Dual Query: Practical Private Query Release for High Dimensional Data

Marco Gaboardi M.GABOARDI@DUNDEE.AC.UK
University of Dundee, Dundee, Scotland, UK

Emilio Jesús Gallego Arias EMILIOGA@CIS.UPENN.EDU
Justin Hsu JUSTHSU@CIS.UPENN.EDU
Aaron Roth AAROTH@CIS.UPENN.EDU
Zhiwei Steven Wu WUZHIWEI@CIS.UPENN.EDU
University of Pennsylvania, Philadelphia, USA

Abstract

We present a practical, differentially private algorithm for answering a large number of queries on high dimensional datasets. Like all algorithms for this task, ours necessarily has worst-case complexity exponential in the dimension of the data. However, our algorithm packages the computationally hard step into a concisely defined integer program, which can be solved non-privately using standard solvers. We prove accuracy and privacy theorems for our algorithm, and then demonstrate experimentally that our algorithm performs well in practice. For example, our algorithm can efficiently and accurately answer millions of queries on the Netflix dataset, which has over 17,000 attributes; this is an improvement on the state of the art by multiple orders of magnitude.[1]

1. Introduction

Privacy is becoming a paramount concern for machine learning and data analysis tasks, which often operate on personal data. For just one example of the tension between machine learning and data privacy, Netflix released an anonymized dataset of user movie ratings for teams competing to develop an improved recommendation mechanism. The competition was a great success (the winning team improved on the existing recommendation system by more than 10%), but the ad hoc anonymization was not as successful: Narayanan & Shmatikov (2008) were later able to re-identify individuals in the dataset, leading to a lawsuit and the cancellation of subsequent competitions.

Differentially private query release is an attempt to solve this problem. Differential privacy is a strong formal privacy guarantee (that, among other things, provably prevents re-identification attacks), and the problem of *query release* is to release accurate answers to a set of statistical queries. As observed early on by Blum et al. (2005), performing private query release is sufficient to simulate any learning algorithm in the *statistical query model* of Kearns (1998).

Since then, the query release problem has been extensively studied in the differential privacy literature. While simple perturbation can be used to privately answer a small number of queries (Dwork et al., 2006), more sophisticated approaches can accurately answer nearly exponentially many queries in the size of the private database (Blum et al., 2013; Dwork et al., 2009; 2010; Roth & Roughgarden; Hardt & Rothblum, 2010; Gupta et al., 2012; Hardt et al., 2012). A natural approach, employed by many of these algorithms, is to answer queries by generating *synthetic data*: a safe version of the dataset that approximates the real dataset on every statistical query of interest.

Unfortunately, even the most efficient approaches for query release have a per-query running time linear in the size of the *data universe*, which is exponential in the dimension of the data (Hardt & Rothblum, 2010).

[1] This is an extended abstract of the full version of this paper (Gaboardi et al., 2014), which contains full details of our algorithm and experiments.

Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

Moreover, this running time is necessary in the worst case (Ullman, 2013), especially if the algorithm produces synthetic data (Ullman & Vadhan, 2011).

This exponential runtime has hampered practical evaluation of query release algorithms. One notable exception is due to Hardt et al. (2012), who perform a thorough experimental evaluation of one such algorithm, which they called MWEM. They find that MWEM has quite good accuracy in practice and scales to higher dimensional data than suggested by a theoretical (worst-case) analysis. Nevertheless, running time remains a problem, and the approach does not seem to scale to high dimensional data (with more than 30 or so attributes for general queries, and more when the queries satisfy special structure[2]). The critical bottleneck is the size of the state maintained by the algorithm: MWEM, like many query release algorithms, needs to manipulate an object that has size linear in the size of the data universe (i.e., exponential in the dimension). This quickly becomes impractical as the record space grows more complex.

We present DualQuery, an alternative algorithm which is *dual* to MWEM in a sense that we will make precise. Rather than manipulating an object of exponential size, DualQuery solves a concisely represented (but NP-hard) optimization problem. Critically, the optimization step does not require a solution that is private or exact, so it can be handled by existing, highly optimized solvers. Except for this step, all parts of our algorithm are extremely efficient. As a result, DualQuery requires (worst-case) space and (in practice) time only linear in the number of *queries* of interest, which is often significantly smaller than the number of possible records. Like existing algorithms for query release, DualQuery has a provable accuracy guarantee and satisfies the strong differential privacy guarantee.

We evaluate DualQuery on a variety of datasets by releasing *3-way marginals* (also known as *conjunctions* or *contingency tables*), demonstrating that it solves the query release problem accurately and efficiently even when the data includes hundreds of thousands of features. We know of no other algorithm to perform accurate, private query release for rich classes of queries on real data with more than even 100 features.

[2]Hardt et al. (2012) are able to scale up to 1000 features on synthetic data when the features are partitioned into a number of small buckets, and the queries are chosen to never depend on features in more than one bucket.

Related work. Differentially private learning has been studied since Blum et al. (2005) showed how to convert learning algorithms in the SQ model of Kearns (1998) into differentially private learning algorithms with similar accuracy guarantees. Since then, private machine learning has become a very active field with both foundational sample complexity results (Kasiviswanathan et al., 2011; Chaudhuri & Hsu, 2011; Beimel et al., 2013; Duchi et al., 2013) and numerous efficient algorithms for particular learning problems (Chaudhuri & Monteleoni, 2008; Chaudhuri et al., 2011; Rubinstein et al., 2012; Kifer et al., 2012; Chaudhuri et al., 2012; Thakurta & Smith, 2013).

In parallel, there has been a significant amount of work on privately releasing synthetic data based on a true dataset while preserving the answers to large numbers of statistical queries (Blum et al., 2013; Dwork et al., 2009; Roth & Roughgarden; Dwork et al., 2010; Hardt & Rothblum, 2010; Gupta et al., 2012). These results are extremely strong in an information theoretic sense: they ensure the consistency of the synthetic data with respect to an exponentially large family of statistics. But, all of these algorithms (including the notable multiplicative weights algorithm of Hardt & Rothblum (2010), which achieves the theoretically optimal accuracy and runtime) have running time exponential in the dimension of the data. With standard cryptographic assumptions, this is necessary in the worst case for mechanisms that answer arbitrary statistical queries (Ullman, 2013).

Nevertheless, there have been some experimental evaluations of these approaches on real datasets. Most related to our work is the evaluation of the MWEM mechanism by Hardt et al. (2012), which is based on the private multiplicative weights mechanism (Hardt & Rothblum, 2010). This algorithm is inefficient (it manipulates a probability distribution over a set exponentially large in the dimension of the data space) but with some heuristic optimizations, Hardt et al. (2012) were able to implement the multiplicative weights algorithm on several real datasets with up to 77 attributes (and even more when the queries are restricted to take positive values only on a small number of disjoint groups of features). However, it seems difficult to scale this approach to higher dimensional data.

Another family of query release algorithms are based on the Matrix Mechanism (Li et al., 2010; Li & Miklau, 2012). The runtime guarantees of the matrix mechanism are similar to the approaches based on multiplicative weights—the algorithm manipulates a "matrix" of queries with dimension exponential in the

number of features. Yaroslavtsev et al. (2013) evaluate an approach based on this family of algorithms on low dimensional datasets, but scaling to high dimensional data also seems challenging. A recent work by Zhang et al. (2014) proposes a low-dimensional approximation for high-dimensional data distribution by privately constructing Bayesian networks, and shows that such a representation gives good accuracy on some real datasets.

Our algorithm is inspired by the view of the synthetic data generation problem as a zero-sum game, first proposed by Hsu et al. (2013). In this interpretation, Hardt et al. (2012) solves the game by having a *data player* use a no-regret learning algorithm, while the *query player* repeatedly best responds by optimizing over queries. In contrast, our algorithm swaps the roles of the two players: the query player now uses the no-regret learning algorithm, whereas the data player now finds best responses by solving an optimization problem. This is reminiscent of "Boosting for queries," proposed by Dwork et al. (2010); the main difference is that our optimization problem is over single records rather than sets of records. As a result, our optimization can be handled non-privately.

2. Differential privacy background

Differential privacy has become a standard algorithmic notion for protecting the privacy of individual records in a statistical database. It formalizes the requirement that the addition or removal of a data record does not change the probability of any outcome of the mechanism by much.

To begin, databases are multisets of elements from an abstract domain $\mathcal{X}$, representing the set of all possible data records. Two databases $D, D' \subset \mathcal{X}$ are *neighboring* if they differ in a single data element ($\|D \triangle D'\| \leq 1$).

Definition 2.1 (Dwork et al. (2006))**.** *A mechanism $M \colon \mathcal{X}^n \to R$ satisfies (ε, δ)-differential privacy if for every $S \subseteq R$ and for all neighboring databases $D, D' \in \mathcal{X}^n$, the following holds:*

$$\Pr[M(D) \in S] \leq e^{\varepsilon} \Pr[M(D') \in S] + \delta$$

Definition 2.2. *For any predicate $\varphi \colon \mathcal{X} \to \{0,1\}$, the* linear query *$Q_\varphi \colon \mathcal{X}^n \to [0,1]$ is defined by*

$$Q_\varphi(D) = \sum_{x \in D} \varphi(x)/|D|.$$

We will use a fundamental tool for private data analysis: we can bound the privacy cost of an algorithm as a function of the privacy costs of its subcomponents.

Lemma 2.3 (Dwork et al. (2010))**.** *Let $M_1, \ldots, M_k$ be such that each M_i is $(\varepsilon_i, 0)$-private with $\varepsilon_i \leq \varepsilon'$. Then $M(D) = (M_1(D), \ldots, M_k(D))$ is $(\varepsilon, 0)$-private for $\varepsilon = \sum_{i=1}^k \varepsilon_i$, and (ε, δ)-private for*

$$\varepsilon = \sqrt{2\log(1/\delta)k}\varepsilon' + k\varepsilon'(e^{\varepsilon'} - 1)$$

for any $\delta \in (0,1)$.

3. The query release game

The analysis of our algorithm relies on the interpretation of query release as a two player, zero-sum game (Hsu et al., 2013). In the present section, we review this idea and related tools.

Game definition. Suppose we want to answer a set of queries $\mathcal{Q}$. For each query $q \in \mathcal{Q}$, we can form the *negated query* $\overline{q}$, which takes values $\overline{q}(D) = 1 - q(D)$ for every database D. For the remainder, we will assume that $\mathcal{Q}$ is closed under negation; if not, we may add negated copies of each query to $\mathcal{Q}$.

Let there be two players, whom we call the *data* player and *query* player. The data player has action set equal to the data universe $\mathcal{X}$, while the query player has action set equal to the query class $\mathcal{Q}$. Given a play $x \in \mathcal{X}$ and $q \in \mathcal{Q}$, we let the payoff be

$$A(x, q) := q(D) - q(x), \tag{1}$$

where D is the true database. As a zero sum game, the data player will try to minimize the payoff, while the query player will try to maximize the payoff.

Equilibrium of the game. Let $\Delta(\mathcal{X})$ and $\Delta(\mathcal{Q})$ be the set of probability distributions over $\mathcal{X}$ and $\mathcal{Q}$. We consider how well each player can do if they randomize over their actions, i.e., if they play from a probability distribution over their actions. By von Neumann's minimax theorem,

$$\min_{u \in \Delta(\mathcal{X})} \max_{w \in \Delta(\mathcal{Q})} A(u, w) = \max_{w \in \Delta(\mathcal{Q})} \min_{u \in \Delta(\mathcal{X})} A(u, w),$$

for any two player zero-sum game, where

$$A(u, w) := \mathbb{E}_{x \sim u, q \sim w} A(x, q)$$

is the expected payoff. The common value is called the *value of the game*, which we denote by v_A.

This suggests that each player can play an optimal strategy, assuming best play from the opponent—this is the notion of equilibrium strategies, which we now define. We will soon interpret these strategies as solutions to the query release problem.

Definition 3.1. *Let $\alpha > 0$. Let A be the payoffs for a two player, zero-sum game with action sets $\mathcal{X}, \mathcal{Q}$. Then, a pair of strategies $u^* \in \Delta(\mathcal{X})$ and $w^* \in \Delta(\mathcal{Q})$ form an α-*approximate mixed Nash equilibrium *if*

$$A(u^*, w) \leq v_A + \alpha \quad \text{and} \quad A(u, w^*) \geq v_A - \alpha$$

for every strategy $u \in \Delta(\mathcal{X}), w \in \Delta(\mathcal{Q})$.

If the true database D is normalized to be a distribution $\widehat{D}$ in $\Delta(\mathcal{X})$, then $\widehat{D}$ always has zero payoff:

$$A(\widehat{D}, w) = \mathbb{E}_{x \sim \widehat{D}, q \sim w}[q(x) - q(D)] = 0.$$

Hence, the value of the game v_A is at most 0. Also, for any data strategy u, the payoff of query q is the negated payoff of the negated query $\overline{q}$:

$$A(u, \overline{q}) = \mathbb{E}_{x \sim u}[\overline{q}(x) - \overline{q}(D)] = \mathbb{E}_{x \sim u}[q(D) - q(x)],$$

which is $A(u, \overline{q})$. Thus, any query strategy that places equal weight on q and $\overline{q}$ has expected payoff zero, so v_A is at least 0. Hence, $v_A = 0$.

Now, let (u^*, w^*) be an α-approximate equilibrium. Suppose that the data player plays u^*, while the query player always plays query q. By the equilibrium guarantee, we then have $A(u^*, q) \leq \alpha$, but the expected payoff on the left is simply $q(D) - q(u^*)$. Likewise, if the query player plays the negated query $\overline{q}$, then

$$-q(D) + q(u^*) = A(u^*, \overline{q}) \leq \alpha,$$

so $q(D) - q(u^*) \geq -\alpha$. Hence for every query $q \in \mathcal{Q}$, we know $|q(u^*) - q(D)| \leq \alpha$. This is precisely what we need for query release: we just need to privately calculate an approximate equilibrium.

Solving the game. To construct the approximate equilibrium, we will use the multiplicative weights update algorithm (MW). This algorithm maintains a distribution over actions (initially uniform) over a series of steps. At each step, the MW algorithm receives a (possibly adversarial) loss for each action. Then, MW reweights the distribution to favor actions with less loss. The algorithm can be found in the full version of this paper.

For our purposes, the most important application of MW is to solving zero-sum games. Freund & Schapire (1996) showed that if one player maintains a distribution over actions using MW, while the other player selects a *best-response* action versus the current MW distribution (i.e., an action that maximizes his expected payoff), the average MW distribution and empirical best-response distributions will converge to an approximate equilibrium rapidly.

Theorem 3.2 (Freund & Schapire (1996))**.** *Let $\alpha > 0$, and let $A(i, j) \in [-1, 1]^{m \times n}$ be the payoff matrix for a zero-sum game. Suppose the first player uses multiplicative weights over their actions to play distributions $p^1, \ldots, p^T$, while the second player plays $(\alpha/2)$-approximate best responses $x^1, \ldots, x^T$, i.e.,*

$$A(p^t, x^t) \geq \max_x A(p^t, x) - \alpha/2.$$

Setting $T = 16 \log n / \alpha^2$ and $\eta = \alpha/4$ in the MW algorithm, the empirical distributions

$$\frac{1}{T} \sum_{i=1}^{T} p^i \quad \text{and} \quad \frac{1}{T} \sum_{i=1}^{T} x^i$$

form an α-approximate mixed Nash equilibrium.

4. Dual query release

By the game interpretation, the algorithm of Hardt & Rothblum (2010) (and the MWEM algorithm of Hardt et al. (2012)) uses MW for the data player, while the query player plays best responses. For privacy, their algorithm selects the query best-responses privately via the exponential mechanism of McSherry & Talwar (2007). Our algorithm simply reverses the roles.

While MWEM uses a no-regret algorithm to maintain the data player's distribution, we will instead use a no-regret algorithm for the query player's distribution. Likewise, instead of finding a maximum payoff query at each round, our algorithm selects a minimum payoff record at each turn. The full algorithm can be found in Algorithm 1.

Our privacy argument differs slightly from the analysis for MWEM. There, the data distribution is public, and finding a query with high error requires access to the private data. Our algorithm instead maintains a distribution Q over queries which depends directly on the private data, so we cannot use Q directly. Instead, we argue that *queries sampled from Q* are privacy preserving. Then, we can use a non-private optimization method to find a minimal error record versus queries sampled from Q. We then trade off privacy (which degrades as we take more samples) with accuracy (which improves as we take more samples, since the distribution of sampled queries converges to Q).

Given known hardness results for the query release problem (Ullman, 2013), our algorithm must have worst-case runtime polynomial in the universe size $|\mathcal{X}|$, so is not theoretically more efficient than prior approaches. In fact, even compared to prior work on query release (e.g., Hardt & Rothblum (2010)), our algorithm has a weaker accuracy guarantee. However, our approach has an important practical benefit: the computationally hard step can be handled with standard, non-private solvers.

The iterative structure of our algorithm, combined with our use of constraint solvers, also allows for several heuristics improvements. For instance, we may run for fewer iterations than predicted by theory. Or, if the optimization problem turns out to be hard (even in practice), we can stop the solver early at a suboptimal (but often still good) solution. These heuristic tweaks can improve accuracy beyond what is guaranteed by our accuracy theorem, while always maintaining a strong *provable* privacy guarantee.

Algorithm 1 DualQuery

Input: Database $D \in \mathbb{R}^{|\mathcal{X}|}$ (normalized) and linear queries $q_1, \ldots, q_k \in \{0,1\}^{|\mathcal{X}|}$.

Initialize: Let $\mathcal{Q} = \bigcup_{j=1}^{k} q_j \cup \overline{q_j}$, Q^1 uniform distribution on $\mathcal{Q}$,

$$T = \frac{16 \log |\mathcal{Q}|}{\alpha^2}, \quad \eta = \frac{\alpha}{4}, \quad s = \frac{48 \log (2|\mathcal{X}|T/\beta)}{\alpha^2}.$$

For $t = 1, \ldots, T$:
  Sample s queries $\{q_i\}$ from $\mathcal{Q}$ according to Q^t.
  Let $\widetilde{q} := \frac{1}{s}\sum_i q_i$.
  Find x^t with $A(\widetilde{q}, x^t) \geq \max_x A(\widetilde{q}, x) - \alpha/4$.
  Update: For each $q \in \mathcal{Q}$:
    $Q_q^{t+1} := \exp(-\eta A(q, x^t)\rangle) \cdot Q_q^t$.
  Normalize Q^{t+1}.
Output synthetic database $\widehat{D} := \bigcup_{t=1}^{T} x^t$.

Privacy. The privacy proofs are largely routine, based on the composition theorems. Rather than fixing ε and solving for the other parameters, we present the privacy cost ε as function of parameters T, s, η. Later, we will tune these parameters for our experimental evaluation. DualQuery satisfies the following privacy guarantee. (All proofs can be found in the full version of this paper (Gaboardi et al., 2014).)

Theorem 4.1. *Let $\delta \in (0,1)$. Algorithm 1 is (ε, δ)-private for*

$$\varepsilon = \frac{4\eta T \sqrt{2sT \log(1/\delta)}}{n}.$$

Accuracy. We show accuracy in two steps. First, we show that the "average query" formed from the samples is close to the average query weighted by Q^t. Next, we show that our algorithm finds an approximate equilibrium of the query release game.

Theorem 4.2. *With probability at least $1 - \beta$,* DualQuery *finds a synthetic database that answers all queries in $\mathcal{Q}$ within additive error α.*

Remark 4.3. *The guarantee in Theorem 4.2 may seem a little unusual, since the convention in the literature is to treat ε, δ as inputs to the algorithm. We can do the same: from Theorem 4.1 and plugging in for T, η, s,*

$$\alpha = O\left(\frac{\log^{1/2}|\mathcal{Q}| \log^{1/6}(1/\delta) \log^{1/6}(2|\mathcal{X}|/\gamma)}{n^{1/3}\varepsilon^{1/3}}\right),$$

for $\gamma < \beta/T$.

5. Case study: 3-way marginals

In our algorithm, the computationally difficult step is finding the data player's approximate best response against the query player's distribution. As mentioned above, the form of this problem depends on the particular query class $\mathcal{Q}$. In this section, we first discuss the optimization problem in general, and then specifically for the well-studied class of *marginal* queries Thaler et al. (2012); Gupta et al. (2013); Dwork et al. (2014). For instance, in a database of medical information in binary attributes, a particular marginal query may be: What fraction of the patients are over 50, smoke, and exercise?

The best-response problem. Recall that the query release game has payoff $A(x, q)$ defined by Equation (1); the data player tries to minimize the payoff, while the query player tries to maximize it. If the query player has distribution Q^t over queries, the data player's best response minimizes the expected loss:

$$\operatorname*{argmin}_{x \in \mathcal{X}} \mathop{\mathbb{E}}_{q \leftarrow Q^t} [q(D) - q(x)].$$

To ensure privacy, the data player actually plays against the distribution of samples $\widehat{q_1}, \ldots, \widehat{q_s}$. Since

the database D is fixed and $\widehat{q}_i$ are linear queries, the best-response problem is

$$\operatorname*{argmin}_{x\in\mathcal{X}} \frac{1}{s}\sum_{i=1}^{s} \widehat{q}_i(D) - \widehat{q}_i(x) = \operatorname*{argmax}_{x\in\mathcal{X}} \sum_{i=1}^{s} \widehat{q}_i(x).$$

3-way marginal queries. To look at the precise form of the best-response problem, we consider *3-way marginal* queries. We think of records as having d binary attributes, so that the data universe $|\mathcal{X}|$ is all bit-strings of length d. We write x_i for $x \in \mathcal{X}$ to mean the ith bit of record x.

Definition 5.1. *Let $\mathcal{X} = \{0,1\}^d$. A* 3-way marginal query *is a linear query specified by 3 integers $a \neq b \neq c \in [d]$, taking values*

$$q_{abc}(x) = \begin{cases} 1 & : x_a = x_b = x_c = 1 \\ 0 & : \textit{otherwise.} \end{cases}$$

Recall that the query class $\mathcal{Q}$ includes each query and its negation. So, we also have negated conjunctions:

$$\overline{q_{abc}}(x) = \begin{cases} 0 & : x_a = x_b = x_c = 1 \\ 1 & : \text{otherwise.} \end{cases}$$

Given sampled conjunctions $\{\widehat{u}_i\}$ and negated conjunctions $\{\widehat{v}_i\}$, the best-response problem is

$$\operatorname*{argmax}_{x\in\mathcal{X}} \sum_i \widehat{u}_i(x) + \sum_j \widehat{v}_j(x).$$

In other words, this is a MAXCSP problem—we can associate a clause to each conjunction:

$$q_{abc} \Rightarrow (x_a \wedge x_b \wedge x_c) \quad \text{and} \quad \overline{q_{abc}} \Rightarrow (\overline{x_a} \vee \overline{x_b} \vee \overline{x_c}),$$

and we want to find $x \in \{0,1\}^d$ satisfying as many clauses as possible.

Since most solvers do not directly handle MAXCSP problems, we convert this optimization problem into a more standard, integer program form. We introduce a variable x_i for each literal x_i, a variable c_i for each sampled conjunction $\widehat{u}_i$, a variable d_i for each sampled negated conjunction $\widehat{v}_i$, and we form the following integer program.

$$\max \sum_i c_i + \sum_j d_j$$

$$\text{with } \forall \widehat{u}_i = q_{abc}: \; x_a + x_b + x_c \geq 3c_i$$
$$\forall \widehat{v}_j = \overline{q_{abc}}: \; (1-x_a) + (1-x_b) + (1-x_c) \geq d_j$$
$$x_i, c_i, d_i \in \{0,1\}$$

Note that $x_i, 1-x_i$ corresponds to the literals $x_i, \overline{x_i}$, and $c_i = 1$, $d_i = 1$ exactly when their respective clauses are satisfied. Thus, the objective is the number of satisfied clauses. The resulting integer program can be solved using any standard solver; we use CPLEX.

Dataset	Size	Attributes	Binary attributes
Adult	30162	14	235
KDD99	494021	41	396
Netflix	480189	17,770	17,770

Figure 1. Test Datasets

6. Experimental evaluation

We evaluate DualQuery on a large collection of 3-way marginal queries on several real datasets (Figure 1) and high dimensional synthetic data. Adult and KDD99 are from the UCI repository (Bache & Lichman, 2013), and have a mixture of discrete (but non-binary) and continuous attributes, which we discretize into binary attributes. We also use the (in)famous Netflix movie ratings dataset, with more than 17,000 binary attributes.

Rather than set the parameters as in Algorithm 1, we experiment with a range of parameters. For instance, we frequently run for fewer rounds (lower T) and take fewer samples (lower s). As such, the accuracy guarantee (Theorem 4.2) need not hold for our parameters. However, we find that our algorithm gives good error, often much better than predicted. In all cases, our parameters satisfy the privacy guarantee Theorem 4.1.

Accuracy. We evaluate the accuracy of the algorithm on 500,000 3-way marginals on Adult, KDD99 and Netflix. We report maximum error in Figure 2, averaged over 5 runs. (Marginal queries have range $[0,1]$, so error 1 is trivial.) The runs are $(\varepsilon, 0.001)$-differentially private, with ε ranging from 0.25 to 5.[3]

Scaling to More Queries. Next, we evaluate accuracy and runtime when varying the number of queries. We use a set of 40,000 to 2 million randomly generated marginals $\mathcal{Q}$ on the KDD99 dataset and run DualQuery with $(1, 0.001)$-privacy. As shown in Figure 3, both average and max error remain mostly stable, demonstrating improved error compared to simpler perturbation approaches. For example, the

[3] By Lemma 2.3, our algorithm actually satisfies (ε, δ)-privacy for smaller values of δ. For example, our algorithm is also $(\sqrt{2}\varepsilon, \delta')$-private for $\delta' = 10^{-6}$.

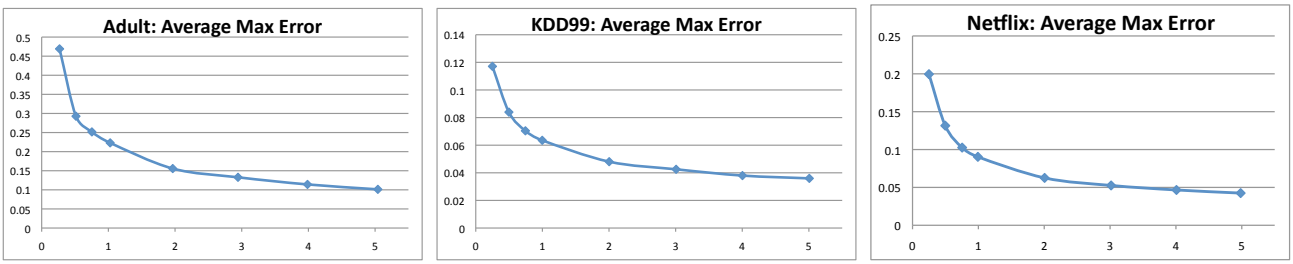

Figure 2. Average max error of $(\varepsilon, 0.001)$-private DualQuery on 500,000 3-way marginals versus ε.

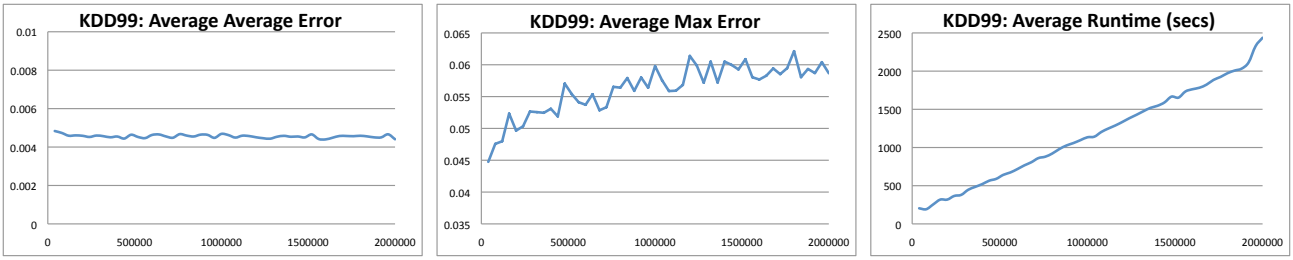

Figure 3. Error and runtime of $(1, 0.001)$-private DualQuery on KDD99 versus number of queries.

Laplace mechanism's error growth rate is $O(\sqrt{|\mathcal{Q}|})$ under (ε, δ)-differential privacy.

Scaling to Higher Dimensional Data. Finally, we evaluate accuracy and runtime behavior for data dimension ranging from 50 to 512,000. We evaluate DualQuery under $(1, 0.001)$-privacy on 100,000 3-way marginals on synthetically genearted datasets. We report runtime, max, and average error over 3 runs in Figure 4; note the logarithmic scale for attributes axis. We do not include query evaluation in our time measurements—this overhead is common to all approaches that answer a set of queries.

When generating the synthetic data, one possibility is to set each attribute to be 0 or 1 uniformly at random. However, this generates very uniform synthetic data: a record satisfies any 3-way marginal with probability $1/8$, so most marginals will have value near $1/8$. To generate more challenging and realistic data, we pick a separate bias $p_i \in [0, 1]$ uniformly at random for each attribute i. For each data point, we then set attribute i to be 1 independently with probability equal to p_i. As a result, different 3-way marginals have different answers on our synthetic data.

Implementation details. The implementation is written in OCaml, using the CPLEX constraint solver. We ran the experiments on a mid-range desktop machine with a 4-core Intel Xeon processor and 12 Gb of RAM. Heuristically, we set a timeout for each CPLEX call to 20 seconds, accepting the best current solution if we hit the timeout. For the experiments shown, the timeout was rarely reached.

Data discretization. We discretize KDD99 and Adult datasets into binary attributes by mapping each possible value of a discrete attribute to a new binary feature. We bucket continuous attributes, mapping each bucket to a new binary feature. We also ensure that our randomly generated 3-way marginal queries are sensible (i.e., they don't require an original attribute to take two different values).

Setting free attributes. Since the collection of sampled queries may not involve all of the attributes, CPLEX often finds solutions that leave some attributes unspecified. We set these *free* attributes heuristically: for real data, we set the attributes to 0 as these datasets are fairly sparse; for synthetic data, we set attributes to 0 or 1 uniformly at random.

Parameter tuning. DualQuery has three parameters that can be set in a wide variety of configurations without altering the privacy guarantee (Theorem 4.1): number of iterations (T), number of samples (s), and learning rate (η), which controls how aggressively to

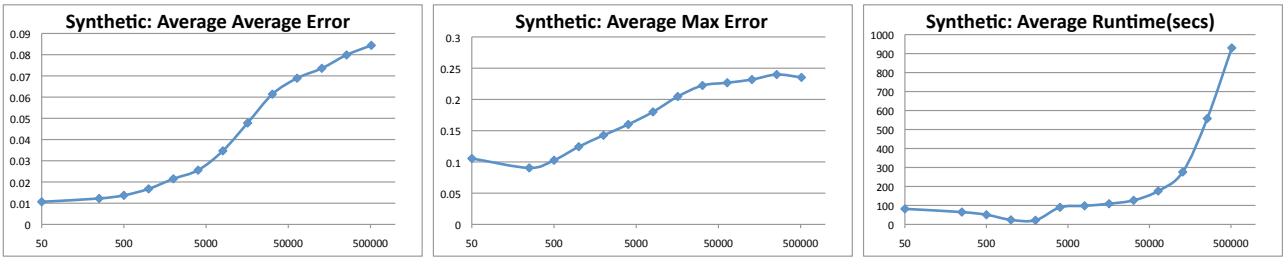

Figure 4. Error and runtime of $(1, 0.001)$-private DualQuery on 100,000 3-way marginal queries versus number of attributes.

update the distribution. For a fixed level of ε and δ, there are many feasible private parameter settings.

Performance depends strongly on the choice of parameters: T has an obvious impact, increasing s increases the number of constraints in the integer program for CPLEX. We have investigated a range of parameters; for the experiments we have used some informal heuristics coming from our observations (parameters details deferred to our full version).

Parameter setting should be done under differential privacy for a truly realistic evaluation. Overall, we do not know of a principled approach to handle this issue; private parameter tuning is an area of active research (see e.g., Chaudhuri & Vinterbo (2013)).

7. Discussion and conclusion

We have given a new private query release mechanism that can handle datasets with dimensionality multiple orders of magnitude larger than what was previously possible. Indeed, it seems we have not reached the limits of our approach—even on synthetic data with more than 500,000 attributes, DualQuery continues to generate useful answers with about 30 minutes of overhead on top of query evaluation (which by itself is on the scale of hours). We believe that DualQuery makes private analysis of high dimensional data practical for the first time.

However, this remarkable improvement in running time is not free: our theoretical accuracy bounds are worse than those of previous approaches (Hardt & Rothblum, 2010; Hardt et al., 2012). For low dimensional datasets for which it is possible to maintain a distribution over records, the MWEM algorithm of Hardt et al. (2012) likely remains the state of the art (for an experimental comparison, see our full version of the paper). Our work complements MWEM by allowing private data analysis on higher-dimensional data sets.

Acknowledgments

We thank Adam Smith, Cynthia Dwork and Ilya Mironov for productive discussions, and for suggesting the Netflix dataset. This work was supported in part by NSF grants CCF-1101389 and CNS-1065060. Marco Gaboardi has been supported by the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement No. 272487.

References

Bache, K. and Lichman, M. UCI machine learning repository, 2013.

Beimel, Amos, Nissim, Kobbi, and Stemmer, Uri. Characterizing the sample complexity of private learners. In *ACM SIGACT Innovations in Theoretical Computer Science (ITCS), Berkeley, California*, pp. 97–110, 2013.

Blum, A., Ligett, K., and Roth, A. A learning theory approach to noninteractive database privacy. *Journal of the ACM*, 60(2): 12, 2013.

Blum, Avrim, Dwork, Cynthia, McSherry, Frank, and Nissim, Kobbi. Practical privacy: the sulq framework. In *ACM SIGACT–SIGMOD–SIGART Symposium on Principles of Database Systems (PODS), Baltimore, Maryland*, pp. 128–138, 2005.

Chaudhuri, Kamalika and Hsu, Daniel. Sample complexity bounds for differentially private learning. *Journal of Machine Learning Research*, 19:155–186, 2011.

Chaudhuri, Kamalika and Monteleoni, Claire. Privacy-preserving logistic regression. In *Conference on Neural Information Processing Systems (NIPS), Vancouver, British Colombia*, pp. 289–296, 2008.

Chaudhuri, Kamalika and Vinterbo, Staal A. A stability-based validation procedure for differentially private machine learning. pp. 2652–2660, 2013.

Chaudhuri, Kamalika, Monteleoni, Claire, and Sarwate, Anand D. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12:1069–1109, 2011.

Chaudhuri, Kamalika, Sarwate, Anand, and Sinha, Kaushik. Near-optimal differentially private principal components. In *Conference on Neural Information Processing Systems (NIPS), Lake Tahoe, California*, pp. 998–1006, 2012.

Duchi, J.C., Jordan, M.I., and Wainwright, M.J. Local privacy and statistical minimax rates. In *IEEE Symposium on Foundations of Computer Science (FOCS), Berkeley, California*, 2013.

Dwork, C., Naor, M., Reingold, O., Rothblum, G.N., and Vadhan, S.P. On the complexity of differentially private data release: efficient algorithms and hardness results. In *ACM SIGACT Symposium on Theory of Computing (STOC), Bethesda, Maryland*, pp. 381–390, 2009.

Dwork, C., Rothblum, G.N., and Vadhan, S. Boosting and differential privacy. In *IEEE Symposium on Foundations of Computer Science (FOCS), Las Vegas, Nevada*, pp. 51–60, 2010.

Dwork, Cynthia, McSherry, Frank, Nissim, Kobbi, and Smith, Adam. Calibrating noise to sensitivity in private data analysis. In *IACR Theory of Cryptography Conference (TCC), New York, New York*, pp. 265–284, 2006.

Dwork, Cynthia, Nikolov, Aleksandar, and Talwar, Kunal. Using convex relaxations for efficiently and privately releasing marginals. In *SIGACT – SIGGRAPH Symposium on Computational Geometry (SOCG), Kyoto, Japan*, pp. 261, 2014.

Freund, Y. and Schapire, R.E. Game theory, on-line prediction and boosting. In *Conference on Computational Learning Theory (CoLT), Desenzano sul Garda, Italy*, pp. 325–332, 1996.

Gaboardi, Marco, Gallego Arias, Emilio Jesús, Hsu, Justin, Roth, Aaron, and Wu, Zhiwei Steven. Dual query: Practical private query release for high dimensional data. Technical report, 2014. `http://arxiv.org/abs/1402.1526`.

Gupta, A., Roth, A., and Ullman, J. Iterative constructions and private data release. In *IACR Theory of Cryptography Conference (TCC), Taormina, Italy*, pp. 339–356, 2012.

Gupta, Anupam, Hardt, Moritz, Roth, Aaron, and Ullman, Jonathan. Privately releasing conjunctions and the statistical query barrier. *SIAM Journal on Computing*, 42(4):1494–1520, 2013.

Hardt, Moritz and Rothblum, Guy N. A multiplicative weights mechanism for privacy-preserving data analysis. In *IEEE Symposium on Foundations of Computer Science (FOCS), Las Vegas, Nevada*, pp. 61–70, 2010.

Hardt, Moritz, Ligett, Katrina, and McSherry, Frank. A simple and practical algorithm for differentially private data release. In *Conference on Neural Information Processing Systems (NIPS), Lake Tahoe, California*, pp. 2348–2356, 2012.

Hsu, Justin, Roth, Aaron, and Ullman, Jonathan. Differential privacy for the analyst via private equilibrium computation. In *ACM SIGACT Symposium on Theory of Computing (STOC), Palo Alto, California*, pp. 341–350, 2013.

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

Kearns, Michael J. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45(6):983–1006, 1998.

Kifer, Daniel, Smith, Adam, and Thakurta, Abhradeep. Private convex empirical risk minimization and high-dimensional regression. *Journal of Machine Learning Research*, 1:41, 2012.

Li, Chao and Miklau, Gerome. An adaptive mechanism for accurate query answering under differential privacy. volume 5, pp. 514–525, 2012.

Li, Chao, Hay, Michael, Rastogi, Vibhor, Miklau, Gerome, and McGregor, Andrew. Optimizing linear counting queries under differential privacy. In *ACM SIGACT–SIGMOD–SIGART Symposium on Principles of Database Systems (PODS), Indianapolis, Indiana*, pp. 123–134, 2010.

McSherry, F. and Talwar, K. Mechanism design via differential privacy. In *IEEE Symposium on Foundations of Computer Science (FOCS), Providence, Rhode Island*, pp. 94–103, 2007.

Narayanan, A. and Shmatikov, V. Robust de-anonymization of large sparse datasets. In *IEEE Symposium on Security and Privacy (S&P), Oakland, California*, pp. 111–125, 2008.

Netflix. Netflix prize.

Roth, Aaron and Roughgarden, Tim. Interactive privacy via the median mechanism. In *ACM SIGACT Symposium on Theory of Computing (STOC), Cambridge, Massachusetts*, pp. 765–774.

Rubinstein, Benjamin I. P., Bartlett, Peter L., Huang, Ling, and Taft, Nina. Learning in a large function space: Privacy-preserving mechanisms for SVM learning. *Journal of Privacy and Confidentiality*, 4(1):4, 2012.

Thakurta, Abhradeep G. and Smith, Adam. (Nearly) optimal algorithms for private online learning in full-information and bandit settings. In *Conference on Neural Information Processing Systems (NIPS), Lake Tahoe, California*, pp. 2733–2741, 2013.

Thaler, Justin, Ullman, Jonathan, and Vadhan, Salil. Faster algorithms for privately releasing marginals. In *International Colloquium on Automata, Languages and Programming (ICALP), Warwick, England*, pp. 810–821, 2012.

Ullman, J. Answering $n^{2+o(1)}$ counting queries with differential privacy is hard. In *ACM SIGACT Symposium on Theory of Computing (STOC), Palo Alto, California*, pp. 361–370, 2013.

Ullman, J. and Vadhan, S.P. PCPs and the hardness of generating private synthetic data. In *IACR Theory of Cryptography Conference (TCC), Providence, Rhode Island*, pp. 400–416, 2011.

Yaroslavtsev, Grigory, Cormode, Graham, Procopiuc, Cecilia M., and Srivastava, Divesh. Accurate and efficient private release of datacubes and contingency tables. In *IEEE International Conference on Data Engineering (ICDE), Brisbane, Australia*, pp. 745–756, 2013.

Zhang, Jun, Cormode, Graham, Procopiuc, Cecilia M., Srivastava, Divesh, and Xiao, Xiaokui. Privbayes: Private data release via bayesian networks. In *ACM SIGMOD International Conference on Management of Data (SIGMOD), Snowbird, Utah*, 2014.