A Privacy Attack That Removes the Majority of the Noise From Perturbed Data

Michal Sramka

***Abstract*— Data perturbation is a sanitization method that helps restrict the disclosure of sensitive information from published data. We present an attack on the privacy of the published data that has been sanitized using data perturbation. The attack employs data mining to remove some noise from the perturbed sensitive values. Our attack is practical, can be launched by non-expert adversaries, and it does not require any background knowledge. Extensive experiments were performed on four databases derived from UCI's Adult and IPUMS census-based data sets sanitized with noise addition that satisfies ϵ-differential privacy. The experimental results confirm that our attack presents a significant privacy risk to published perturbed data. The results show that up to 93% of the noise added during perturbation can be effectively removed using general-purpose data miners from the Weka software package. Interestingly, the higher the aimed privacy, the higher the percentage of noise can be removed. This suggests that adding more noise does not always increase the real privacy.**

I. Introduction

Data perturbation is a well-studied sanitization method that simultaneously allows access to the data by publishing them and at the same time preserves privacy of the data. Typically it is achieved using *noise addition*, a technique that adds noise to the data values or query answers which are then released. It can be also achieved through different techniques, but we concentrate on noise addition. Although extensive research has been done in the area of data perturbation, and especially in the context of privacy in statistical databases and statistical disclosure control, released data is still prone to sensitive data disclosures and other privacy breaches.

The additive noise used to perturb data is typically chosen from some distribution. Uniform, normal, exponential and other distributions were considered in the literature. Initially, it was unclear how much noise needs to be added to achieve privacy, and so many ad-hoc approaches were proposed. A systematic treatment came with the introduction of the *differential privacy* notion, which limits the amount of exponential noise that needs to be added while providing strong theoretical privacy guarantees.

Data mining is a method to extract implicit, previously unknown, and potentially useful information from data. It has long been used in the area of data privacy, for example, for measuring the usefulness of the released data or more recently for comparing the trade-off between privacy and data utility of different sanitization methods. We have also used it for privacy attacks against anonymized data, in particular for predicting values that may lead to privacy breaches. Likewise, data mining can be used to remove some or all of the added noise from the perturbed released data. We explore this idea, specifically, we use data mining to partially remove noise from perturbed data that satisfies ϵ-differential privacy. We focus on establishing that, in some scenarios, it is possible to remove the majority of the added noise using data mining techniques alone, without using or assuming any background knowledge (sometimes also referred to as external knowledge or auxiliary information) that is usually available to the adversary.

We illustrate the scenario and attack on the following example: A data owner wants to release its data collection about individuals while protecting the privacy of their ages. To achieve the privacy, the data owner sanitizes the data by adding random noise limited by the ϵ-differential privacy to all the ages individually, and releases the sanitized data. An adversary launches the attack by applying one or more data mining algorithms to uncover hidden trends and patterns in the data. The mining results allow the adversary to make predictions of ages. The predictions should be seen as estimates of the original ages and thus added noise can be partially removed.

A. Our Contribution

We propose a noise removal process based on data mining and fusion methods. Basically, multiple data mining algorithms are applied to the released sanitized data in order to obtain multiple predictions of each of the sensitive values (e.g., age). These multiple predictions are fused together and the combined value then represents an estimate of the original sensitive value. An estimate that is possibly closer to the original value, in other words, a value that may have some of the noise removed.

The proposed attack relies on data mining [1] and fusion techniques [2], [3] and does not assume any background knowledge available to the adversary. Furthermore, this attack is very practical – it can be launched by adversaries that have no or very little knowledge about data mining and fusion, e.g., reporters. We recognize that expert adversaries using specific and tailored data mining and fusion techniques can achieve higher success in the attack. However, we still concentrate on the non-expert adversaries, because their

Michal Sramka is with the UNESCO Chair in Data Privacy at the Department of Computer Engineering and Maths, Rovira i Virgili University, Av. Paisos Catalans, 26, 43007 Tarragona, Spain (email: michal.sramka@urv.cat).

The author is with the UNESCO Chair in Data Privacy, but he is solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organization.

This work was partly supported by the Spanish Government through projects TSI2007-65406-C03-01 "E-AEGIS" and CONSOLIDER INGENIO 2010 CSD2007-00004 "ARES", and by the Government of Catalonia under grant 2009 SGR 01135.

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

techniques are more general and cover a larger class of attacks. Plus, we are still able to demonstrate that even these non-expert adversaries and their simple techniques breach privacy.

Our extensive experiments are performed on four different data sets derived from the UCI Machine Learning Repository's Adult and IPUMS97 census-based databases [4]. The ages of the individuals are perturbed using ϵ-differential privacy [5], [6] that adds noise chosen from a Laplace distribution. We simulate the adversary by applying six data mining algorithms from the Weka software package [1] and fuse the outputs using a simple fusion strategy. We evaluate the success of the proposed attack by measuring the amount of noise that is removed by considering the fused values as age estimates, and the "nearness" of these estimates to the original values of age. We require a high privacy from the perturbation: $\epsilon \in \{0.5, 0.1, 0.05, 0.01\}$. The similar leakages between 0.0001 to 1 have been used by one of the differential privacy co-inventors in [7], although some other illustrations in literature only use weaker privacy with leakage $\epsilon = 2$.

Even considering only the non-expert adversaries that use general-purpose data mining techniques and a simple fusion that computes the average, the experiments show that 77%–93% of the added noise can be removed. An important observation we make from the experiments is that the higher the privacy, the higher percentage of the noise can be removed. In other words, adding more noise does not necessary increase privacy in this scenario. Overall, the experimental results confirm that our attack presents a significant risk to statistical disclosure and privacy. The risk is substantial also because no background knowledge is used in the attack, and thus any background knowledge can only aid the adversary.

We note that we do not understand the experimental results as a fallacy of the theoretical guarantees of differential privacy. The differential privacy notion is an important and a strong theoretical privacy definition. It is based on probabilistic concepts. As with every probabilistic and statistical tools, one has to be careful about how are the tools applied. We consider scenarios with specific query functions that may be typically chosen by data owners to protect privacy. We show that in these scenarios it is possible to remove the majority of the added noise from perturbed released data. Our attack is applicable to ϵ-differentially perturbed data as well as to data sanitized with other perturbation techniques. Finally, we briefly discuss some countermeasures that should be considered in real-world applications to avoid these kinds of attacks.

B. Related Work

Data perturbation and noise addition has been identified early for achieving data privacy [8], [9]. Uniform and normal (Gaussian) distributions are used in [10] to sample the additive noise, while exponential (Laplace) distribution is used to achieve differential privacy [5], [6]. Differential privacy is a well-studied notion [11], [12], [13], [14], [15], [16], which provides strong theoretical guarantees about the privacy of the released data. Differential privacy is usually achieved by noise addition, however a recent research claims that noise addition is not enough [17]. We use differential privacy throughout this paper as well as in our experiments.

Data mining has been traditionally considered in the area of data privacy for measuring the usefulness of the released data [10], [18], [19]. It has also been used to measure not just the usefulness of the released data but also the amount of privacy various sanitization methods provide [20]. Data mining has also been used to attack the privacy of the data [21], [22]. Similarly, we use data mining to attack the privacy by removing portions of the added noise. In other words, we use data mining not to evaluate the usefulness of the released sanitized data, but to attack the privacy of the data.

It is possible to statistically estimate original values [8], [9] when repeating queries and obtaining different noisy values. In contrast, if the noise is fixed for the same value [10], an adversary is unable to improve the estimate using just the simple statistical methods – we propose an attack to estimate the original values in this case using data mining.

Similar to our attack, [23] presents an attack that obtains disclosures by approximating the original data values from matrices distorted by addition of random values. The authors use matrices and a specific matrix-based spectral filtering techniques to obtain disclosures, while we use relational databases and simple general-purpose data mining techniques.

Combining data from several measurements is referred to in the literature as data fusion, data aggregation, data consolidation, and data integration. We only use a simple fusion strategy to combine outputs of multiple data miners, but there is a lot of research in the field of fusion and verification methods [2], [3], [24], [25].

II. Differential Privacy

A *database* DB is a set of tuples, where a *tuple* $x = (x_1, \ldots, x_t)$ consists of *fields* x_i's defined over some finite domains (categorical data) or infinite domains (numerical data). This concept conforms to the traditional relational database definition. We refer to the database DB as the *original data* and assume it is in possession and control of a *data owner*.

A *sanitization mechanism* $\mathcal{S}$ is a randomized algorithm that for a database DB and query f returns $\mathcal{S}(f(DB))$, a transformation of the query result $f(DB)$ that conforms to the "privacy guarantee" of $\mathcal{S}$. A *privacy guarantee* of $\mathcal{S}$ is a notion, a description of what is meant and provided by the sanitization mechanism and how sensitive data, identities, and sensitive relations are protected by $\mathcal{S}$.

For simplicity, in the rest of the paper, we assume that the data owner uses a sanitization mechanism $\mathcal{S}$ to release the original data DB without suppressing any tuples. We further suppose, although this time without loss of generality, that suppressed fields are replaced with dummy values. Hence, there is a one-to-one correspondence between the fields and tuples of DB and their sanitized versions. Then by $DB^\star$,

$x^\star$, and $x_i^\star$, respectively, we denote the database DB, tuple x, and field x_i after sanitization. We refer to $DB^\star$ as the *sanitized data* or *perturbed data*, and we assume this data is public – available to *data users* that can be legitimate (analysts) or malicious (adversaries).

The *differential privacy* is a notion based on the indistinguishability of transcripts (messages and outputs of a data user interacting with a sanitization mechanism). In the database setting, it is typically achieved by adding random Laplace noise to numeric values, and it effectively limits the amount of necessary noise needed for achieving privacy. The privacy guarantee of the differential privacy notion is semantic, that is, it considers the shifts and incremental gain in the adversary's knowledge.

Definition (ϵ-differential privacy [5]). A randomized function $\mathcal{S}$ gives *ϵ-differential privacy*, if for all databases DB_1 and DB_2 differing on at most one element and all $R \subseteq Range(\mathcal{S})$ holds that

$$\Pr[\mathcal{S}(DB_1) \in R] \leq \exp(\epsilon) \cdot \Pr[\mathcal{S}(DB_2) \in R] \ .$$

The ϵ-differential privacy notion limits the probability that the randomized function (sanitization mechanism) $\mathcal{S}$ would leak information from a database that is differing by at most one element. Therefore a data leak and a disclosure of a private data through $\mathcal{S}$ is possible, but the probability of such leak is limited through the *leakage* parameter ϵ.

In practice, the sanitization mechanism achieving ϵ-differential privacy for numerical values works by adding noise to the query result $f(DB)$, a noise that is randomly chosen from the Laplace (exponential) distribution $\mathrm{Lap}(0, \Delta f/\epsilon)$ with the probability density function

$$h(z) = \frac{\epsilon}{2} \cdot \exp(-\frac{\epsilon|z|}{\Delta f}) \ ,$$

where Δf is the sensitivity [5] of the query function f defined as

$$\Delta f = \max_{DB_1, DB_2} \|f(DB_1) - f(DB_2)\|_1$$

with DB_1 and DB_2 again ranging over all databases differing on at most one element.

If fresh noise is sampled and added to each query result, then it is possible to statistically estimate the original values [8], [9] after repeating the same queries. Therefore, it has been suggested (e.g., in [9], [10]) to fix the noise for the already seen values and only add freshly sampled noise for the new, previously unseen values. Similarly, our noise-adding function "noise", for a fixed query f and a fixed leakage ϵ, depends only on the original value, and returns the same noise for already seen input values and for new input values it returns a fresh sample from the Laplace distribution $\mathrm{Lap}(0, \Delta f/\epsilon)$. If v is the original sensitive value that needs to be protected and $v^\star$ denotes the perturbed protected value, then

$$v^\star := v + \mathrm{noise}(v) \ .$$

This thwarts the statistical attack, as it does not allow an adversary to improve the estimates by repeating queries. We also focus on this scenario and show that data mining can improve the estimates of the original values.

III. The Attack Model

To make the explanation more concrete, we present the attack model on an example that uses ages. However, the attack would work for any attribute.

A data owner wants to release its data collection about individuals. The data owner considers the individuals' ages to be a sensitive information. The owner protects the privacy of the ages by adding random noise limited by the ϵ-differential privacy to all the ages individually, and then releases the sanitized data. Formally, suppose that the original data is DB, and that noise is added during sanitization to x_m, the m-th field of every tuple $x \in DB$ that represents age. The added noise is different for distinct ages, but fixed for the same values of age. The newly formed sanitized tuples $x^\star$'s are then released as the sanitized data $DB^\star$. The sanitized data can then be accessed by the data users – legitimate data users (analysts) as well as malicious data users (adversaries).

An *adversary* is an end user of the sanitized data with malicious intentions, namely trying to make disclosures and breach privacy. The adversary launches the attack by applying one or more data mining algorithms to uncover hidden trends and patterns in the data that would allow him/her to partially remove the added noise and obtain quality estimates of the ages.

The attack consists of first obtaining predictions for the ages that have been perturbed and that are of the interest to the adversary. Data miners can provide such predictions. However, multiple data miners produce multiple predictions. Moreover, there may be different predictions, based on individual records in the database, for the same original value of age. Therefore the adversary combines the different predictions obtained from multiple miners using fusion methods. The adversary's intention is to use all these predictions and combinations to obtain reasonable estimates of the ages – estimates that contain much less noise than the perturbed values and possibly estimates that breach privacy.

A. Noise Removal Using Data Mining and Fusion

A data miner or simply a *miner* $\mathcal{M}$ is an algorithm that takes the sanitized data $DB^\star$ as the input together with one sanitized tuple $x^\star$. The miner determines trends and patterns in this data and outputs a prediction for the m-th field $x_m^\star$ of the tuple

$$x^\star = (x_1^\star, \ldots, x_m^\star, \ldots, x_t^\star) \ .$$

We consider m to be an implicit parameter of the miner $\mathcal{M}$. The adversary possesses a set of n miners $M = \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$, each predicting the same field m of any input tuple. The *prediction* for the m-th field of the tuple $x^\star$ obtained by the miner $\mathcal{M}_i$ is denoted by p_i, that is,

$$p_i \leftarrow \mathcal{M}_i(DB^\star, x^\star) \ .$$

After the adversary performs all the mining, the adversary will obtain n predictions for the m-th field for each tuple $x^\star \in DB^\star$.

Note that the miners $\mathcal{M}_i$'s can be any data mining algorithms, for example, they can be selected and parametrized specifically for the type of data in $DB^\star$. However, a selection of a few general-purpose miners that are freely available and are easy to use would suffice in practice to launch a successful attack.

At this step of the attack, when considering any single miner $\mathcal{M}_i$, the adversary possesses a prediction p_i for the field m for every tuple $x^\star \in DB^\star$. A single original value x_m, that is perturbed into

$$x^\star_m := x_m + \text{noise}(x_m) \ ,$$

may appear in multiple tuples in the database DB. For example, think about age – there are about 120 ages that are repeated for individuals in the database. Consequently, because of the noise function, the same perturbed value $x^\star_m$ appears in multiple tuples in the database $DB^\star$. Therefore predictions for the same original value x_m may be different over different tuples $x^\star$ having the same value $x^\star_m$, because the predictions are based on all the fields of the tuple $x^\star$. Thus the adversary possesses different predictions for each single $x^\star_m$. The adversary proceeds with the attack by first fusing these different predictions for each distinct value of $x^\star_m$ and then the adversary fuses the values from different miners.

Let Y be the set of all distinct values x_m from all the tuples in DB. Then clearly

$$Y^\star := \{x^\star_m \mid x^\star_m = x_m + \text{noise}(x_m) \ : \ x_m \in Y\}$$

is the set of all distinct perturbed values. Because of the randomness of the noise selection in practice, we can assume that no two distinct original values are perturbed into the same perturbed value, which is the same as assuming that $|Y| = |Y^\star|$.

A *fusion* is an algorithm that takes multiple inputs, fuses them, and outputs a single combined value. The adversary uses a fusion algorithm $\mathcal{F}_1$ to combine the different predictions by a single miner that were obtained for the same value of $x^\star_m$. The adversary does this for all distinct perturbed values and for all miners. Algorithmically, for each miner $\mathcal{M}_i \in M$, and for each each distinct perturbed value $y \in Y^\star$, the adversary computes

$$q_{i,y} \leftarrow \mathcal{F}_1(\{\ p_i \mid p_i = \mathcal{M}_i(DB^\star, x^\star) \ : \\ x^\star \in DB^\star \text{ having } x^\star_m = y\ \}) \ .$$

This computation results in n predictions (one for each miner) for each distinct perturbed value $x^\star_m$.

Finally, the adversary fuses the multiple predictions from multiple miners into a single prediction r_y for each perturbed value $y \in Y^\star$ using a fusion algorithm $\mathcal{F}_2$ as follows:

$$r_y \leftarrow \mathcal{F}_2(\{\ q_{i,y} \mid i = 1, \ldots, n\ \}) = \mathcal{F}_2(q_{1,y}, \ldots, q_{n,y}) \ .$$

The fused value r_y represents an *estimate* of the perturbed value y.

The adversary ends with a database derived from $DB^\star$, where $x^\star_m$ of each tuple is replaced with its estimation $r_{x^\star_m}$. This concludes the attack.

The fusion algorithms $\mathcal{F}_1$ and $\mathcal{F}_2$ can be based on any fusion strategy [2], [3], [24], [25]. We use a simple average for all fusions in our experiments, while still able to demonstrate significant negative impact on privacy.

B. Evaluation of the Attack

The adversary is clueless with regard to which predictions and estimates are correct, if any at all. We evaluate the success of the attack from the privacy viewpoint, that is, if there are privacy breaches and disclosures. This evaluation covers the impact the attack has for the data owner trying to protect the privacy of the data as well as for the adversary who is interested to remove the noise from the perturbed data.

There are several possibilities to evaluate the success of the attack. We focus on how much of the privacy protection provided by the data perturbation is lost due to the attack. We have previously developed [20], [22] a comprehensive way to measure success of the data mining methods over sanitized/perturbed data based on the data mining utility. The previously proposed utility functions could measure the success of predictions, and are flexible enough to measure the success of this attack. But contrary to our needs in this attack, the previously proposed utility functions measure and evaluate the success of predictions per each tuple (per each record) in the database. Here we concentrate on measuring of how much noise is being removed by predictions from the perturbed values that may be the same for many tuples. We borrow the "nearness" concept from [20], [22] and also evaluate the success by looking at the amounts of added and remaining noise.

First, we focus on the success of the estimation of the distinct values $x^\star_m$'s rather than the impact for all the tuples in $DB^\star$ individually. An estimation $r_{x^\star_m}$ is *δ-nearer* with respect to the original value x_m, if

$$d(r_{x^\star_m}, x_m) \leq (1 - \delta) \cdot d(x^\star_m, x_m) \ ,$$

where d is a distance function, for example, the Euclidean distance function. We say an estimation $r_{x^\star_m}$ is *nearer* w.r.t. x_m, if it is δ-nearer for any $\delta > 0$, which translates into strict inequality:

$$d(r_{x^\star_m}, x_m) < d(x^\star_m, x_m) \ .$$

In other words, the "nearness" tells us whether the estimation $r_{x^\star_m}$ is a better representation for x_m than the perturbed value $x^\star_m$. For example, age 20 is perturbed into 31.4. Then 25 is a nearer estimate, and so is 18. Age 32 is not a nearer estimate. Age 21 is a 0.9-nearer (or 90%-nearer) estimate, because it lies within the $1/10^{\text{th}}$ of the distance $|31.4 - 20|$ from the original age 20. Our measure of success of the noise-removing attack is the number of estimates that are

nearer in comparison to all nearer and non nearer estimates, and we will also look at the δ factor of these estimates.

For our other measure of success we compute the amount of noise that has been added to the different values during perturbation and compare it with the amount of noise that remains in the estimated values. The *cumulative added noise* is

$$\text{added noise} := \sum_{x_m \in Y} |x_m^\star - x_m| \ ,$$

where we decided to sum over all distinct values rather than over all tuples $x \in DB$, and the *cumulative remaining noise* after the attack is

$$\text{remaining noise} := \sum_{x_m \in Y} |r_{x_m^\star} - x_m| \ .$$

Our second measure of the success of the noise-removing attack is the ratio of the remaining noise to the added noise.

Existing verification and validation methods for data mining and fusion techniques [2] can help the adversary to evaluate the success of the attack, even without having access to the original data. Note that our evaluation cannot be computed by the adversary because it uses the original values not available to the adversary. The same techniques can suggest to the adversary which of the data miner is the best for the noise removal. However, we abstracted for the need to identify the best miner, as we fuse the various predictions. In practice, a simple fusion consisting of computing averages turns out to be very successful.

IV. Experimental Results

Our experiments demonstrate the power of our simple privacy attack. The attack is successful regardless of the choice of a database and even for privacy parameters believed to provide very strong privacy guarantees. More than 84% of all estimates are "nearer", so privacy-impacting. These estimates are more than 70%-nearer on average and they collectively remove between 77%–93% of the noise from perturbed values. Interestingly, the smaller the leakage ϵ (the higher the privacy), the higher the percentage of the removed noise. Furthermore, our attack uses only simple tools and techniques, so we believe even more nearer estimations can be obtained and more noise can be removed if specific tailored miners and fusion strategies are used.

We performed the experiments on four databases derived from two census-based UCI's Machine Learning Repository [4] databases, namely the Adult and IPUMS databases. We sanitized the age of the individuals in these databases using data perturbation (noise addition) limited by the ϵ-differential privacy notion [5]. We used $\epsilon = 0.5, 0.1, 0.05$, even 0.01, which represents increasingly stronger privacy, similar to [7] and much stronger than $\epsilon = 2$ used in some other experiments.

We simulated the adversary by using six miners from the Weka software package [1] and by using simple average as the fusion strategy to combine multiple values. Thus the comprehensive simulation of attacks demonstrate that even simple techniques (average) and readily-available tools (Weka miners) can remove the majority of the noise that have been added to individuals' ages during perturbation.

A. Data and Sanitization

We derived four different databases from the UCI Machine Learning Repository's Adult and IPUMS97 [4] census-based data:

1) *Adult-UCI* consisting of 45,222 records was the combination of learning and testing data of the Adult data set, excluding the records containing missing values.
2) *Adult-Iyengar* was derived from Adult-UCI by removing some attributes as proposed in [18]. This data is a "benchmark" for experimental testing in the privacy field.
3) *IPUMS97-num* consisting of 70,187 records was a sample of Los Angeles and Long Beach area citizens from 1997. All attributes were considered to be numerical.
4) *IPUMS97-cat* was the same as IPUMS97-num, except all the attributes but age were considered to be categorical.

The databases contained census-based information about individuals. In particular, individuals are described using several attributes that differ over the databases, but in all cases the attributes include *age*. All the data and attributes are described in detail in [4] and [18], respectively.

To obtain the perturbed sanitized data, we used ϵ-differential privacy in order to limit the noise that was being added to the attribute *age* of each of the four databases. We used leakage $\epsilon = 0.5$, 0.1, 0.05, and 0.01, which all represent a strong privacy protection. Then the additive noise was chosen randomly from the Laplace distribution $\text{Lap}(0, \Delta f/\epsilon)$ with the probability density function $h(z) = (\epsilon/2)\cdot\exp(-\epsilon|z|/\Delta f)$, where Δf is the sensitivity [5] of the query function f, which basically describes how much can query results differ on two queries. The query function f in our case was a function which "selects" age, so $\Delta f = 73$ for the two Adult databases, where the ages ranged between $17 - 90$, and $\Delta f = 100$ for the two IPUMS97 databases, where the ages ranged between $0 - 100$. The noise was fixed for each age, so that observations of multiple records cannot improve an adversary's estimate of the age [9], [10]. Using the notation developed in the previous sections, the original age in any database tuple x is denoted as x_{age}, and the perturbed age in a sanitized tuple $x^\star$ is denoted as $x^\star_{age}$.

B. Attacks

The following six miners representing three different types of miners were selected from the Weka [1] software package, version 3.6.1, and used in the attack:

- function-based: RBFNetwork ($\mathcal{M}_1$)
- rule-based: ConjunctiveRule ($\mathcal{M}_2$), DecisionTable ($\mathcal{M}_3$), and ZeroR ($\mathcal{M}_4$)
- tree-based: DecisionStump ($\mathcal{M}_5$) and REPTree ($\mathcal{M}_6$)

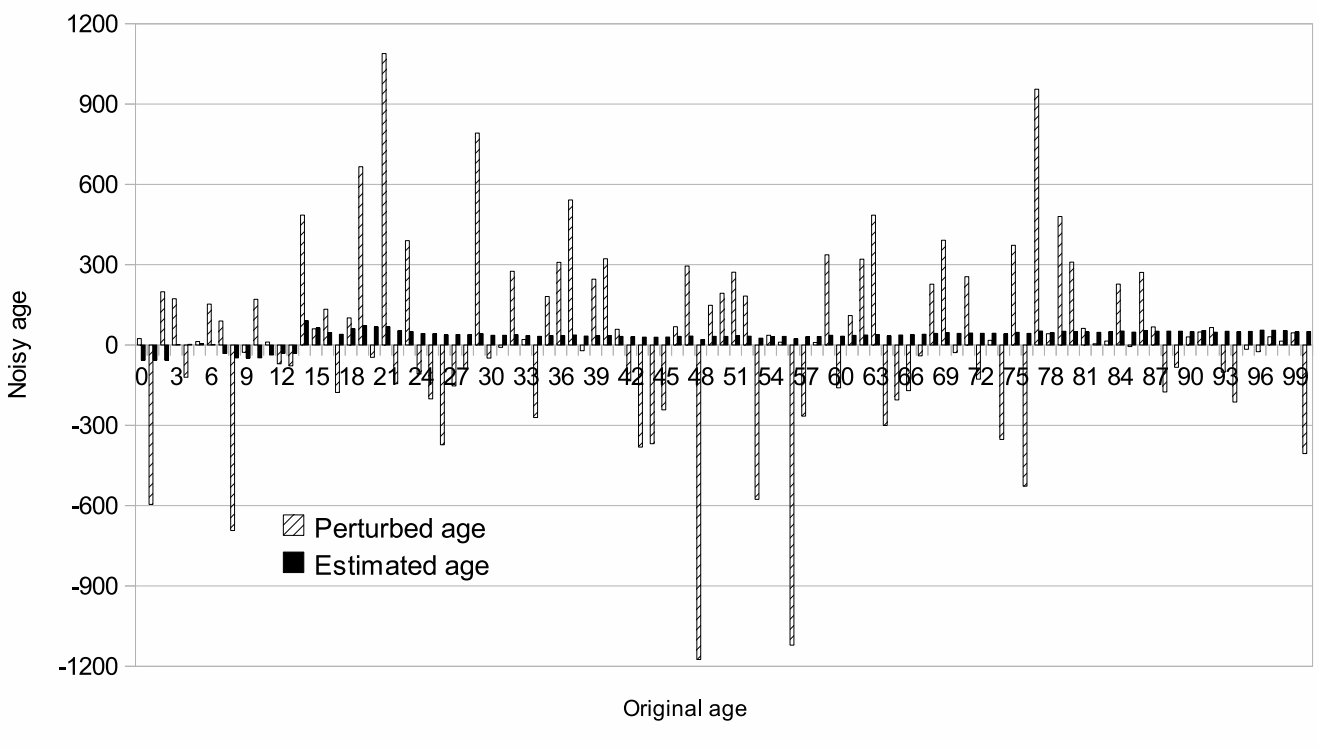


Fig. 1. Original, perturbed and estimated ages of the IPUMS97-cat database sanitized with perturbation satisfying ($\epsilon = 0.5$)-differential privacy.

As mentioned earlier, these are general-purpose miners. The Weka user interface allows adversaries to use them with no or minimum knowledge about data mining.

Because fresh noise was added only to new values and fixed for the same values, an adversary could identify the records $x^\star$ in the perturbed released data that have the same age $x^\star_{age}$ and compute the average of the corresponding predictions. Note that although the added noise was same for the same age values, the data mining predictions may have varied even for the same age, because the predictions were based on the whole tuple $x^\star$.

Using the notation from Section III-B, the set Y consisted of all ages in the database and the set $Y^\star$ of all the perturbed ages: $|Y| = |Y^\star| = 74$ for the two Adult databases and $|Y| = |Y^\star| = 101$ for the the IPUMS97 databases. The fusion algorithm $\mathcal{F}_1$ simply took all the possibly various predictions p_i's by one miner $\mathcal{M}_i$ that were obtained by this miner over all those tuples $x^\star$ that have the same perturbed age $y := x^\star_{age}$ and computed the average. Formally, $q_{i,y}$ is computed as follows:

$$q_{i,y} := \sum \frac{p_i}{c} \ ,$$

where the sum is over all those p_i's such that

$$p_i := \mathcal{M}_i(DB^\star, x^\star)$$

computed for all those $x^\star$ that have the age attribute value equal to y ($x^\star_{age} = y$) and where c is the number of such $x^\star$'s. This was repeatedly computed for all distinct perturbed ages y from among $x^\star_{age}$'s, and for all miners $\mathcal{M}_i$, $i = 1, \ldots, 6$, resulting in six predictions $q_{1,y}, \ldots, q_{6,y}$ for each distinct perturbed age $y = x^\star_{age} \in Y^\star$.

The second fusion algorithm $\mathcal{F}_2$ took the six combined predictions obtained as a result of mining with six miners, and fused them once more, again using simple average:

$$r_y := \frac{q_{1,y} + \cdots + q_{6,y}}{6} \ .$$

The average $r_{x^\star_{age}}$ represented a very simple fusion of multiple predictions, and we considered it to be an estimate of the original age x_{age}.

C. Evaluation and Discussion

Nearer estimates comprise a privacy breach. We considered an estimate $r_{x^\star_{age}}$ to be nearer with respect to the original age x_{age} if

$$|r_{x^\star_{age}} - x_{age}| < |x^\star_{age} - x_{age}| \ .$$

Table II presents the number of nearer estimates from the attack. Between 84%–94% out of all estimates are nearer. Not all estimates are nearer, because probably there was not enough data for some ages in the database and so the data miners could not always produce highly accurate and confident predictions. Some additional evidence about this is explained next.

Figure 1 shows the scenario in the case of the IPUMS97-cat database sanitized with perturbation satisfying ($\epsilon = 0.5$)-differential privacy. The x-axis consist of the original ages x_{age}'s. The y-axis shows the perturbed ages $x^\star_{age}$'s and the estimated ages from the attack $r_{x^\star_{age}}$'s. From

TABLE I

AMOUNT OF NOISE OF PERTURBED AGES (ADDED NOISE) AND PREDICTED AGES (REMAINING NOISE)

Perturbation	**Amount of**	*Adult-UCI*	*Adult-Iyengar*	*IPUMS97-cat*	*IPUMS97-num*
$\epsilon = 0.5$	added noise	12,617	12,617	23,970	23,970
	remaining noise	2,832	2,788	2,811	3,431
	reduction	77.55%	77.90%	88.27%	85.69%
$\epsilon = 0.1$	added noise	63,087	63,087	119,848	119,848
	remaining noise	9,291	9,057	9,222	10,819
	reduction	85.27%	85.64%	92.31%	90.97%
$\epsilon = 0.05$	added noise	126,174	126,174	239,695	239,695
	remaining noise	17,423	16,947	16,696	21,128
	reduction	86.19%	86.57%	93.03%	91.19%
$\epsilon = 0.01$	added noise	630,869	630,869	1,198,477	1,198,477
	remaining noise	82,415	80,323	79,046	105,524
	reduction	86.94%	87.27%	93.40%	91.20%

TABLE II

NUMBER AND PERCENTAGE OF "NEARER" ESTIMATES (OUT OF POSSIBLE 74 FOR THE ADULT AND 101 FOR IPUMS97 DATABASES)

DB \ Perturb.	$\epsilon = 0.5$	$\epsilon = 0.1$	$\epsilon = 0.05$	$\epsilon = 0.01$
Adult-UCI	62 (84%)	67 (91%)	67 (91%)	66 (89%)
Adult-Iyengar	62 (84%)	67 (91%)	68 (92%)	67 (91%)
IPUMS97-cat	93 (92%)	94 (93%)	94 (93%)	95 (94%)
IPUMS97-num	88 (87%)	94 (93%)	94 (93%)	94 (93%)

TABLE III

AVERAGE δ FACTOR OF "NEARER" ESTIMATES

DB \ Perturb.	$\epsilon = 0.5$	$\epsilon = 0.1$	$\epsilon = 0.05$	$\epsilon = 0.01$
Adult-UCI	70.54%	74.73%	76.42%	79.08%
Adult-Iyengar	71.04%	75.51%	76.08%	78.28%
IPUMS97-cat	78.89%	85.82%	88.13%	88.50%
IPUMS97-num	77.91%	87.10%	87.08%	86.28%

the chart, one can see that the estimation of ages between 0–13 (solid black columns) was mostly problematic, likely due to the insufficiency of rich information in the data about these ages and consequently the inability of data miners to produce quality estimates. Similar situation can be seen for ages between 14–23, where the estimates are too high. For the rest of the ages, the estimates are pretty close to the original age, considering the amount of noise that has been added to some of them. Similar charts can be shown for other databases and stronger privacy parameters, but stronger privacy means that the perturbed ages would dwarf the estimated ages too much.

In addition to counting the nearer estimates, we also looked at the quality of the "nearness". We considered an estimate $r_{x^\star_{age}}$ to be δ-nearer w.r.t. x_{age} if

$$|r_{x^\star_{age}} - x_{age}| \leq (1-\delta) \cdot |x^\star_{age} - x_{age}| \ .$$

We computed the average δ of the nearer estimates. We present the results in Table III. The higher the δ, the better the estimate. For example, a 70.54%-nearer prediction of an age is one that is within radius of 29.46% of the added noise to that age. Except for the IPUMS97-num database, that is 3 times out of 4, the higher privacy (smaller ϵ) resulted in closer estimates on average.

Table I shows cumulative amount of noise that was added for each database and each leakage parameter of the ϵ-differential perturbation, together with the amount of noise that the data contained after perturbed ages $x^\star_{age}$'s were estimated with predicted and averaged values $r_{x^\star_{age}}$'s. Overall, between 77.55%–93.4% of the added noise was removed due to our attack. It was more for the two IPUMS97 databases than for the two Adult databases, perhaps because the IPUMS97 databases contained more attributes and records, so the data miners provided better predictions. More interestingly, the smaller the leakage ϵ ($\epsilon : 0.5 \rightarrow 0.1 \rightarrow 0.05 \rightarrow 0.01$), the increasingly larger percentage of noise reduction. A smaller leakage ϵ means higher privacy in the terms of the differential privacy notion, that is, more noise is being added to mask the ages. But the more noise is being added during perturbation, the higher fraction of the added noise is being removed by our attack. So a major observation we make is: The higher the privacy, the more successful is our attack, which means that in the presented experimental scenario, adding more noise does not increase privacy.

V. CONCLUSIONS

Data perturbation is a sanitization method that helps restrict the disclosure of sensitive information from published data, usually achieved by noise addition. We presented an attack which removes the majority of such added noise from the perturbed released data. The attack employs multiple data miners to make prediction about sensitive perturbed values, and it uses fusion algorithms to combine the like data and multiple predictions.

Our proposed attack is practical. It can be launched by non-expert adversaries – adversaries that have no deep

knowledge about data mining or fusion techniques. The attack is also significant, because it does not require any additional knowledge (background knowledge). Although a simple form of the attack is used in experiments, we are still able to show the negative privacy implications and vulnerabilities of data perturbation in the presented scenarios.

Stronger adversaries, using tailored data mining techniques, advanced fusion strategies, or having and using background knowledge about the data at hand will be obviously able to obtain better results and make more disclosures. Our contribution is in showing that even simple attacks and non-expert adversaries are successful. The consequent question is then not how to improve the attack, but rather how to avoid it and make data publication private from malicious parties yet useful to legal data analysts.

Our experimental results indicate that the higher the privacy, the higher the percentage of noise can be removed. This suggests that adding more noise does not always increase privacy in these scenarios. The only countermeasure we have at this time is to completely mask not just the sensitive attributes of a database, but also other attributes in the data. Improper masking of relations in the records in a database results in the ability of data mining methods to make privacy breaches. On the other hand, sanitizing the data too much may render it unusable for legal data analysts. Hence, the fundamental question about balancing the amount of sanitization and the potential usefulness of the sanitized data remains an open problem.

REFERENCES

[1] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd ed. San Francisco, CA, USA: Morgan Kaufmann, 2005.

[2] V. Torra and Y. Narukawa, *Modeling Decisions: Information Fusion and Aggregation Operators*, ser. Cognitive Technologies. Springer-Verlag, 2007.

[3] J. Bleiholder and F. Naumann, "Data fusion," *ACM Computing Surveys*, vol. 41, no. 1, pp. 1–41, 2008.

[4] A. Asuncion and D. J. Newman, "UCI Machine Learning Repository," 2007. [Online]. Available: http://www.ics.uci.edu/∼mlearn/MLRepository.html

[5] C. Dwork, "Differential Privacy," in *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP 2006), Part II*, ser. LNCS 4052. Venice, Italy: Springer-Verlag, 2006, pp. 1–12.

[6] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating Noise to Sensitivity in Private Data Analysis," in *Proceedings of the 3rd Theory of Cryptography Conference (TCC 2006)*, ser. LNCS 3876. New York, NY, USA: Springer-Verlag, 2006, pp. 265–284.

[7] F. McSherry, "Preserving privacy in large-scale data analysis," A presentation at Workshop on Algorithms for Modern Massive Data Sets (MMDS 2006), Stanford, CA, USA, June 21-24, 2006. [Online]. Available: http://www.stanford.edu/group/mmds/slides/mcsherry-mmds.pdf

[8] T. Dalenius, "Towards a Methodology for Statistical Disclosure Control," *Statistisk Tidskrift*, vol. 15, pp. 429–444, 1977.

[9] N. A. Adam and J. C. Wortmann, "Security-control methods for statistical databases," *ACM Computing Surveys*, vol. 21, no. 4, pp. 515–556, 1989.

[10] R. Agrawal and R. Srikant, "Privacy-Preserving Data Mining," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. Dallas, Texas, USA: ACM Press, 2000, pp. 439–450.

[11] A. Blum, C. Dwork, F. McSherry, and K. Nissim, "Practical privacy: the SuLQ framework," in *Proceedings of the 24nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2005)*. Baltimore, Maryland, USA: ACM Press, 2005, pp. 128–138.

[12] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith, "What Can We Learn Privately?" in *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008)*. Philadelphia, PA, USA: IEEE Computer Society, 2008, pp. 531–540.

[13] C. Dwork, "Differential Privacy: A Survey of Results," in *Proceedings of the 5th International Conference on Theory and Applications of Models of Computation (TAMC 2008)*, ser. LNCS 4978. Xi'an, China: Springer-Verlag, 2008, pp. 1–19.

[14] ——, "The Differential Privacy Frontier," in *Proceedings of the 6th Theory of Cryptography Conference (TCC 2009)*, ser. LNCS 5444. San Francisco, CA, USA: Springer-Verlag, 2009, pp. 496–502.

[15] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. P. Vadhan, "On the complexity of differentially private data release: efficient algorithms and hardness results," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC 2009)*. Bethesda, MD, USA: ACM Press, 2009, pp. 381–390.

[16] I. Mironov, O. Pandey, O. Reingold, and S. Vadhan, "Computational Differential Privacy," in *Proceedings of 29th Annual International Cryptology Conference, Advances in Cryptology (CRYPTO 2009)*, ser. LNCS 5677. Santa Barbara, CA, USA: Springer-Verlag, 2009, pp. 126–142.

[17] K. Muralidhar and R. Sarathy, "Differential Privacy for Numeric Data," Joint UNECE/Eurostat work session on statistical data confidentiality, Bilbao, Spain,, 2009.

[18] V. S. Iyengar, "Transforming data to satisfy privacy constraints," in *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2002)*. Edmonton, Alberta, Canada: ACM Press, 2002, pp. 279–288.

[19] M. Sramka, R. Safavi-Naini, J. Denzinger, M. Askari, and J. Gao, "Utility of Knowledge Extracted from Unsanitized Data when Applied to Sanitized Data," in *Proceedings of the 6th Annual Conference on Privacy, Security and Trust (PST 2008)*. Fredericton, New Brunswick, Canada: IEEE Computer Society, 2008, pp. 227–231.

[20] M. Sramka, R. Safavi-Naini, J. Denzinger, and M. Askari, "A Practice-oriented Framework for Measuring Privacy and Utility in Data Sanitization Systems," in *Proceedings of the 12th International Conference on Extending Database Technology Workshops (EDBT-PAIS 2010)*. Lausanne, Switzerland: ACM Press, 2010.

[21] A. Valls, V. Torra, and J. Domingo-Ferrer, "Semantic based aggregation for statistical disclosure control," *International Journal of Intelligent Systems*, vol. 18, no. 9, pp. 393–951, 2003.

[22] M. Sramka, R. Safavi-Naini, and J. Denzinger, "An Attack on the Privacy of Sanitized Data That Fuses the Outputs of Multiple Data Miners," in *Proceedings of the 9th IEEE International Conference on Data Mining Workshops (ICDM-PADM 2009)*. Miami Beach, FL, USA: IEEE Computer Society, 2009, pp. 130–137.

[23] H. Dutta, H. Kargupta, S. Datta, and K. Sivakumar, "Analysis of privacy preserving random perturbation techniques: further explorations," in *Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society (WPES 2003)*. Washington, DC, USA: ACM Press, 2003, pp. 31–38.

[24] V. Torra, Ed., *Information Fusion in Data Mining*, ser. Studies in Fuzziness and Soft Computing. Springer-Verlag, 2003, vol. 123.

[25] I. R. Goodman, R. P. Mahler, and H. T. Nguyen, *Mathematics of Data Fusion*. Norwell, MA, USA: Kluwer Academic Publishers, 1997.