Differentially Private Data Release for Data Mining

Noman Mohammed
Concordia University
Montreal, QC, Canada
no_moham@encs.concordia.ca

Rui Chen
Concordia University
Montreal, QC, Canada
ru_che@encs.concordia.ca

Benjamin C. M. Fung
Concordia University
Montreal, QC, Canada
fung@ciise.concordia.ca

Philip S. Yu
University of Illinois at Chicago
IL, USA
psyu@cs.uic.edu

ABSTRACT

Privacy-preserving data publishing addresses the problem of disclosing sensitive data when mining for useful information. Among the existing privacy models, ϵ-*differential privacy* provides one of the strongest privacy guarantees and has no assumptions about an adversary's background knowledge. Most of the existing solutions that ensure ϵ-differential privacy are based on an *interactive* model, where the data miner is only allowed to pose aggregate queries to the database. In this paper, we propose the first anonymization algorithm for the *non-interactive* setting based on the generalization technique. The proposed solution first probabilistically generalizes the raw data and then adds noise to guarantee ϵ-differential privacy. As a sample application, we show that the anonymized data can be used effectively to build a decision tree induction classifier. Experimental results demonstrate that the proposed non-interactive anonymization algorithm is scalable and performs better than the existing solutions for classification analysis.

Categories and Subject Descriptors

H.2.7 [**Database Administration**]: [Security, integrity, and protection]; H.2.8 [**Database Applications**]: [Data mining]

General Terms

Algorithms, Performance, Security

Keywords

Differential privacy, anonymization, data mining

1. INTRODUCTION

Due to the rapid advancement in storing, processing, and networking capabilities of computing devices, there has been a tremendous growth in the collection of digital information about individuals. And the emergence of new computing paradigms, such as cloud computing, increases the possibility of large-scale distributed data collection from multiple sources. While the collected data offer tremendous opportunities for mining useful information, there is also a threat to privacy because data in raw form often contain sensitive information about individuals. *Privacy-preserving data publishing* (*PPDP*) studies how to transform raw data into a version that is immunized against privacy attacks but that still supports effective data mining tasks. In this paper, we present an anonymization algorithm to transform a raw data table into a version that satisfies ϵ-*differential privacy* [7] and supports effective classification analysis.

Defining privacy is a difficult task. One of the key challenges is how to model the background knowledge of an adversary. Simply removing explicit identifiers (e.g., name) does not preserve privacy, given that the adversary has some background knowledge about the victim. Sweeney [37] illustrates that 87% of the U.S. population can be uniquely identified based on 5-digit zip code, gender, and date of birth. These attributes are called *quasi-identifier* (*QID*) and the adversary may know these values from publicly available sources such as a voter list. An individual can be identified from published data by simply joining the QID attributes with an external data source.

To limit such disclosure, Samarati and Sweeney [36, 37] propose the k-*anonymity* privacy model, which requires that an individual should not be identifiable from a group of size smaller than k based on the QID. However, Machanavajjhala et al. [28] point out that with additional knowledge about the victim, k-anonymous data is vulnerable against background knowledge attacks. To prevent such attacks, ℓ-*diversity* requires that every QID group should contain at least ℓ "well-represented" values for the sensitive attribute. Similarly, there are a number of other partition-based privacy models such as (α, k)-*anonymity* [41], t-*closeness* [26], and (c, k)-*safety* [29] that model the adversary differently and have different assumptions about its background knowledge.

To transform a raw data table to satisfy a specified privacy requirement, one of the most popular techniques is *generalization* [36, 37]. Generalization replaces a specific value with a more general value to make the information less precise while preserving the "truthfulness" of information. Let Table 1.a be a raw data table (ignore the *class* attribute for now). Generalization can be used to create a 4-anonymous table, as shown in Table 1.b, according to the taxonomy trees given in Figure 1. A large number of anonymization

KDD'11, August 21–24, 2011, San Diego, California, USA.

Table 1: A raw data table and its anonymized versions

(a) Raw data table

Job	Age	Class
Engineer	34	Y
Lawyer	50	N
Engineer	38	N
Lawyer	33	Y
Dancer	20	Y
Writer	37	N
Writer	32	Y
Dancer	25	N

(b) 4-anonymous table

Job	Age
Professional	[18-65)
Professional	[18-65)
Professional	[18-65)
Professional	[18-65)
Artist	[18-65)
Artist	[18-65)
Artist	[18-65)
Artist	[18-65)

(c) Contingency table

Job	Age	Count
Engineer	[18-40)	2
Engineer	[40-65)	0
Lawyer	[18-40)	1
Lawyer	[40-65)	1
Dancer	[18-40)	2
Dancer	[40-65)	0
Writer	[18-40)	2
Writer	[40-65)	0

(d) Generalized contingency table

Job	Age	Count
Professional	[18-40)	3
Professional	[40-65)	1
Artist	[18-40)	4
Artist	[40-65)	0

algorithms [2, 13, 24, 23, 36], tailored for both general and specific data mining tasks, have been proposed based on generalization.

Recently, Wong et al. [39] and Zhang et al. [45] show that these algorithms are vulnerable against minimality attack and do not provide the claimed privacy guarantee. Although several fixes against minimality attack have been proposed [5, 19, 43], new attacks such as *composition attack* [14], *deFinetti attack* [21], and *foreground knowledge attack* [40] have emerged against these algorithms [2, 13, 24, 23, 36]. One way to handle these attacks is to revise the existing algorithms or propose new algorithms while keeping the current privacy models and hoping no other attack will be discovered. Another way is to choose a privacy model that is robust enough to provide a provable privacy guarantee and that is, by definition, immune against all these attacks. We adopt the latter approach in this paper.

Differential privacy [7] is a rigorous privacy model that makes no assumption about an adversary's background knowledge. A differentially-private mechanism ensures that the probability of any output (released data) is equally likely from all nearly identical input data sets and thus guarantees that all outputs are insensitive to any individual's data. In other words, an individual's privacy is not at risk because of her participation in the data set.

1.1 Motivation

Existing algorithms that provide differential privacy guarantee are based on two approaches: *interactive* and *non-interactive*. In an interactive framework, a data miner can pose aggregate queries through a private mechanism, and a database owner answers these queries in response. Most of the proposed methods for ensuring differential privacy are based on an interactive framework [6, 9, 35, 11]. In a non-interactive framework the database owner first anonymizes the raw data and then releases the anonymized version for public use. In this paper we adopt the non-interactive framework and argue that this approach has a number of advantages for data mining.

In an interactive framework privacy is ensured by adding noise to each query response. To ensure privacy a database owner can answer only a limited number of queries before she has to increase the noise level to a point that the answer is no longer useful. Thus, the database can only support a fixed number of queries for a given privacy budget. This is a big problem when there are a large number of data miners because each user (data miner) can only ask a small number of queries. Even for a small number of users, it is not possible to explore the data for testing various hypotheses.

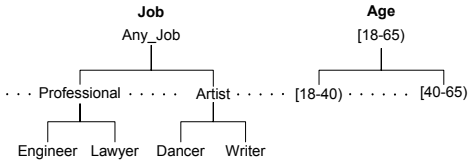


Figure 1: Taxonomy tree of attributes

On the other hand, by releasing the data all data miners get full access to the anonymized data. This gives researchers greater flexibility in performing the required data analysis, and they can fine-tune the data mining results for their research purposes.

Current techniques that adopt the non-interactive approach publish contingency tables or marginals of the raw data [9, 1, 44, 16] (see Section 2 for more discussion). The general structure of these approaches is to first derive a frequency matrix[1] of the raw data over the database domain. For example, Table 1.c shows the contingency table of Table 1.a. After that, noise is added to each count to satisfy the privacy requirement. Finally, the noisy frequency matrix is published. However, this approach is not suitable for high-dimensional data with a large domain because when the added noise is relatively large compared to the count, the utility of the data is significantly destroyed. We also confirm this point in our experimental results (Section 5).

1.2 Contributions

In this paper, we propose a novel technique for privacy-preserving data publishing that provides an ϵ-*differential privacy* [7] guarantee. While protecting privacy is a critical element in data publishing, it is equally important to preserve the utility of the published data because this is the primary reason for data release. Taking the decision tree induction classifier as an example, we show that our anonymization algorithm can be effectively tailored for preserving information for the data mining task. The contributions of this paper are summarized as follow:

1. We present the first generalization-based algorithm for differentially private data release that preserves information for classification analysis. Previous work [27] suggests that generalization technique cannot be used to achieve ϵ-differential privacy as it heavily depends on the underlying data. Yet, we show that differentially private data can be released by adding uncertainty in the generalization procedure. The proposed

[1]For a contingency table, a frequency matrix is computed over all the attributes, whereas a marginal is derived by projecting some of the attributes.

Table 2: Characteristics of PPDP algorithms

Algorithms	Dimension		Attribute Domain		Privacy Model	
	Single	Multi	Leaf Level	Hierarchical	Differential Privacy	Partition-based Privacy
Mondrian [23], TDS [13], etc.	✓	✓		✓		✓
Barak *et al.* [1]	✓	✓	✓		✓	
Hay *et al.* [16]	✓		✓	✓	✓	
Privelet [44]	✓	✓	✓		✓	
Our proposal	✓	✓	✓	✓	✓	

solution first probabilistically generates a generalized contingency table and then adds noise to the counts. For example, Table 1.d is a generalized contingency table of Table 1.a. Thus the count of each partition is typically much larger than the added noise.

2. The proposed algorithm can handle both categorical and numerical attributes. Unlike existing methods [44], it does not require the numerical attribute to be pre-discretized. The algorithm adaptively determines the split points for numerical attributes and partitions the data based on the workload, while guaranteeing ϵ-differential privacy. This is an essential requirement for getting accurate classification, as we show in Section 5. Moreover, the algorithm is very efficient and scales for large data sets.

3. It is well acknowledged that ϵ-differential privacy provides strong privacy guarantee. However, the utility aspect of the differentially-private algorithms has received much less study. Does the interactive approach offer better data mining results than the non-interactive approach? Does differentially private data provide less utility than k-anonymous data? Experimental results demonstrate that our algorithm outperforms the recently proposed differentially-private interactive algorithm for building classifier [11] and the *top-down specialization (TDS)* approach[13] that publishes k-anonymous data for classification analysis.

The rest of the paper is organized as follows. Section 2 reviews related literature. Section 3 overviews ϵ-differential privacy and generalization techniques. Our anonymization algorithm is explained in Section 4. Section 5 experimentally evaluates the performance of our solution. Section 6 concludes the paper.

2. RELATED WORK

Partition-based approach divides a given data set into disjoint groups and releases some general information about the groups. The two most popular anonymization techniques are generalization and bucketization. Generalization [2, 24, 36] makes information less precise while preserving the "truthfulness" of information. Unlike generalization, bucketization [42, 29] does not modify the QID and the sensitive attribute (SA) values but instead de-associates the relationship between the two. However, it thus also disguises the correlation between SA and other attributes and, therefore, hinders data analysis that depends on such correlation.

Many algorithms have been proposed to preserve privacy, but only a few have considered the goal for classification [12]. Iyengar [18] presents the anonymity problem for classification and proposes a genetic algorithmic solution. Bayardo and Agrawal [2] also address the classification problem using the same classification metric of [18]. Fung et al. [13] propose a top-down specialization (TDS) approach to generalize a data table. Recently, LeFevre et. al. [24] propose another anonymization technique for classification using multidimensional recoding [23]. All these algorithms adopt k-anonymity [36, 37] or its extensions [28, 38] as the underlying privacy principle and, therefore, are vulnerable to the recently discovered privacy attacks [39, 14, 21, 40]. More discussion about the partition-based approach can be found in a survey paper [12].

Differential privacy has received considerable attention recently as a substitute for partition-based privacy models for PPDP. However, most of the research on differential privacy so far concentrates on the interactive setting with the goal of reducing the magnitude of added noise [6, 9, 35], releasing certain data mining results [3, 11], or determining the feasibility and infeasibility results of differentially-private mechanisms [4, 20]. A general overview of various research works on differential privacy can be found in the recent survey [8]. Below, we briefly review the results relevant to this paper.

Barak *et al.* [1] address the problem of releasing a set of consistent marginals of a contingency table. Their method ensures that each count of the marginals is non-negative and their sum is consistent for a set of marginals. Xiao et al. [44] propose *Privelet*, a wavelet-transformation-based approach that lowers the magnitude of noise needed to ensure differential privacy to publish a multidimensional frequency matrix. Hay *et al.* [16] propose a method to publish differentially private histograms for a one-dimensional data set. Although Privelet and Hay *et al.*'s approach can achieve differential privacy by adding polylogarithmic noise variance, the latter is only limited to a one-dimensional data set.

Some works [15, 25] address how to compute the results of a number of given queries while minimizing the added noise. However, these methods require the set of queries to be given first altogether to compute the results. In contrast, our method complements the above works by determining how to partition the data adaptively so that the released data can be useful for a given data mining task. In addition, a number of recent works propose differentially-private mechanisms for different applications such as record linkage [17], and recommender systems [31]. Though closely related, all these works do not address the problem of privacy-preserving data publishing for classification analysis, the primary theme of this paper. Table 2 summaries different characteristics of the PPDP algorithms discussed above.

The most relevant work to this paper is *DiffP-C4.5* [11], an interactive algorithm for building a classifier while guaranteeing differential privacy. We have already discussed the shortcomings of an interactive framework and will ex-

perimentally compare our algorithm designed for the non-interactive setting with *DiffP-C4.5* in Section 5.

3. PRELIMINARIES

In this section, we first present an overview of ϵ-differential privacy and the core mechanisms to achieve ϵ-differential privacy. We then introduce the notion of generalization in the context of microdata publishing, followed by a problem statement.

3.1 Differential Privacy

Differential privacy is a recent privacy definition that provides a strong privacy guarantee. Partition-based privacy models ensure privacy by imposing syntactic constraints on the output. For example, the output is required to be indistinguishable among k records, or the sensitive value to be well represented in every equivalent group. Instead, differential privacy guarantees that an adversary learns nothing more about an individual, regardless of whether her record is present or absent in the data. Informally, a differentially private output is insensitive to any particular record. Therefore, from an individual's point of view, the output is computed as if from a data set that does not contain her record.

DEFINITION 3.1 (ϵ-*differential privacy*). A randomized algorithm Ag is differentially private if for all data sets D and D' where their symmetric difference contains at most one record (i.e., $|D \triangle D'| \leq 1$), and for all possible anonymized data sets $\hat{D}$,

$$\Pr[Ag(D) = \hat{D}] \leq e^{\epsilon} \times \Pr[Ag(D') = \hat{D}], \quad (1)$$

where the probabilities are over the randomness of the Ag. ■

The parameter $\epsilon > 0$ is public and specified by a data owner. Lower values of ϵ provide a stronger privacy guarantee. Typically, the values of ϵ should be small, such as 0.01, 0.1, or in some cases $\ln 2$, or $\ln 3$ [8]. When ϵ is very small, we have $e^{\epsilon} \approx 1 + \epsilon$.

A standard mechanism to achieve differential privacy is to add random noise to the true output of a function. The noise is calibrated according to the *sensitivity* of the function. The sensitivity of a function is the maximum difference of its outputs from two data sets that differ only in one record.

DEFINITION 3.2 (*Sensitivity*). For any function $f : D \rightarrow \mathbb{R}^d$, the sensitivity of f is

$$\Delta f = \max_{D,D'} ||f(D) - f(D')||_1 \quad (2)$$

for all D, D' differing in at most one record. ■

EXAMPLE 1. Consider the raw data set of Table 1.a. Let f be a function that counts the number of records with Age less than 40. Then, the Δf is 1 because $f(D)$ can differ at most 1 due to the addition or removal of a single record. ■

Laplace Mechanism. Dwork et al. [9] propose the Laplace mechanism. The mechanism takes a data set D, a function f, and the parameter λ that determines the magnitude of noise as inputs. It first computes the true output $f(D)$, and then perturbs the output by adding noise. The noise is generated according to a Laplace distribution with probability density function $\Pr(x|\lambda) = \frac{1}{2\lambda}\texttt{exp}(-|x|/\lambda)$; its variance is $2\lambda^2$ and mean is 0. The following theorem connects the sensitivity to the magnitude of noise and guarantees that perturbed output $f(\hat{D}) = f(D) + \texttt{Lap}(\lambda)$ satisfies ϵ-differential privacy, where $\texttt{Lap}(\lambda)$ is a random variable sampled from the Laplace distribution.

THEOREM 3.1. *[9] For any function $f : D \rightarrow \mathbb{R}^d$, the algorithm Ag that adds independently generated noise with distribution $\texttt{Lap}(\Delta f/\epsilon)$ to each of the d outputs satisfies ϵ-differential privacy.*

EXAMPLE 2. Continue from Example 1. The mechanism first computes the true count $f(D)$ and then outputs the noisy answer $f(\hat{D}) = f(D) + \texttt{Lap}(1/\epsilon)$. ■

Exponential Mechanism. McSherry and Talwar [32] propose the exponential mechanism that can choose an output $t \in \mathcal{T}$ that is close to the optimum with respect to a utility function while preserving differential privacy. The exponential mechanism takes as inputs a data set D, output range $\mathcal{T}$, privacy parameter ϵ, and a utility function $u : (D \times \mathcal{T}) \rightarrow \mathbb{R}$ that assigns a real valued score to every output $t \in \mathcal{T}$, where a higher score means better utility.

The mechanism induces a probability distribution over the range $\mathcal{T}$ and then samples an output t. Let $\Delta u = \max_{\forall t, D, D'} |u(D,t) - u(D',t)|$ be the sensitivity of the utility function. The probability associated with each output is proportional to $\texttt{exp}(\frac{\epsilon u(D,t)}{2\Delta u})$; that is, the output with a higher score is exponentially more likely to be chosen.

THEOREM 3.2. *[32] For any function $u : (D \times \mathcal{T}) \rightarrow \mathbb{R}$, an algorithm Ag that chooses an output t with probability proportional to $\texttt{exp}(\frac{\epsilon u(D,t)}{2\Delta u})$ satisfies ϵ-differential privacy.*

3.2 Generalization

Let $D = \{r_1, \ldots, r_n\}$ be a multiset of records, where each record r_i represents the information of an individual with d attributes $\mathcal{A} = \{A_1, \ldots, A_d\}$. We represent the data set D in a tabular form and use the terms "data set" and "data table" interchangeably. We assume that each attribute A_i has a finite domain, denoted by $\Omega(A_i)$. The domain of D is defined as $\Omega(D) = \Omega(A_1)\times, \ldots, \times\Omega(A_d)$. To anonymize a data set D, generalization replaces a value of an attribute with a more general value. The exact general value is determined according to the attribute partition.

DEFINITION 3.3 (*Attribute Partition*). The partitions $P(A_i)$ of a numerical attribute are the intervals $\langle I_1, I_2, \ldots, I_k\rangle$ in $\Omega(A_i)$ such that $\bigcup_{j=1}^{k} I_j = \Omega(A_i)$. For categorical attribute, partitions are defined by a set of nodes from the taxonomy tree such that it covers the whole tree, and each leaf node belongs to exactly one partition. ■

For example, *Artist* is the general value of *Dancer* according to the taxonomy tree of *Job* in Figure 1. And, *Age* 23 can be represented by the interval $[18 - 40)$. For numerical attributes, these intervals are determined adaptively from the data set.

DEFINITION 3.4 (*Generalization*). Generalization is defined by a function $\Phi = \{\phi_1, \phi_2, \ldots, \phi_d\}$, where $\phi_i : v \rightarrow p$ maps each value $v \in \Omega(A_i)$ to a $p \in P(A_i)$. ■

Clearly, given a data set D over a set of attributes $\mathcal{A} = \{A_1, \ldots, A_d\}$, many alternative generalization functions are feasible. Each generalization function partitions the attribute domains differently. To satisfy the ϵ-differential privacy requirement the algorithm must determine a generalization function that is insensitive to the underlying data. More formally, for any two data sets D and D', where $D \triangle D' = 1$, the algorithm must ensure that the ratio of $\Pr[Ag(D) = \Phi]$ and $\Pr[Ag(D') = \Phi]$ is bounded.

One naive solution that satisfies ϵ-differential privacy is to have a fixed generalization function, irrespective of the input database. However, a proper choice of generalization function is very crucial since the data mining result varies significantly for different choices of partitioning. In Section 4 we present an efficient algorithm for determining an adaptive partitioning technique for classification analysis that guarantees ϵ-differential privacy.

3.3 Problem Statement

Suppose a data owner wants to release a data table $D(A_1^{pr}, \ldots, A_d^{pr}, A^{cls})$ to the public for classification analysis. The attributes in D are classified into three categories: (1) An *explicit identifier* A^i attribute that explicitly identifies an individual, such as *SSN*, and *Name*. These attributes are removed before releasing the data. (2) A *class* attribute A^{cls} that contains the class value, and the goal of the data miner is to build a classifier to accurately predict the value of this attribute. (3) A set of d *predictor* attributes $\mathcal{A}^{pr} = \{A_1^{pr}, \ldots, A_d^{pr}\}$, whose values are used to predict the class attribute.

We require the class attribute to be categorical, and the predictor attribute can be either numerical or categorical. Further, we assume that for each categorical-predictor attribute A_i^{pr}, a taxonomy tree is provided. The taxonomy tree of an attribute A_i^{pr} specifies the hierarchy among the values in $\Omega(A_i^{pr})$. Next, we give our problem statement.

Given a data table D and the privacy parameter ϵ, our objective is to generate an anonymized data table $\hat{D}$ such that (1) $\hat{D}$ satisfies ϵ-differential privacy, and (2) preserves as much information as possible for classification analysis.

4. THE ALGORITHM

In this section, we first present an overview of our *Diff*erentially-private anonymization algorithm based on *Gen*eralization (*DiffGen*). We then elaborate the key steps, and prove that the algorithm is ϵ-differential private. Finally, we present the implementation details and analyze the complexity of the algorithm.

4.1 Overview

Algorithm 1 first generalizes the predictor attributes $\mathcal{A}^{pr}$ and thus divides the raw data into several equivalence groups, where all the records within a group have the same attribute values. Then the algorithm publishes the noisy counts of the groups. The general idea is to anonymize the raw data by a sequence of specializations, starting from the topmost general state as shown in Figure 2. A *specialization*, written $v \rightarrow child(v)$, where $child(v)$ denotes the set of child values of v, replaces the parent value v with a child value. The specialization process can be viewed as pushing the "cut" of each taxonomy tree downwards. A *cut* of the taxonomy tree for an attribute A_i^{pr}, denoted by Cut_i, contains exactly one value on each root-to-leaf path. Figure 1 shows a solution cut indicated by the dashed curve representing the anonymous Table 1.d.

Algorithm 1 DiffGen

Input: Raw data set D, privacy budget ϵ, and number of specializations h.
Output: Generalized data set $\hat{D}$

1: Initialize every value in D to the topmost value;
2: Initialize Cut_i to include the topmost value;
3: $\epsilon' \leftarrow \frac{\epsilon}{2(|A_n^{pr}|+2h)}$;
4: Determine the split value for each $v_n \in \cup Cut_i$ with probability $\propto \texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v_n))$;
5: Compute the score for $\forall v \in \cup Cut_i$;
6: **for** $i = 1$ to h **do**
7: Select $v \in \cup Cut_i$ with probability $\propto \texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v))$;
8: Specialize v on D and update $\cup Cut_i$;
9: Determine the split value for each new $v_n \in \cup Cut_i$ with probability $\propto \texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v_n))$;
10: Update score for $v \in \cup Cut_i$;
11: **end for**
12: **return** each group with count $(C + \texttt{Lap}(2/\epsilon))$

The specialization starts from the topmost cut and pushes down the cut iteratively by specializing some value in the current cut. Initially, all values in $\mathcal{A}^{pr}$ are generalized to the topmost value in their taxonomy trees (Line 1), and Cut_i contains the topmost value for each attribute A_i^{pr} (Line 2). At each iteration *DiffGen* probabilistically selects a candidate $v \in \cup Cut_i$ for specialization (Line 7). Candidates are selected based on their score values, and different heuristics (e.g., information gain) can be used to determine the score of the candidates. Then, the algorithm specializes v and updates $\cup Cut_i$ (Line 8). Finally, it updates the score of the affected candidates due to the specialization (Line 10). *DiffGen* terminates after a given number of specializations. The proposed algorithm can also be used to publish a contingency table by allowing the specialization to continue until it reaches the leaf level of the attribute domains.

Example 3. Consider the raw data set of Table 1.a. Initially the algorithm creates one root partition containing all the records that are generalized to ⟨*Any_Job, [18-65)*⟩. $\cup Cut_i$ includes {*Any_Job*, *[18-65)*}. Let the first specialization be *Any_Job* → {*Professional*, *Artist*}. The algorithm creates two new partitions under the root, as shown in Figure 2, and splits data records between them. $\cup Cut_i$ is updated to {*Professional*, *Artist*, *[18-65)*}. Suppose that the next specialization is *[18-65)* → {*[18-40)*,*[40-65)*}, which creates further specialized partitions. Finally, the algorithm outputs the equivalence groups of each leaf partition along with their noisy counts. ■

4.2 Privacy Analysis

We next elaborate the key steps of the algorithm: (1) selecting a candidate for specialization, (2) determining the split value, and (3) publishing the noisy counts. We show that each of these steps preserves privacy, and then we use the composition properties of differential privacy to guarantee that *DiffGen* is ϵ-differentially private.

(1) Candidate Selection. We use an exponential mechanism (see Section 3) to select a candidate for specialization in each round. We define two utility functions to calculate the score

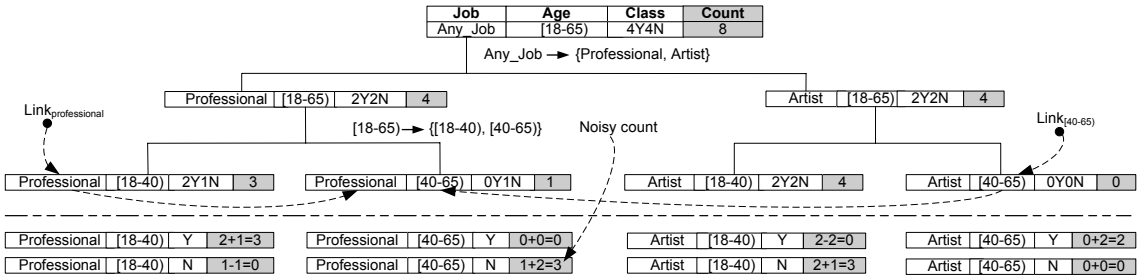


Figure 2: Tree for partitioning records

of each candidate $v \in \cup Cut_i$. The first utility function is *information gain*. Let D_v denote the set of records in D generalized to the value v. Let $|D_v^{cls}|$ denote the number of records in D_v having the class value $cls \in \Omega(A^{cls})$. Note that $|D_v| = \sum_c |D_c|$, where $c \in child(v)$. Then, we get

$$\texttt{InfoGain}(D, v) = H_v(D) - H_{v|c}(D), \tag{3}$$

where $\texttt{H}_v(D) = -\sum_{cls} \frac{|D_v^{cls}|}{|D_v|} \times \log_2 \frac{|D_v^{cls}|}{|D_v|}$ is the *entropy* of candidate v with respect to the class attribute A^{cls} and $\texttt{H}_{v|c}(D) = \sum_c \frac{|D_c|}{|D_v|}\texttt{H}_c(D)$ is the conditional entropy given the candidate is specialized. The sensitivity of $\texttt{InfoGain}(D, v)$ is $\log_2 |\Omega(A^{cls})|$, where $|\Omega(A^{cls})|$ is the domain size of the class attribute A^{cls}. It is because the value of the entropy $\texttt{H}_v(D)$ must be between 0 and $\log_2 |\Omega(A^{cls})|$. And, the value of the conditional entropy $\texttt{H}_{v|c}(D)$ lies between 0 and $\texttt{H}_v(D)$. Therefore, the maximum change of $\texttt{InfoGain}(D, v)$ due to the addition or removal of a record is bounded by $\log_2 |\Omega(A^{cls})|$.

The second utility function is:

$$\texttt{Max}(D, v) = \sum_{c \in child(v)} (\max_{cls}(|D_c^{cls}|)). \tag{4}$$

$\texttt{Max}(D, v)$ is the summation of the highest class frequencies over all child values and the sensitivity of this function is 1 because the value of $\texttt{Max}(D, v)$ can vary at most 1 due to the change of a record.

Given the scores of all the candidates, exponential mechanism selects a candidate v_i with the following probability,

$$\frac{\texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v_i))}{\sum_{v \in \cup Cut_i} \texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v))}, \tag{5}$$

where the $\texttt{u}(D, v)$ is either $\texttt{InfoGain}(D, v)$ or $\texttt{Max}(D, v)$ and the sensitivity of the function Δu is $\log_2 |\Omega(A^{cls})|$ and 1, respectively. Thus, from Theorem 3.2, Line 7 of Algorithm 1 satisfies ϵ'-differential privacy. The beauty of the exponential mechanism is that while it ensures privacy, it also exponentially favors a candidate with a high score.

(2) Split Value. Once a candidate is determined, *DiffGen* splits the records into child partitions. The split value of a categorical attribute is determined according to the taxonomy tree of the attribute. Since the taxonomy tree is fixed, the sensitivity of the split value is 0. Therefore, splitting the records according to the taxonomy tree does not violate differential privacy.

For numerical attributes, a split value cannot be directly chosen from the attribute values that appear in the data set D, because the probability of selecting the same split value from a different data set D' not containing this value is 0. We again use an exponential mechanism to determine the split value. We first partition the domain into intervals $I_1, \ldots, I_k$ such that all values within an interval have the same score. Then, the exponential mechanism is used to select an interval I_i with the following probability,

$$\frac{\texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v_i)) \times |\Omega(I_i)|}{\sum_{j=1}^{k}(\texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v_j)) \times |\Omega(I_j)|)}, \tag{6}$$

where $v_i \in \Omega(I_i)$, and $|\Omega(I_i)|$ is the length of the interval. After selecting the interval, the split value is determined by sampling a value uniformly from the interval. Thus, the probability of selecting a value $v_i \in \Omega(A_i)$ is

$$\frac{\texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v_i))}{\int_{v \in \Omega(A_i)} \texttt{exp}(\frac{\epsilon'}{2\Delta \texttt{u}}\texttt{u}(D, v))\, dv} \tag{7}$$

This satisfies ϵ'-differential privacy because the probability of choosing any value is proportional to $\texttt{exp}(\frac{\epsilon' u(D, v_i)}{2\Delta u})$.

(3) Noisy Counts. Each leaf partition contains $|\Omega(A^{cls})|$ equivalence groups. Publishing the exact counts of these groups does not satisfy differential privacy since for a different data set D', the counts may change. This change can be easily offset by adding noise to the count of each group according to the Laplace mechanism (See Theorem 3.1). As discussed earlier, the sensitivity of count query is 1; therefore, to satisfy $\frac{\epsilon}{2}$-differential privacy, *DiffGen* adds $\texttt{Lap}(2/\epsilon)$ noise to each true count C of the groups (Line 12). We post-process the noisy counts by rounding each count to the nearest non-negative integer. Note that post-processing does not violate the differential privacy [22].

Next, we use composition properties of differential privacy to guarantee that the proposed algorithm satisfies ϵ-differential privacy as a whole.

LEMMA 4.1 (*Sequential composition* [30]). *Let each Ag_i provide ϵ_i-differential privacy. A sequence of $Ag_i(D)$ over the data set D provides $(\sum_i \epsilon_i)$-differential privacy.*

LEMMA 4.2 (*Parallel composition* [30]). *Let each Ag_i provide ϵ-differential privacy. A sequence of $Ag_i(D_i)$ over a set of disjoint data sets D_i provides ϵ-differential privacy.*

Any sequence of computations that each provides differential privacy in isolation also provides differential privacy in sequence, which is known as *sequential composition*. However, if the sequence of computations is conducted on *disjoint* data sets, the privacy cost does not accumulate but depends only on the worst guarantee of all computations. This is known as *parallel composition*.

THEOREM 4.1. *DiffGen is ϵ-differentially private.*

PROOF. (Sketch) The algorithm first determines the split value for each numerical attribute using the exponential mechanism (Line 4). Since the cost of each exponential

mechanism is ϵ', Line 4 of the algorithm preserves $\epsilon'|A_n^{pr}|$-differential privacy, where $|A_n^{pr}|$ is the number of numerical attributes.

In Line 7, the algorithm selects a candidate for specialization. This step uses the exponential mechanism and thus, candidate selection step guarantees ϵ'-differential privacy for each iteration. In Line 9, the algorithm determines the split value for each new numerical candidate $v_n \in \cup Cut_i$. All records in the same partition have the same generalized values on $\mathcal{A}^{pr}$; therefore, each partition can only contain at most one candidate value v_n. Thus, determining the split value for the new candidates requires at most ϵ' privacy budget for each iteration due to the parallel composition property. Note that this step does not take place in every iteration.

Finally, the algorithm outputs the noisy count of each group (Line 12) using the Laplace mechanism and guarantees $\frac{\epsilon}{2}$-differential privacy. Therefore, for $\epsilon' = \frac{\epsilon}{2(|A_n^{pr}|+2h)}$, *DiffGen* is ϵ-differentially private. ■

4.3 Implementation

A simple implementation of *DiffGen* is to scan *all* data records to compute scores for all candidates in $\cup Cut_i$. Then scan all the records again to perform the specialization. A key contribution of this work is an efficient implementation of the proposed algorithm that computes scores based on some information maintained for candidates in $\cup Cut_i$ and provides *direct access* to the records to be specialized, instead of scanning all data records. We briefly explain the efficient implementation of the algorithm as follows.

Initial Steps (Lines 1-5). Initially, we determine split points for all numerical candidates (Line 4). First, the data is sorted with respect to the split attribute, which requires $O(|D| \log |D|)$. Then the data is scanned once to determine the score for all attribute values that appear in the data set D. An interval is represented by two successive different attribute values. Finally, the exponential mechanism is used to determine the split point. We also compute the scores for all candidates $v \in \cup Cut_i$ (Line 5). This can be done by scanning the data set once. However, for each subsequent iteration, information needed to calculate scores comes from the update of the previous iteration (Line 10). Thus the worst-case runtime of this step is $O(|\mathcal{A}^{pr}| \times |D| \log |D|)$.

Perform Specialization (Line 8). To perform a specialization $v \rightarrow child(v)$, we need to retrieve D_v, the set of data records generalized to v. To facilitate this operation we organize the records in a tree structure, with each root-to-leaf path representing a generalized record over $\mathcal{A}^{pr}$, as shown in Figure 2. Each leaf partition (node) stores the set of data records having the same generalized record for $\mathcal{A}^{pr}$ attributes. For each v in $\cup Cut_i$, P_v denotes a leaf partition whose generalized record contains v, and $Link_v$ provides direct access to all P_v partitions generalized to v.

Initially, the tree has only one leaf partition containing all data records, generalized to the topmost value on every attribute in $\mathcal{A}^{pr}$. In each iteration we perform a specialization v by refining the leaf partitions on $Link_v$. For each value $c \in child(v)$, a new partition P_c is created from P_v, and data records in P_v are split among the new partitions. This is the *only* operation in the whole algorithm that requires scanning data records. In the same scan, we also collect the following information for each c: $|D_c|$, $|D_g|$, $|D_c^{cls}|$ and $|D_g^{cls}|$, where $g \in child(c)$ and cls is a class label. These pieces of information are used in Line 10 to update scores. Thus, the total runtime of this step is $O(|D|)$.

Determine the Split Value (Line 9). If a numerical candidate v_n is selected in Line 7, then we need to determine the split points for two new numerical candidates $c_n \in child(v_n)$. This step takes time $O(|D| \log |D|)$.

Update Score (Line 10). Both `InfoGain` and `Max` scores of the other candidates $x \in \cup Cut_i$ are not affected by $v \rightarrow child(v)$, except that we need to compute the scores of each newly added value $c \in child(v)$. The scores of the new candidates are computed using the information collected in Line 8. Thus, this step can be done in constant $O(1)$ time.

Exponential Mechanism (Lines 4, 7 and 9). The cost of the exponential mechanism is proportional to the number of discrete alternatives from which it chooses a candidate. For Line 7, the cost is $O(|\cup Cut_i|)$, and for Lines 4 and 9 the cost is $O(|\mathbf{I}|)$, where $|\mathbf{I}|$ is the number of intervals. Usually both $|\cup Cut_i|$ and $|\mathbf{I}|$ are much smaller than $|D|$.

In summary, the cost of the initial steps and Lines 7-10 are $O(|\mathcal{A}^{pr}| \times |D| \log |D|)$ and $O(h \times |D| \log |D|)$, respectively. Hence, for a fixed number of attributes the total runtime of *DiffGen* is $O(h \times |D| \log |D|)$.

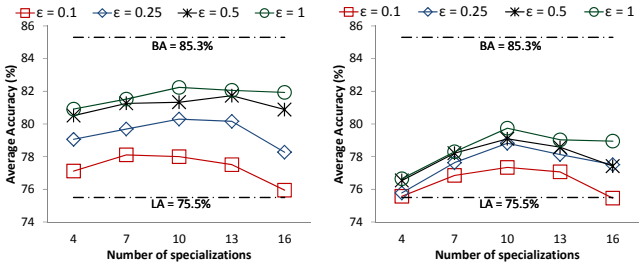
5. EXPERIMENTAL EVALUATION

In this section our objectives are to study the impact of enforcing differential privacy on the data quality in terms of classification accuracy, and to evaluate the scalability of the proposed algorithm for handling large data sets. We also compare *DiffGen* with *DiffP-C4.5* [11], a differentially-private interactive algorithm for building a classifier, and with the *top-down specialization (TDS)* approach [13] that publishes k-anonymous data for classification analysis.

We employ the publicly available *Adult* [10] data set, a real-life census data set that has been used for testing many anonymization algorithms [2, 13, 18, 28, 38, 33]. *Adult* has 45,222 census records with 6 numerical attributes, 8 categorical attributes, and a binary *class* column representing two income levels, ≤50K or >50K. See [13] for the description of attributes. All experiments were conducted on an Intel Core *i*7 2.7GHz PC with 12GB RAM.

Data Quality. To evaluate the impact on classification quality we divide the data into training and testing sets. First, we apply our algorithm to anonymize the training set and to determine the $\cup Cut_i$. Then, the same $\cup Cut_i$ is applied to the testing set to produce a generalized testing set. Next, we build a classifier on the anonymized training set and measure the *classification accuracy* (CA) on the generalized records of the testing set. For classification models we use the well-known C4.5 classifier [34]. To better visualize the cost and benefit of our approach we provide additional measures: *Baseline Accuracy* (BA) is the classification accuracy measured on the raw data without anonymization. $BA - CA$ represents the cost in terms of classification quality for achieving a given ϵ-differential privacy requirement. On the other extreme, we measure *Lower bound Accuracy* (LA), which is the accuracy on the raw data with all attributes (except for the *class* attribute) removed. $CA - LA$ represents the benefit of our method over the naive non-disclosure approach.

Figure 3.a depicts the classification accuracy CA for the utility function `Max`, where the privacy budget $\epsilon = 0.1$, 0.25, 0.5, 1, and the number of specializations $4 \leq h \leq 16$. The BA and LA are 85.3% and 75.5%, respectively, as shown in



(a) Max (b) Information gain

Figure 3: Classification accuracy of *DiffGen*

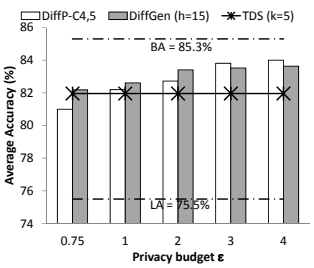


Figure 4: Comparison

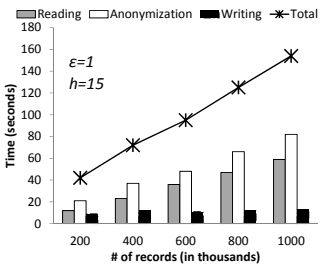


Figure 5: Scalability

the figure by the dotted lines. We use 2/3 of the records to build the classifier and measure the accuracy on the remaining 1/3 of the records. For each experiment we executed 10 runs and averaged the results over the runs. For $\epsilon = 1$ and $h = 10$, $BA - CA$ is around 3% and $CA - LA$ is 6.74%. For $\epsilon = 0.5$, $BA - CA$ spans from 3.57% to 4.8%, and $CA - LA$ spans from 5% to 6.23%. However, as ϵ decreases to 0.1, CA quickly decreases to about 78% (highest point), the cost increases to about 7%, and the benefit decreases to about 3%. These results suggest that for an acceptable privacy budget such as 1, the cost for achieving ϵ-differential privacy is small, while the benefit of our method over the naive method is large. Figure 3.b depicts the classification accuracy CA for the utility function `InfoGain`. The performance of the `InfoGain` is not as good as `Max` because the difference between the scores of a good and a bad attribute is much smaller for `InfoGain` as compared to `Max`. Therefore, exponential mechanism does not work effectively in the case of `InfoGain` as it does for `Max`.

We observe two general trends from the experiments. First, the privacy budget has a direct impact on the classification accuracy. A higher budget results in better accuracy since it ensures better attribute partitioning and lowers the magnitude of noise that is added to the count of each equivalence group. Second, the classification accuracy initially increases with the increase of the number of specializations. However, after a certain threshold the accuracy decreases with the increase of the number of specializations. This is an interesting observation. The number of equivalence groups increases quite rapidly with an increase in the number of specializations, resulting in a smaller count per group. Up to a certain threshold it has a positive impact due to more precise values; however, the influence of the Laplace noise gets stronger as the number of specializations grows. Note that if the noise is as big as the count, then the data is useless. This confirms that listing all the possible combination of values (i.e., contingency table) and then adding noise to their counts is not a good approach for high-dimensional data since the noise will be as big as the count.

Since this is a non-interactive approach, the data owner can try different values of h to find the threshold and then release the anonymized data. Determining a good value of h adaptively, given the data set and the privacy budget, is an interesting future work.

Comparison. Figure 4 shows the classification accuracy CA of *DiffGen*, *DiffP-C4.5*, and *TDS*. For *DiffGen*, we use utility function `Max` and fix the number of specializations $h = 15$. *DiffP-C4.5* also uses *Adult* data set and all the results of the *DiffP-C4.5* are taken from their paper [11]. For *TDS* we fixed the anonymity threshold $k = 5$ and conducted the experiment ourselves. Following the same setting of [11], we executed 10 runs of 10-fold cross-validation to measure the CA. 10-fold cross-validation yields higher CA since more training records are available.

The accuracy of *DiffGen* is clearly better than *DiffP-C4.5* for privacy budget $\epsilon \leq 2$. Note that the privacy budget should be typically smaller than 1 [11, 7, 8]. Even for a higher budget, the accuracy of *DiffGen* is comparable to *DiffP-C4.5*. The major advantage of our algorithm is that we publish data and the data miner has much better flexibility to perform the required data analysis. On the other hand, in *DiffP-C4.5* the classifier is built through interactive queries; therefore, the database has to be permanently shut down to satisfy the privacy requirement after generating only *one* classifier.

The experimental result also shows that *DiffGen* performs better than *TDS*. For a higher anonymity threshold k, the accuracy of *TDS* will be lower. One advantage of *DiffGen* is that, unlike *TDS*, it does not need to ensure that every equivalence group contains k records; therefore, *DiffGen* is able to provide more detailed information than *TDS*. This result demonstrates for the first time that, if designed properly, a differentially private algorithm can provide better utility than a partition-based approach.

Scalability. All the previous experiments can finish the anonymization process within 30 seconds. We further study the scalability of our algorithm over large data sets. We generate different data sets of different sizes by randomly adding records to the Adult data set. For each original record r, we create $\alpha - 1$ variations of the record by replacing some of the attribute values randomly from the same domain. Here α is the blowup scale and thus the total number of records is $\alpha \times 45,222$ after adding random records. Figure 5 depicts the runtime from 200,000 to 1 million records for $h = 15$ and $\epsilon = 1$.

6. CONCLUSIONS

This paper presents a new anonymization algorithm that achieves differential privacy and supports effective classification analysis. The proposed solution connects the classical generalization technique with output perturbation to effectively anonymize raw data. Experimental results suggest that the proposed solution provides better utility than the interactive approach and the k-anonymous data, and that it is more effective than publishing a contingency table.

7. ACKNOWLEDGMENTS

We sincerely thank the reviewers for their insightful comments. The research is supported in part by the Discovery Grants (356065-2008), Strategic Project Grants (130570020), and Canada Graduate Scholarships from the Natural Sciences and Engineering Research Council of Canada, and the NSF grant IIS-0914934.

8. REFERENCES

[1] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: A holistic solution to contingency table release. In *PODS*, 2007.

[2] R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE*, 2005.

[3] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta. Discovering frequent patterns in sensitive data. In *SIGKDD*, 2010.

[4] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *STOC*, 2008.

[5] G. Cormode, D. Srivastava, N. Li, and T. Li. Minimizing minimality and maximizing utility: Analyzing methodbased attacks on anonymized data. In *VLDB*, 2010.

[6] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *PODS*, 2003.

[7] C. Dwork. Differential privacy. In *ICALP*, 2006.

[8] C. Dwork. A firm foundation for private data analysis. *Commun. ACM*, 54(1):86–95, 2011.

[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

[10] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[11] A. Friedman and A. Schuster. Data mining with differential privacy. In *SIGKDD*, 2010.

[12] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys*, 42(4):1–53, June 2010.

[13] B. C. M. Fung, K. Wang, and P. S. Yu. Anonymizing classification data for privacy preservation. *IEEE TKDE*, 19(5):711–725, May 2007.

[14] S. R. Ganta, S. Kasiviswanathan, and A. Smith. Composition attacks and auxiliary information in data privacy. In *SIGKDD*, 2008.

[15] M. Hardt and K. Talwar. On the geometry of differential privacy. In *STOC*, 2010.

[16] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. In *VLDB*, 2010.

[17] A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino. Private record matching using differential privacy. In *EDBT*, 2010.

[18] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *SIGKDD*, 2002.

[19] X. Jin, N. Zhang, and G. Das. Algorithm-safe privacy-preserving data publishing. In *EDBT*, 2010.

[20] S. P. Kasiviswanathan, M. Rudelson, A. Smith, and J. Ullman. The price of privately releasing contingency tables and the spectra of random matrices with correlated rows. In *STOC*, 2010.

[21] D. Kifer. Attacks on privacy and de finetti's theorem. In *SIGMOD*, 2009.

[22] D. Kifer and B. Lin. Towards an axiomatization of statistical privacy and utility. In *PODS*, 2010.

[23] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *ICDE*, 2006.

[24] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Workload-aware anonymization techniques for large-scale data sets. *ACM TODS*, 33(3), 2008.

[25] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing linear counting queries under differential privacy. In *PODS*, 2010.

[26] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and ℓ-diversity. In *ICDE*, 2007.

[27] A. Machanavajjhala, J. Gehrke, and M. Gotz. Data publishing against realistic adversaries. In *VLDB*, 2009.

[28] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. ℓ-diversity: Privacy beyond k-anonymity. *ACM TKDD*, 2007.

[29] D. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J. Halpern. Worst-case background knowledge in privacy-preserving data publishing. In *ICDE*, 2007.

[30] F. McSherry. Privacy integrated queries. In *SIGMOD*, 2009.

[31] F. McSherry and I. Mironov. Differentially private recommender systems: building privacy into the net. In *SIGKDD*, 2009.

[32] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, 2007.

[33] N. Mohammed, B. C. M. Fung, P. C. K. Hung, and C. Lee. Anonymizing healthcare data: A case study on the blood transfusion service. In *SIGKDD*, 2009.

[34] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[35] A. Roth and T. Roughgarden. Interactive privacy via the median mechanism. In *STOC*, 2010.

[36] P. Samarati. Protecting respondents' identities in microdata release. *IEEE TKDE*, 2001.

[37] L. Sweeney. k-anonymity: A model for protecting privacy. In *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 2002.

[38] K. Wang, B. C. M. Fung, and P. S. Yu. Handicapping attacker's confidence: An alternative to k-anonymization. *KAIS*, 11(3):345–368, April 2007.

[39] R. C. W. Wong, A. W. C. Fu, K. Wang, and J. Pei. Minimality attack in privacy preserving data publishing. In *VLDB*, 2007.

[40] R. C. W. Wong, A. W. C. Fu, K. Wang, Y. Xu, and P. S. Yu. Can the utility of anonymized data be used for privacy breaches? *ACM TKDD, to appear*.

[41] R. C. W. Wong, J. Li., A. W. C. Fu, and K. Wang. (α,k)-anonymity: An enhanced k-anonymity model for privacy preserving data publishing. In *SIGKDD*, 2006.

[42] X. Xiao and Y. Tao. Personalized privacy preservation. In *SIGMOD*, 2006.

[43] X. Xiao, Y. Tao, and N. Koudas. Transparent anonymization: Thwarting adversaries who know the algorithm. *ACM TODS*, 35(2), 2010.

[44] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In *ICDE*, 2010.

[45] L. Zhang, S. Jajodia, and A. Brodsky. Information disclosure under realistic assumptions: Privacy versus optimality. In *CCS*, 2007.