Private and Communication-Efficient Algorithms for Entropy Estimation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Modern statistical estimation is often performed in a distributed setting where each sample belongs to single user who shares their data with a central server. Users are typically concerned with preserving the privacy of their sample, and also with minimizing the amount of data they must transmit to the server. We give improved private and communication-efficient algorithms for estimating several popular measures of the entropy of a distribution. All of our algorithms have constant communication cost and satisfy local differential privacy. For a joint distribution on several variables whose conditional independence graph is a tree, we describe algorithms for estimating Shannon entropy that require a number of samples that is linear in the number of variables, compared to the quadratic sample complexity of prior work. We also describe an algorithm for estimating Gini entropy whose sample complexity has no dependence on the support size of the distribution and can be implemented using a single round of concurrent communication between the users and the server, while the previously best-known algorithm has high communication cost and requires the server to facilitate interaction between the users. Finally, we describe an algorithm for estimating collision entropy that matches the space and sample complexity of the best known algorithm but generalizes it to the private and communication-efficient setting.

1 Introduction

Statistical estimation has traditionally focused on minimizing the number of samples needed to estimate properties of a distribution. In the 'big data' era, statisticians and computer scientists have also tried to minimize the space complexity of estimation algorithms, particularly in the streaming setting. More recently, the increasing prevalence of mobile computing has led to a focus on the privacy and communication costs of statistical estimation. In this paper, we focus on a setting in which a set of users each draw one sample from a distribution and share information about their sample with a central server, who uses the collected data to estimate a property of the distribution. Users are concerned with preserving the privacy of their sample, and also with minimizing the amount of data that is transmitted to the server.

For example, consider a wearable electronic device that records health metrics about a user (*e.g.*, heart rate, sleep quality, *etc.*) and reports that information to the device's manufacturer, who uses the collected data to estimate aggregate statistics about the user population. Individual health metrics are obviously very sensitive and should be kept private from the manufacturer, and if the number of metrics is large, then transmitting them can be prohibitive for a device with low power or limited bandwidth. Similar concerns also arise in the study of federated learning [24, 27].

In this paper, we study private and communication-efficient algorithms for estimating the entropy of a distribution. Specifically, we give algorithms for estimating the following properties that are

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

widely-used in many scientific fields to quantify the uncertainty, diversity and spread of a discrete distribution:

- ***Shannon entropy***, the most fundamental quantity in information theory.
- ***Gini entropy*** (also known as *Tsallis entropy of entropic-index 2* or *second frequency moment*), which has been used to measure ecological diversity [28], market competition among firms [19], the effective size of political parties [23], and the suitability of features to split on during decision tree learning [26].
- ***Collision entropy*** (also known as *Rényi entropy of order 2*), which has been used to measure the quality of random number generators [29] and to determine the number of reads needed to reconstruct a DNA sequence [25].

Our algorithms are implemented in either the *non-interactive* model, in which all users simultaneously exchange information with the server during a single round of communication, or the stronger *sequentially interactive* model, in which the server queries users one at a time, possibly in an adaptive manner [20]. When analyzing the communication complexity of an algorithm we prove bounds on the number of bits that each user transmits to the server. However, the server is allowed to broadcast an arbitrary amount of information to the users (this is also called the *blackboard* model [15]), including shared random bits (also known as the *public coin* model [1, 2, 21]). When analyzing the privacy of our algorithms we use the framework of *local differential privacy* [16] which ensures that the server learns very little about each user's data.

Our contributions:

- A sequentially interactive α-local differentially private algorithm for estimating the Shannon entropy of a joint distribution on d variables within ϵd error using $\tilde{O}(d/\alpha^2\epsilon^3)$ samples and $O(1)$ bits per sample. Our analysis assumes that each of the d variables has a constant support size and that their conditional independence graph is a tree. We also describe algorithms that have better dependence on $1/\epsilon$ in certain special cases, such as when the tree has low diameter or is a chain. Our algorithms achieve $O(1)$ communication complexity by observing only two or three variables from any single sample; we call these *pair* and *triplet* observations. The only previously known algorithm for estimating the Shannon entropy of a tree-structured distribution from pair observations is a non-interactive algorithm due to Chou and Liu [11]. We prove that any non-interactive algorithm requires $\Omega(d^2)$ pair observations to achieve $O(d)$ error. We also prove that $\Omega(d/\epsilon)$ pair observations are necessary to achieve $O(\epsilon d)$ error.
- A non-interactive α-local differentially private algorithm for estimating the Gini entropy of a distribution within ϵ error using $O(b^2 \max\{1-g, 2^{-b}\}/(\alpha^2\epsilon^2))$ samples, b bits per sample, and $\tilde{O}(b)$ space, where $g \in [0,1]$ is the Gini entropy of the distribution. The best previous algorithm [9] has the same sample complexity, even when $b = 1$, but is sequentially interactive, and also requires $\Omega(k)$ bits per sample and $\Omega(k)$ space, where k is the support size of the distribution. Our error bound also holds with high probability instead of only in expectation.
- A non-interactive α-local differentially private algorithm for estimating the collision entropy of a distribution with support size k within ϵ error using $\tilde{O}(b^2k^2/(\alpha^2\epsilon^2 \min\{k, 2^b\}))$ samples, b bits per sample, and $\tilde{O}(b)$ space. Setting $b = \log k$ and $\alpha = O(1)$ recovers the sample and space complexity guarantees of the non-interactive algorithm from [29] up to logarithmic factors, and thus, our algorithm generalizes the previous one to the private and communication-efficient setting.

2 Related Work

There is a very extensive literature on distributed statistical estimation under communication constraints (see [33] for the paper that appears to have started this thread). Variations on the problem include whether communication is allowed between users, whether communication happens in one or multiple rounds, whether there is a shared source of randomness among the users, and whether communication is limited per-user or only cumulatively for all users.

Many previous results in this area bound the sample and communication complexity of estimating the parameters of a distribution P_θ, where $\theta \in \Theta$ (see *e.g.* [18]). This problem class includes discrete distribution estimation, where the guarantees are usually stated as bounds on the relative entropy or

total variation distance between the estimated and true distribution (see *e.g.* [5]). Other problems of interest are mean estimation [30] and heavy hitter estimation [4].

There has also been significant interest in differentially private statistical estimation, and of particular relevance is work by [3], who gave private algorithms for estimating certain functionals of a distribution, including Shannon entropy. However, they used the central model of differential privacy, while in this paper we prove guarantees using the local model.

3 Entropy Measures

The *Shannon*, *Tsallis*, and *Rényi* entropy of a discrete random variable X are defined as

$$\text{(Shannon)} \qquad H(X) = -\sum_x \Pr[X = x] \log \Pr[X = x], \tag{1}$$

$$\text{(Tsallis)} \qquad T_\gamma(X) = \frac{1}{\gamma - 1}\big(1 - \sum_x \Pr[X = x]^\gamma\big), \tag{2}$$

$$\text{(Rényi)} \qquad R_\gamma(X) = \frac{1}{1-\gamma} \log\big(\sum_x (\Pr[X = x])^\gamma\big), \tag{3}$$

where γ in (2) and (3) is a free parameter satisfying $\gamma > 0$ and $\gamma \neq 1$. Both Tsallis and Rényi entropy are generalizations of Shannon entropy in the sense that $\lim_{\gamma\to 1} T_\gamma(X) = \lim_{\gamma\to 1} R_\gamma(X) = H(X)$.

In this paper, we describe algorithms for estimating Shannon entropy and special cases of Tsallis and Rényi entropy that are widely used in many scientific fields: $T_2(X)$, also called the *Gini entropy*, and $R_2(X)$, also called the *collision entropy*. Substituting $\gamma = 2$ into the definitions above and using the abbreviations $G(X) \equiv T_2(X)$ and $C(X) \equiv R_2(X)$ we have

$$G(X) = 1 - \sum_x \Pr[X = x]^2, \qquad C(X) = -\log\big(\sum_x \Pr[X = x]^2\big).$$

Gini entropy is so-called because it is equivalent to the Gini index [32], and collision entropy takes its name from the observation that if X and X' are independent and identically distributed then $C(X) = -\log \Pr[X = X']$.

The *joint Shannon entropy* $H(X_1, \ldots, X_d)$ of a set of random variables $X_1, \ldots X_d$ is the Shannon entropy $H(X)$ of the random variable $X = (X_1, \ldots, X_d)$. We write the abbreviated term *joint entropy* when the use of Shannon entropy is obvious from context.

The *mutual information* between two random variables X and Y and their *conditional mutual information* given the random variable Z are defined as

$$I(X;Y) = H(X) + H(Y) - H(X,Y),$$
$$I(X;Y \mid Z) = H(X,Z) + H(Y,Z) - H(X,Y,Z) - H(Z).$$

4 Estimation Algorithms and Evaluation Criteria

A set of n users and a central server cooperate according to the following protocol to estimate the entropy of a random variable X:

1. Each user $i \in [n]$ draws an independent sample x_i according to the distribution of X.
2. For several rounds:
   (a) The server sends information to a subset of the users.
   (b) A subset of the users send information back to the server.
3. The server outputs an estimate of the entropy of X.

An *estimation algorithm* specifies the steps that each user and the server perform to implement the above protocol. The algorithm is *non-interactive* if the protocol consists of a single round in which all users participate. In a non-interactive algorithm the server cannot adapt its queries to users based on responses from other users, since the server communicates with all the users concurrently. An

algorithm is *sequentially interactive* if each round consists of communication with a single user, who is never contacted again. Sequential interactivity enables the server to query users adaptively [20].

We evaluate estimation algorithms according to the following criteria:

- *Sample complexity*: The number of users from whom the server requests data.
- *Space complexity*: The space used by the server when executing the algorithm.
- *Communication complexity*: The maximum number of bits transmitted by any single user to the server. Note that the amount of information sent by the server to the users is not counted when determining communication complexity.
- *Privacy*: Let x_i be the sample belonging to user i and o_i be the data observed by the server from user i. We say that an algorithm satisfies α-*local differential privacy* if

$$\Pr[o_i \in O \mid x_i = x] \leq e^{\alpha} \Pr[o_i \in O \mid x_i = x']$$

  for any user i, measurable set O, and possible sample values x, x'.
- *Error*: The difference between the true entropy and the estimate output by the server.

5 Estimating Shannon Entropy of Tree-structured Distributions

In this section we assume that $X = (X_1, \ldots, X_d)$ is a vector of d discrete variables, and that the support size of each variable X_i is a constant (*e.g.*, each variable is Boolean). We also assume that X is *tree-structured*, which means that there exists a rooted tree T with d nodes such that for any node $i \in [d]$ we have

$$\Pr[X_i \mid X_{-i}] = \Pr[X_i \mid X_{\mathrm{pa}_T(i)}],$$

where $X_{-i} = (X_1, \ldots X_{i-1}, X_{i+1}, \ldots, X_d)$ and the node $\mathrm{pa}_T(i)$ is the parent of node i in tree T. If i is the root of the tree then we define $\Pr[X_i \mid X_{\mathrm{pa}_T(i)}] = \Pr[X_i]$. Loosely speaking, the tree-structured assumption implies that the only correlations among the X_i's are pairwise correlations. If T is a chain or a star we say that X is *chain-structured* and *star-structured*, respectively.

5.1 Estimating Shannon Entropy When Support Size Is Small

Before proceeding to describe algorithms for estimating the Shannon entropy of tree-structured distributions, we use existing results for private distribution estimation to devise a local differentially private estimator for Shannon entropy that is sample and communication efficient when the support size of the distribution is small. The server will repeatedly invoke this algorithm as a subroutine in the sections below.

First let us recall that the Shannon entropy can be upper bounded according to Theorem 17.3.3 of [12] as

$$|H(X_{\mathbf{p}}) - H(X_{\mathbf{p}'})| \leq \|\mathbf{p} - \mathbf{p}'\|_1 \log \frac{c}{\|\mathbf{p} - \mathbf{p}'\|_1}$$

where $X_{\mathbf{p}}$ and $X_{\mathbf{p}'}$ are two discrete random variables with support size c and parameters $\mathbf{p}$ and $\mathbf{p}'$. Next, we apply a local differentially private learning algorithm for discrete distribution due to [4] that learns the parameters of a discrete distribution with small L_1 error. The following theorem combines these two results.

Theorem 5.1. *For any discrete distribution X with support size c and for any $\epsilon > 0$, there exists an estimator satisfying α-local differential privacy that estimates $H(X)$ within ϵ error using $n = O(c^2 \log \frac{1}{\delta}/(\epsilon^2\alpha^2))$ samples with probability $1 - \delta$ when $\alpha \in (0, 1)$.*

This algorithm from Theorem 5.1 can be used to privately estimate the entropy $H(X_i)$, mutual information $I(X_i; X_j)$, and conditional mutual information $I(X_i; X_j \mid X_k)$ of any variables X_i, X_j and X_k by using $O\left(\frac{\log \frac{1}{\delta}}{\alpha^2\epsilon^2}\right)$ samples per estimate and $O(1)$ bits per sample, since each of these variables has constant support size, and both mutual information and conditional mutual information can be expressed in terms of entropies (see Section 3). We call such an estimate $(\alpha, \epsilon, \delta)$-*good*.

5.2 Algorithm for General Trees

Note that the support size of X can be exponential in d. In the worst case, estimating the entropy of a distribution with support size k within constant error requires $\tilde{\Theta}(k)$ samples [17]. However the tree-structure of X can be exploited to significantly reduce the sample complexity. In their seminal paper, Chow and Liu [11] proved the identity

$$H(X) = \sum_{i=1}^{d} H(X_i) - \max_{T} \sum_{i=1}^{d} I(X_i; X_{\mathrm{pa}_T(i)}), \tag{4}$$

for any tree-structured random variable X, where the maximization is taken over all possible trees connecting the d variables.

Eq. (4) suggests a communication-efficient algorithm for estimating the entropy of X, which is known as the *Chow-Liu algorithm*: First, estimate each marginal entropy $H(X_i)$ and each mutual information $I(X_i; X_j)$. Next, compute a maximum spanning tree on the d variables, where the weight of each edge (X_i, X_j) is the estimate of the mutual information $I(X_i; X_j)$. Finally, plug all of the $H(X_i)$ estimates, the maximum spanning tree, and the corresponding $I(X_i; X_j)$ estimates, into Eq. (4).

The Chow-Liu algorithm requires $\Omega(d^2)$ samples, since it computes the mutual information between every pair of variables in order to compute a maximum spanning tree. However, estimating the right-hand side of Eq. (4) only requires estimating the *weight* of the maximum spanning tree, which is significantly easier than finding the tree itself. Algorithm 1 adapts a technique from [10] that estimates the weight of the maximum spanning tree of a graph in time that is sublinear in the number of edges in the graph. The basic idea is to select nodes of the graph at random and use breadth-first search to determine the size of each of their connected components if we were to drop edges that do not meet a weight threshold, short-circuiting the search when the size becomes too large. These quantities are combined to estimate the weight of the maximum spanning tree. In our case, an edge weight is a mutual information between a pair of variables, which we estimate from pair observations.

Algorithm 1 Shannon entropy estimation for tree-structured distribution

```
Let M = ⌈2/ε⌉ and R = ⌈1/ε²⌉
for m = 1, ..., M do
    for r = 1, ..., R do
        Choose positive integer Z randomly according to Pr[Z ≥ z] = 1/z.
        Choose i* uniformly at random from [d].
        if Z ≥ 2/ε then γ_mr ← 0.
        else                                                    ▷ Breadth-first search
            Initialize queue to contain i*.
            while queue length is non-zero and shorter than Z do
                Remove i from front of queue.
                for j = [d] \ {i} do
                    Server computes (α, ε, δ)-good estimate Î_ij of I(X_i; X_j).
                    if Î_ij ≥ εm then add j to back of queue.
            if queue length is zero then γ_mr ← 0 else γ_mr ← 1.
    ĉ_m ← (d/R) Σ_{r=1}^{R} γ_mr.
Ŵ ← εMd − ε Σ_{m=1}^{M} ĉ_m
Server computes (α, ε, δ)-good estimate of each entropy in the first sum in Eq. (4).
Let Ŝ be the sum of the entropy estimates.
Return Ĥ = Ŝ − Ŵ.
```

Theorem 5.2. *Algorithm 1 is α-locally differentially private and has $O(1)$ communication complexity. The number of samples requested by the server is $O\left(\frac{d\log(\frac{1}{\delta})}{\alpha^2\epsilon^3}\right)$. Let $\hat{H}$ be the output of the algorithm. If X is tree-structured then $|\hat{H} - H(X)| \leq \epsilon d$ with probability $1-\delta$.*

5.3 Algorithm for a Chain

Verma and Pearl [31] observed that if X is chain-structured with chain T then for any triplet (X_i, X_j, X_k), if X_k is on the unique path in T between X_i and X_j, then $I(X_i; X_j|X_k) = 0$. This observation alone does not help to recover the chain, since the conditional mutual information $I(X_i; X_j|X_k)$ can also be zero for X_i, X_j and X_k when X_k is not on the path between X_i and X_j in the chain. Nevertheless, under the mild assumption the mutual information between every pair of variables is distinct, we can recover the chain T by estimating the conditional mutual information of triplets of variables.

The algorithm that is similar to a sorting algorithm like merge sort which requires $O(d \log_2 d)$ pairwise comparisons over d items. However, we cannot compare pairs explicitly like in a sorting problem, but for any triplets (X_i, X_j, X_k), we can decide locally which item is between the other two: *i.e.*, $X_i \leftrightarrow X_j \leftrightarrow X_k$, $X_i \leftrightarrow X_k \leftrightarrow X_j$ or $X_k \leftrightarrow X_i \leftrightarrow X_j$ in the chain, by estimating conditional mutual information. If X is chain-structured then one of $I(X_i; X_j|X_k)$ or $I(X_i; X_k|X_j)$ or $I(X_j; X_k|X_i)$ has to be zero (due to the observation from Verma and Pearl). This suggests Algorithm 2, which inserts the variables in a chain one by one in a sequential manner. Algorithm 3 is called by Algorithm 2 as a subroutine which seeks to find the position where to insert.

Theorem 5.3. *Algorithm 4 is α-locally differentially private and has $O(1)$ communication complexity. The number of samples requested by the server is $O\left(\frac{d \log \frac{d}{\delta}}{\alpha^2 \epsilon^2}\right)$. Let $\hat{H}$ be the output of the algorithm. If X is chain-structured and $|I(X_i; X_j) - I(X_j; X_k)| \geq \epsilon$ then $|\hat{H} - H(X)| \leq \epsilon d$ with probability $1 - \delta$.*

Algorithm 2 Shannon entropy estimation for chain-structured distribution

$S = [d]$, $C = \emptyset$, pick an arbitrary $i, j, k \in S$ and set $S = S \setminus \{i, j, k\}$.
Server computes $(\alpha, \epsilon, \delta)$-good estimates $\hat{I}(X_i; X_j \mid X_k)$, $\hat{I}(X_i; X_k \mid X_j)$ and $\hat{I}(X_k; X_j \mid X_i)$.
if $\hat{I}(X_i; X_j \mid X_k) > \epsilon$ **then** $\boldsymbol{x}_1 = (i, k, j)$
else if $\hat{I}(X_i; X_k \mid X_j) > \epsilon$ **then** $\boldsymbol{x}_1 = (i, j, k)$
else if $\hat{I}(X_k; X_j \mid X_i) > \epsilon$ **then** $\boldsymbol{x}_1 = (j, i, k)$
for $i \in (1, \ldots, d-3)$ **do**
  Pick item j from S and set $S = S \setminus \{j\}$ and set $r = x_{i,1}$ and $p = x_{i,i+2}$
  Server computes $(\alpha, \epsilon, \delta)$-good estimates $\hat{I}(X_j; X_p \mid X_o)$, $\hat{I}(X_r; X_j \mid X_p)$.
  if $\hat{I}(X_j; X_p \mid X_r) > \epsilon$ **then** $\boldsymbol{x}_{i+1} = (j, \boldsymbol{x}_i)$ ▷ Attach X_j to the head of the chain
  else if $\hat{I}(X_o; X_j \mid X_p) > \epsilon$ **then** $\boldsymbol{x}_{i+1} = (\boldsymbol{x}_i, j)$ ▷ Attach X_j to the tail of the chain
  else ▷ Insert X_j into the chain defined by $\boldsymbol{x}_i$
    $\ell = \textsc{TernarySearch}(\boldsymbol{x}_i, 1, i+2, j)$ ▷ Defined in Algorithm 3
    $\boldsymbol{x}_{i+1} = (\boldsymbol{x}_i[1, \ldots, \ell], j, \boldsymbol{x}_i[\ell+1, \ldots, i+2])$
Create chain T according to the order defined by $\boldsymbol{x}_{d-2}$.
Server computes $(\alpha, \epsilon, \delta)$-good estimate of each term in Eq. (4) using T and returns their sum $\hat{H}$.

Algorithm 3 $\textsc{TernarySearch}(\boldsymbol{x}, \ell_l, \ell_h, j)$

if $\ell_l = \ell_h - 1$ **then return** ℓ_l
Pick the median element $k = \lceil (\ell_h + \ell_l) \rceil$, and set $i = x_{\ell_l}$ and $o = x_{\ell_h}$
Server computes $(\alpha, \epsilon, \delta)$-good estimate $\hat{I}(X_i; X_k \mid X_j)$.
if $\hat{I}(X_i; X_k \mid X_j) > \epsilon$ **then return** $\textsc{TernarySearch}(\boldsymbol{x}, i, k, j)$
else return $\textsc{TernarySearch}(\boldsymbol{x}, k, o, j)$

5.4 Algorithm for a Star

If X is star-structured then recovering the star T is a matter of identifying the center of the star, which can be done by computing the mutual information between only $\tilde{O}(d)$ pairs of variables. The algorithm picks a random marginals X_i and takes a Prim step, *i.e.*, choosing the neighboring node, say X_j, for which the mutual information is the largest. The edge between X_i and X_i is in the

maximal spanning tree, when we assume that each edges weight is different. Next, the algorithm estimates $\sum_{k \neq i} I(X_i, X_k)$ and $\sum_{k \neq j} I(X_j, X_k)$ to decide whether X_i or X_j is the center node. The Algorithm is presented in Algorithm 4 and the sample complexity in 5.4.

Theorem 5.4. *Algorithm 4 is α-locally differentially private and has $O(1)$ communication complexity. The number of samples requested by the server is $O\left(\frac{d \log \frac{d}{\delta}}{\alpha^2 \epsilon^2}\right)$. Let $\hat{H}$ be the output of the algorithm. If X is star-structured and $|I(X_i; X_j) - I(X_j; X_k)| \geq \epsilon$ then $|\hat{H} - H(X)| \leq \epsilon d$ with probability $1 - \delta$.*

Algorithm 4 Shannon entropy estimation for star-structured distribution

1: Pick a $i \in [d]$ uniformly at random.
2: Server computes $(\alpha, \epsilon, \delta)$-good estimate $\hat{I}(X_i, X_j)$ for all $j \in [d] \setminus \{i\}$.
3: Find $k = \arg\max_{j \in [d] \setminus \{i\}} \hat{I}(X_i, X_j)$
4: Server computes $(\alpha, \epsilon, \delta)$-good estimate $\hat{I}(X_k, X_j)$.
5: **if** $\sum_j \hat{I}(X_i, X_j) > \sum_j \hat{I}(X_k, X_j)$ **then** let T be a star with X_i as center
6: **else** let T be a star with X_k as the center.
7: Server computes $(\alpha, \epsilon, \delta)$-good estimate of each term in Eq. (4) using T and returns their sum $\hat{H}$.

5.5 Lower Bounds

We prove sample complexity lower bounds for estimating the Shannon entropy of a tree-structured distribution from pair observations. Our first lower bound focuses on the non-interactive case, when the algorithm must select all the pairs in advance. The second claim is more general, and holds for all sequentially interactive algorithms.

Theorem 5.5. *For any non-interactive algorithm that uses $o(d^2)$ pair observations to estimate Shannon entropy, there exists a tree-structured distribution over $\{0,1\}^d$ such that the error of the algorithm is $\Omega(d)$ with constant probability.*

Theorem 5.6. *For any $\epsilon > 0$ and for any sequentially interactive algorithm that uses $o(d/\epsilon)$ pair observations to estimate Shannon entropy, there exists a tree-structured distribution on $\{0,1\}^d$ such the error of the algorithm is $\Omega(\epsilon \cdot d)$ with constant probability.*

The lower bound given in Theorem 5.5 is based on Turán's theorem [8], which we use to show that for any algorithm with sub-quadratic sample complexity and for any constant $C \in (0,1)$, there is a graph with d nodes containing $C \cdot d$-clique – when d is large enough – such that the algorithm does not observe any edge of that clique. This implies that the additive error of the algorithm is linear in d. The lower bound for sequentially interactive algorithms in Theorem 5.6 is based on a information theoretical approach. Interestingly, our construction of problem instances for which we applied Le Cam's theorem is fairly simple, since it contains d independent random variables in every case. Nevertheless, this lower bound shows that Algorithm 1 is optimal in d. Moreover, note that Theorem 5.6 also holds for non-interactive algorithms.

5.6 Comparison to Prior Work

To the best of our knowledge, the Chow-Liu algorithm is the only published method for estimating the entropy of a distribution that takes advantage of its tree structure. Since the algorithm is non-interactive, the lower bound in Theorem 5.5 shows that our algorithms have provably better sample complexity. The Chow-Liu algorithm can also be used to estimate the distribution itself, not just its entropy, and it has recently been shown [7, 14] that the algorithm has optimal sample complexity when given full observations (*i.e.*, samples of the entire vector $(X_1, \ldots, X_d)$ and not just pairs or triplets of the variables). The root cause of this difference appears to be the fact that it is significantly easier to estimate the weight of the maximum spanning tree than finding the tree itself.

6 Estimating Gini and collision entropy

Algorithm 5 below estimates both the Gini entropy and collision entropy of a random variable X using samples from the distribution of X while observing only b bits per sample. In the algorithm, the server first partitions all users into pairs (we assume for simplicity that the number of users is even). The server then distributes a b-bit hash function to each user, along with a distinct salt to each user pair. Each user then hashes their sample along with their salt, and returns the hash value to the server. The server computes entropy estimates based on the number of observed hash collisions across all pairs. In Algorithm 5 we let $\langle x, y\rangle$ denotes a binary string that encodes x, followed by a delimiter, followed by y.

Algorithm 5 Gini and collision entropy estimation

1: Each user $i \in [n]$ draws sample x_i independently from the distribution of X.
2: Server partitions the n users into $\frac{n}{2}$ disjoint pairs.
3: Let $q_i \in \left[\frac{n}{2}\right]$ be the index of the pair containing user i.
4: Server transmits q_i and hash function $h : \{0,1\}^* \mapsto \{0,1\}^b$ to each user i.
5: Each user i generates a b-bit hash value $h_i = h(\langle q_i, x_i\rangle)$ for their sample x_i.
6: Each user i lets $\hat{h}_i = h_i$ with probability $\lambda = \frac{e^\alpha}{2^b + e^\alpha}$ and otherwise draws $\hat{h}_i$ uniformly from $[2^b]$.
7: Server receives $\hat{h}_i$ from each user i.
8: If pair q contains users i and j then let $c_q = \mathbf{1}[\hat{h}_i = \hat{h}_j]$ indicate whether a hash collision was observed for pair q.
9: Server computes $\bar{c} = \left(\frac{2}{(1-\lambda)n}\sum_q c_q\right) - \frac{\lambda}{2^b}$.
10: Server outputs $\hat{G} = \frac{2^b}{2^b-1}\bar{c} - \frac{1}{2^b-1}$ and $\hat{C} = -\log\left(1 - \hat{G}\right)$.

The analysis of Algorithm 5 is based on the observation that if X and X' are independent and identically distributed then the Gini entropy is equal to $1 - \Pr[X = X']$ and the collision entropy is equal to $-\log \Pr[X = X']$. If the server observed each sample directly then it could estimate $\Pr[X = X']$ using collision frequency, *i.e.*, the fraction of sample pairs (x_i, x_j) such that $x_i = x_j$. However, the server only observes a b-bit hash of each sample. Among sample pairs in which there is a true collision, all of them also produce a hash collision. Among samples pairs in which there is not a true collision, about a $\frac{1}{2^b}$ fraction of them produce a hash collision. Therefore the true collision frequency can be estimated using an appropriately bias-corrected hash collision frequency, and the server uses this estimate to approximate the Gini entropy and collision entropy.

The analysis of Algorithm 5 is given in Theorem 6.1 below. As is customary, for the analysis we assume that the hash function h is constructed by assigning each element of its domain to a uniform random element of its range [6]. See the Appendix for the proof of the theorem.

Theorem 6.1. *Algorithm 5 is α-locally differentially private and has $\tilde{O}(b)$ communication complexity and $\tilde{O}(b)$ space complexity. Let $\hat{G}$ and $\hat{C}$ be the outputs of the algorithm. Let $\alpha, \epsilon, \delta \in (0,1)$. If $n = \Omega\left(\frac{b^2 \max\{1-G(X), 2^{-b}\}\log\frac{1}{\delta}}{\alpha^2\epsilon^2}\right)$ then $|\hat{G} - G(X)| \le \epsilon$ with probability at least $1-\delta$. Also, if X has support size k and $n = \Omega\left(\frac{b^2 k^2 \log\frac{1}{\delta}}{\alpha^2\epsilon^2 \min\{k, 2^b\}}\right)$ then $|\hat{C} - C(X)| \le \epsilon$ with probability at least $1-\delta$.*

6.1 Comparison to Prior Work

Recall that Gini entropy is equivalent to the second frequency moment (up to a constant). Local differentially private algorithms for estimating frequency moments were recently studied in [9]. Letting $b = 1$ in Algorthm 5 yields a sample complexity of $\tilde{O}(1/\alpha^2\epsilon^2)$, which matches the sample complexity of the sequentially interactive algorithm for estimating the second frequency moment from [9]. However our algorithm is non-interactive, which is a much weaker communication model, and also only uses 1 bit per sample and $\tilde{O}(1)$ space, while the previous algorithm uses $\Omega(k)$ bits per sample and $\Omega(k)$ space, where k is the support size of the distribution. The authors in [9] asked whether there is a non-interactive algorithm for privately estimating frequency moments with a sample

complexity that is independent of the distribution's support size, and we affirmatively answer this open question for the second frequency moment.

The best known algorithm for estimating collision entropy using $\tilde{O}(1)$ space is due to [29]. The sample complexity of their algorithm is $\tilde{O}\left(k/\epsilon^2\right)$ and its communication complexity is $O(\log k)$ communication per user, where k is the support size of the distribution. Letting $b = \log k$ and $\alpha = O(1)$ in Algorithm 5 recovers these results (up to logarithmic factors), and using smaller values for b or α generalizes the previous algorithm to the private and communication-efficient setting. Also, it was shown in [13] that (conditioned on a plausible conjecture) any algorithm that estimates collision entropy to within $O(1)$ error using $O(1)$ space requires $\Omega(k)$ samples. Therefore our algorithm is likely to be Pareto optimal with respect to the sample complexity-space complexity trade-off.

7 Conclusion and future work

Estimating entropy is of importance in many practical applications. In this paper, we studied three widely used entropy measures: Shannon, Gini and collision entropy. We described estimation algorithms for each entropy that require minimal communication and satisfy local differential privacy.

Our sequentially interactive algorithm for estimating Shannon entropy observes only second order marginals, and it has a sample complexity $O(d/\epsilon^3)$. Our approach relies on the celebrated Chow-Liu approximation [11]. This is an improvement of the sample complexity of the original non-interactive Chow-Liu algorithm whose sample complexity is $\Omega(d^2)$. We also identified some special cases when the underlying graphical model of the joint distribution is either a chain or star graph. In these cases, the proposed algorithms have a sample complexity of $\tilde{O}(d \log d/\epsilon^2)$.

Our work on on Gini and collision entropy estimation also improved on the state-of-the-art, either by matching the sample complexity of previous work but in a weaker communication model and with significantly better communication complexity, or by generalizing the best known algorthm to the private and commnication-efficient setting.

A natural extension of our work on Shannon entropy estimation is to consider higher order marginals in the Chow-Liu decomposition [22]. When higher order marginals is used, discover the underlying structure of the joint distribution is already computationally challenging, unlike in the second order case when it reduces to a maximum spanning tree problem.

References

[1] Jayadev Acharya, Clement Canonne, Cody Freitag, and Himanshu Tyagi. Test without trust: Optimal locally private distribution testing. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 2067–2076. PMLR, 2019.

[2] Jayadev Acharya, Clément L Canonne, and Himanshu Tyagi. Inference under information constraints: Lower bounds from chi-square contraction. In *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 3–17. PMLR, 2019.

[3] Jayadev Acharya, Gautam Kamath, Ziteng Sun, and Huanyu Zhang. Inspectre: Privately estimating the unseen. In *International Conference on Machine Learning*, pages 30–39. PMLR, 2018.

[4] Jayadev Acharya and Ziteng Sun. Communication complexity in locally private distribution estimation and heavy hitters. In *International Conference on Machine Learning*, pages 51–60. PMLR, 2019.

[5] Jayadev Acharya, Ziteng Sun, and Huanyu Zhang. Hadamard response: Estimating distributions privately, efficiently, and with little communication. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1120–1129. PMLR, 2019.

[6] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[7] Arnab Bhattacharyya, Sutanu Gayen, Eric Price, and NV Vinodchandran. Near-optimal learning of tree-structured distributions by chow-liu. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 147–160, 2021.

[8] Béla Bollobás. *Modern Graph Theory*. Graduate Texts in Mathematics 184. 1998.

[9] Cristina Butucea and Yann ISSARTEL. Locally differentially private estimation of functionals of discrete distributions. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 24753–24764. Curran Associates, Inc., 2021.

[10] Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the minimum spanning tree weight in sublinear time. *SIAM Journal on computing*, 34(6):1370–1379, 2005.

[11] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

[12] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, New York, NY, USA, 2006.

[13] Michael Crouch, Andrew McGregor, Gregory Valiant, and David P Woodruff. Stochastic streams: Sample complexity vs. space complexity. In *24th Annual European Symposium on Algorithms (ESA 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[14] Constantinos Daskalakis and Qinxuan Pan. Tree-structured ising models can be learned efficiently. *arXiv e-prints*, pages arXiv–2010, 2020.

[15] John Duchi and Ryan Rogers. Lower bounds for locally private estimation via communication complexity. In *Conference on Learning Theory*, pages 1161–1191. PMLR, 2019.

[16] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[17] Kazuto Fukuchi and Jun Sakuma. Minimax optimal estimators for additive scalar functionals of discrete distributions. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 2103–2107. IEEE, 2017.

[18] Yanjun Han, Ayfer Özgür, and Tsachy Weissman. Geometric lower bounds for distributed parameter estimation under communication constraints. In *Conference On Learning Theory*, pages 3163–3188. PMLR, 2018.

[19] Oc Herfindahl. Concentration in the us steel industry. 1950.

[20] Matthew Joseph, Jieming Mao, Seth Neel, and Aaron Roth. The role of interactivity in local differential privacy. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 94–105. IEEE, 2019.

[21] Matthew Joseph, Jieming Mao, Seth Neel, and Aaron Roth. The role of interactivity in local differential privacy. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 94–105. IEEE Computer Society, 2019.

[22] Edith Kovács and Tamás Szántai. *On the Approximation of a Discrete Multivariate Probability Distribution Using the New Concept of t-Cherry Junction Tree*, pages 39–56. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[23] Markku Laakso and Rein Taagepera. “effective” number of parties: A measure with application to west europe. *Comparative Political Studies*, 12(1):3–27, 1979.

[24] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[25] Abolfazl S Motahari, Guy Bresler, and NC David. Information theory of DNA shotgun sequencing. *IEEE Transactions on Information Theory*, 59(10):6273–6289, 2013.

[26] Laura Elena Raileanu and Kilian Stoffel. Theoretical comparison between the gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence*, 41(1):77–93, 2004.

[27] Osama Shahid, Seyedamin Pouriyeh, Reza M Parizi, Quan Z Sheng, Gautam Srivastava, and Liang Zhao. Communication efficiency in federated learning: Achievements and challenges. *arXiv preprint arXiv:2107.10996*, 2021.

[28] Edward H Simpson. Measurement of diversity. *nature*, 163(4148):688–688, 1949.

[29] Maciej Skorski. Evaluating entropy for true random number generators: Efficient, robust and provably secure. In *International Conference on Information Security and Cryptology*, pages 526–541, 03 2017.

[30] Ananda Theertha Suresh, X Yu Felix, Sanjiv Kumar, and H Brendan McMahan. Distributed mean estimation with limited communication. In *International Conference on Machine Learning*, pages 3329–3337. PMLR, 2017.

[31] Thomas Verma and Judea Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, UAI '90, page 255–270, USA, 1990. Elsevier Science Inc.

[32] Shlomo Yitzhaki and Edna Schechtman. *More Than a Dozen Alternative Ways of Spelling Gini*, pages 11–31. Springer New York, New York, NY, 2013.

[33] Yuchen Zhang, John C Duchi, Michael I Jordan, and Martin J Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. In *NIPS*, pages 2328–2336. Citeseer, 2013.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] There is a gap between upper and lower bounds.
   (c) Did you discuss any potential negative societal impacts of your work? No social negative impact.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] In the appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? No experiments.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? No experiments.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? No experiments.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? No experiments.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? We used results published in peer reviewed venues and we cited these results properly.
   (b) Did you mention the license of the assets? No licence is needed.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]