mathcal{O}] + \beta.$$

By the definition of β-approximate max-information, we obtain that $I_\infty^\beta(\boldsymbol{S}; \boldsymbol{Y}) \leq \log(2^k/\beta)$. $\qquad\square$

We next show that if the output of an algorithm $\mathcal{A}$ has low approximate max-information about its input then there exists a (different) algorithm whose output is statistically close to that of $\mathcal{A}$ while having short randomized description. We remark that this reduction requires the knowledge of the marginal distribution $\mathcal{A}(\boldsymbol{S})$.

Lemma 31. *Let $\mathcal{A}$ be a randomized algorithm taking as an input a dataset of n points from $\mathcal{X}$ and outputting a value in $\mathcal{Y}$. Let $\boldsymbol{Z}$ be a random variable over $\mathcal{Y}$. For $k > 0$ and a dataset S let $\beta_S = \min\{\beta \mid D_\infty^\beta(\mathcal{A}(S)\|\boldsymbol{Z}) \leq k\}$. There exists an algorithm $\mathcal{A}'$ that given $S \in \mathcal{X}^n$, β, k and any $\beta' > 0$,*

  1. the output of $\mathcal{A}'$ has randomized description length $k + \log\ln(1/\beta')$.

  2. for every S, $\Delta(\mathcal{A}'(S), \mathcal{A}(S)) \leq \beta_S + \beta'$.

Proof. Let S denote the input dataset. By definition of β_S, $D_\infty^{\beta_S}(\mathcal{A}(S)\|\boldsymbol{Z}) \leq k$. By the properties of approximate divergence (*e.g.*[DR14]), $D_\infty^{\beta_S}(\mathcal{A}(S)\|\boldsymbol{Z}) \leq k$ implies that there exists a random variable $\boldsymbol{Y}$ such that $\Delta(\mathcal{A}(S), \boldsymbol{Y}) \leq \beta_S$ and $D_\infty(\boldsymbol{Y}\|\boldsymbol{Z}) \leq k$.

For $t = 2^k \ln(1/\beta')$ the algorithm $\mathcal{A}'$ randomly and independently chooses t samples from $\boldsymbol{Z}$. Denote them by $y_1, y_2, \ldots, y_t$. For $i = 1, 2, \ldots, t$, $\mathcal{A}'$ outputs y_i with probability $p_i = \frac{\mathbb{P}[\boldsymbol{Y} = y_i]}{2^k \cdot \mathbb{P}[\boldsymbol{Z} = y_i]}$ and goes to the next sample otherwise. Note that $p_i \in [0, 1]$ and therefore this is a legal choice of probability. When all samples are exhausted the algorithm outputs y_1.

We first note that by the definition of this algorithm its output has randomized description length $\log t = k + \log \ln(1/\beta')$. Let T denote the event that at least one of the samples was accepted. Conditioned on this event the output of $\mathcal{A}'(S)$ is distributed according to $\boldsymbol{Y}$. For each i,

$$\mathbb{E}_{y_i \sim p(\boldsymbol{Z})}[p_i] = \mathbb{E}_{y_i \sim p(\boldsymbol{Z})}\left[\frac{\mathbb{P}[\boldsymbol{Y} = y_i]}{2^k \cdot \mathbb{P}[\boldsymbol{Z} = y_i]}\right] = \sum_{y_i \in \mathcal{Y}} \frac{\mathbb{P}[\boldsymbol{Y} = y_i]}{2^k} = \frac{1}{2^k}.$$

This means that the probability that none of t samples will be accepted is $(1 - 2^{-k})^t \leq e^{t/2^k} \leq \beta'$. Therefore $\Delta(\mathcal{A}'(S), \boldsymbol{Y}) \leq \beta'$ and, consequently, $\Delta(\mathcal{A}'(S), \mathcal{A}(S)) \leq \beta_S + \beta'$. $\qquad\square$

We can now use Lemma 31 to show that if for a certain random choice of a dataset $\boldsymbol{S}$, the output of $\mathcal{A}$ has low approximate max-information then there exists an algorithm $\mathcal{A}'$ whose output on $\boldsymbol{S}$ has low randomized description length and is statistically close to the output distribution of $\mathcal{A}$.

Theorem 32. *Let $\boldsymbol{S}$ be a random dataset in $\mathcal{X}^n$ and $\mathcal{A}$ be an algorithm taking as an input a dataset in $\mathcal{X}^n$ and having a range $\mathcal{Y}$. Assume that for some $\beta \geq 0$, $I_\infty^\beta(\boldsymbol{S}; \mathcal{A}(\boldsymbol{S})) = k$. For any $\beta' > 0$, there exists an algorithm $\mathcal{A}'$ taking as an input a dataset in $\mathcal{X}^n$ such that*

1. *the output of $\mathcal{A}'$ has randomized description length $k + \log \ln(1/\beta')$;*

2. $\Delta((\boldsymbol{S}, \mathcal{A}'(\boldsymbol{S})), (\boldsymbol{S}, \mathcal{A}(\boldsymbol{S}))) \leq \beta + \beta'$.

Proof. For a dataset S let $\beta_S = \min\{\beta \mid D_\infty^\beta(\mathcal{A}(S)\|\mathcal{A}(\boldsymbol{S})) \leq k\}$. To prove this result it suffices to observe that $\mathbb{E}[\beta_{\boldsymbol{S}}] \leq \beta$ and then apply Lemma 31 with $\boldsymbol{Z} = \mathcal{A}(\boldsymbol{S})$ To show that $\mathbb{E}[\beta_{\boldsymbol{S}}] \leq \beta$ let $\mathcal{O}_S \subseteq \mathcal{Y}$ denote an event such that $\mathbb{P}[\mathcal{A}(S) \in \mathcal{O}_S] = 2^k \cdot \mathbb{P}[\mathcal{A}(\boldsymbol{S}) \in \mathcal{O}_S] + \beta_S$. Let $\mathcal{O} = \bigcup_{S \in \mathcal{X}^n} \{(S, \mathcal{O}_S)\}$. Then,

$$\begin{aligned} \mathbb{P}[(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S}) \in \mathcal{O}] &= \mathbb{E}_{S \sim p(\boldsymbol{S})}[\mathbb{P}[(S, \mathcal{A}(S) \in \mathcal{O}_S]] \\ &= \mathbb{E}_{S \sim p(\boldsymbol{S})}\left[2^k \cdot \mathbb{P}[\mathcal{A}(\boldsymbol{S}) \in \mathcal{O}_S] + \beta_S\right] \\ &= 2^k \cdot \mathbb{P}[\boldsymbol{S} \times \mathcal{A}(\boldsymbol{S}) \in \mathcal{O}] + \mathbb{E}[\beta_{\boldsymbol{S}}]. \end{aligned}$$

If $\mathbb{E}[\beta_{\boldsymbol{S}}] > \beta$ then it would hold for some $k' > k$ that $\mathbb{P}[(\boldsymbol{S}, \mathcal{A}(\boldsymbol{S}) \in \mathcal{O}] = 2^{k'} \cdot \mathbb{P}[\boldsymbol{S} \times \mathcal{A}(\boldsymbol{S}) \in \mathcal{O}] + \beta$ contradicting the assumption $I_\infty^\beta(\boldsymbol{S}; \mathcal{A}(\boldsymbol{S})) = k$. We remark that, it is also easy to see that $\mathbb{E}[\beta_{\boldsymbol{S}}] = \beta$. $\qquad\square$

It is important to note that Theorem 32 is not the converse of Theorem 30 and does not imply equivalence between max-information and randomized description length. The primary difference is that Theorem 32 defines a new algorithm rather than arguing about the original algorithm. In addition, the new algorithm requires samples of $\mathcal{A}(\boldsymbol{S})$, that is, it needs to know the marginal distribution on $\mathcal{Y}$. As a more concrete example, Theorem 32 does not allow us to obtain a description-length-based equivalent of Theorem 20 for all i.i.d. datasets. On the other hand, any algorithm that has bounded max-information for all distributions over datasets can be converted to an algorithm with low randomized description length.

Theorem 33. *Let $\mathcal{A}$ be an algorithm over $\mathcal{X}^n$ with range $\mathcal{Y}$ and let $k = I_\infty(\mathcal{A}, n)$. For any $\beta > 0$, there exists an algorithm $\mathcal{A}'$ taking as an input a dataset in $\mathcal{X}^n$ such that*

1. *the output of $\mathcal{A}'$ has randomized description length $k + \log \ln(1/\beta)$;*

2. *for every dataset $S \in \mathcal{X}^n$, $\Delta(\mathcal{A}'(S), \mathcal{A}(S)) \leq \beta$.*

Proof. Let $S_0 = (x, x, \ldots, x)$ be an n-element dataset for an arbitrary $x \in \mathcal{X}$. By Lemma 3 we know that for every $S \in \mathcal{X}^n$, $D_\infty(\mathcal{A}(S)\|\mathcal{A}(S_0)) \leq k$. We can now apply Lemma 31 with $\boldsymbol{Z} = \mathcal{A}(S_0)$, and $\beta' = \beta$ to obtain the result. $\qquad\square$

The conditions of Theorem 33 are satisfied by any ε-differentially private algorithm with $k = \log e \cdot \varepsilon n$. This immediately implies that the output of any ε-differentially private algorithm is β-statistically close to the output of an algorithm with randomized description length of $\log e \cdot \varepsilon n + \log \ln(1/\beta)$. Special cases of this property have been derived (using a technique similar to Lemma 31) in the context of proving lower bounds for learning algorithms [BNS13] and communication complexity of differentially private protocols [MMP+10].

B   Answering Queries via Description Length Bounds

In this section, we show a simple method for answering any adaptively chosen sequence of m statistical queries, using a number of samples that scales only polylogarithmically in m. This is an exponential improvement over what would be possible by naively evaluating the queries exactly on the given samples. Algorithms that achieve such dependence were given in [DFH+14] and [BSSU15] using differentially private algorithms for answering queries and the connection between generalization and differential privacy (in the same way as we do in Section 4.1). Here we give a simpler algorithm which we analyze using description length bounds. The resulting bounds are comparable to those achieved in [DFH+14] using pure differential privacy but are somewhat weaker than those achieved using approximate differential privacy [DFH+14, BSSU15, NS15].

The mechanism we give here is based on the *Median Mechanism* of Roth and Roughgarden [RR10]. A differentially private variant of this mechanism was introduced in [RR10] to show that it was possible to answer exponentially many adaptively chosen *counting* queries (these are queries for an estimate of the empirical mean of a function $\phi : \mathcal{X} \to [0,1]$ on the dataset). Here we analyze a noise-free version and establish its properties via a simple description length-based argument. We remark that it is possible to analogously define and analyze the noise-free version of the Private Multiplicative Weights Mechanism of Hardt and Rothblum [HR10]. This somewhat more involved approach would lead to better (but qualitatively similar) bounds.

Recall that statistical queries are defined by functions $\phi : \mathcal{X} \to [0,1]$, and our goal is to correctly estimate their expectation $\mathcal{P}[\phi]$. The Median Mechanism takes as input a dataset S and an adaptively chosen sequence of such functions $\phi_1, \ldots, \phi_m$, and outputs a sequence of answers $a_1, \ldots, a_m$.

Algorithm Median Mechanism
Input: An upper bound m on the total number of queries, a dataset S and an accuracy parameter τ

1. **Let** $\alpha = \frac{\tau}{3}$.

2. **Let** $\text{Consistent}_0 = \mathcal{X}^{\frac{\log m}{\alpha^2}}$

3. **For** a query ϕ_i **do**

   (a) **Compute** $a_i^{\text{pub}} = \text{median}(\{\mathcal{E}_{S'}[\phi_i] : S' \in \text{Consistent}_{i-1}\})$.

   (b) **Compute** $a_i^{\text{priv}} = \mathcal{E}_S[\phi_i]$.

   (c) **If** $\left| a_i^{\text{pub}} - a_i^{\text{priv}} \right| \leq 2\alpha$ **Then**:

      i. **Output** $a_i = a_i^{\text{pub}}$.
      ii. **Let** $\text{Consistent}_i = \text{Consistent}_{i-1}$.

   (d) **Else**:

      i. **Output** $a_i = \lfloor a_i^{\text{priv}} \rfloor_\alpha$.
      ii. **Let** $\text{Consistent}_i = \{S' \in \text{Consistent}_{i-1} : |a_i - \mathcal{E}_{S'}[\phi_i]| \leq 2\alpha\}$.

Figure 5: Noise-free version of the Median Mechanism from [RR10]

The guarantee we get for the Median Mechanism is as follows:

Theorem 34. *Let $\beta, \tau > 0$ and $m \geq B > 0$. Let $\boldsymbol{S}$ denote a dataset of size n drawn i.i.d. from a distribution $\mathcal{P}$ over $\mathcal{X}$. Consider any algorithm that adaptively chooses functions $\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_m$ while interacting with the Median Mechanism which is given $\boldsymbol{S}$ and values τ, β. For every $i \in [m]$, let $\boldsymbol{a}_i$ denote the answer of the Median Mechanism on function $\boldsymbol{\phi}_i : \mathcal{X} \to [0, 1]$. Then*

$$\mathbb{P}\left[\exists i \in [m], \ |\boldsymbol{a}_i - \mathcal{P}[\boldsymbol{\phi}_i]| \geq \tau\right] \leq \beta$$

whenever

$$n \geq n_0 = \frac{81 \cdot \log |\mathcal{X}| \cdot \log m \cdot \ln(3m/\tau)}{2\tau^4} + \frac{9 \ln(2m/\beta)}{2\tau^2}.$$

Proof. We begin with a simple lemma which informally states that for every distribution $\mathcal{P}$, and for every set of m functions $\phi_1, \ldots, \phi_m$, there exists a small dataset that approximately encodes the answers to each of the corresponding statistical queries.

Lemma 35 ([DR14] Theorem 4.2). *For every dataset S over $\mathcal{X}$, any set $C = \{\phi_1, \ldots, \phi_m\}$ of m functions $\phi_i : \mathcal{X} \to [0, 1]$, and any $\alpha \in [0, 1]$, there exists a data set $S' \in \mathcal{X}^t$ of size $t = \frac{\log m}{\alpha^2}$ such that:*

$$\max_{\phi_i \in C} |\mathcal{E}_S[\phi_i] - \mathcal{E}_{S'}[\phi_i]| \leq \alpha.$$

Next, we observe that by construction, the Median Mechanism (as presented in Figure 5) always returns answers that are close to the empirical means of respective query functions.

Lemma 36. *For every sequence of queries $\phi_1, \ldots, \phi_m$ and dataset S given to the Median Mechanism, we have that for every i:*

$$|a_i - \mathcal{E}_S[\phi_i]| \leq 2\alpha.$$

Finally, we give a simple lemma from [RR10] that shows that the Median Mechanism only returns answers computed using the dataset S in a small number of rounds – for any other round i, the answer returned is computed from the set Consistent$_{i-1}$.

Lemma 37 ([RR10], see also Chapter 5.2.1 of [DR14]). *For every sequence of queries $\phi_1, \ldots, \phi_m$ and a dataset S given to the Median Mechanism:*

$$\left|\{i : a_i \neq a_i^{pub}\}\right| \leq \frac{\log |\mathcal{X}| \log m}{\alpha^2}.$$

Proof. We simply note several facts. First, by construction, $|\text{Consistent}_0| = |\mathcal{X}|^{\log m/\alpha^2}$. Second, by Lemma 35, for every i, $|\text{Consistent}_i| \geq 1$ (because for every set of m queries, there is at least one dataset S' of size $\log m/\alpha^2$ that is consistent up to error α with S on every query asked, and hence is never removed from Consistent$_i$ on any round). Finally, by construction, on any round i such that $a_i \neq a_i^{\text{pub}}$, we have $|\text{Consistent}_i| \leq \frac{1}{2} \cdot |\text{Consistent}_{i-1}|$ (because on any such round, the median dataset S' – and hence at least half of the datasets in Consistent$_{i-1}$ were inconsistent with the answer given, and hence removed.) The lemma follows from the fact that there can therefore be at most $\log\left(|\mathcal{X}|^{\log m/\alpha^2}\right)$ many such rounds. $\square$

Our analysis proceeds by viewing the interaction between the data analyst and the Median Mechanism (Fig. 5) as a single algorithm $\mathcal{A}$. $\mathcal{A}$ takes as input a dataset S and outputs a set of queries and answers $\mathcal{A}(S) = \{\phi_i, a_i\}_{i=1}^m$. We will show that $\mathcal{A}$'s output has short randomized description length (the data analyst is a possibly randomized algorithm and hence $\mathcal{A}$ might be randomized).

Lemma 38. *Algorithm $\mathcal{A}$ has randomized description length of at most*

$$b \leq \frac{\log |\mathcal{X}| \cdot \log m}{\alpha^2} \cdot \left(\log m + \log \frac{1}{\alpha}\right)$$

bits.

Proof. We observe that for every fixing of the random bits of the data analyst the entire sequence of queries asked by the analyst, together with the answers he receives, can be reconstructed from a record of the *indices* i of the queries ϕ_i such that $a_i \neq a_i^{\mathrm{pub}}$, together with their answers a_i (*i.e.* it is sufficient to encode $M := \{(i, a_i) \mid a_i \neq a_i^{\mathrm{pub}}\}$. Once this is established, the lemma follows because by Lemma 37, there are at most $\left(\frac{\log |\mathcal{X}| \log m}{\alpha^2}\right)$ such queries, and for each one, its index can be encoded with $\log m$ bits, and its answer with $\log \frac{1}{\alpha}$ bits.

To see why this is so, consider the following procedure for reconstructing the sequence $(\phi_1, a_1, \ldots, \phi_m, a_m)$ of queries asked and answers received. For every fixing of the random bits of the data analyst, her queries can be expressed as a sequence of functions $(f_1, \ldots, f_m)$ that take as input the queries previously asked to the Median Mechanism, and the answers previously received, and output the next query to be asked. That is, we have:

$$f_1() := \phi_1, \quad f_2(\phi_1, a_1) := \phi_2, \quad f_3(\phi_1, a_1, \phi_2, a_2) := \phi_3, \quad \ldots, \quad f_m(\phi_1, a_1, \ldots, \phi_{m-1}, a_{m-1}) := \phi_m.$$

Assume inductively that at stage i, the procedure has successfully reconstructed $(\phi_1, a_1, \ldots, \phi_{i-1}, a_{i-1}, \phi_i)$, and the set $\mathrm{Consistent}_{i-1}$ (This is trivially satisfied at stage $i = 1$). For the inductive case, we need to recover a_i, ϕ_{i+1}, and $\mathrm{Consistent}_i$. There are two cases we must consider at stage i. In the first case, i is such that $a_i \neq a_i^{\mathrm{pub}}$. But in this case, $(i, a_i) \in M$ by definition, and so we have recovered a_i, and we can compute $\phi_{i+1} = f_{i+1}(\phi_1, a_1, \ldots, \phi_1, a_i)$, and can compute $\mathrm{Consistent}_i = \{S' \in \mathrm{Consistent}_{i-1} : |a_i - \mathcal{E}_{S'}[\phi_i]| \leq 2\alpha\}$. In the other case, $a_i = a_i^{\mathrm{pub}}$. But in this case, by definition of a_i^{pub}, we can compute $a_i = \mathrm{median}(\{\mathcal{E}_{S'}[\phi_i] : S' \in \mathrm{Consistent}_{i-1}\})$, $\phi_{i+1} = f_{i+1}(\phi_1, a_1, \ldots, \phi_i, a_i)$, and $\mathrm{Conisistent}_i = \mathrm{Consistent}_{i-1}$. This completes the argument – by induction, M is enough to reconstruct the entire query/answer sequence. $\quad\square$

Finally, we can complete the proof. By Hoeffding's concentration inequality and the union bound we know that for any every sequence of queries $\phi_1, \ldots, \phi_m$ and a dataset $\boldsymbol{S}$ of size n drawn from the distribution $\mathcal{P}^n$:

$$\mathbb{P}\left[\exists i, \ |\mathcal{E}_{\boldsymbol{S}}[\phi_i] - \mathcal{P}[\phi_i]| \geq \alpha\right] \leq 2m \cdot \exp\left(-2n\alpha^2\right).$$

Applying Theorem 29 to the set $R(\phi_1, a_1, \ldots, \phi_m, a_m) = \{S \mid \exists i, \ |\mathcal{E}_S[\phi_i] - \mathcal{P}[\phi_i]| \geq \alpha\}$ we obtain that for the queries $\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_m$ generated on the dataset $\boldsymbol{S}$ and corresponding answers of the Median Mechanism $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ we have

$$
\begin{aligned}
\mathbb{P}\left[\exists i, \ |\mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i] - \mathcal{P}[\boldsymbol{\phi}_i]| \geq \alpha\right] &\leq 2^b \cdot 2m \cdot \exp\left(-2n\alpha^2\right). \\
&\leq 2^{\frac{\log |\mathcal{X}| \log m}{\alpha^2} \cdot \log(m/\alpha)} \cdot 2m \cdot \exp\left(-2n\alpha^2\right).
\end{aligned}
$$

Solving, we have that whenever:

$$n \geq \frac{\log |\mathcal{X}| \cdot \log m}{2\alpha^4} \cdot \ln(m/\alpha) + \frac{\ln(2m/\beta)}{2\alpha^2},$$

we have: $\mathbb{P}\left[\exists i, \ |\mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}_i] - \mathcal{P}[\boldsymbol{\phi}_i]| \geq \alpha\right] \leq \beta$. Combining this with Lemma 36 we have:

$$\mathbb{P}\left[\exists i \in [m], \ |\boldsymbol{a}_i - \mathcal{P}[\boldsymbol{\phi}_i]| \geq 3\alpha\right] \leq \beta.$$

Plugging in $\tau = 3\alpha$ gives the theorem. $\quad\square$